phyCORE®-PXA270 with UML

QuickStart Instructions

Using the UML Tool Rhapsody from Telelogic and the GNU Cross Development Tool Chain

Note: The PHYTEC UML-XScale-Disc includes the electronic version of the English phyCORE®-PXA270 Hardware Manual

Edition: November 2006

A product of a PHYTEC Technology Holding company

In this manual are descriptions for copyrighted products that are not explicitly indicated as such. 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.

	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

2st Edition: November 2006

1 Introduction

5 min

In this QuickStart you can find general information on the PHYTEC phyCORE®-PXA270 Single Board Computer (SBC), an overview of Telelogic's UML-Tool and the GNU Cross compiler, and instructions on how to run example programs on the phyCORE®-PXA270, mounted on the PHYTEC phyCORE® Development Board PXA270 in conjunction with UML-Tool Rhapsody from Telelogic.

Please refer to the phyCORE®-PXA270 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" (RDK) includes the following electronic documentation on the enclosed "PHYTEC UML-XScale-Disc":

- the PHYTEC phyCORE®-PXA270 Hardware Manual
- controller User's Manuals and Data Sheets

1.2 Professional Support Packages available

This Kit comes with free installation support. If you do 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.

1.3 Overview of this QuickStart Instruction

This QuickStart Instruction gives a general "Rapid Development Kit" description, as well as software installation hints and an example program enabling quick out-of-the box start-up of the phyCORE®-PXA270 in conjunction with the GNU Cross compiler development tool chain and Rhapsody. It is structured as follows:

1) The *"Getting Started"* section describes how to configure the host platform and how download an embedded application to the PXA-270.

2) The *"Getting More Involved"* section provides a step-by-step instructions on how to create and build new projects, generate and download output files to the phyCORE®-PXA270 using Rhapsody and modify the example.

3) The *"Design Level Debugging"* section provides an example how to debug an application running on the PXA-270 with Rhapsody on the host PC at "design level".

4) The "*Advanced Debugging*" section provides information on how to debug your application at "code level"

In addition to dedicated data for this Rapid Development Kit, the PHYTEC UML-XScale-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 be 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 of the time to pass the following chapter.

You have successfully passed an important part of this QuickStart.

You can find information to solve problems.

1.5 System Requirements

Use of this "Rapid Development Kit" requires:

- the PHYTEC phyCORE®-PXA270
- the phyCORE® Development Board PXA270 with the included DB-9 serial cable, Ethernet crosslink cable and AC adapter supplying 12 VDC /min. 1A
- Rhapsody Demo version
- PHYTEC Distribution based on OSELAS from Pengutronix
- an IBM-compatible host-PC (486 or higher running Linux)

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

PHYTEC

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

1.6 The PHYTEC phyCORE®-PXA270

The phyCORE®-PXA270 represents an affordable yet highly functional Single Board Computer (SBC) solution in sub-miniature dimensions (70 x 57 mm). Its universal design enables its insertion in a wide range of embedded applications.

All controller signals and ports extend from the controller to high-density (0.635 mm) Molex pin header connectors aligning two sides of the board, allowing it to be plugged like a "big chip" into a target application.

The standard module runs at a maximum of 520 MHz internal clock speed and offers 64 MByte (up to 128 MByte) SDRAM and 32 MByte (up to 64 MByte) Flash on-board for DATA and CODE storage.

The module communicates by means of one RS-232 transceiver, one CAN bus interface, and a SMSC LAN91C111 10/100BaseT Ethernet controller which enables implementation of the module in embedded Internet applications. The phyCORE®-PXA270 operates within a temperature range of -40°C to +85°C.

The phyCORE®-PXA270 offers the following features:

- subminiature Single Board Computer (70 x 57 mm) achieved through modern SMD technology
- populated with the Intel PXA270 microcontroller (BGA-360 packaging)
- improved interference safety achieved through multi-layer PCB technology and dedicated Ground pins
- controller signals and ports extend to two 160-pin high-density (0.635 mm) Molex connectors aligning two sides of the board, enabling it to be plugged like a "big chip" into target application
- max. 520 MHz clock frequency
- 32 MByte (up to 64 MByte) on-board Flash[1]
- 64 MByte (up to 256 MByte) RAM on-board
- one CAN transceiver
- RS-232 transceiver
- SMSC 91C111 Ethernet controller with configuration EEPROM
- USB Client Interface and Host Interface
- 2 * PCMCIA / Compact Flash
- 2 * MMC card and MemoryStick
- LCD controller
- Synchronous Serial Protocol ports (SSP and NSSP)
- I^2C Interface
- 4 * PWM
- RTC
- General purpose I/O pins (GPIO)

[1] : Please contact PHYTEC for more information about additional module configurations.

The Development Board PCM-990 is a universal carrier board for start-up and programming of the PHYTEC phyCORE-PXA270 Single Board Computer module with two high-density SMT (160 pins each, 0.63 mm pitch) pin header connectors.
The Development Board is fully equipped with all mechanical and electrical components necessary for the speedy and secure insertion and subsequent programming of the high-density PHYTEC phyCORE module. These components include TFT display connection, touch screen interface, Ethernet transformer and connector, DB-9, USB and power socket connectors.

phyCORE® Development Board PXA270 Technical Highlights

- Molex connectors for mating with phyCORE-PXA270 SBC module
- Connector for 8.4" Sharp TFT display with 640*480 pixel resolution, incl. adjustable inverter for background illumination
- AC97 controller WM9712L with touch and sound interface (Line-IN, Line-OUT, MIC-IN and 0.4 watts speaker)
- Two DB-9 sockets for RS-232 interface from FF-UART and BT-UART, 2*5 pin header for IR-UART
- One DB-9 plug for CAN interface, optical isolation
- RJ-45 Ethernet connector with transformer for 100/10 MBit/s
- USB Host socket for PXA270 USB host interface
- USB client socket for PXA270 USB client interface
- Power supply for unregulated input voltage from 12 V (optional 24V). It supplies regulated +3.3 V for the phyCORE-PXA270. Additional +5 V is created for the IDE and CF card and LCD/inverter circuitry.
- Optional battery charger circuitry for NiMH batteries
- Power magement for MMC, CF, IDE and LCD display
- Two user programmable LEDs
- Optional IDE interface for two 2.5" hard disks (Master / Slave)
- Interface for Matrix keyboards
- JTAG interface
- AC97 expansion port
- Reset and GPIO push button

1.7 Software Development Tools

1.7.1 Telelogic UML Development Tool Rhapsody

As the embedded market grows, the systems are getting more complex. Normaly, embedded systems are designed with a functional approach, using e.g. the language C. With more complex systems, this will lead to problems in quality, reusability. The commercial market has established a standard called Unified Modelling Language for developing complex systems. But UML is also available for the embedded market. With UML you can describe your system graphically by drawing diagrams, showing the components and behaviour of your system.

Today there are many UML tools on the market, most only concentrating on designing the models. But with these tools, you will get not much benefit from using UML because the code needs to be developed seperatly from the model leading to inconsistencies during the development cycles.

Telelogic´s UML software development tool Rhapsody is the next step in software development for the embedded market. It allows to graphically model the behaviour and functionality of the embedded system and will generate productivity code from the models. With this approach, your code and your model are always consistent.

Rhapsody supports most of the UML 2.0 diagrams, including

- Use Case diagrams
- Class diagrams
- Object model diagrams
- Component diagrams
- Sequence diagrams
- Statecharts

- Activity diagrams

Rhapsody will generate code witch consists of the class interfaces that are given in the model and the behavior the user gave these classes. Code can be added to the class definitions.

This code is then compiled with an external cross compiler resulting in an application that will run on the target platform. The application running on the target can be debugged at design and code level on the host PC.

Rhapsody need to know the location of the cross compiler, so be sure to first install the GNU Cross Compiler Tool Chain before installing Rhapsody

1.7.2 GNU Cross Compiler Tool Chain

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 arm architecture with a PXA270 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.

1.7.3 Eclipse Platform

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.

.

2 Getting Started

40 min

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 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 10.0 OSS (x86) and KDE desktop.

When you are installing openSUSE 10.0, you can select *KDE* as *Desktop selection*. The default packages to use openSUSE 10.0 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.

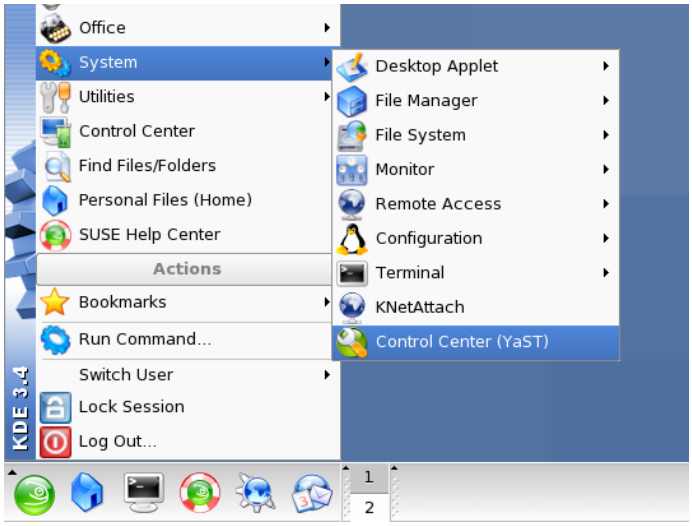
2.2 Configuring the host platform

In this passage you will learn how to configure the host platform. You will install additional Linux software packages that are necessary to accomplish the steps in the QuickStart Instruction. Then you will setup the Network configuration to use the host pc with your target and disable the firewall. If the firewall is enabled, you will have problems with connections to the target.

2.2.1 Installing Software packages:

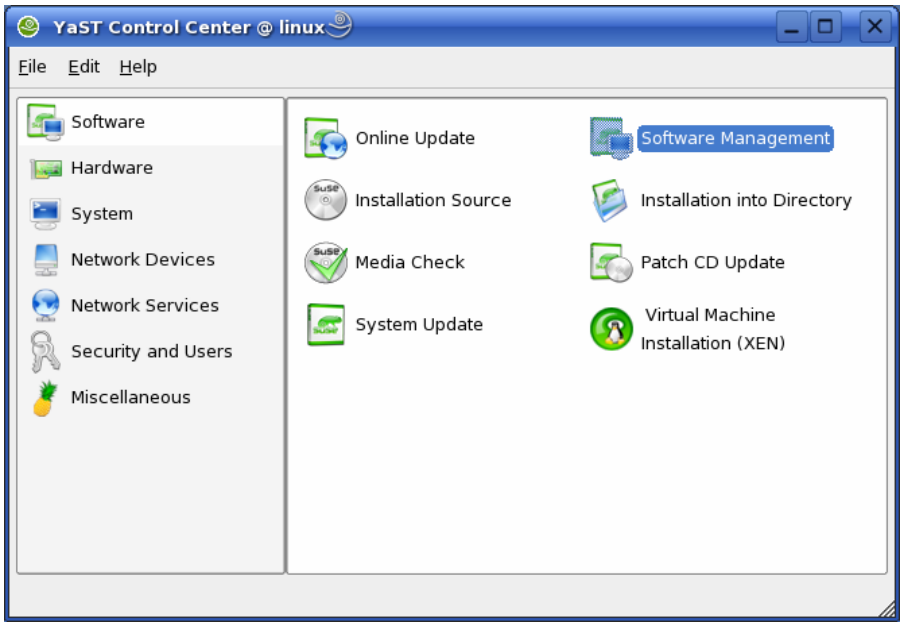
To accomplish the steps in the QuickStart Instruction you will have to install additional packages.

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

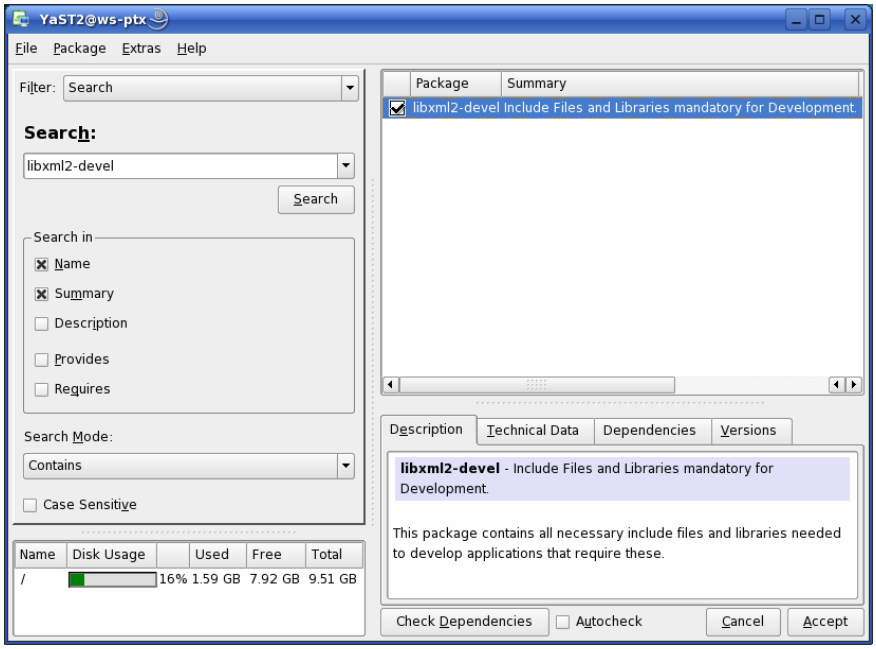


- Open the Control Center (YaST)

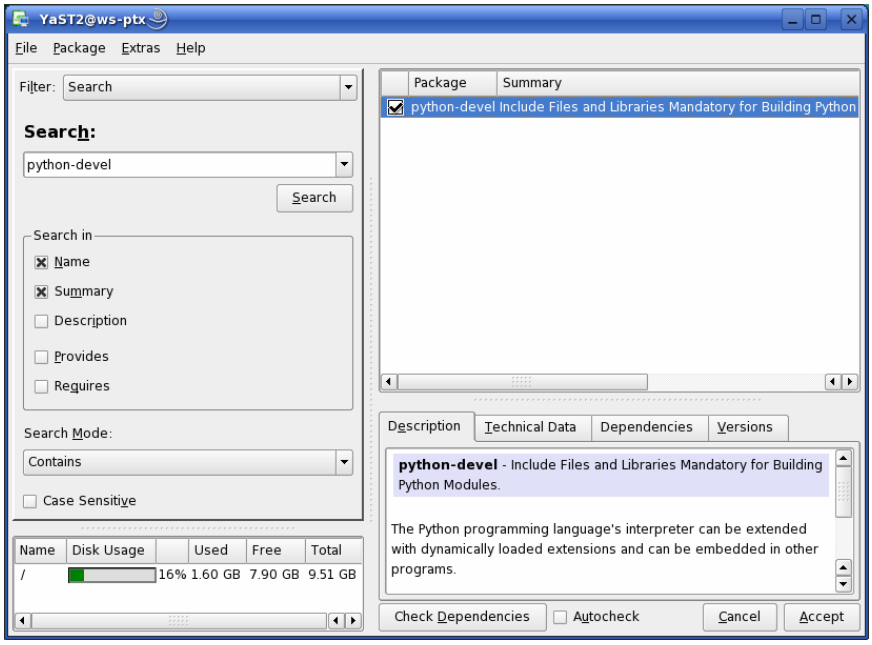
- Enter your root password and click on the *OK* button.



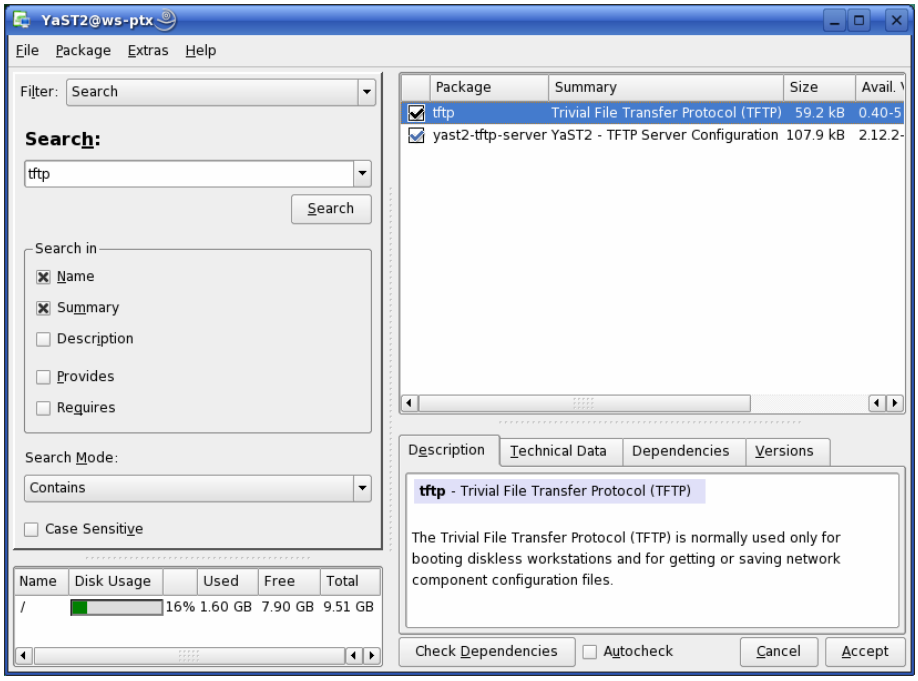
- Open *Software Management* in *Software*.



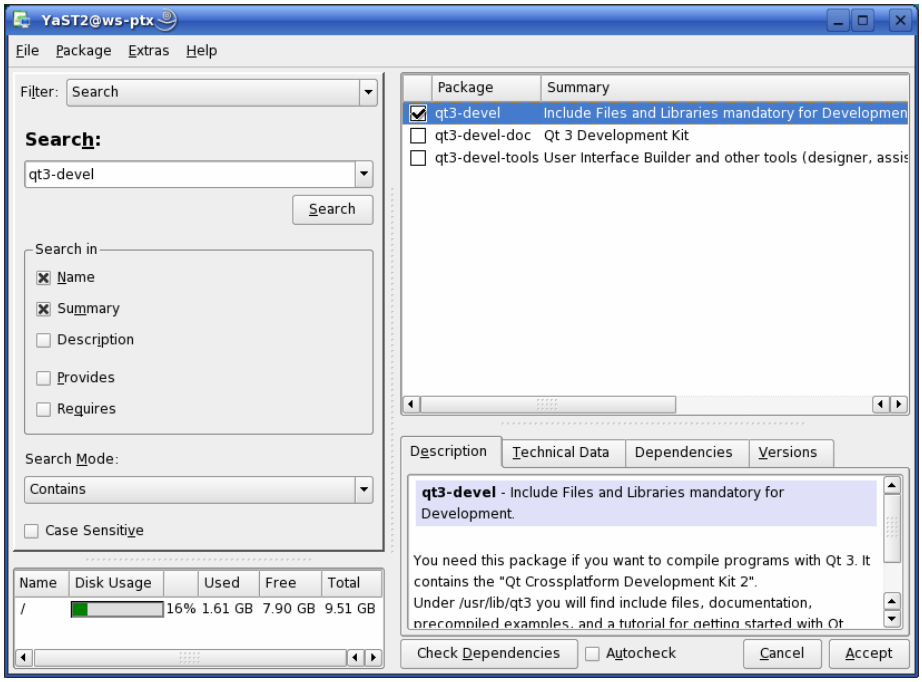
- Select Filter *Search*.
- Type **libxml2-devel** and click *Search* button.
- Check *libxml2-devel.*



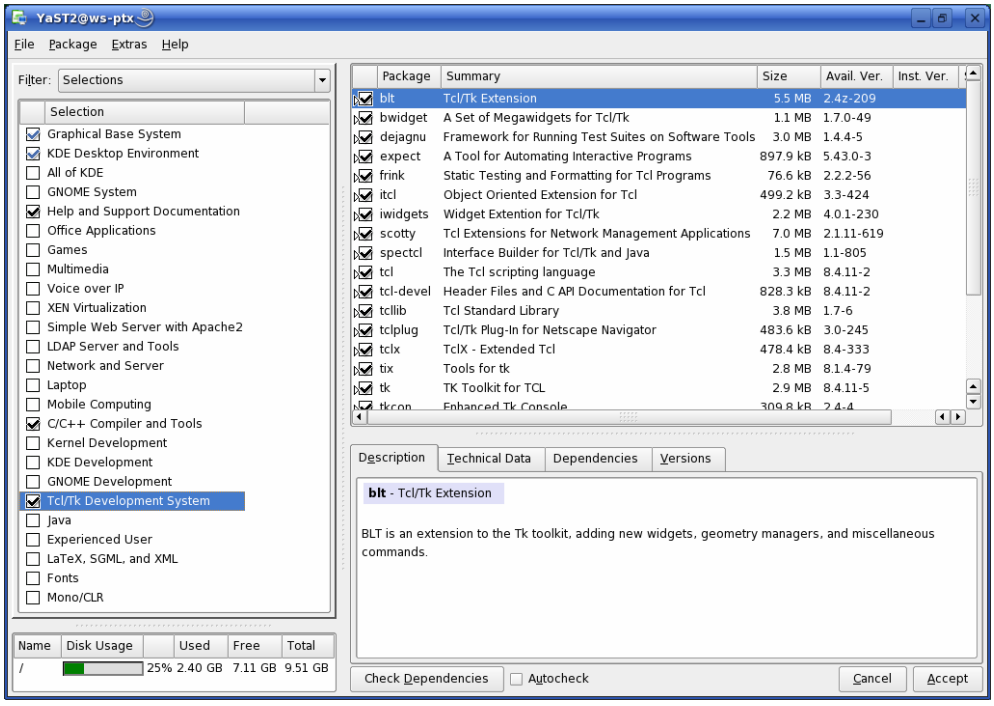
- Type **python-devel** and click *Search* button.
- Check *python-devel*



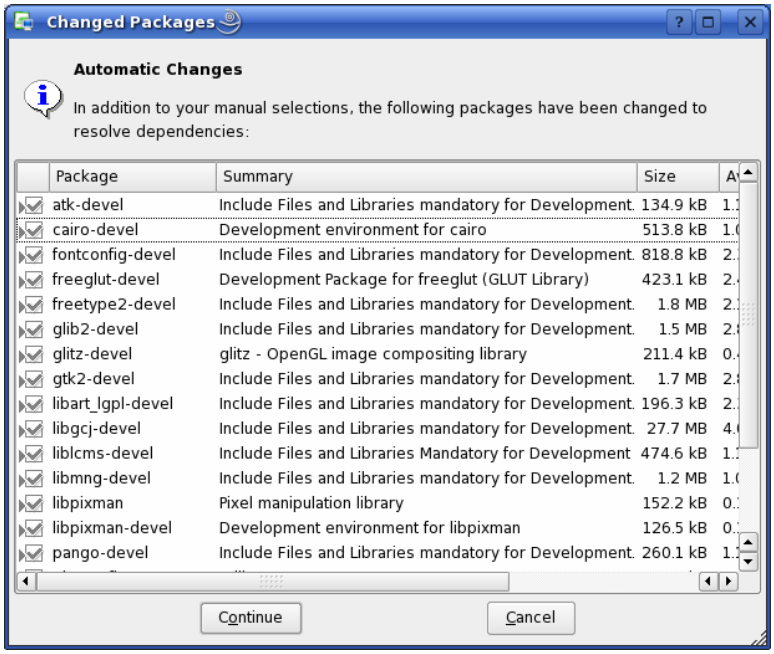
- Type **tftp** and click *Search* button.
- Check the package **tftp**



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



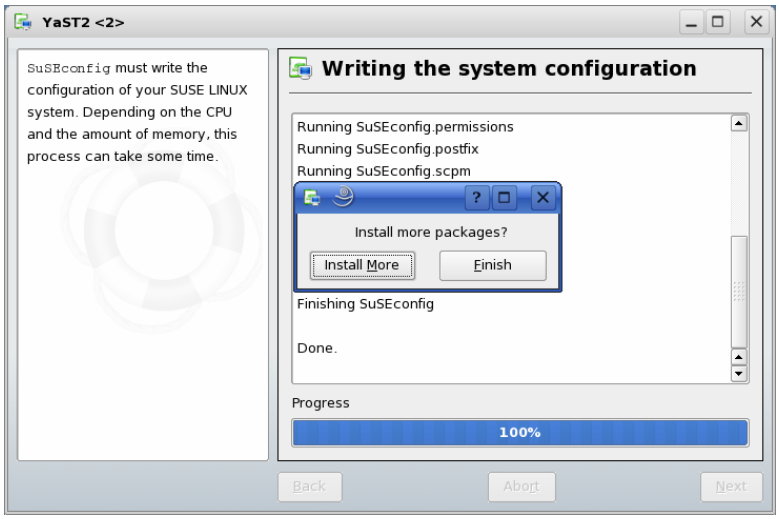
- Select Filter *Selections*.
- Select *C/C++ Compiler and Tools* in the window selection.
- Select *Tcl/Tk Development System* in the window selection.
- Click *Accept* button.



Some additional packages will be selected automatically to resolve dependencies.

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

- Click on *Continue* to install the packages.

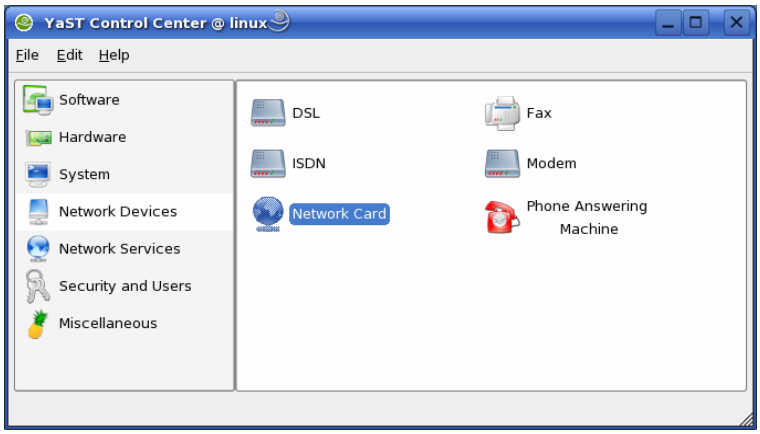


- Click on *Finish.*

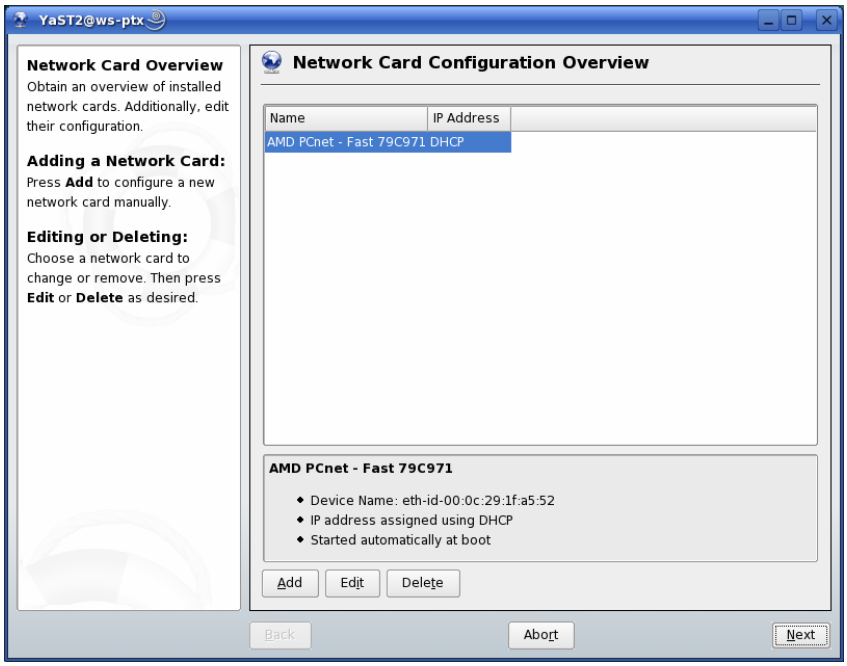
2.2.2 Setup Network Card Configuration

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 ip, problems may occur with existing hosts.

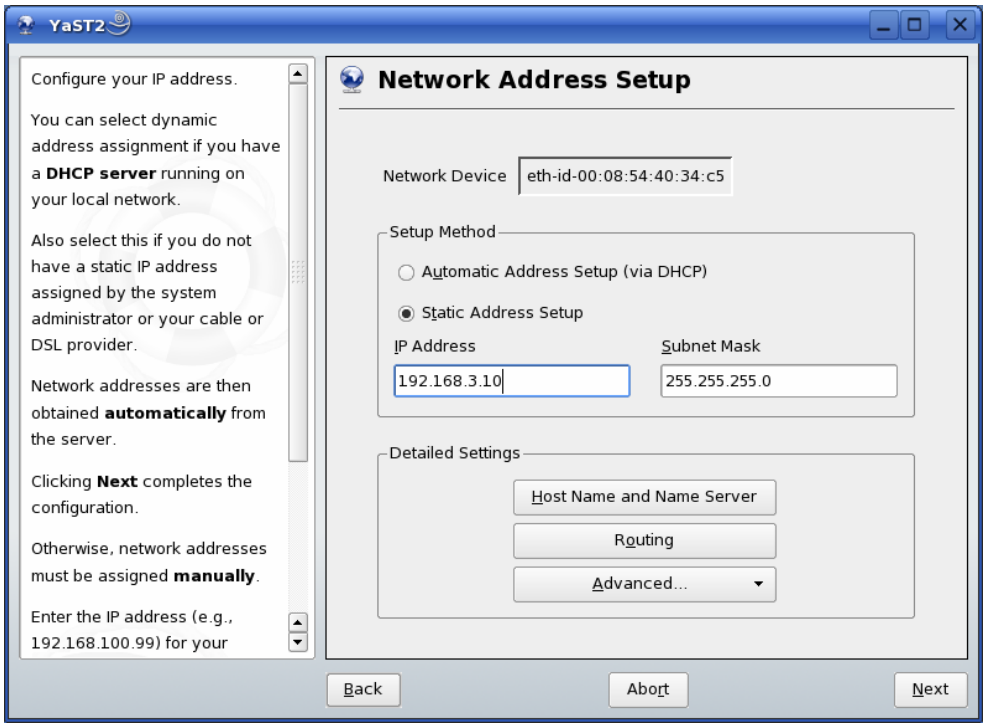
- Open the YasST Control Center if it is not already open.



- Choose *Network Card* in *network devices*.



- Select the right network card, if more than one network card is installed on your host.
- Click on button *Edit* to enter Network Address Setup.

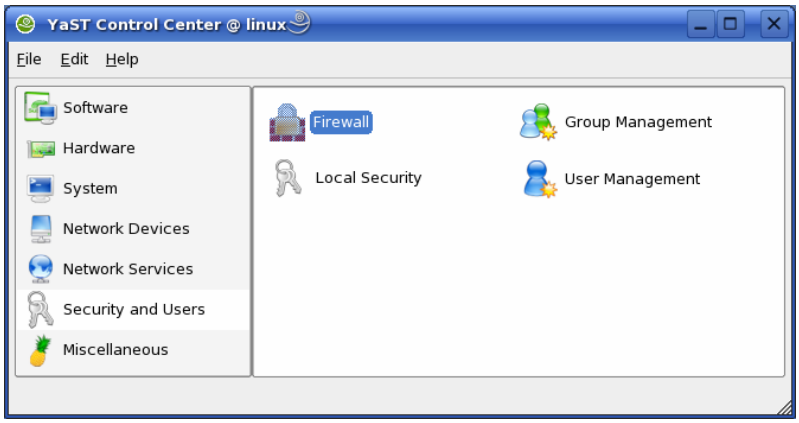


- Choose *Static Address setup*.
- Enter IP Address **192.168.3.10** and Subnet Mask **255.255.255.0**
- Click on the *Next* button.

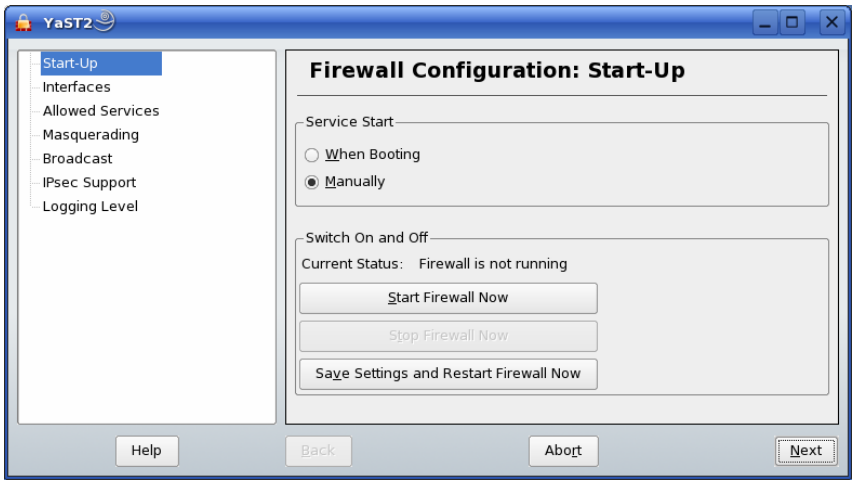
- Click on the *Next* button again to finish Network Card Configuration.

2.2.3 Disabling the Firewall

If you want to create a telnet or ftp session to the target, for example you will have to add rules to allow access to these services. To ensure that there are no problems with connections to target, because of the firewall, the firewall should be disabled.



- Choose *Firewall* in *Security and Users*

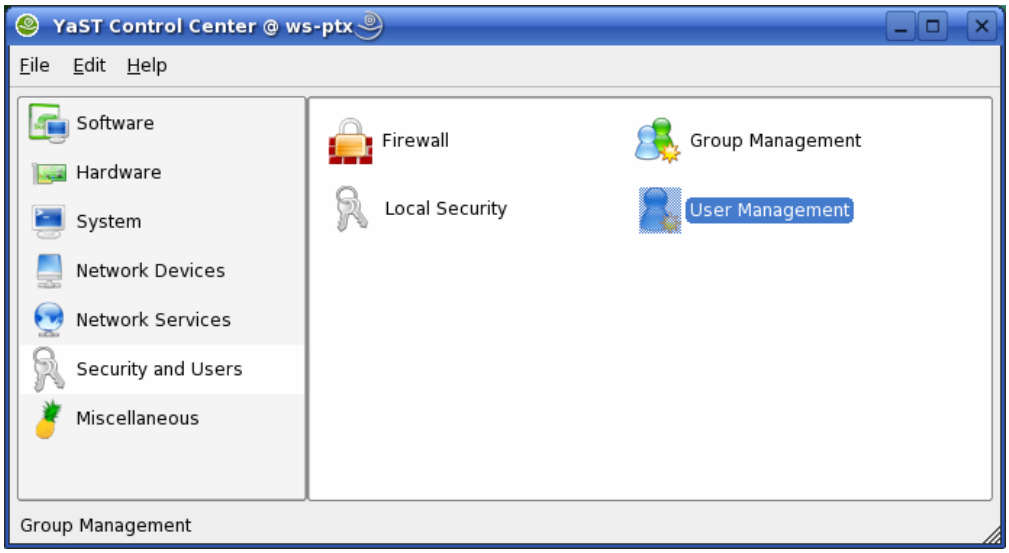


- Choose *Manually* for *Service Start.*
- Click on the Button *Stop Firewall Now.*
- Click on *Next and Accpet*

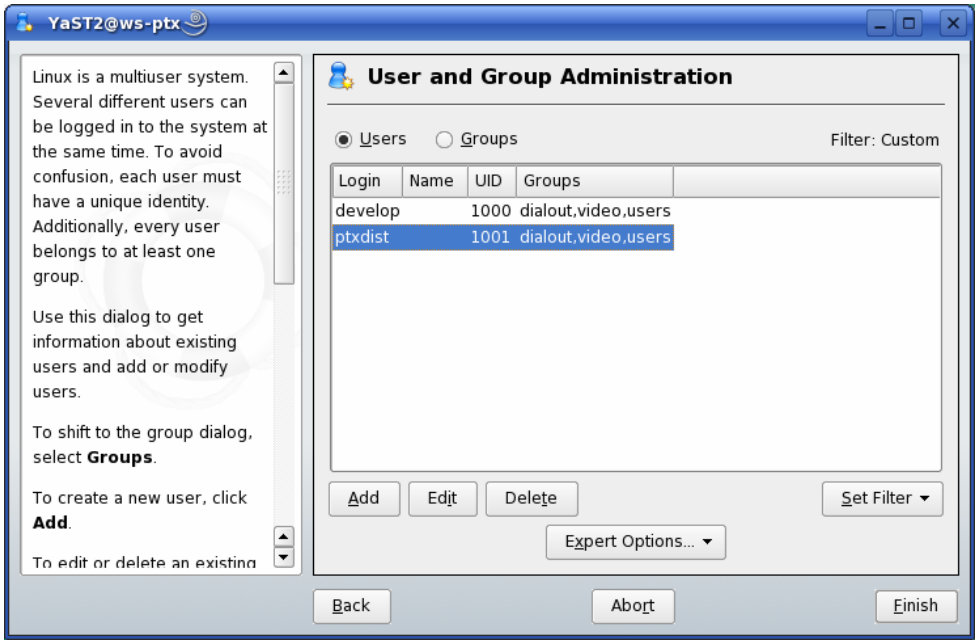
The firewall is now disabled.

2.2.4 Configure User Rights for the Serial Interface

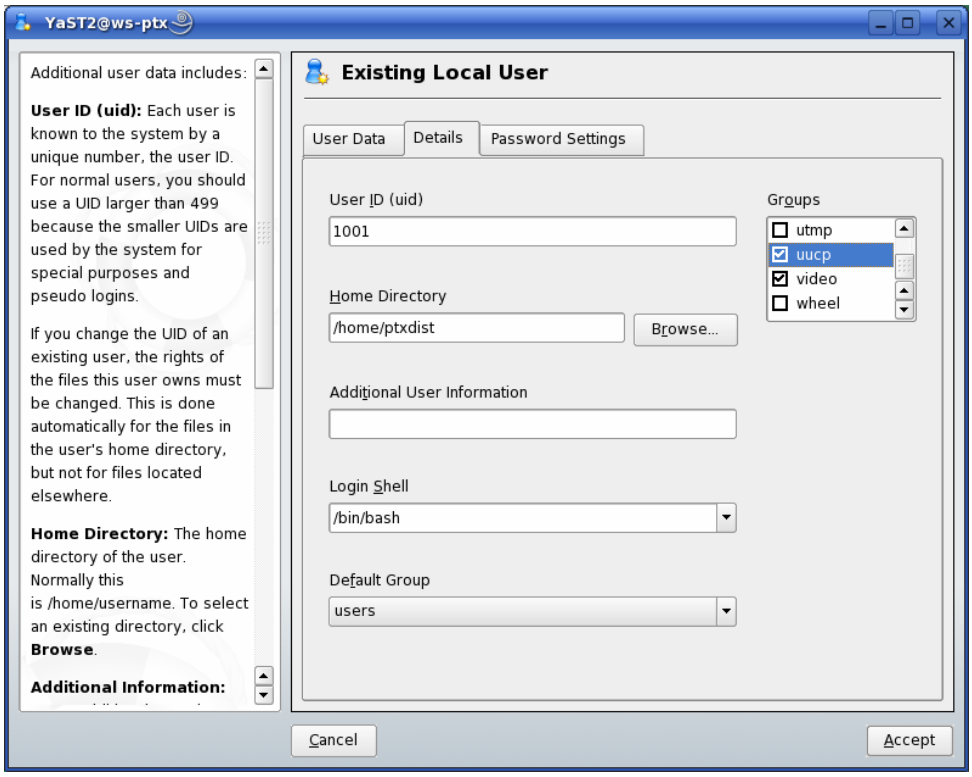
- Open the YasST Control Center if is not already open.



- Choose *User Management* in *Security and Users.*



- Select your username
- Select *Edit*

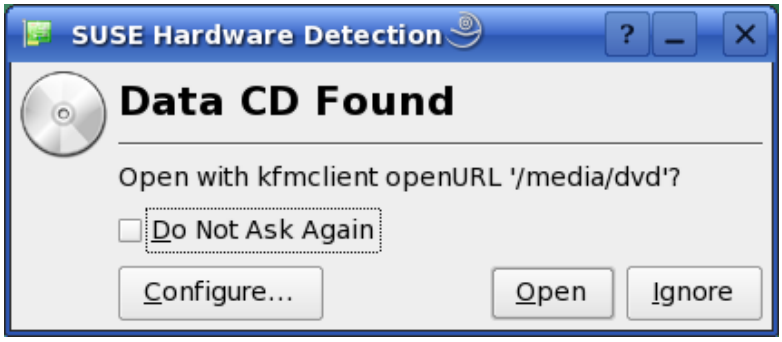


- Select *Details* tab
- Check groups *uucp*
- Click *Accept* button.
- Click *Finish* button.

You have successfully finished the configuration of the host platform.

2.3 UML-Kit Setup

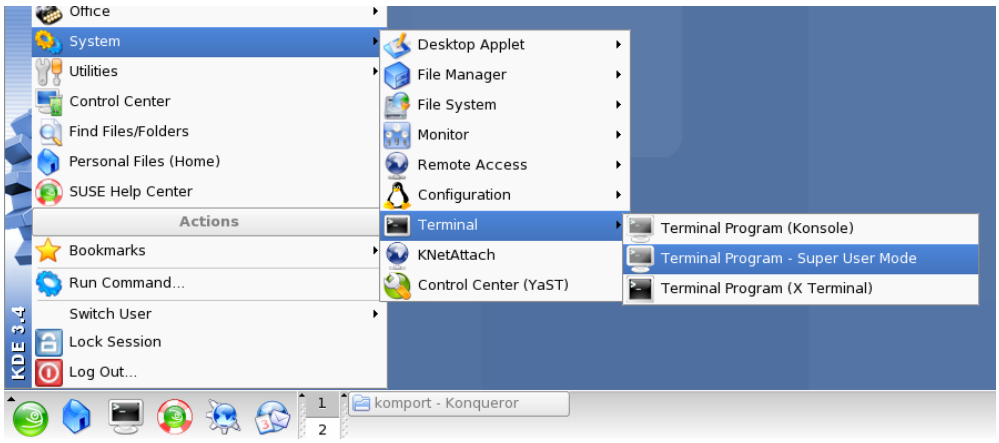
- Insert your UML-Kit CD-ROM in you CD-ROM drive. You will see the following dialog:



The openURL /media/dvd is the directory to access the setup dvd. If you have a cdrom drive you will see the openURL /media/cdrom.

- Click button *Ignore*.

- Open a terminal window in the Super User Mode (Root Shell) and enter the root password. You need root privilege to install the software.



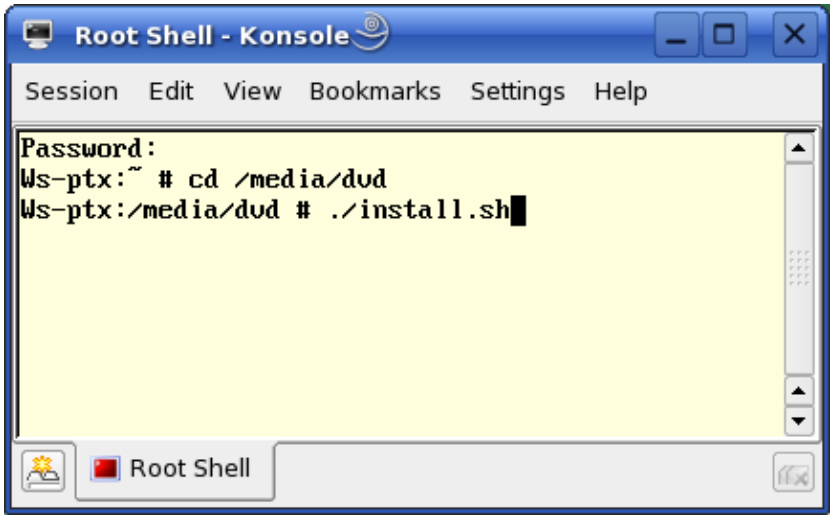
- Change to the cdrom / dvd directory:

  cd /media/cdrom

  or

  cd /media/dvd

- Start the setup programm: *./install.sh*



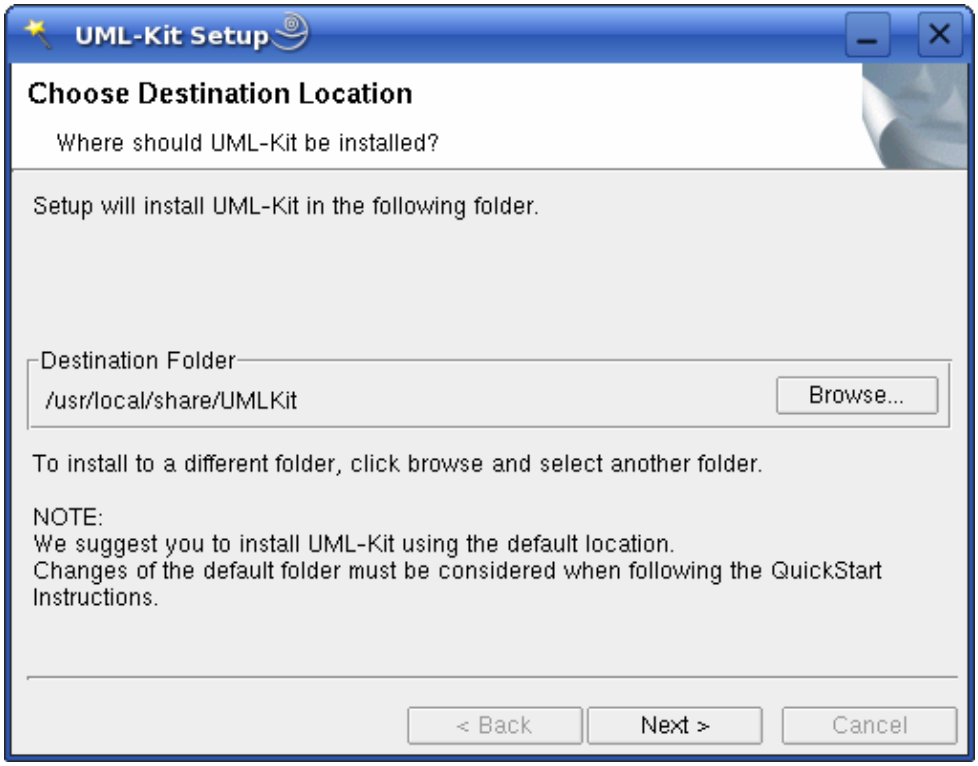
Don’t close the terminal program while the setup is running.

Be sure that you adjusted the right date on your PC. You can see the date in lower right corner of your desktop. If your date is in the past your installation may not work correctly.

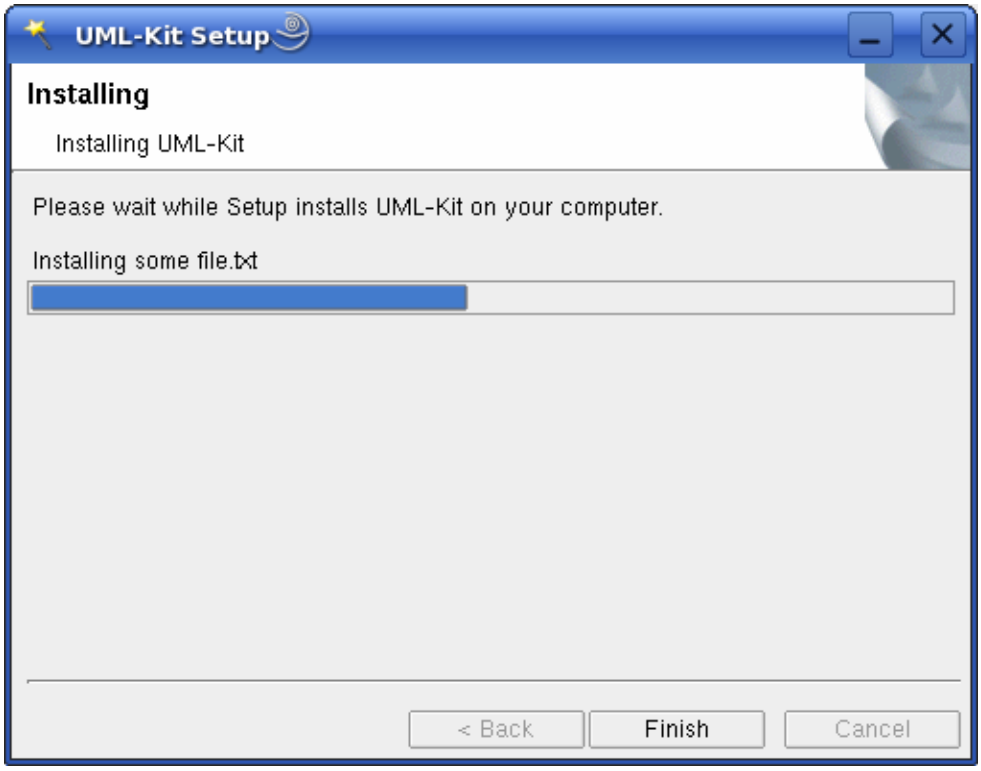
- Click *Yes* to proceed
- After accepting the Welcome window and license agreement select the destination location for the installation of the Linux Kit software and click on *Next*.

The default destination location is */usr/local/share/UMLKit*. 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.

We recommend that you accept the default destination location.

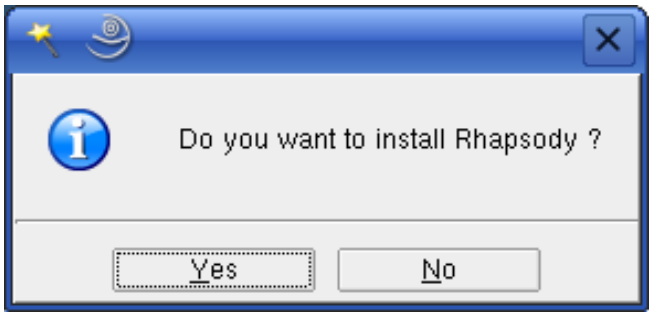


Click *next* to install the setup files to */usr/local/share/UMLKit*. The GNU GCC /GLIBC Tool toolchain will be installed to the standard default directory */opt/arm-softfloat-linux-gnu*.



- Click *Finish.*

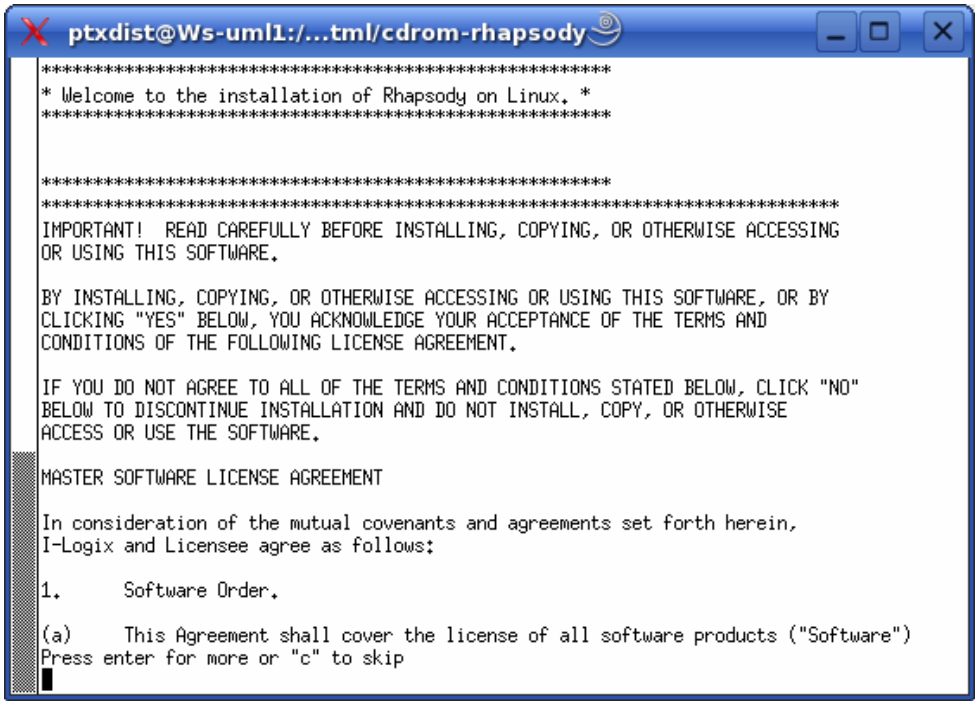
- In the dialog box click *Yes* to install Rhapsody. If you want to skip the installation of Rhapsody choose *No*.



Even if you have Rhapsody already installed on you system, it is recommend to choose *Yes* and to install Rhaspody again into a different directory. The OXF Framework in this Version was specially build for an ARM Xscale target. The configuration site.prp was adapted to use the GNU GCC C/C++ Toolchain with Rhapsody.

If you choose yes, a new window will be opened and the Rhapsody setup will be started.

- Press *Enter* to read more or press "*c*" to skip.



- Accept the license agreement by pressing *Enter*.
- Enter the path of the directory to install the Rhapsody software:

  /usr/local/share/rhapsody

Do not use a slash at the end of the path

- Type "**1**" or press *Enter* to select C++.

```
Language Selection
1: C++
2: C
```

```
3: Java
Please select at least one of the above
languages, as in 1, or
1,2,3 etc.
```

- Press *Enter* to use Rhapsody for C++ code generation

```
using C++ code generation with Rhapsody? [Y/N]Y
```

- Enter the path of your C++ compiler. The default location is /usr/bin. If you are using Suse Linux you can press *Enter*.

```
Please enter the directory where your g++
compiler can be found:
[/usr/bin]:
```

- You will be asked if you are using MontaVista Standard PC with Rhapsody. Press *Enter* for no.

```
Will you be using MontaVista 3.1 Generic x86
Industry Standard PC with Rhapsody ? [Y/N]:N
```

- Install the Rhapsody Help files by pressing *Enter*.

```
Do you want Rhapsody's Help files installed ?
[Y/N]:Y
```

- Install the Rhapsody Sample files by pressing *Enter*.

```
Do you want Rhapsody's Sample files installed ?
[Y/N]:Y
```

- Press *Enter* to install Adobe Acrobat.

```
Do you want to install Adobe Acorbat now ?
[Y/N]:Y
```

- The license agreement will appear. Press *Enter* to read more or press "**c**" to skip.
- Accept the license agreement by pressing *Enter*.
- Press *Enter* to accept Acrobat's install dir.

The script starts to install the Rhapsody files and Adobe Acrobat.

- When the setup is finished press *Enter* to continue.
- Press *Next* to proceed.

- A dialog box appears. Click *Yes* to install Eclipse. If you want to skip the installation of Eclipse choose *No*.

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

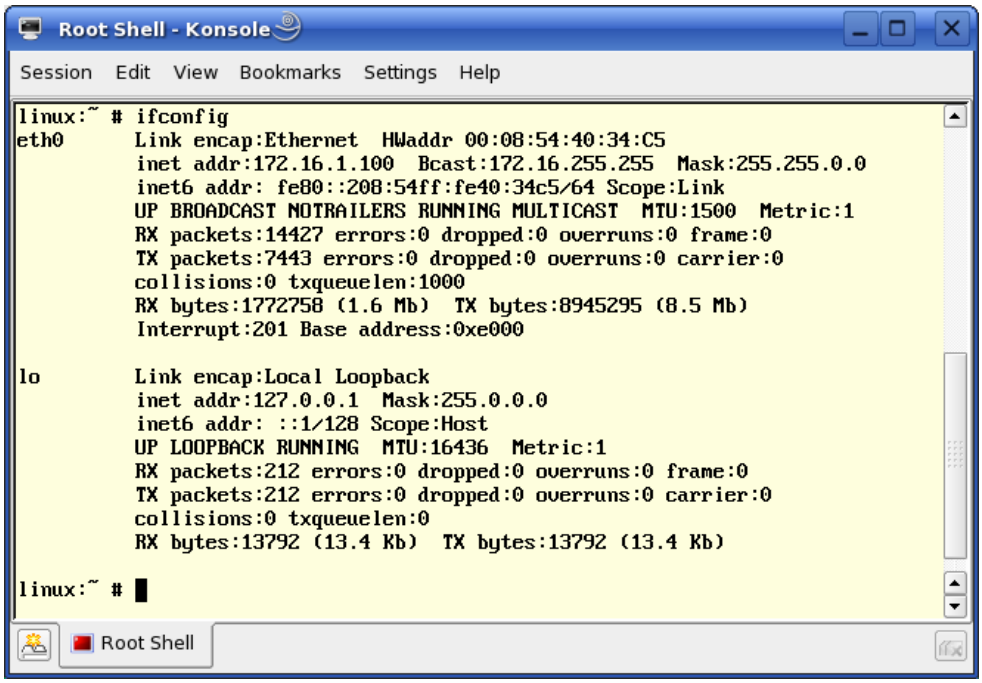
- Click *Next* to continue the setup.

The setup of the PXA270 UML Kit is now complete. To use Rhapsody you will have to obtain and install a license key.

Installing the license key

First you must obtain a license key. The license key is coupled to the MAC address of your network card.

- Open a Terminal in Super User Mode.
- Enter *ifconfig* to get the address



- Eth0 is the first network device. The MAC address of this device is *00:08:54:40:34:C5*.

You will have to use the MAC address of the device **eth0**. If you have more then one network device, be sure to use the address of eht0.

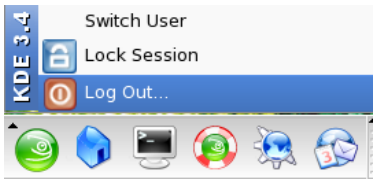
- To obtain the license key send a license request with the information of ifconfig to support@ilogix.com. You will get an E-Mail with a file lic.txt.
- Save this file in your home directory.
- Open a Root Shell.
- Type:

  cd ~
  cp lic.txt /usr/local/share/rhapsody/flexIm/license.dat

The commands are case-sensitive. Type an upper “I” in flexIm.

Now you will have to restart your host system.

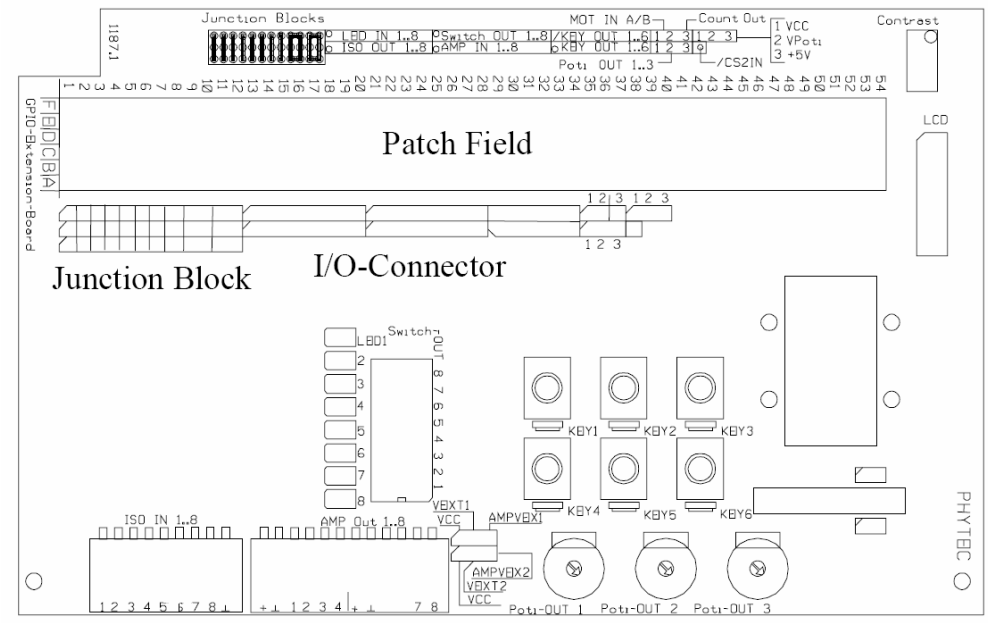
- Select *Log Out*



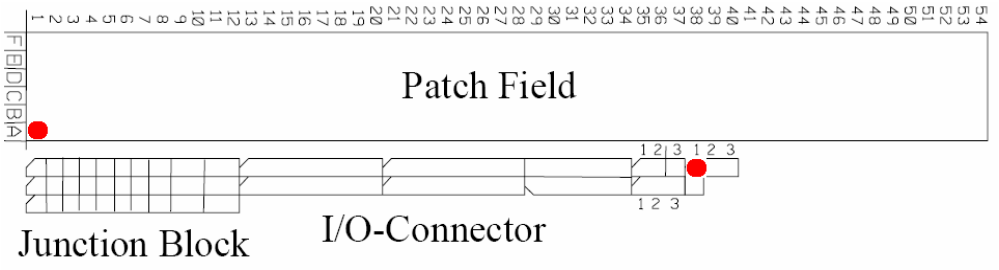
- Select *Restart Computer*

2.4 Setup your GPIO-Extension-Board

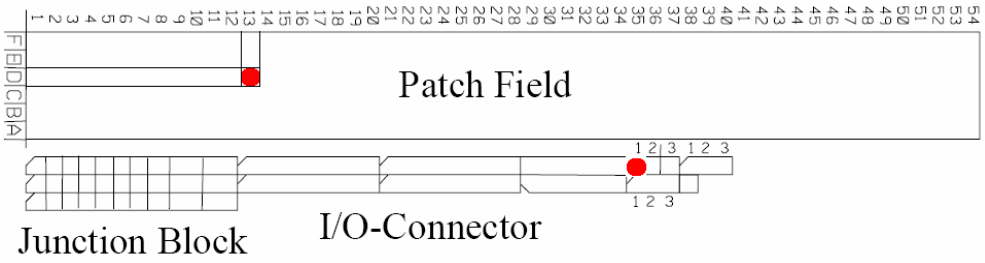
To setup your GPIO-Extension-Board, you have to connect the right pin on the patch field with the corresponding pin on the IO-Connector. The following figure shows you the location of the patch field and the IO-Connector.



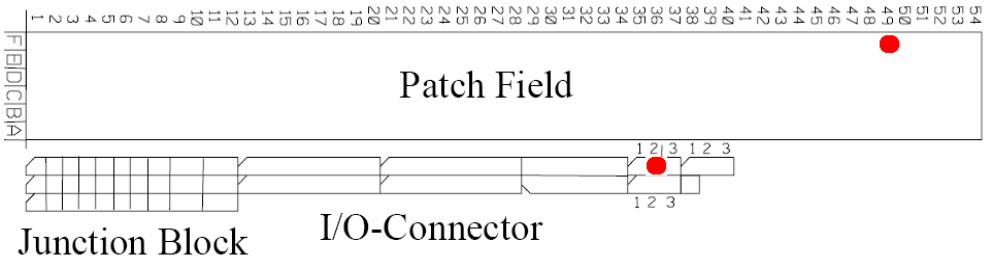
- First put your GPIO-Extension-Board on your baseboard



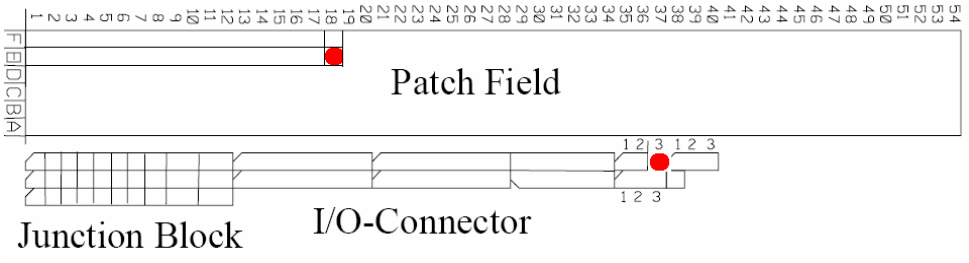
- To supply the GPIO-Extension-Board with power connect pin 1A with IO-Connector VCC



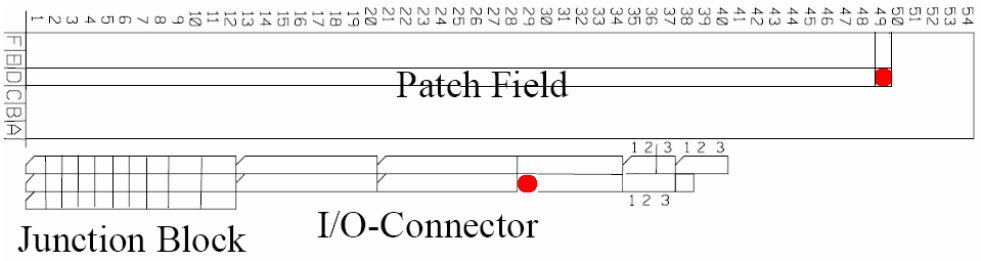
- Connect pin 13D to IO-Connector MOT IN A



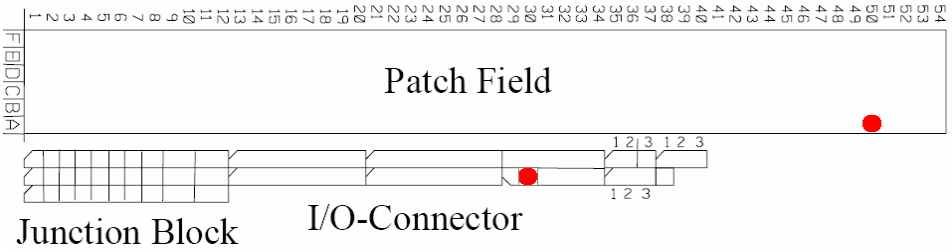
- Connect pin 49F to IO-Connector MOT IN B



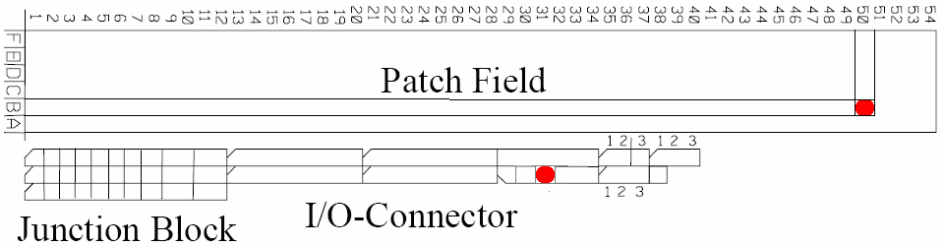
- Connect pin 18E to IO-Connector Count Out



- Connect pin 49D to IO-Connector KEY_OUT1



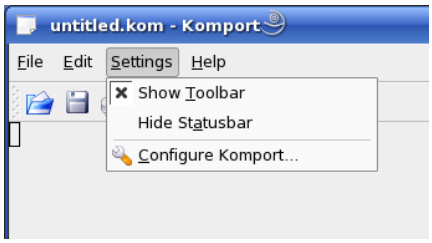
- Connect pin 50A to IO-Connector KEY_OUT2



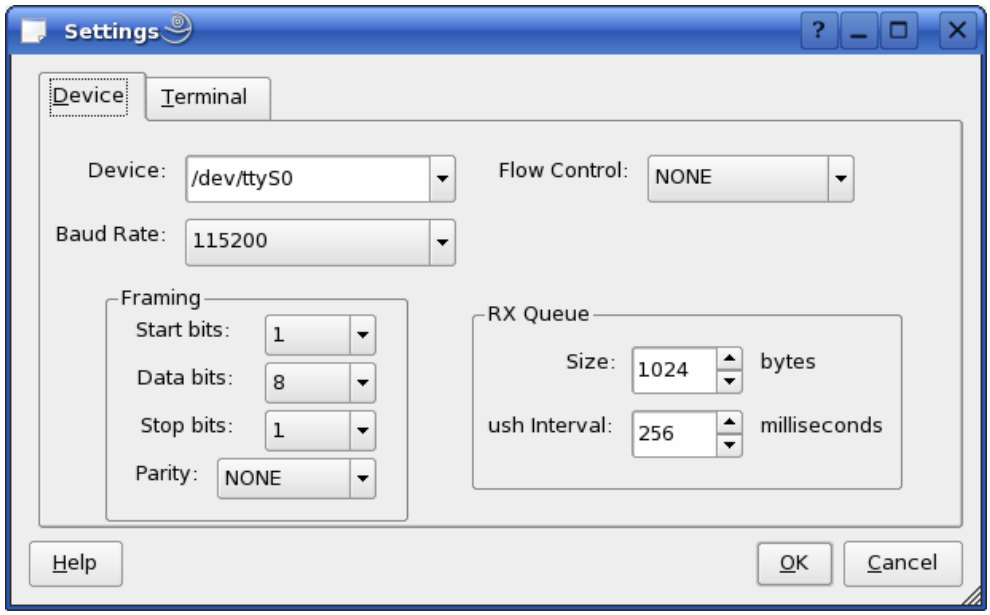
- Connect pin 50B to IO-Connector KEY_OUT3

2.5 Configuring Komport

- Connect the serial interface of the target (Connector P1) with COM1 of your host.
- Open Konqueror
- Navigate to **/usr/local/share/UMLKit/komport** and start the application.
- Choose *Configure Komport* in *Settings*



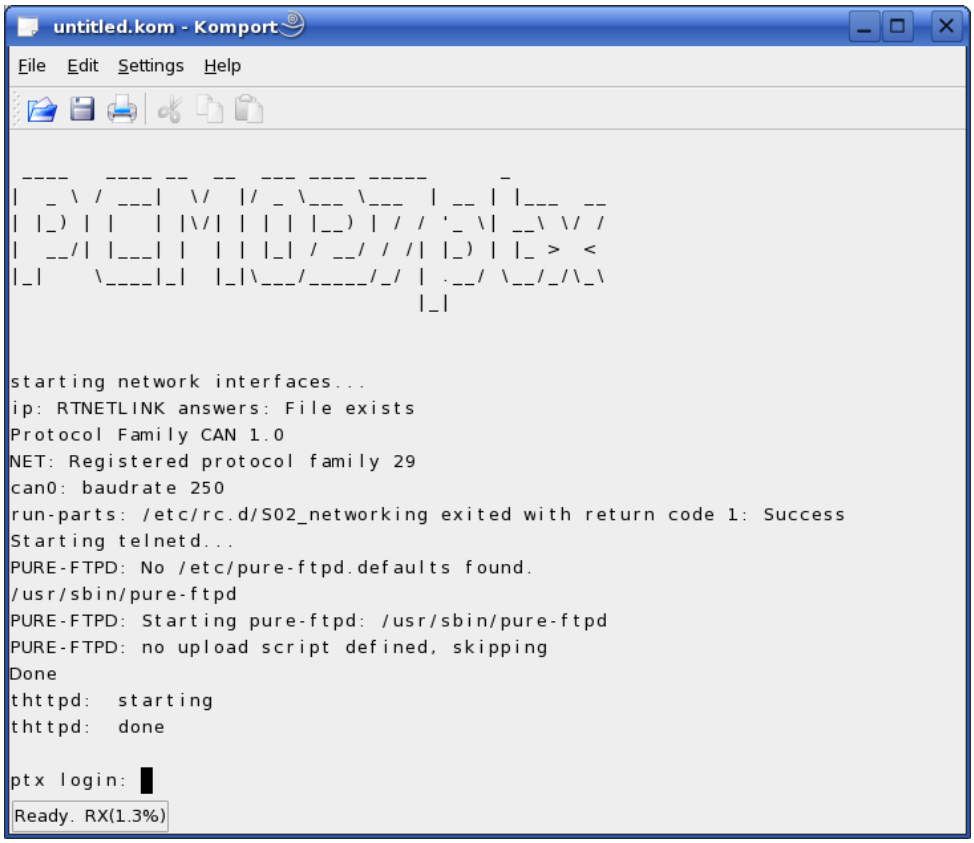
- Choose the device */dev/ttyS0*.
- Configure the following settings to 115200 baud, 1 Start bit, 8 data bits, 1 stop bit, no parity and no flow control.



- Click button *OK*

- Connect the power supply (12 V) with your board.

The target starts booting. When the target finished loading the file system you will see a screen similar to the following screenshot:



- Type **root** to login.

To access the serial interface, you need read / write access rights. How to set these rights, see in section Configure User Rights for the Serial Interface.

2.6 Downloading an Example on the Target

To download an example, you will have to connect the target (connector P1) to your Host PC with an Ethernet crossover patch cord.

- Open a new terminal window
- Change to /usr/local/share/UMLKit/HelloWorld:

  cd /usr/local/share/UMLKit/HelloWorld

- Create a FTP-Session to the target:

  ftp root@192.168.3.11
- Press *Enter* (no password required)

- Copy the application file to the target by typing:

  ftp>put HelloWorld

- End the ftp session: **ftp>exit**

- Open a telnet session to the target.

  telnet -l root 192.168.3.11

- Type **root** and press *Enter*
- Type **./HelloWorld** to start the application.

You have successfully finished the Getting Started part of this QuickStart.

3 Getting more involved

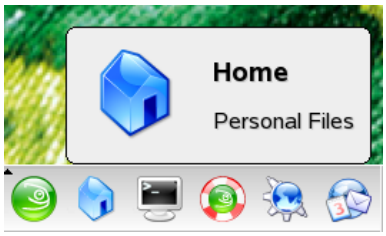
60 min

In this part you learn how to open the example project "Powerwindow" with Rhapsody. After opening the project you will see the model of a control unit of a carwindow. You will generate the source code from the model and build an executable program for the target. The program can be executed on the target and be used to control the motor on the GPIO Expansion board by three pushbuttons.
In the next section of this chapter will find an introduction on design level debugging with Rhapsody. You can use design level debugging to validate your model by tracing and simulating the executable. This is although known as animation.
At the end of this chapter you will add some changes to the model. After these changes you will see the ouput "moving down" / "moving up" when you press a button and the motor starts to move.

3.1 Open and build a project

First copy the Powerwindow project to your home directory.

- Click on the *Home* Icon to start the *Konqueror* browser.

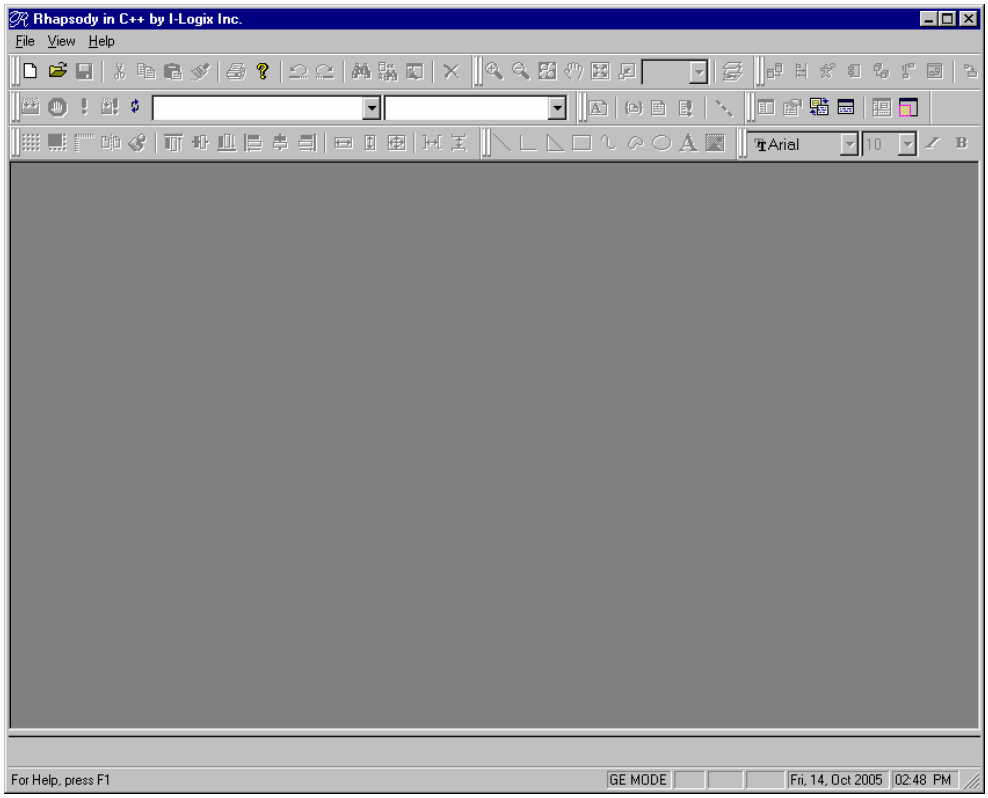


- Browse to the directory */usr/local/share/UMLKit/*
- Right-click the *Powerwindow* directory and select *Copy*.
- Browse to the directory */home/<your Home>*
- Right-click in your home directory and select *Paste*.

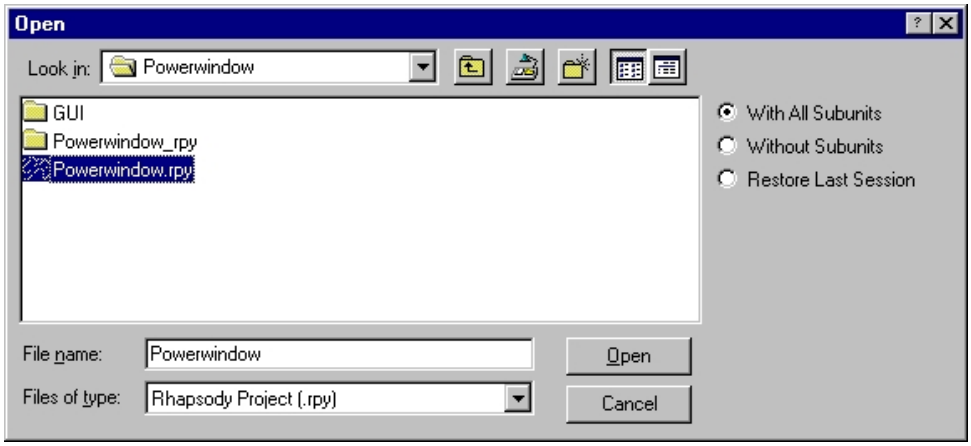
After you have copied the project you can start Rhapsody.

- Open the Konqueror file browser.
- Navigate to the directory */usr/local/share/rhapsody*
- Click on *Rhapsody* to start the application.

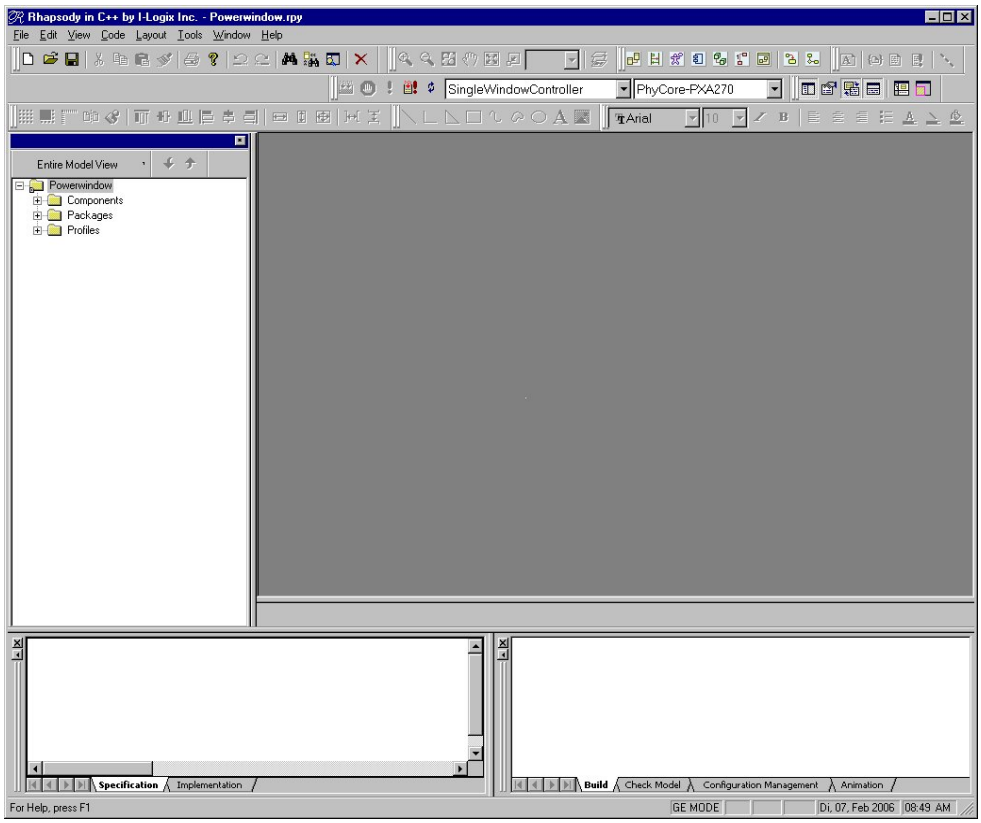
If you start Rhapsody the first time after the UML-Kit Setup, you first have to restart your Computer. If you can't start Rhapsody, select *Logout* in the K Menu and click on *Restart Computer*.

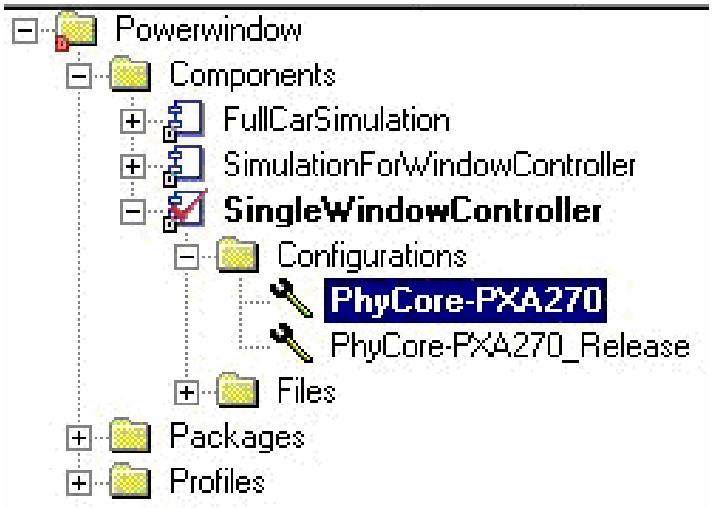


- Select *File* -> *Open* in the menu bar.



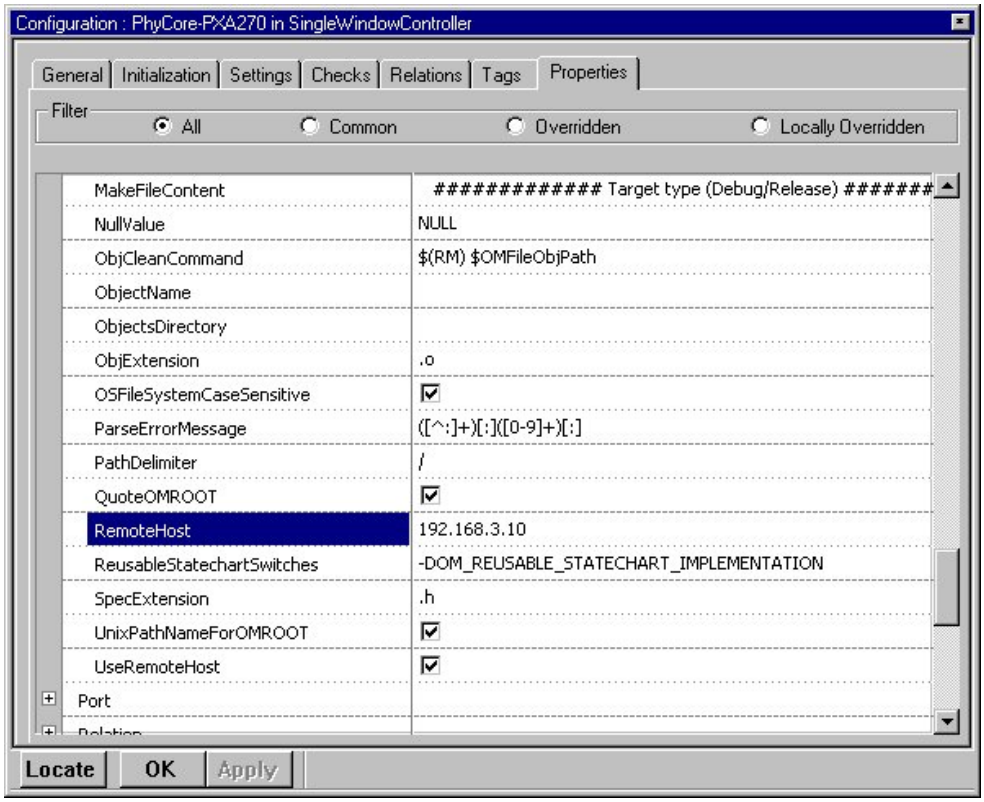
- Choose the directory */home/<your Home>/Powerwindow*
- Select *Powerwindow.rpy* and open the project.





- Right-click on *PhyCore-PXA270* in the Object-Browser: *Components -> SingleWindowController -> Configurations*
- Select *Set as Active Configuration* from the list.
  If the ip address of your system is not 192.168.3.10, you need to configure the host ip address for your configuration
- Right-click on *PhyCore-PXA270* in the Object-Browser *Components -> SingleWindowController -> Configurations*
- Select *Features*.

The feature dialog appears.



- Select the *Properties* Tab and choose Filter *All.*
- Select *CPP_CG -> PhytecLinux -> RemoteHost.*
- Enter the IP address of your host (**not** the target ip address)

To execute the program on the target you need two additional files (window_lift.bin and window_lift.xml).

- Open a new terminal window
- Change to the window_lift directory:

  cd ~/Powerwindow/window_lift/
- Create a FTP-Session to the target: **ftp root@192.168.3.11**
- Press *Enter* (no password required)
- Copy the files *window_lift.bin* and *window_lift.xml* to the target:

  ftp> put window_lift.bin

  ftp> put window_lift.xml

- Type e*xit* to end the ftp session: **ftp> exit**

Now you are ready to generate code.

- Select Code->Re Generate->Configuration File
- Select Code->Re Generate->PhyCore-PXA270
- Select Generate/Make/Run 
- If a dialog appears enter **yes** and press *OK*. The dialog will only open the first time you are creating a SSH connection.

After generating the Code, the program will be compiled and copied to the target.

Copying the files to the target can take a minute.

When the program is copied to the target, the application is started on the target and the XTerm window opens. If you started the application the first time enter **yes** and press Enter.

Make sure that the files *windowlift.bin* and *windowlift.xml* are present in the same directory as the program file SingleWindowController. You need the two files windowlift.bin and windowlift.xml for the hardware access on the target.

Now you can use the push buttons to control the motor:

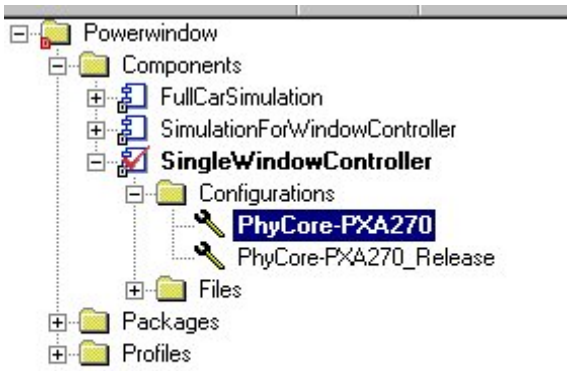
- Press Button 2 on the extension board for down-movement.
- Press Button 1 on the extension board for up-movement.

You can use Button 3 as simulation of a jam detection (resistance caused by an object blocking the way of the window)

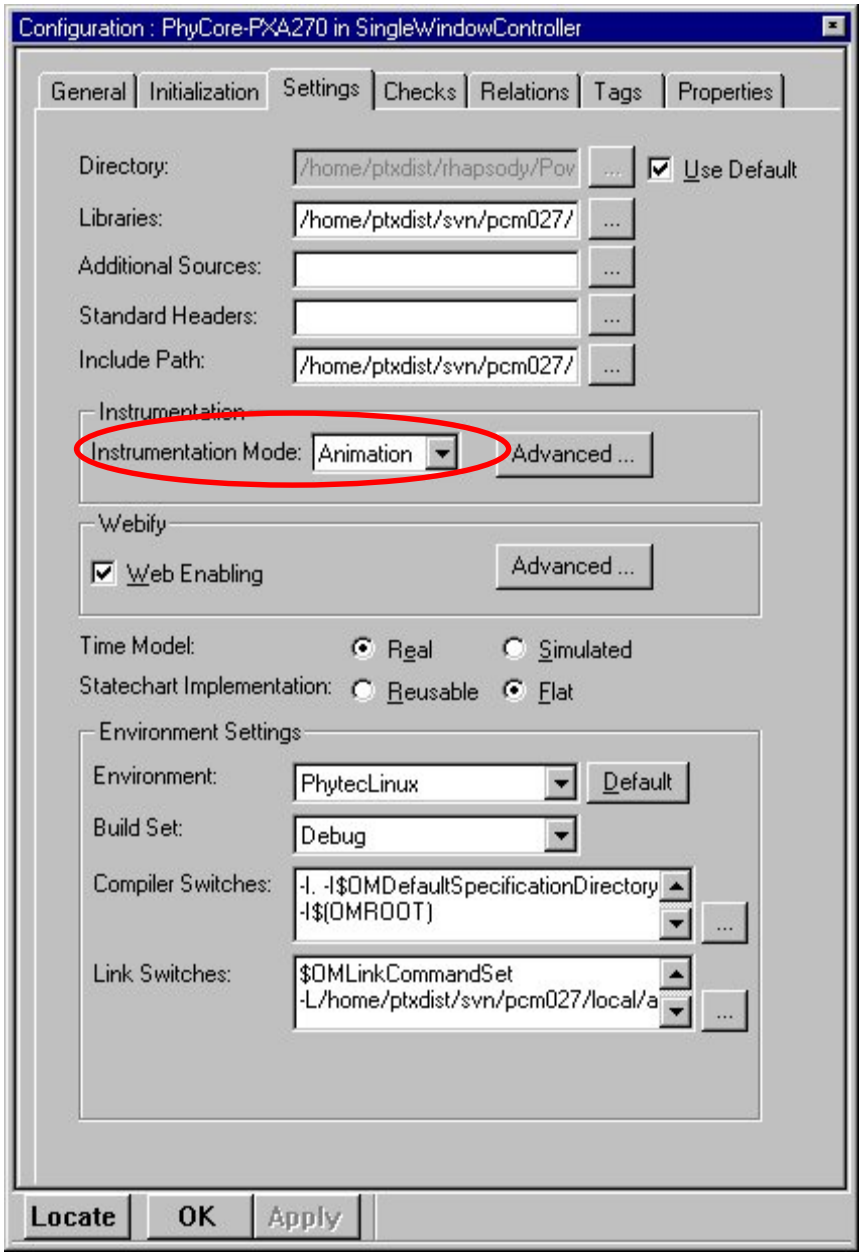
3.2 Design Level Debugging

When a model becomes more complicated, you have to validate the model. You can use design level debugging to validate your model by tracing and simulating the executable. This is although known as animation.

If you want to animate your model, the instrumentation flag must be set to Animation.



- First expand the component in the browser and double-click on PhyCore-PXA270 to open the feature dialog.



- Select the tab *Settings*.
- Select *Animation* as Instrumental Mode.
- Click on OK to accept the settings.
- Save the configuration 
- Click on Generate / Make / Run 

The application is built and copied to the target. When the application starts, the Animation toolbar, the Event Queue and Call Stack are displayed.

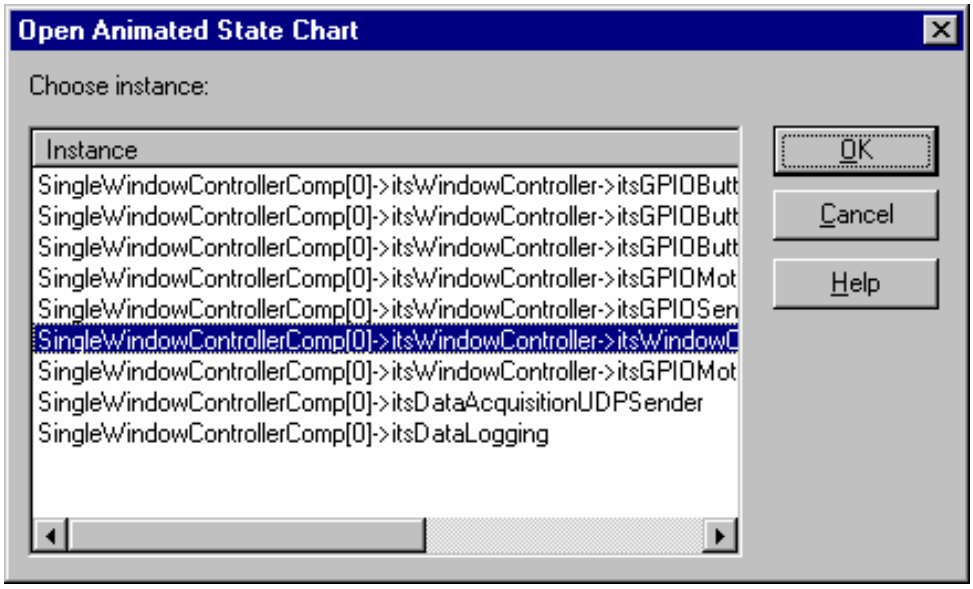
Perhaps there is still an application running on the target and the message “Text file busy appears”. Open a terminal window and type:

- ssh root@192.168.3.11 killall SingleWindowController

Animation Toolbar:

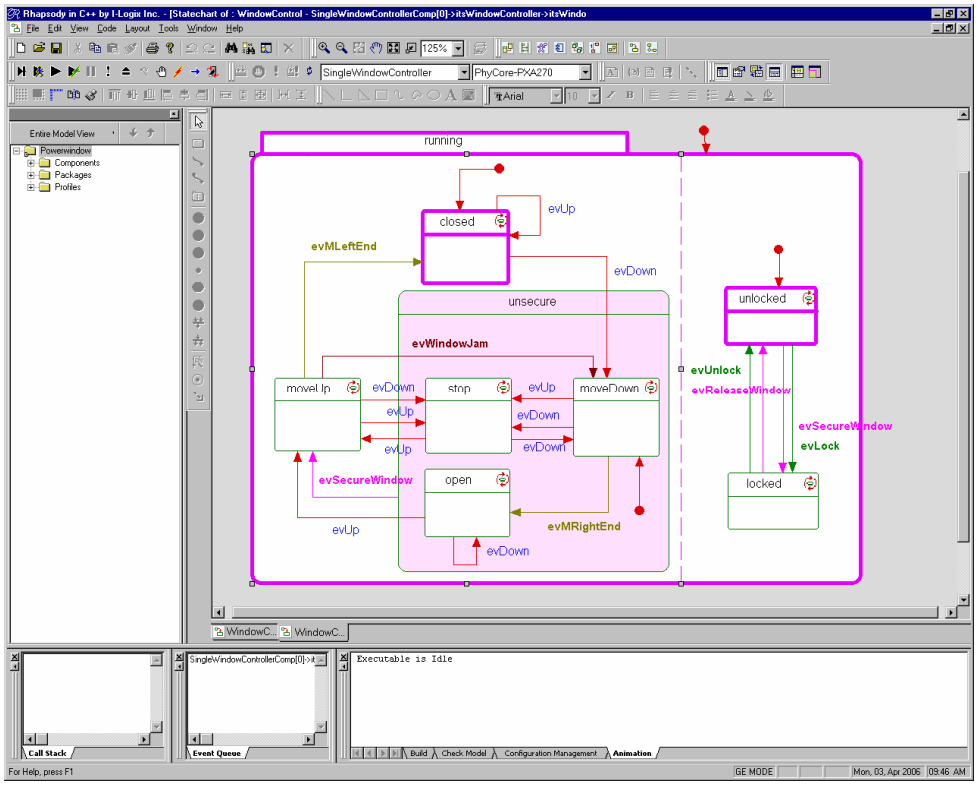
Call Stack and Event Queue:

- Click on *Go Idle* 
- Select in the menu bar *Tools -> Animated Statechart*

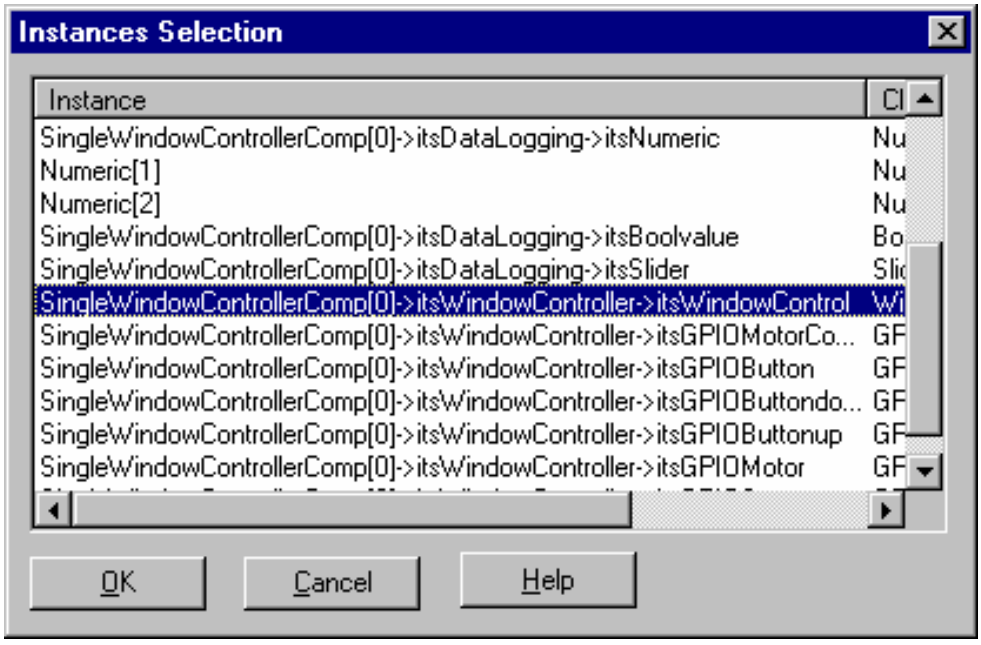


- Select *SingleWindowControllerComp[0]*
  -> itsWindowController->itsWindowControl
- Click *OK* button.

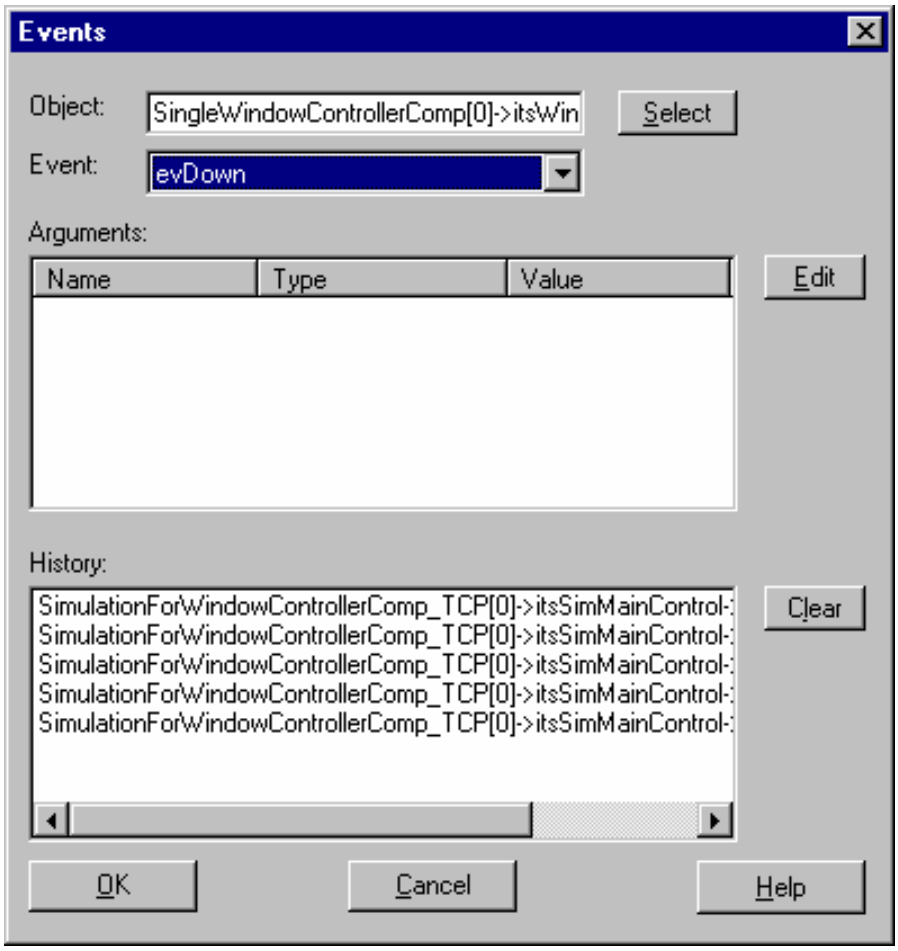
A new window with the animated statechart opens. The active states are highlighted.



- Call *Event Generator*
- Click *Select* Button

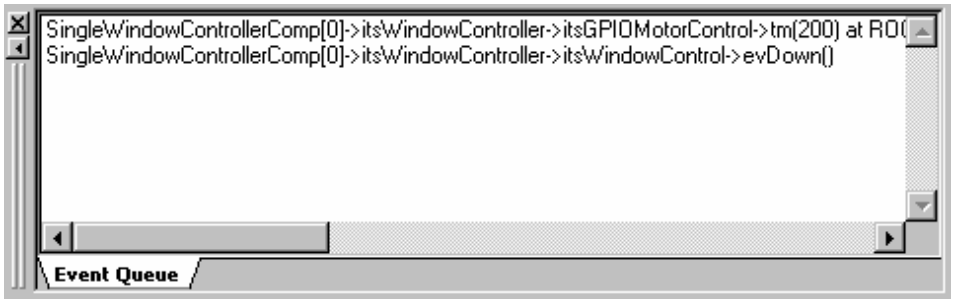


- Select SingleWindowControllerComp[0] ->itsWindowController->itsWindowControl
- Click on *OK* button.



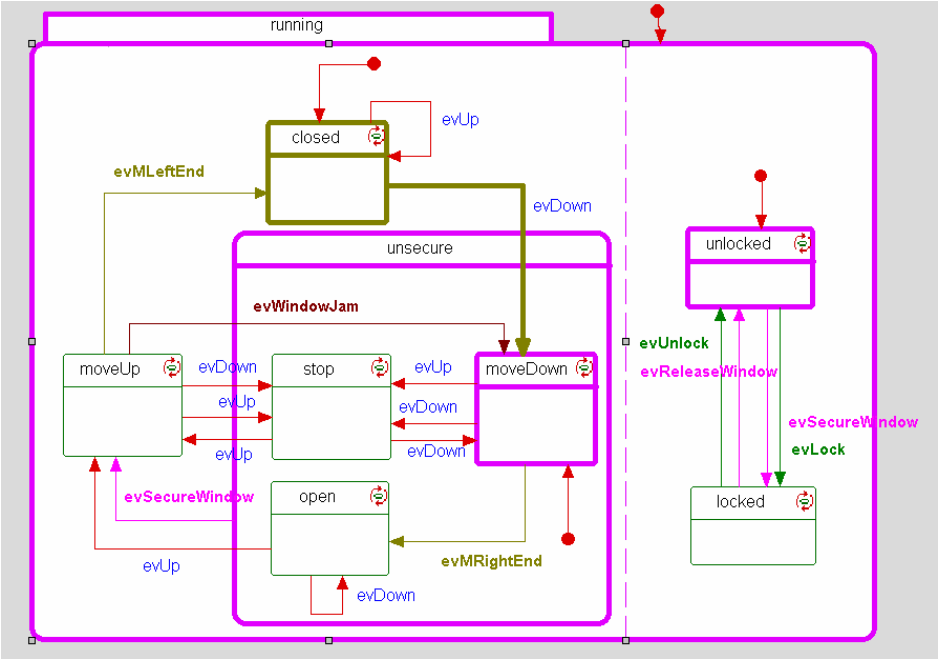
- Select the Event *evDown*
- Click Button *OK*

You see the generated event in the event Queue.



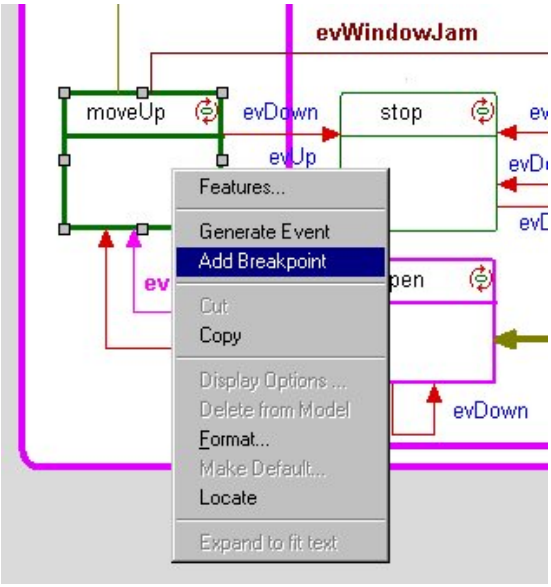
- Click on *Go Idle*

The active state changes to *moveDown*.

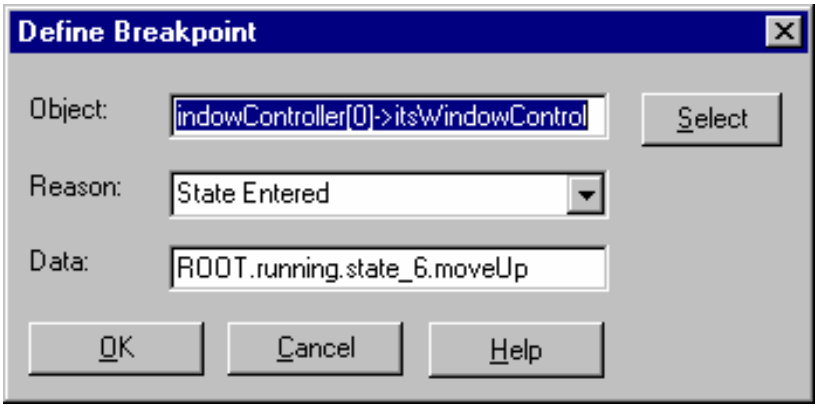


- Click on *Go Idle*

The active state changes to *open*.



- Right-click on the *state moveUp* and select *Add Breakboint.*

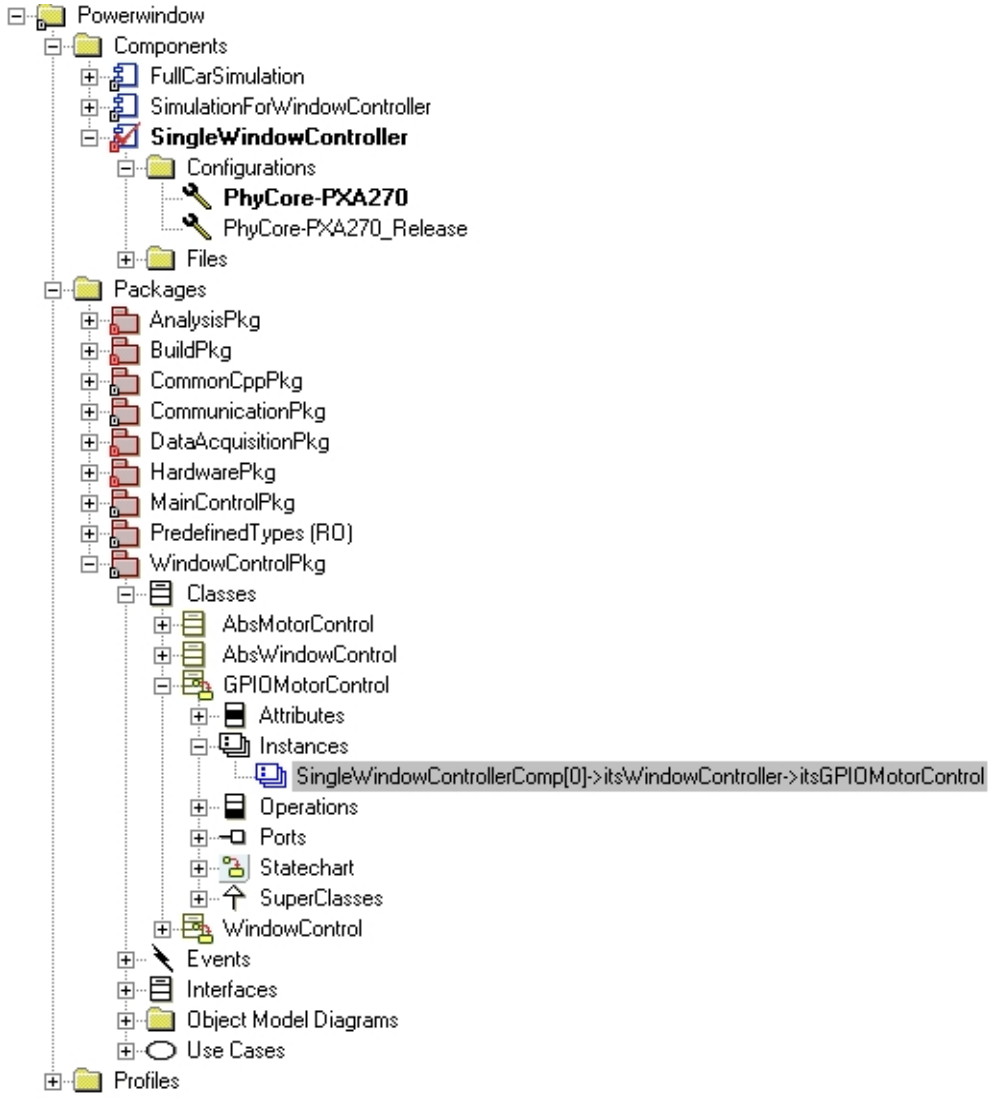


- Select OK in the dialog box to set the breakpoint

You can also add / remove breakpoints with the breakpoint icon in the animation toolbar

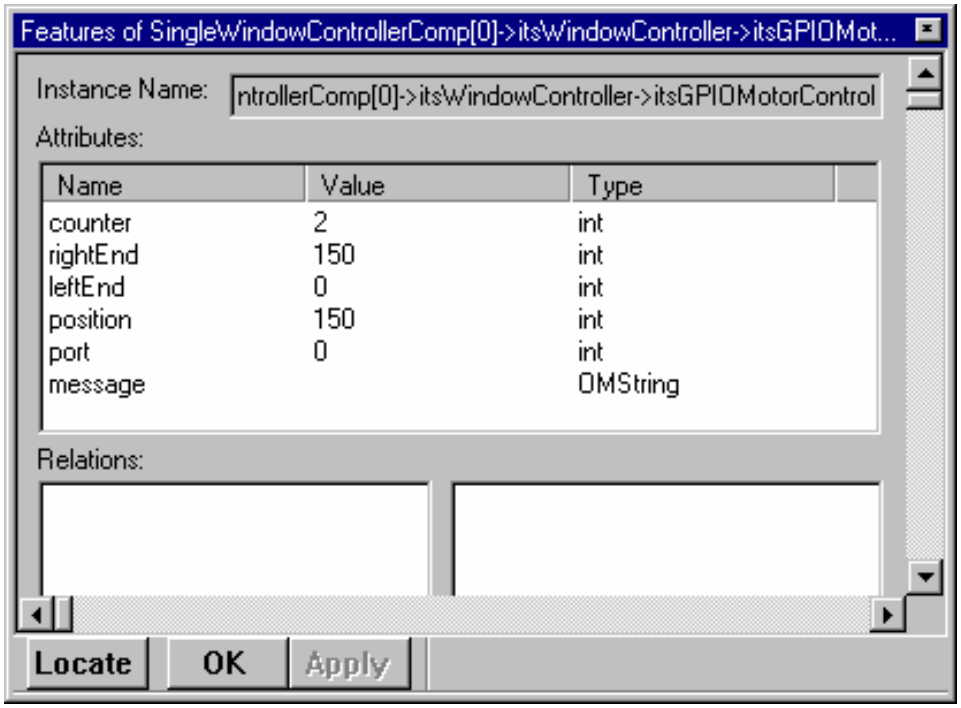
- Click on *Go*
- Push Button *Key1* on the Extension Board

The animation stops when the executable reaches the breakpoint.



- Expand the browser tree.
- Select the class *GPIOMotorControl* in *Packages-> WindowControlPkg->Classes*
- Expand the class *GPIOMotorControl*.
- Double-click in Instances on *SingleWindowControllerComp[0]->itsWindowController ->itsGPIOMotorControl*

A new dialog appears. You can see the names, values and types of the attributes in the selected instance.



- Click Go 

The motor starts to move and the values are changing. The active state changes to closed.

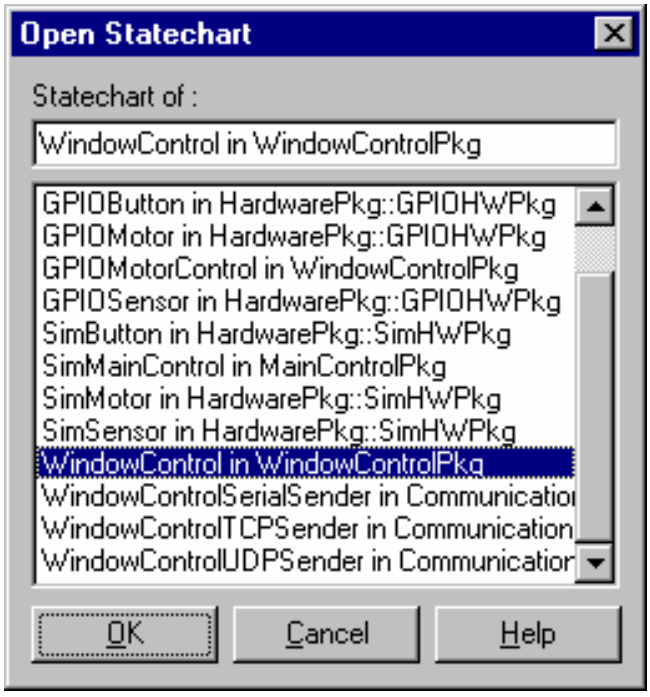
- Click *Animation break* 
- Click *Quit Animation* 
- Press *Enter* to confirm with *Yes*

3.3 Changing the demo application

This section will show how you can change the existing model.

- Select in the menu bar *Tools -> Statechart*.

The *Open Statechart* dialog opens.



- Select *WindowControl in WindowControlPkg*
- Click button *OK*.

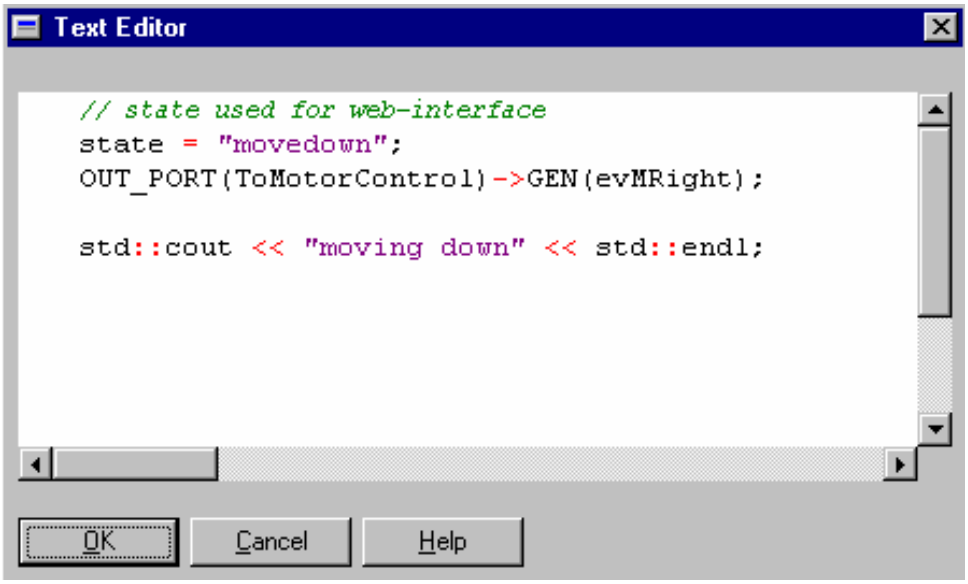
The statechart of the class *GPIOMotorControl* opens.

- Double-click the state moveDown.

A new dialog opens.

- Click on the *Edit* icon.

The Text Editor opens.



- Enter the following line:

  std::cout << “moving down” << std::endl;
- Click on button *OK*.

- Double-click the state moveUp.
- Click on the *Edit* icon.
- Enter the following line:

  std::cout << “moving up” << std::endl;
- Click button *OK*

- Click on *OK* to accept the settings.

- Save the configuration 
- Click on *Generate / Make / Run* 

- Click on *Go* 
- Push Button *Key2* on the Extension Board

The motor starts to move .You will see the output *moving down.*

- Push Button *Key1* on the Extension Board.

The motor starts to move .You will see the output *moving up.*

- Click *Animation break* 
- Click *Quit Animation* 
- Press Enter to confirm with yes

You have successfully passed the Getting Involved part of the QuickStart.

4 Debugging an example project

40 min

In this chapter you will learn using the GNU GDB-Debugger on the Host for Remote Debugging in conjunction with the GDB-Server on the target. The GNU GDB debugger is the symbolic debugger of the GNU project and is arguably the most important debugging tool for any Linux system.

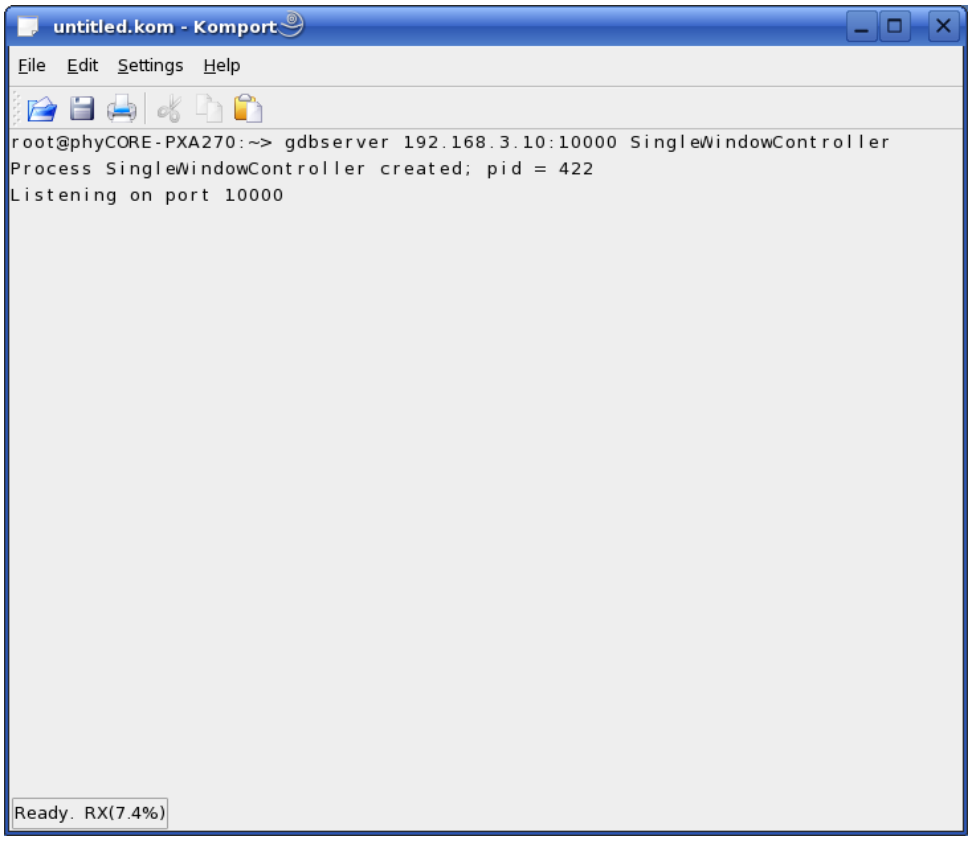
First you will start the GDB-Server on the target. Then you will configure the Eclipse Platform and start the GDB-Debugger out of Eclipse using the Debug View.

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 suspended threads for 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 is running on the host and used to debug. The GDB-Server is running on the target and it is used to start and control the program to debug. The GDB and GDB-Server can communicate over TCP/IP and the serial interface. In this QuickStart we will only describe debugging via TCP.

To debug the example project, Rhapsody has to be executed and the Powerwindow project has to be opened. So if Rhapsody, is closed, start the application and open the Powerwindow project.
Don’t’ start the application on the target.

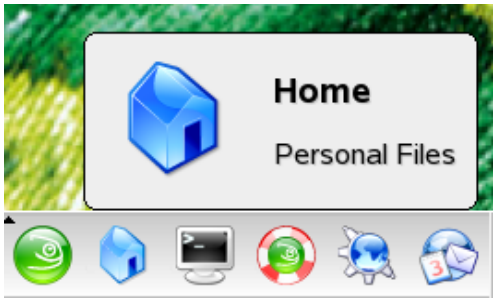
4.1 Starting the GDB Server on the target



- Open Komport.
- Type **root** and press *Enter*.
- Start the GDB Server:

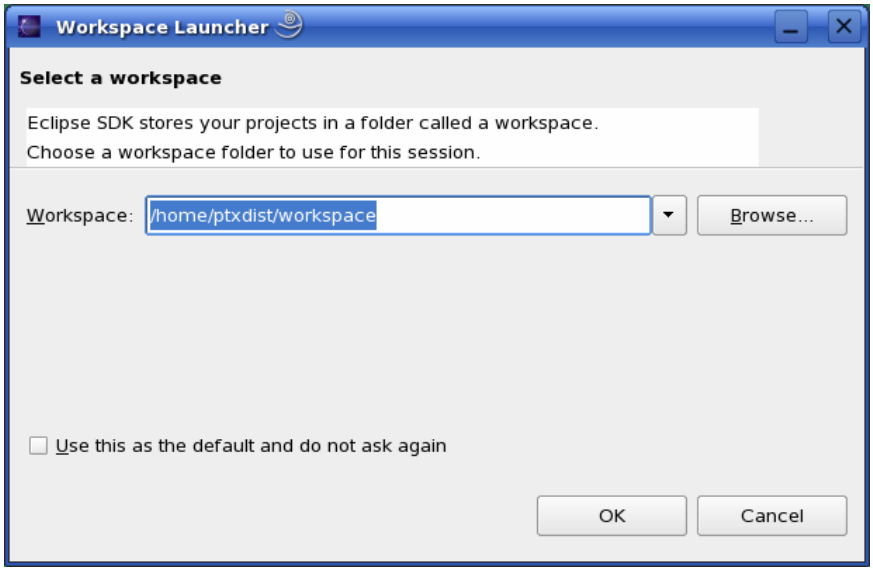
  gdbserver 192.168.3.10:10000 SingleWindowController

4.2 Starting Eclipse



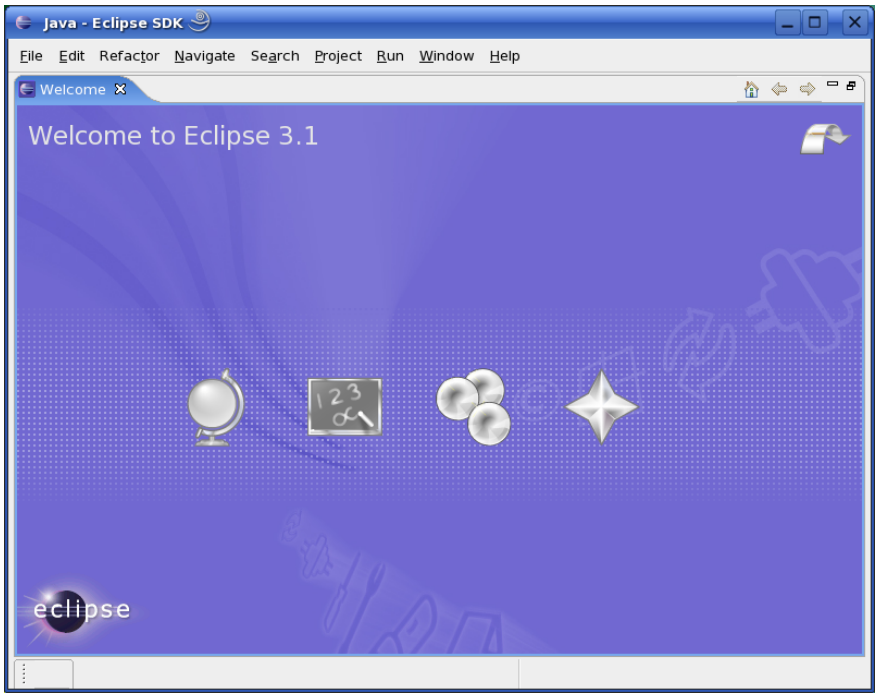
- Click on the *Home* Icon to start the *Konqueror* browser.

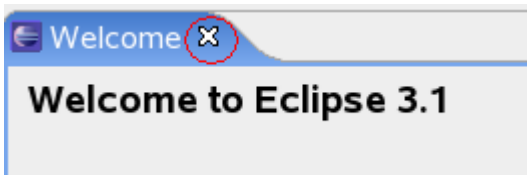
- Browse to the directory */usr/local/share/UMLKit/eclipse.*
- Click on the *Eclipse* icon to start the application.



- Confirm the workspace directory with *OK*.

The welcome screen will appears.

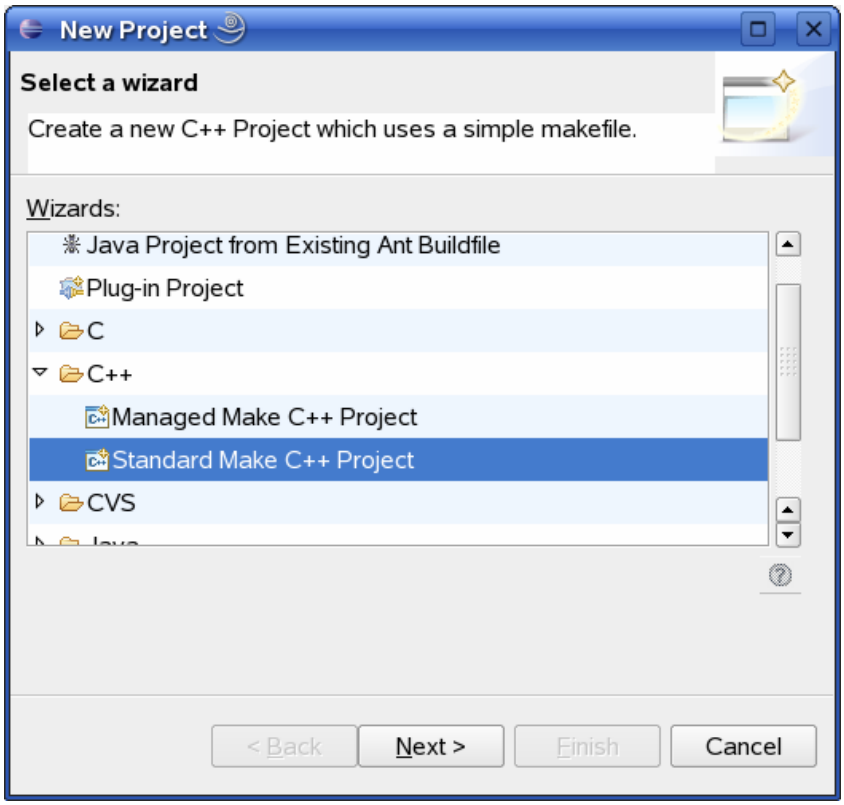




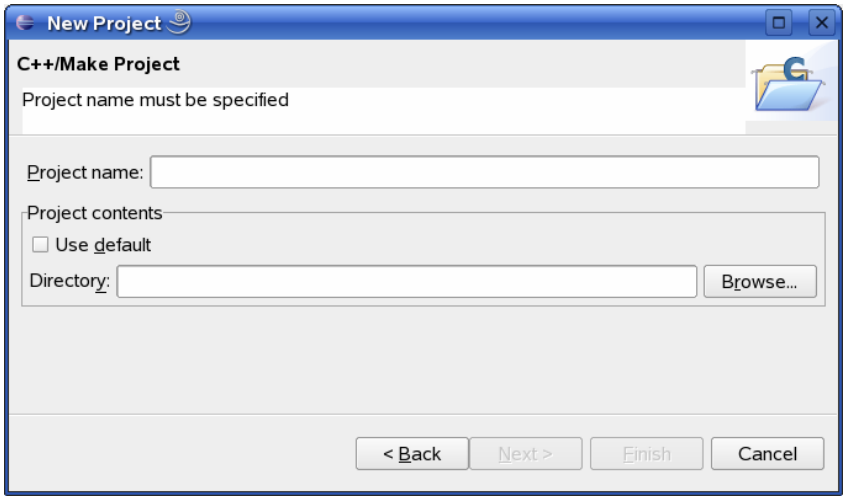
- Click the cross right of welcome to close the welcome screen

4.3 Open the demo project

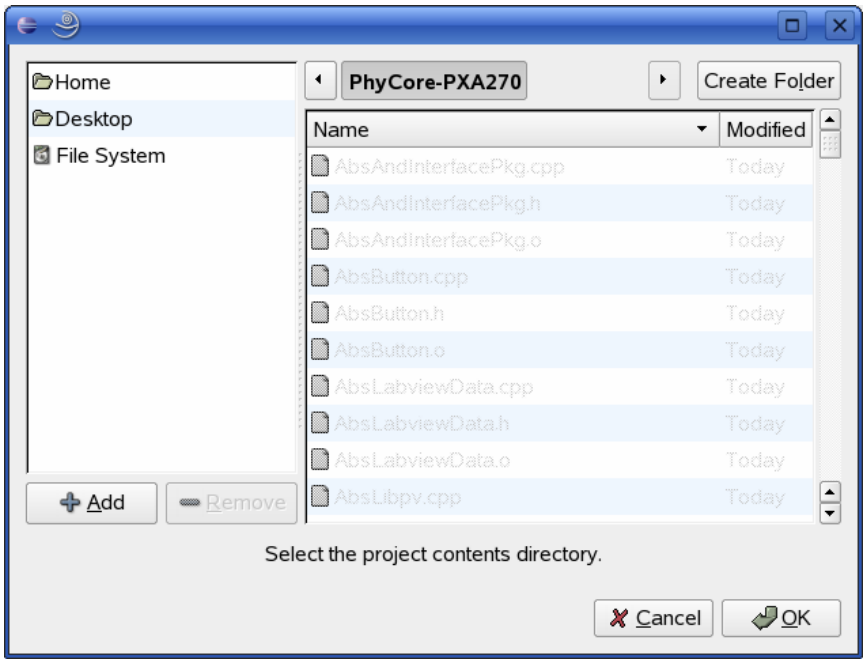
- Select in the menu bar *File -> New -> Project.*



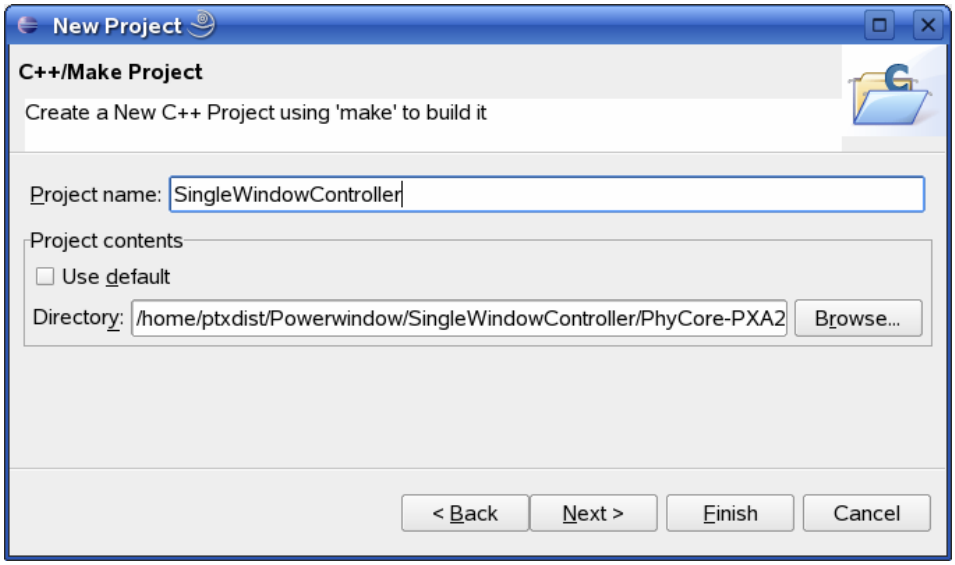
- Select *Standard Make C++ Project* in C++.
- Click the *Next* button.



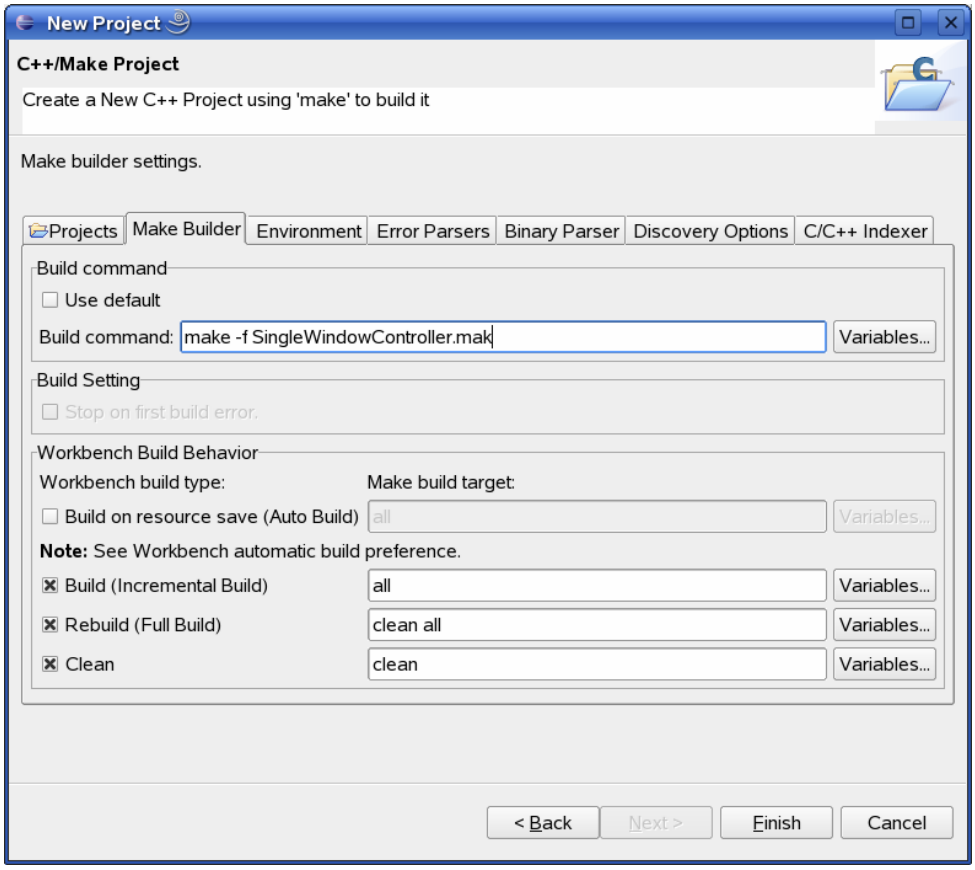
- Uncheck *Use default* in *Project contents*.
- Click on the *Browse* button.



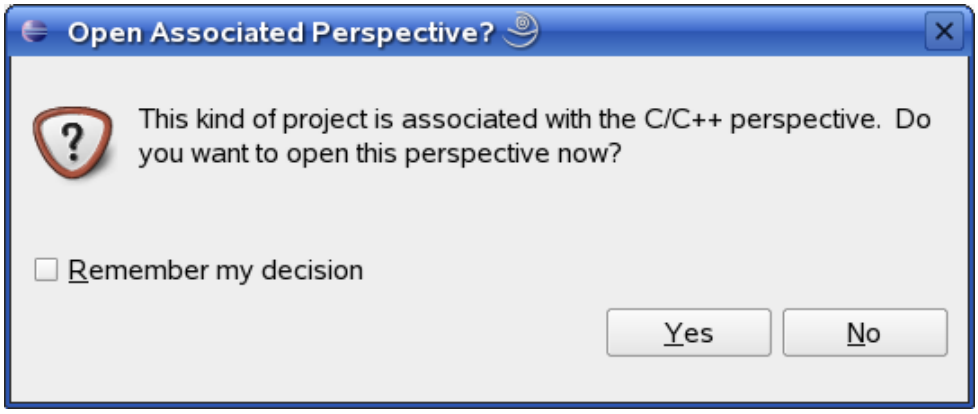
- Navigate to the directory
  Home/Powerwindow/SingleWindowController/PhyCore-PXA270
- Click button *OK*.



- Enter the project name **SingleWindowController**.
- Click button *Next*.

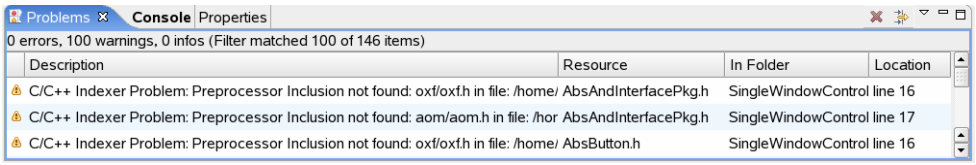


- Select the *Make Builder* tab.
- Uncheck *Use default*.
- Enter **make -f SingleWindowController.mak**.
- Select button *Finish*.



- Select button *Yes* to change to the C/C++ perspective.

The project will be build. After building some Indexer Problems occurs.

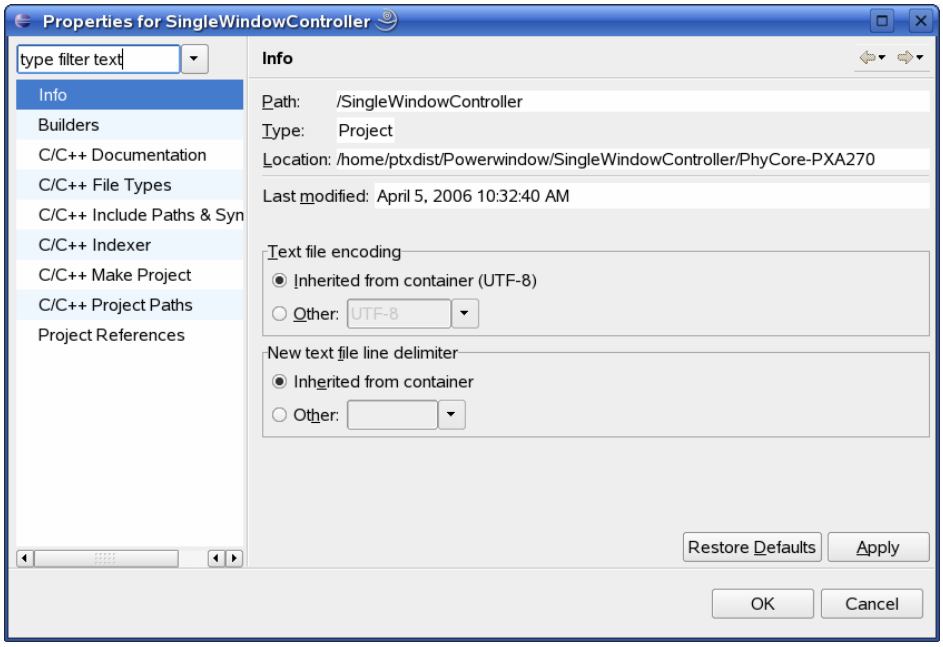


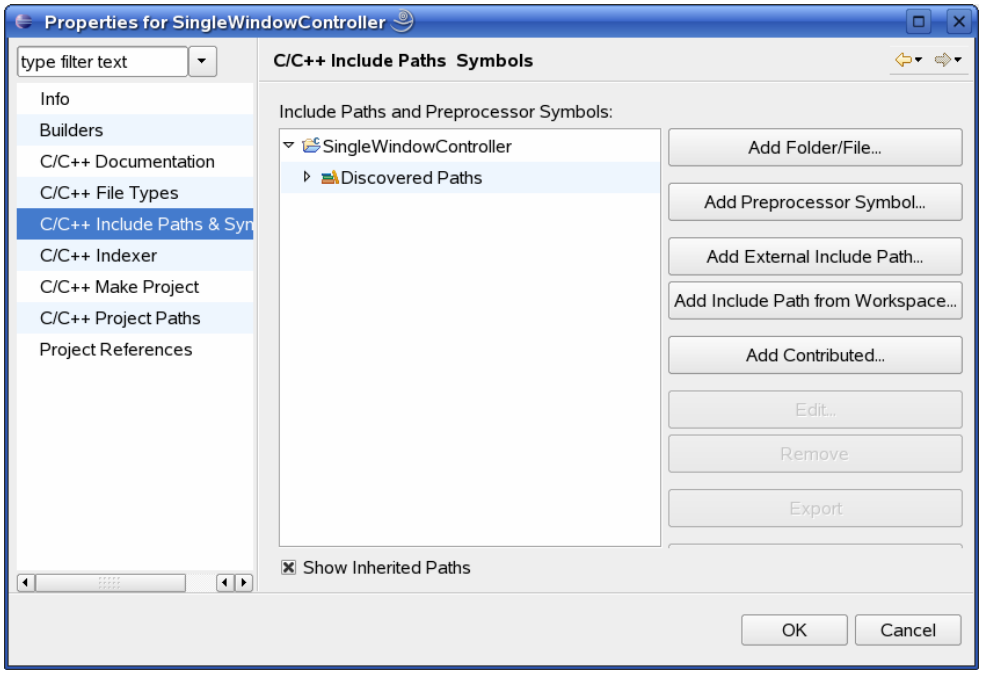
To solve these problems you will have to configure the include paths in the next section.

4.4 Configuring Include - Paths

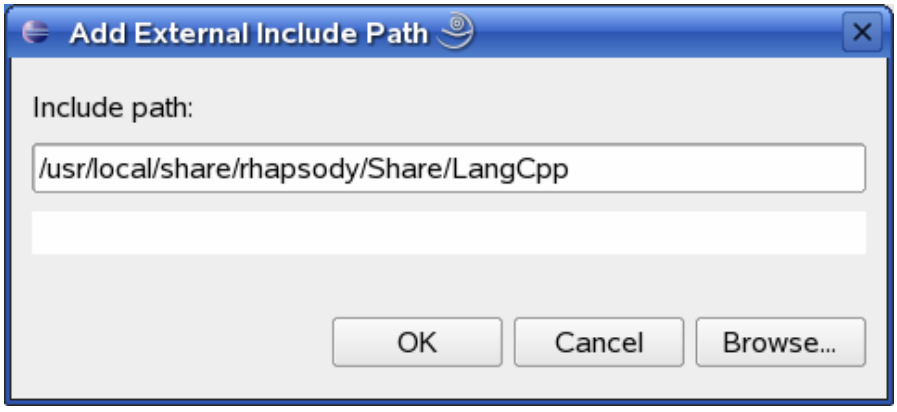
- Right click *SingleWindowController* in C/C++ Projects.
- Select *Properties*.

The properties dialog opens.





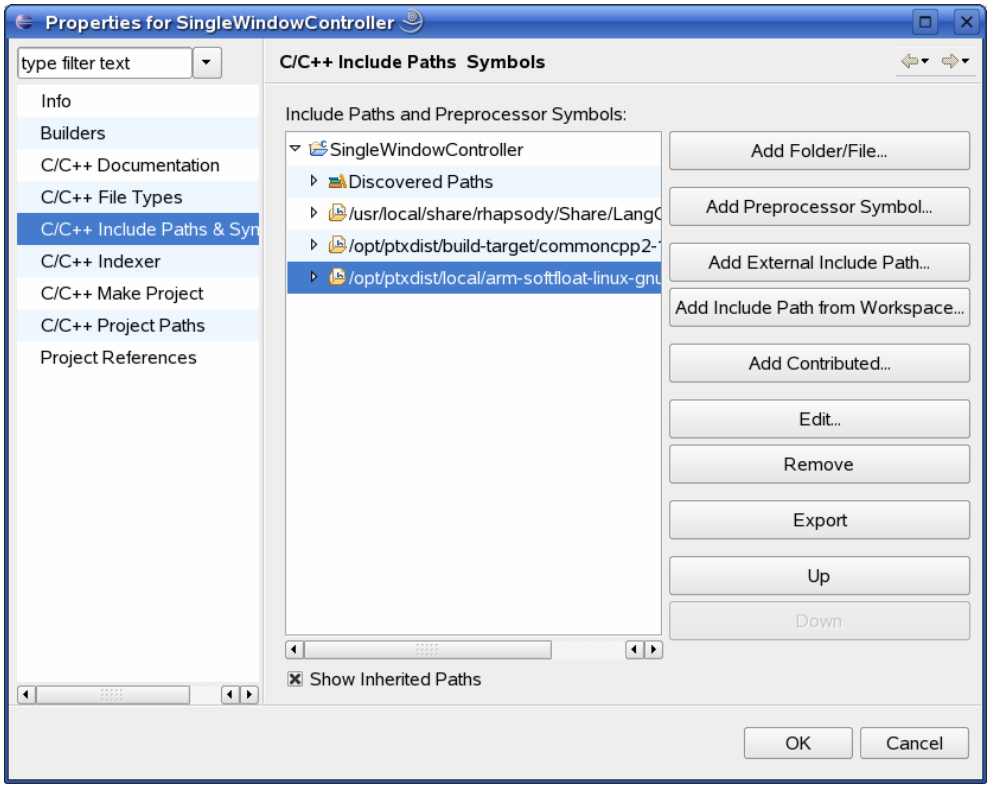
- Select *C/C++ Include Paths & Symbols*.
- Select *Add External Include Path* from Workspace.



- Click Button *Browse*.
- Double-Click *File System*.
- Navigate to **/usr/local/share/rhapsody/Share/LangCpp**.
- Click button OK.
- Click button OK.

- Select *Add External Include* Path from Workspace again.
- Click Button *Browse*.
- Double-Click *File System*.
- Navigate to the directory
  /opt/ptxdist/build-target/commoncpp2-1.3.24/include
- Click button *OK*.
- Click button OK again.

- Select *Add External Include* Path from Workspace again.
- Click Button *Browse*.
- Double-Click *File System*.
- Navigate to the directory
  /opt/ptxdist/local/arm-softfloat-linux-gnu/usr/include
- Click button *OK*.
- Click button *OK* again.



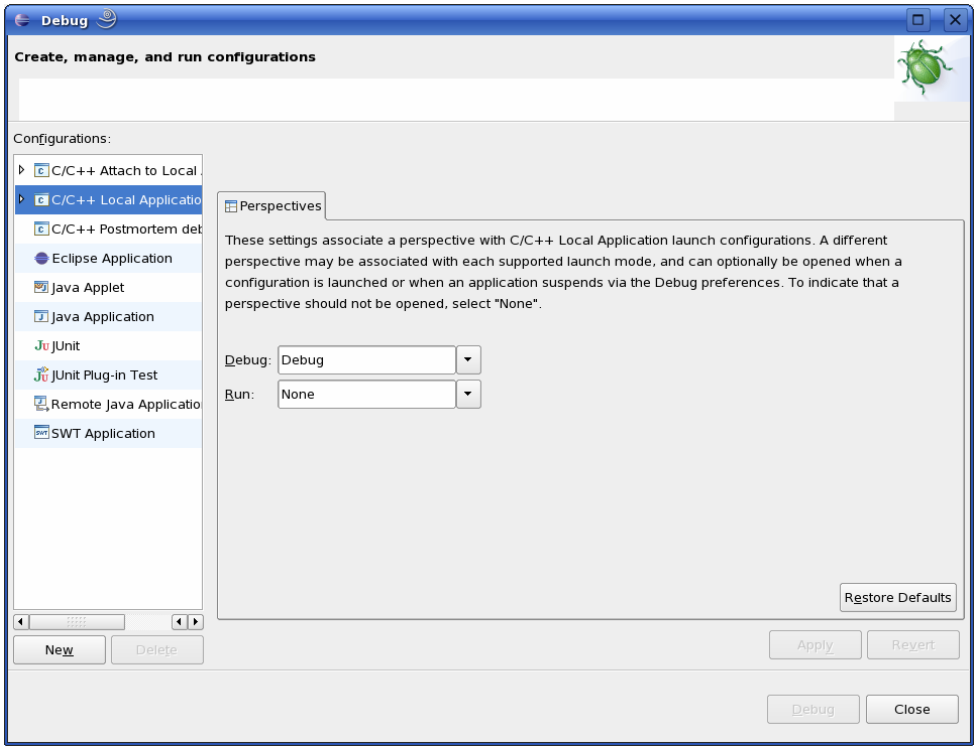
- Click button *OK* to close the properties dialog.

You will see the C/C++ Indexer in the lower right corner. After the indexer is finished you will see following window:

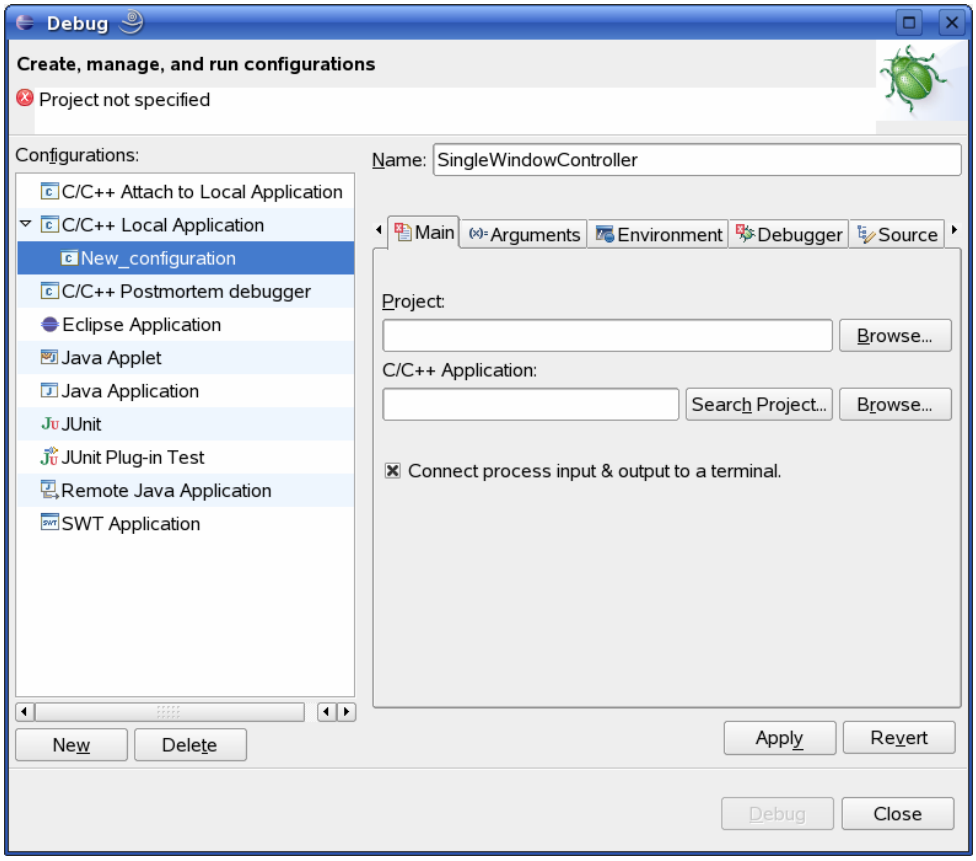
4.5 Configuring and Starting the GDB debugger

- Select in the menu bar *Run->Debug*.

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



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

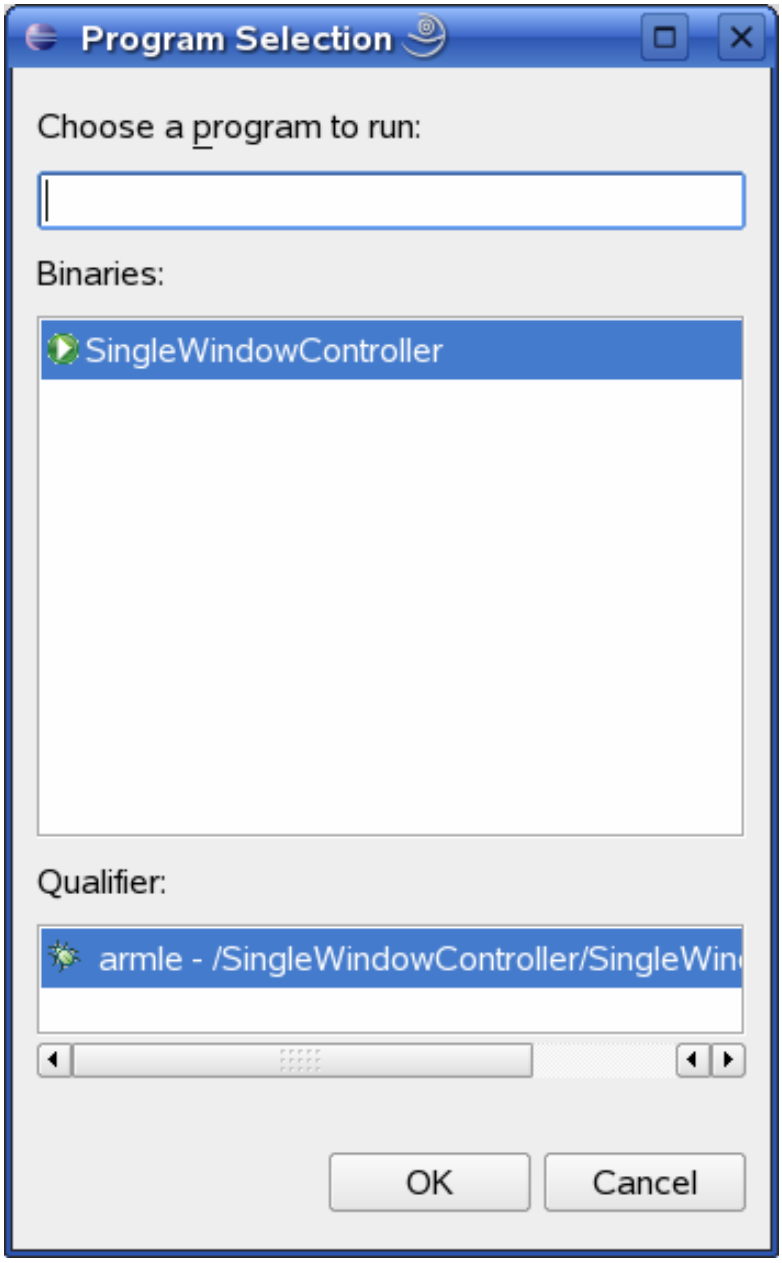


- Enter the Name **SingleWindowController.**
- Select *Browse* right of the Project text field.

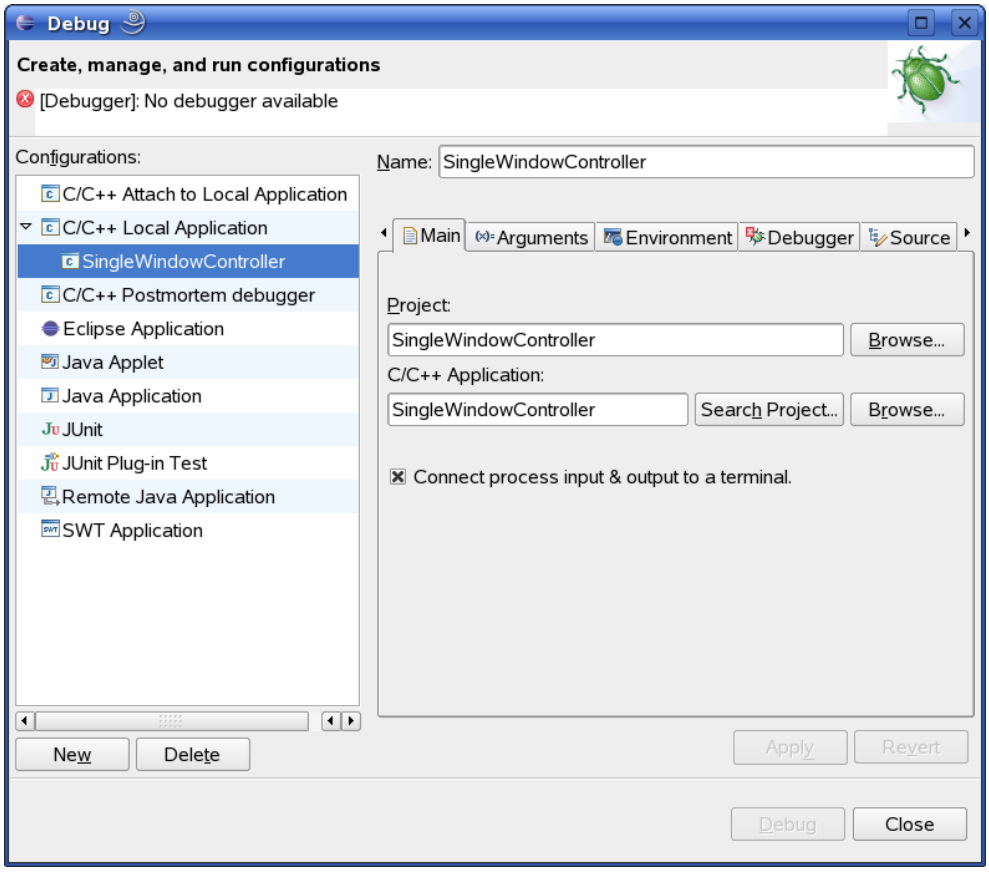
- Select SingleWindowController.
- Click button OK.

- Select Search Project.

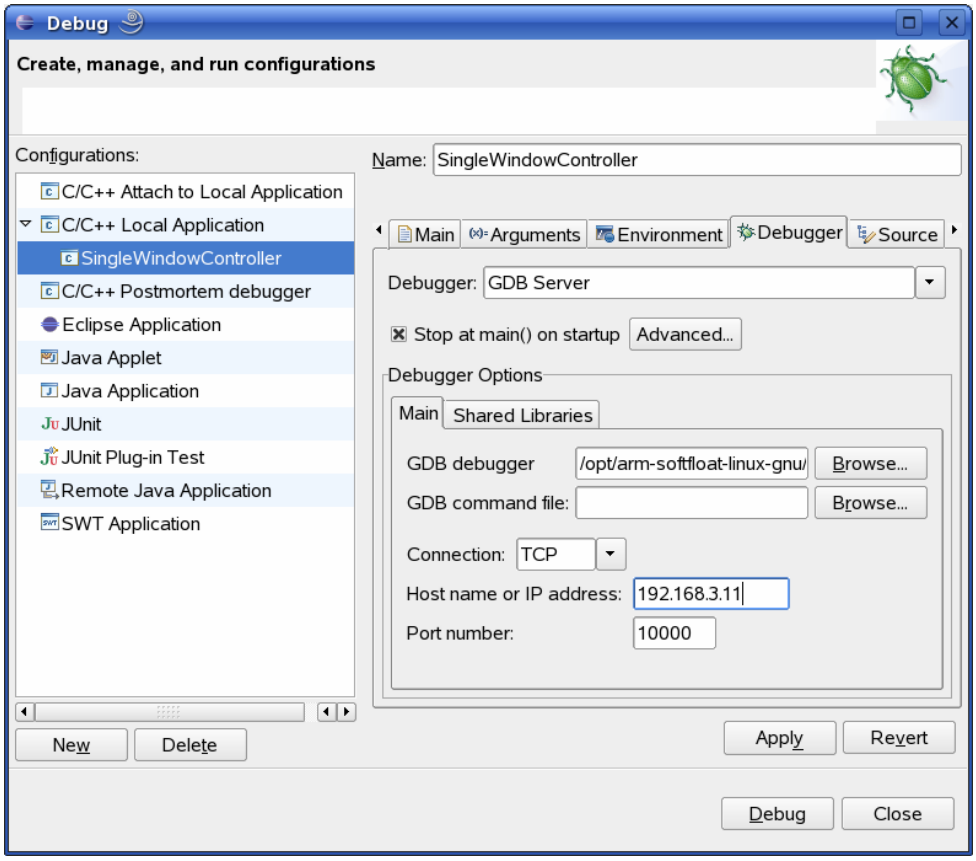
The Program Selection dialog will open.


Program Selection
Choose a program to run:
Binaries:
SingleWindowController
Qualifier:
armle - /SingleWindowController/SingleWin
OK
Cancel

- Click on *OK*



- Click on *Apply*.



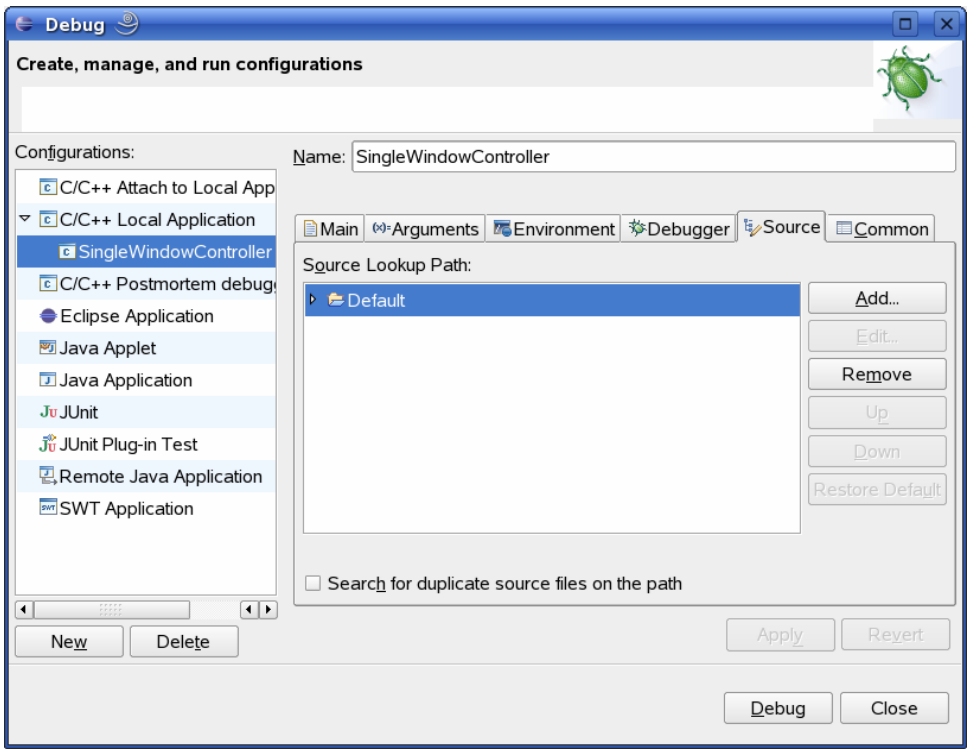
- Select the *Debugger* tab.
- Select Debugger: *GDB Server*.
- Select *Browse*(GDB debugger)

A new dialog opens.

- Double-Click *File System*.
- Navigate to */opt/arm-softfloat-linux-gnu/bin*
- Select *arm-softfloat-linux-gnu-gdb*
- Select *OK*.

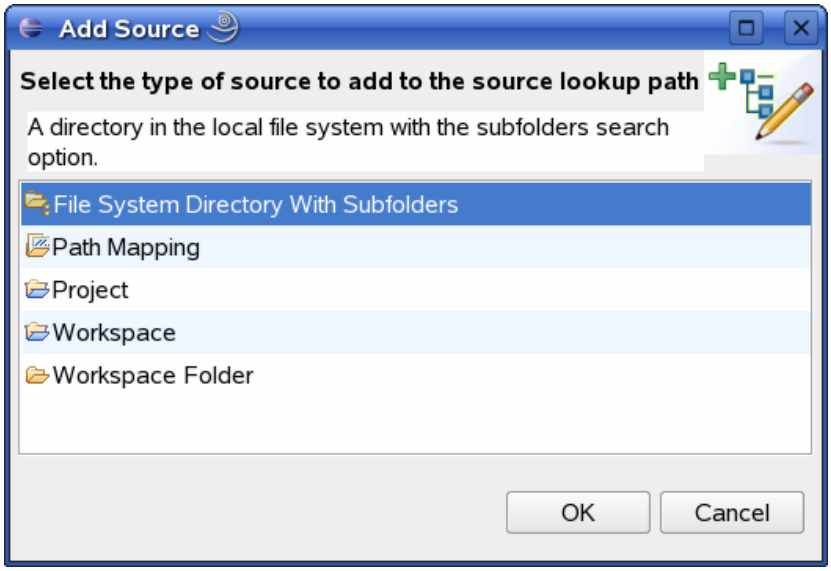
- Select Connection: *TCP*.
- Enter the Host name IP address: **192.168.3.11**
- Click *Apply* button.

The host name IP address, is the IP address of the target.

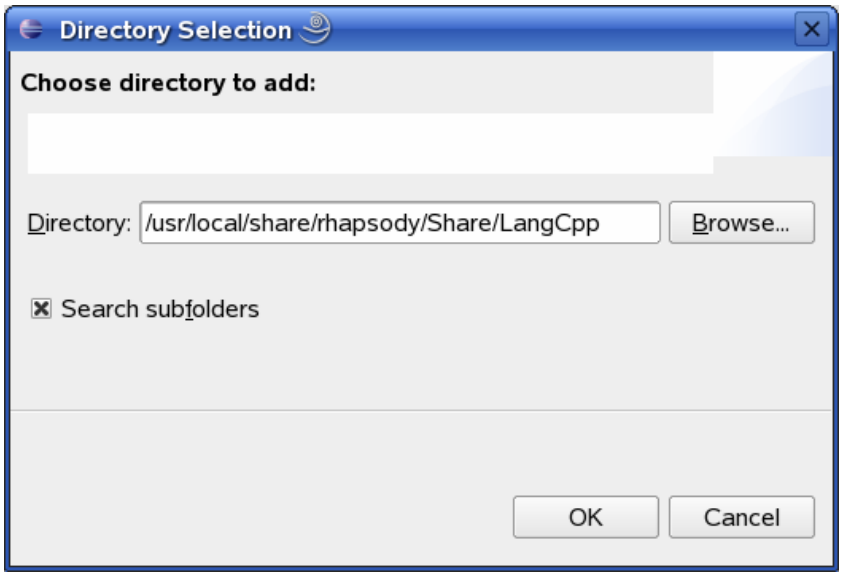


- Select the *Source* tab.
- Click button *Add*.

The Add Source dialog opens.

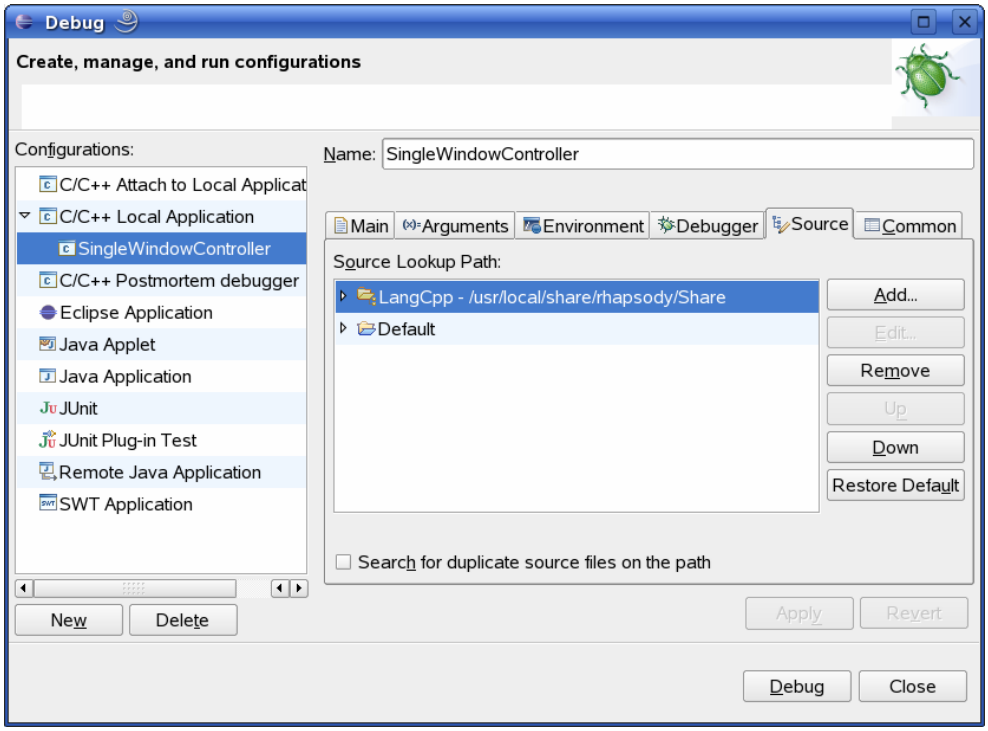


- Select *File System Directory With Subfolders*
- Click button *OK*.



- Click button *Browse*.
- Double-Click *File System*.
- Browse to the directory:
  /usr/local/share/rhapsody/Share/LangCpp/

- Click button *OK*.
- Check *Search subfolders*.
- Click button *OK*.



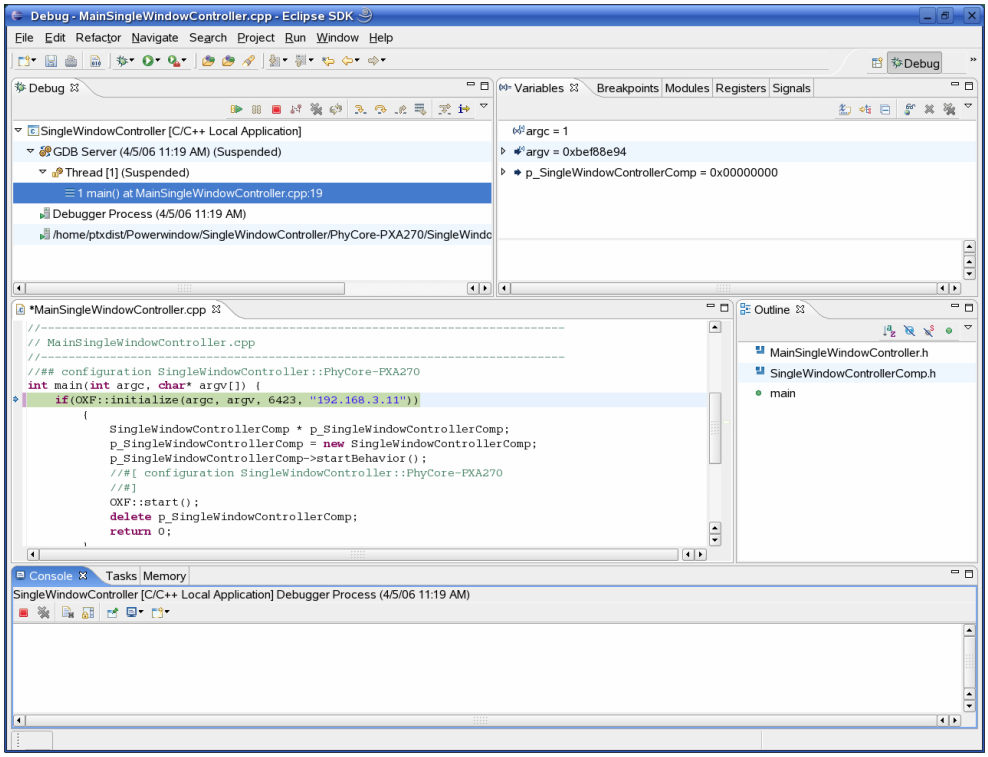
- Click button *Apply*.
- Click *Debug* button.

Be sure that the GDB Server is running on the target.

A new dialog opens.

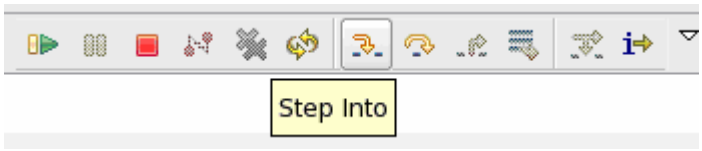
- Select *yes* to switch to the debug perspective.

The debug perspective opens and the debugger stops at the first line automatically.



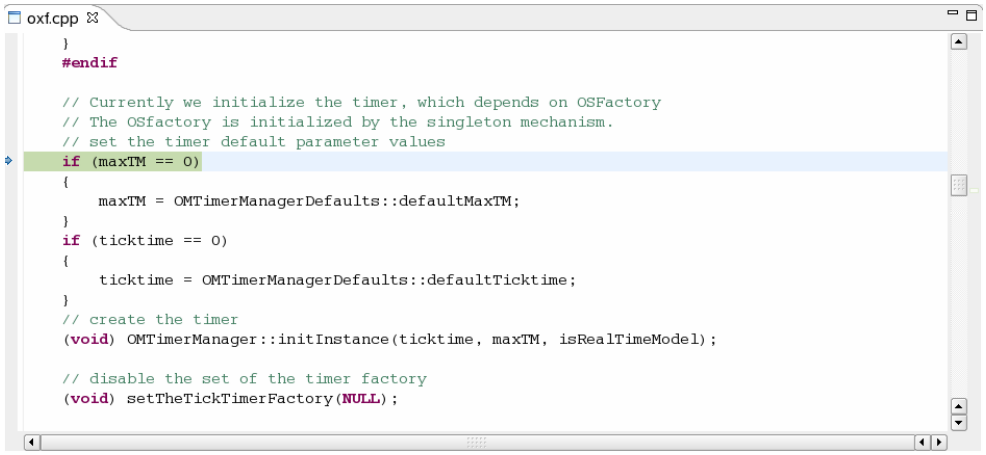
4.6 Stepping and Watching Variables Content

After you have started the GDB Server on the target and the GDB Debugger in Eclipse you can step through the project.

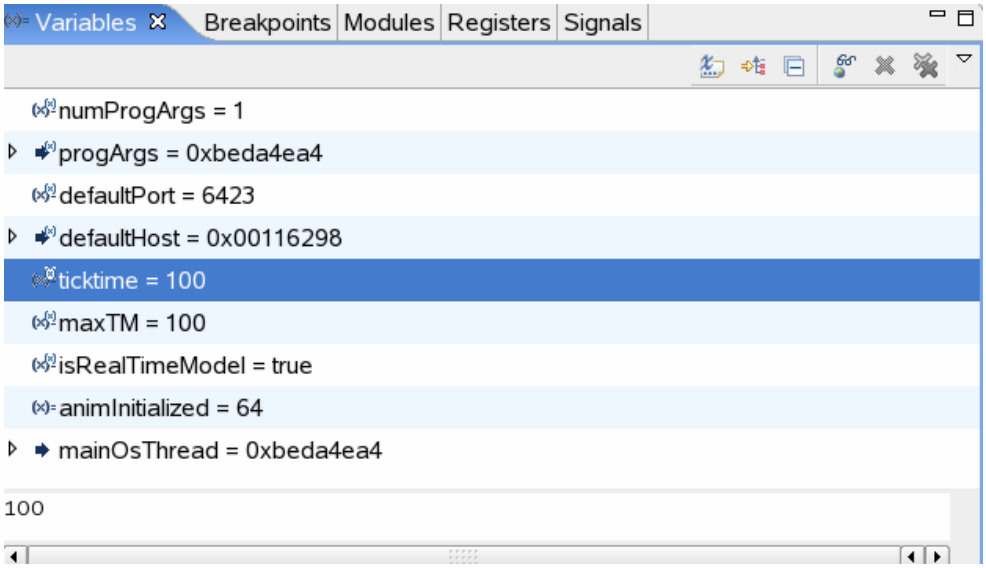


- First click the button *Step Into*.

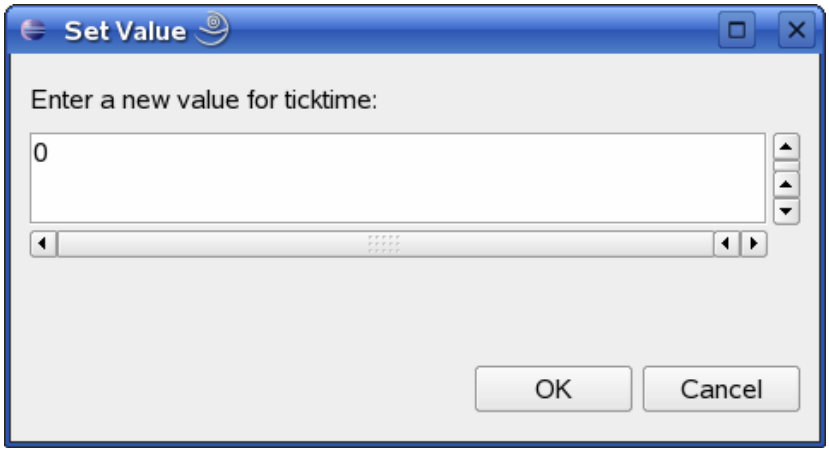
You will see the following screen:



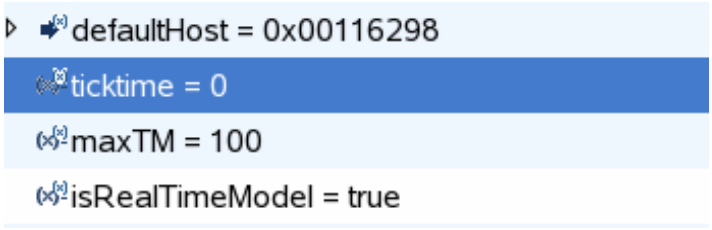
In the upper right corner you can see some variables and their values.



- Select the variable ticktime.
- Right click on the variable ticktime and select *Change Value*.



- Enter the value **0**.
- Click on button *OK*.



The values changes to 0.

```
    }
    if (ticktime == 0)
    {
        ticktime = OMTimerManagerDefaults::defaultTicktime;
    }
    // create the timer
    (void) OMTimerManager::initInstance(ticktime, maxTM, isRealTimeModel);
```

- Right-click on the grey border left of the following line *ticktime = OMTimerManagerDefaults::defaultTicktime;*

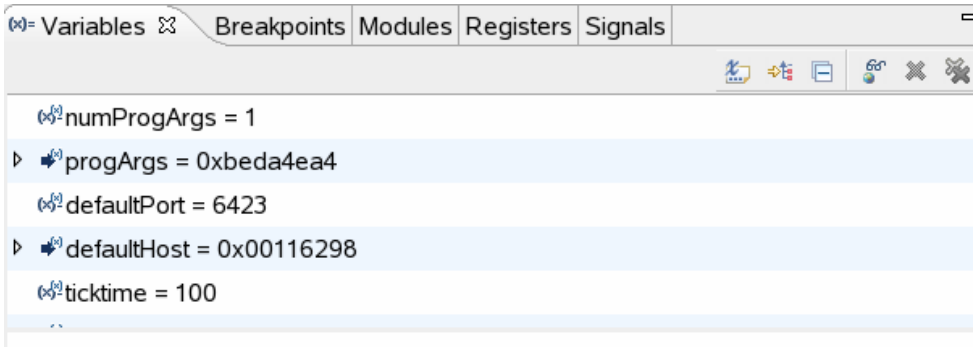
A selection menu opens.

- Select *Toggle Breakpoint* to add a breakpoint

```
    }
    if (ticktime == 0)
    {
        ticktime = OMTimerManagerDefaults::defaultTicktime;
    }
    // create the timer
    (void) OMTimerManager::initInstance(ticktime, maxTM, isRealTimeModel);
```

- Click button *Resume* 

- Click Step Over 



The value of ticktime is set back to 100.

- Select *Terminate*  to stop debugging.

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

Documentation Powerwindow Demoapplication

You can find the documentation for the Powerwindow Demoapplication in the directory */Rhapsody/Demo* on your setup cdrom.

The documentation includes information about the following topics:

- Working with Rhapsody
- IO-Interface classes
- Common-Interface classes
- Labview DataVisualization
- Adapting Rhapsody to the PXA270

PTXdist User Manual

In the PTXdist User Manual you can find further information. You can find the manual in the directory */linux/OSELAS* on your setup cdrom.

The 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-PXA270 from network only
- Running phyCORE-PXA270 stand alone
- U-Boot and phyCORE-PXA270
- phyCORE-PXA270's BSP
- Using CAN on phyCORE-PXA270

6 Summary

This QuickStart Instruction gave 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®-PXA270 in conjunction with Rhapsody.

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 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, create and build new projects and copy output files to the phyCORE -PXA270 using Eclipse.

The Debugging part of this QuickStart gave you information on setting up and using the GDB debugger with the Eclipse IDE. You learned how to set breakpoints, watching and changing variables content and using the memory monitor.

Document: **phyCORE®-PXA270 with UML QuickStart Instructions**
Document number: **L-676_2, November 2006**

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, Germany
Fax: +49 (6131) 9221-33

Published by

PHYTEC

Ordering No. L-676e_2
Printed in Germany