Manual

Development of

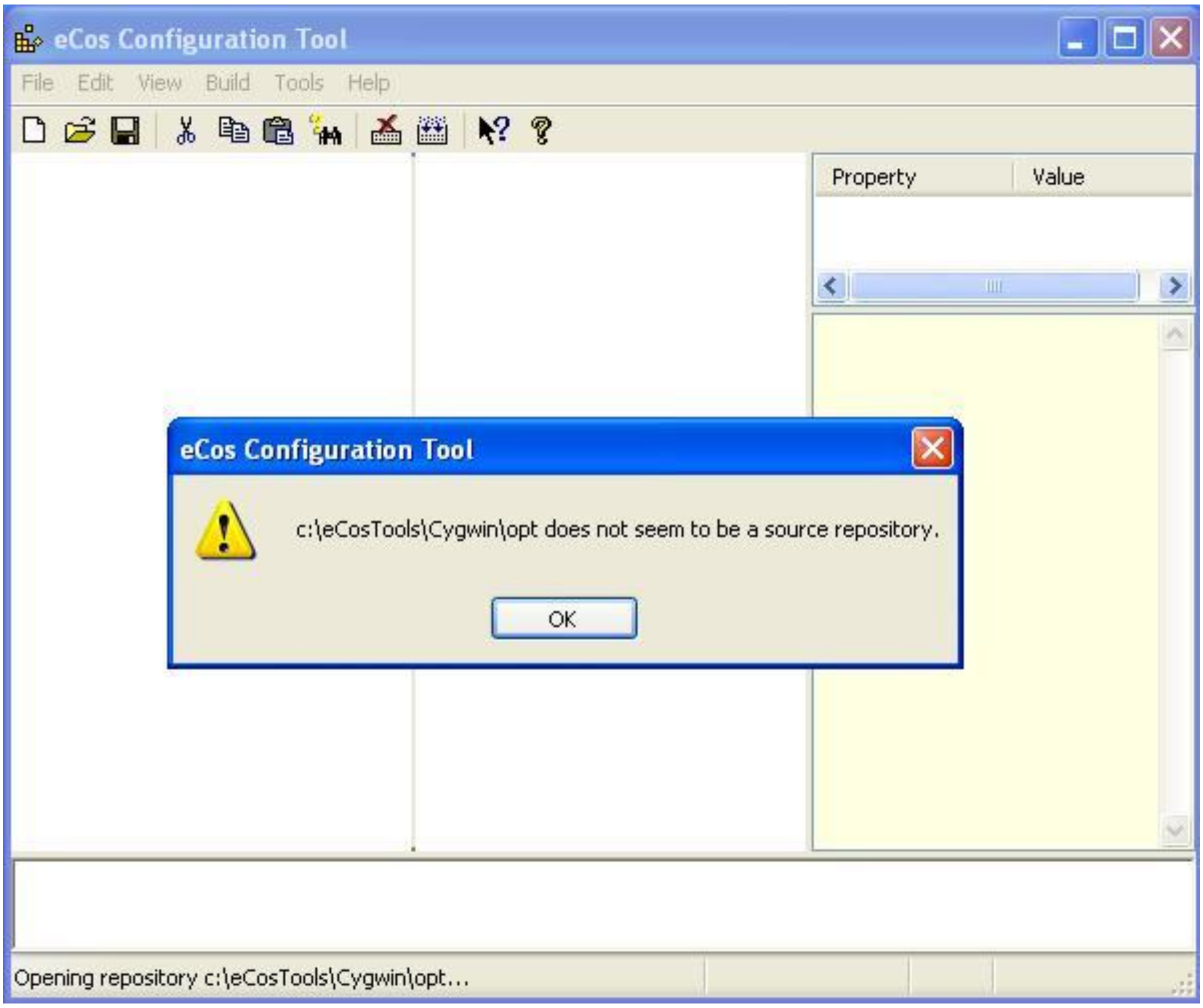
eCos Applications

using Eclipse

TABLE OF CONTENTS

Working with the eCos Configuration Tool

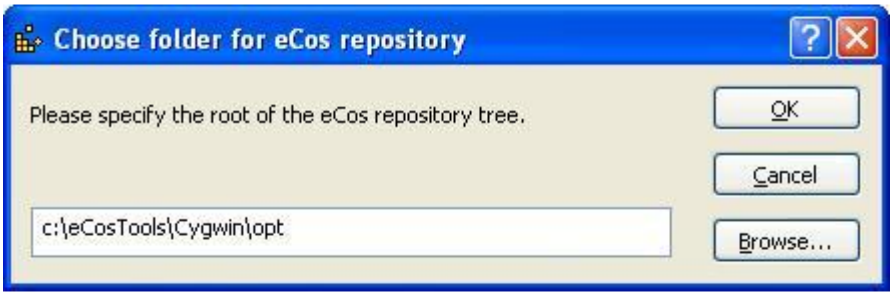
1. Open the **eCos Configuration Tool**:
   At the first time running eCos Configuration Tool a message box might appear if the eCos source repository was not correct:



c:\eCosTools\Cygwin\opt does not seem to be a source repository.

Opening repository c:\eCosTools\Cygwin\opt...

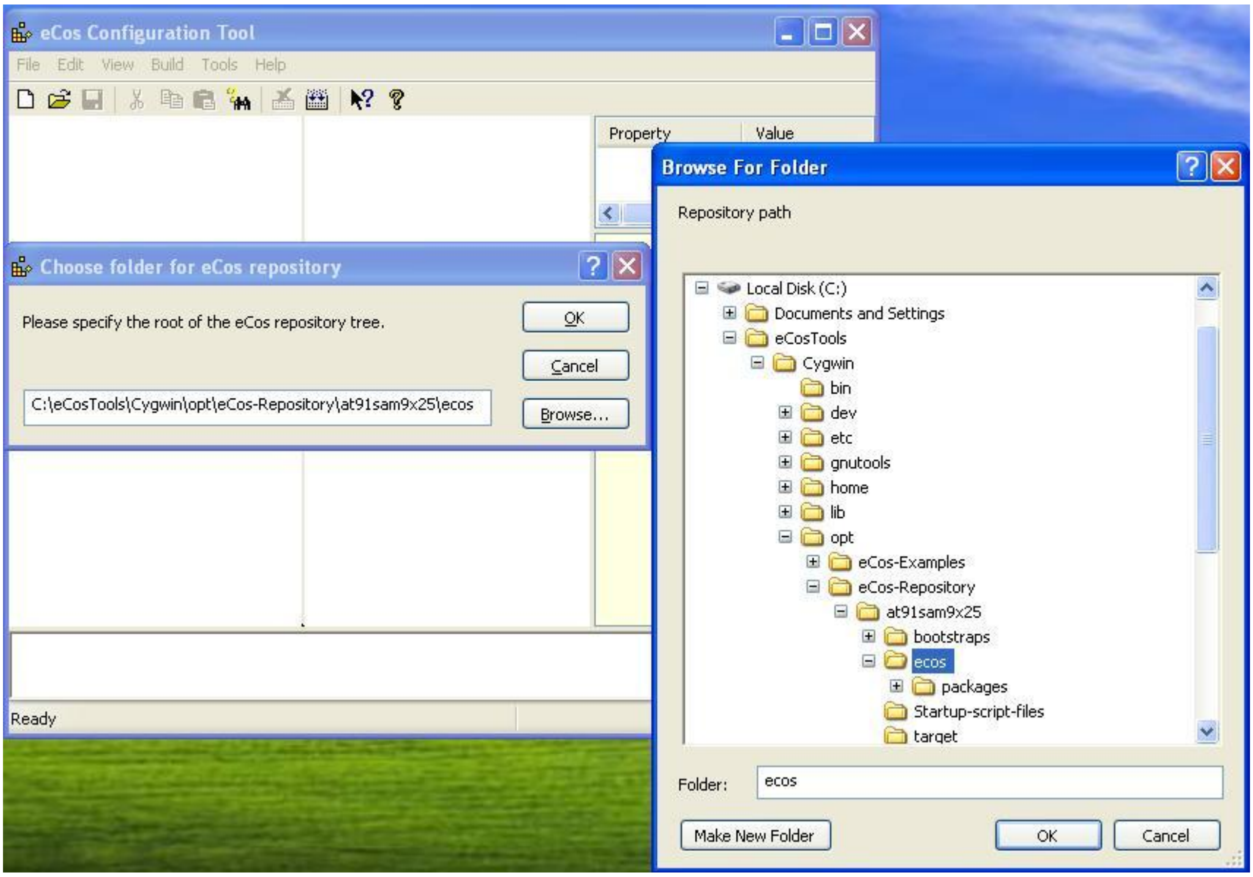
Click **OK**, a dialog for browsing the eCos repository will appear:



Please specify the root of the eCos repository tree.

c:\eCosTools\Cygwin\opt

Click **Browse…** to find the eCos repository:



Please specify the root of the eCos repository tree.

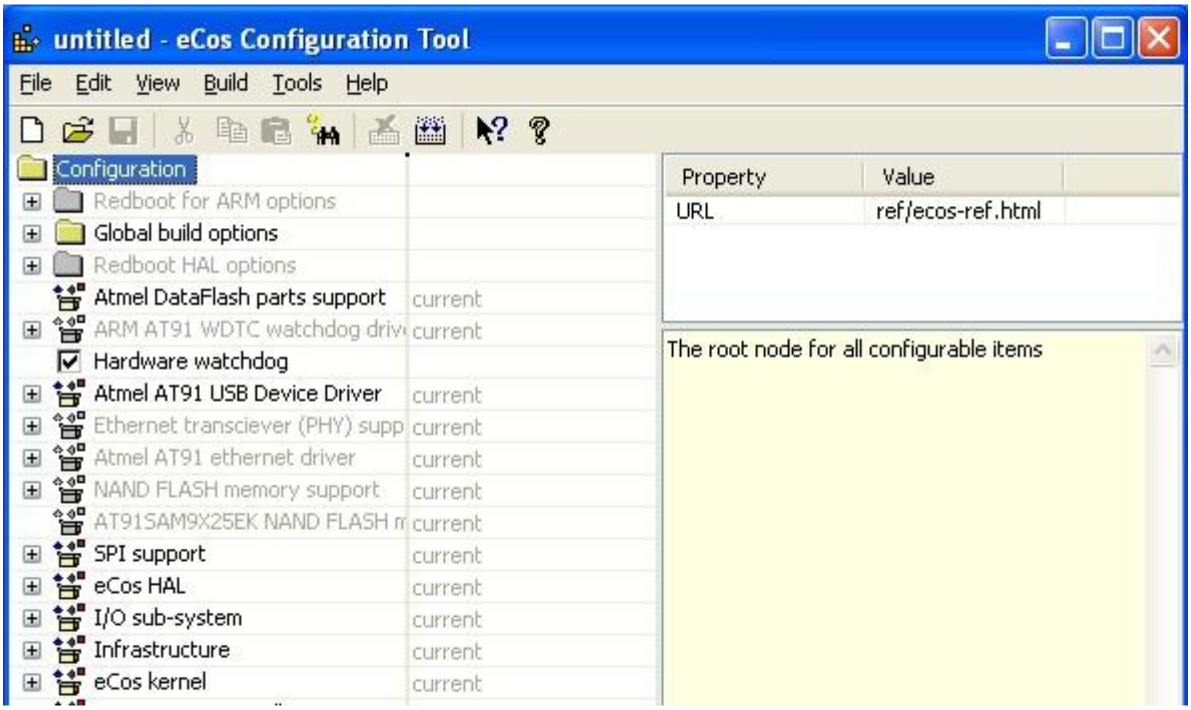
C:\eCosTools\Cygwin\opt\eCos-Repository\at91sam9x25\ecos

Repository path

```
Local Disk (C:)
  Documents and Settings
  eCosTools
    Cygwin
      bin
      dev
      etc
      gnutools
      home
      lib
      opt
        eCos-Examples
        eCos-Repository
          at91sam9x25
            bootstraps
            ecos
              packages
            Startup-script-files
            target
```

Folder: ecos

Click **OK** you can see the eCos Configuration Tools like below:

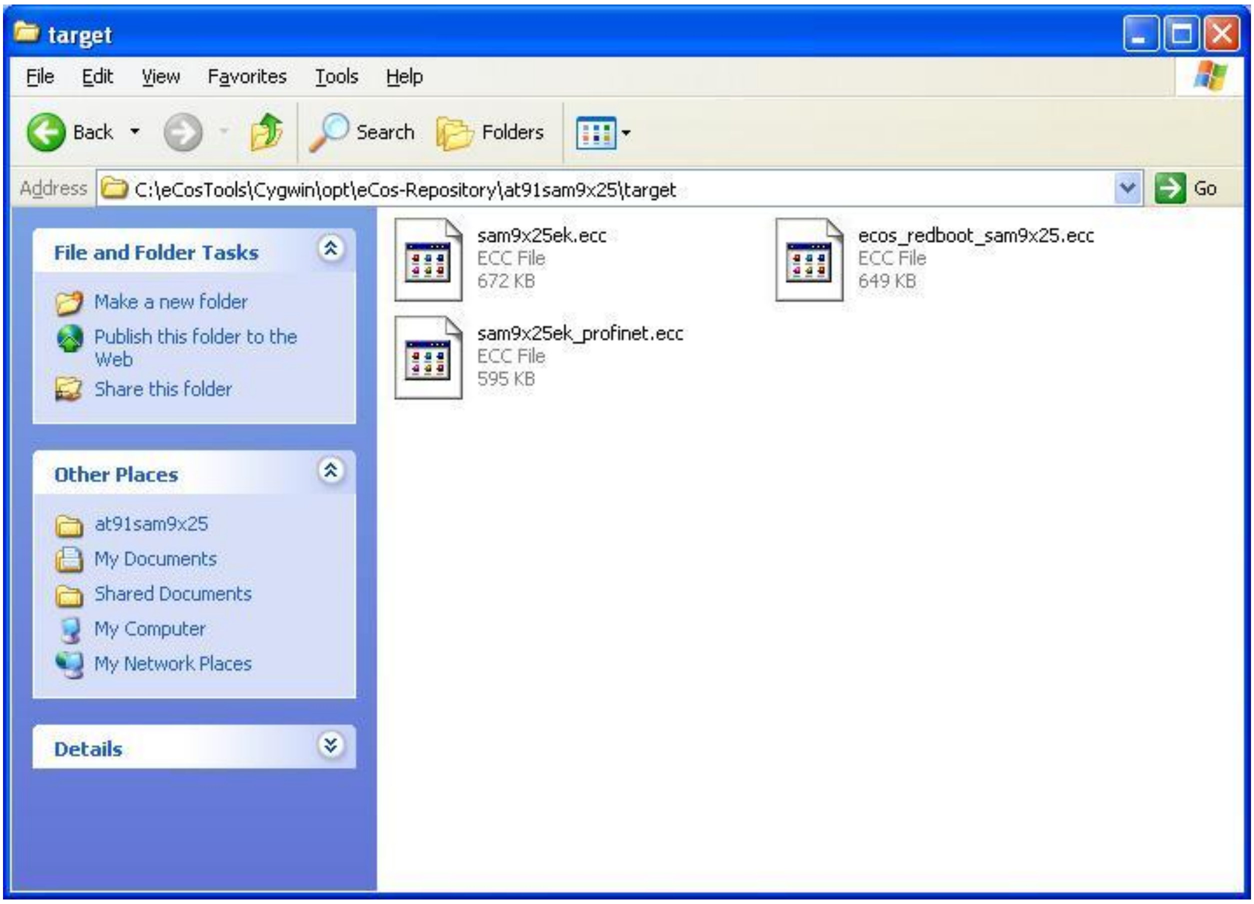

untitled - eCos Configuration Tool
File Edit View Build Tools Help

Configuration	
Redboot for ARM options	
Global build options	
Redboot HAL options	
Atmel DataFlash parts support	current
ARM AT91 WDTC watchdog driv	current
☑ Hardware watchdog	
Atmel AT91 USB Device Driver	current
Ethernet transciever (PHY) supp	current
Atmel AT91 ethernet driver	current
NAND FLASH memory support	current
AT91SAM9X25EK NAND FLASH m	current
SPI support	current
eCos HAL	current
I/O sub-system	current
Infrastructure	current
eCos kernel	current

Property	Value
URL	ref/ecos-ref.html

The root node for all configurable items

2. Inside the **eCos Configuration Tool**, open file **sam9x25ek.ecc**. By default it's located in **C:\eCosTools\Cygwin\opt\eCos-Repository\at91sam9x25\target\**



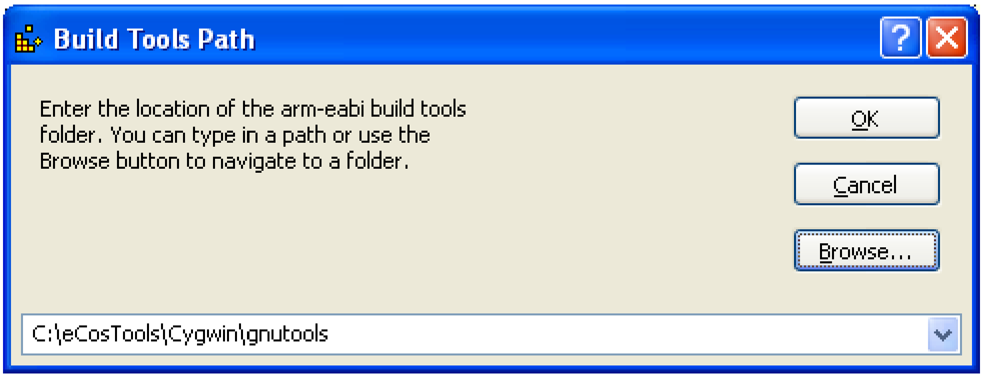
C:\eCosTools\Cygwin\opt\eCos-Repository\at91sam9x25\target

sam9x25ek.ecc
ECC File
672 KB

ecos_redboot_sam9x25.ecc
ECC File
649 KB

sam9x25ek_profinet.ecc
ECC File
595 KB

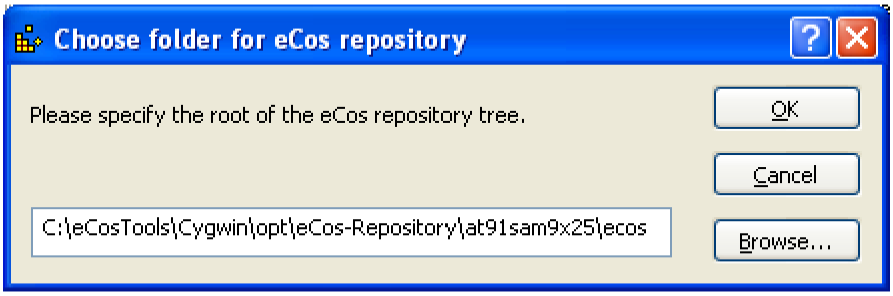
3. Make sure the **Build Tools Path** (Tools>Paths>Build Tools…) and **eCos Repository** (Build>Repository…the repository need contain ecos.db file and all related packages source folder) are configured correctly:



Build Tools Path

Enter the location of the arm-eabi build tools folder. You can type in a path or use the Browse button to navigate to a folder.

C:\eCosTools\Cygwin\gnutools

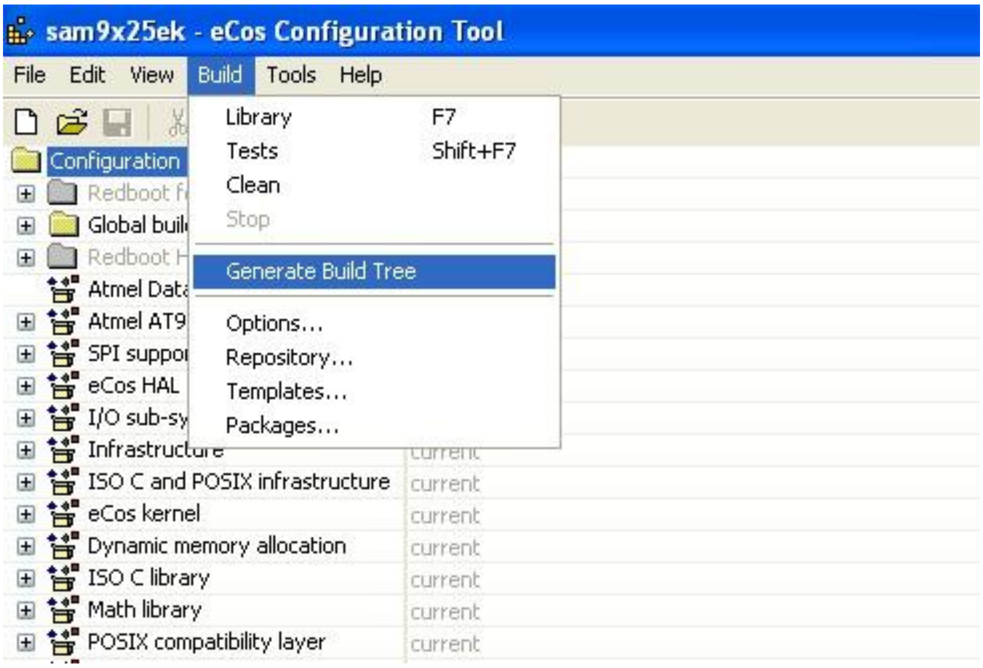


Choose folder for eCos repository

Please specify the root of the eCos repository tree.

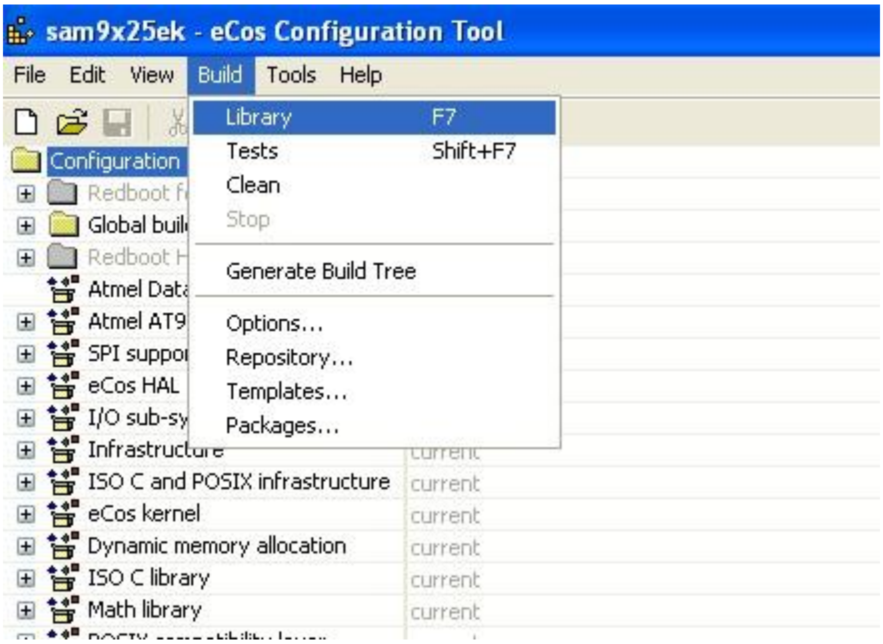
C:\eCosTools\Cygwin\opt\eCos-Repository\at91sam9x25\ecos

4. Run **Generate Build Tree** in menu **Build** of the eCos Configuration Tool:



sam9x25ek - eCos Configuration Tool

5. Run **Build>Library** (or press **F7**):

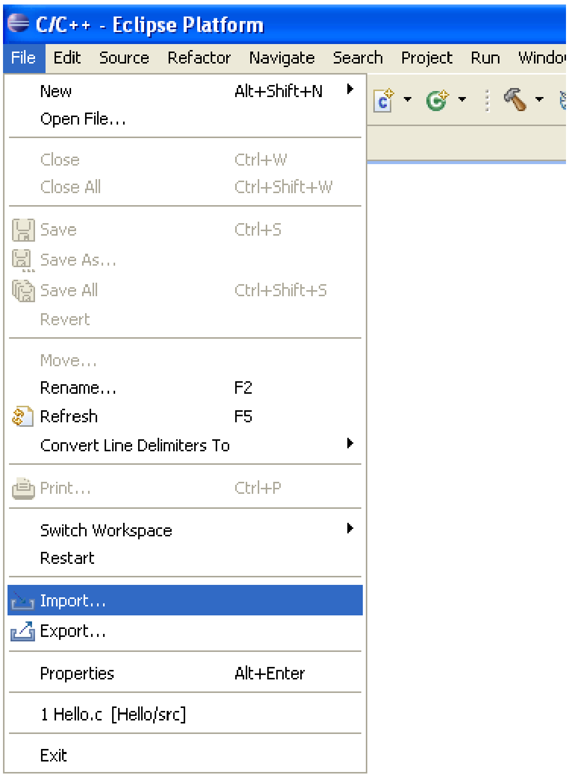


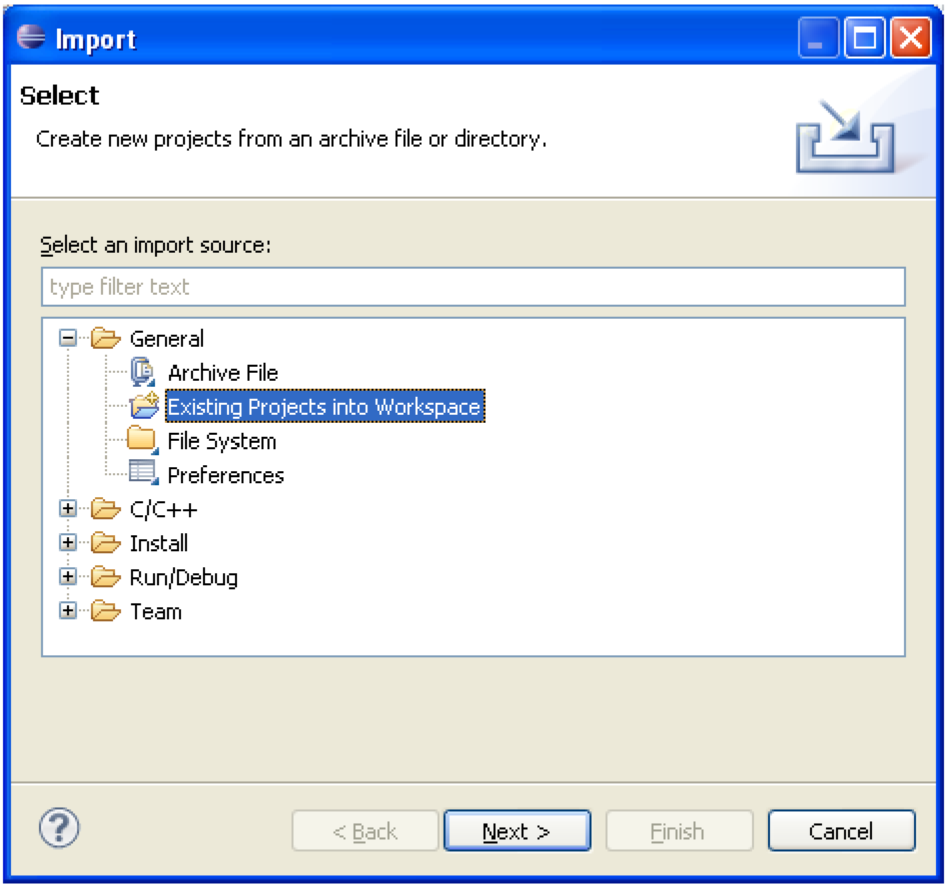
Please wait while the eCos Configuration Tool builds eCos Library. This will take some minutes to complete.

Working with eclipse

Importing an example

1. Open eclipse IDE.
2. Import example project (by default, the project files are located in the subfolders of **C:\eCosTools\Cygwin\opt\eCos-Examples**). To do this run **File>Import…**





Import

Select

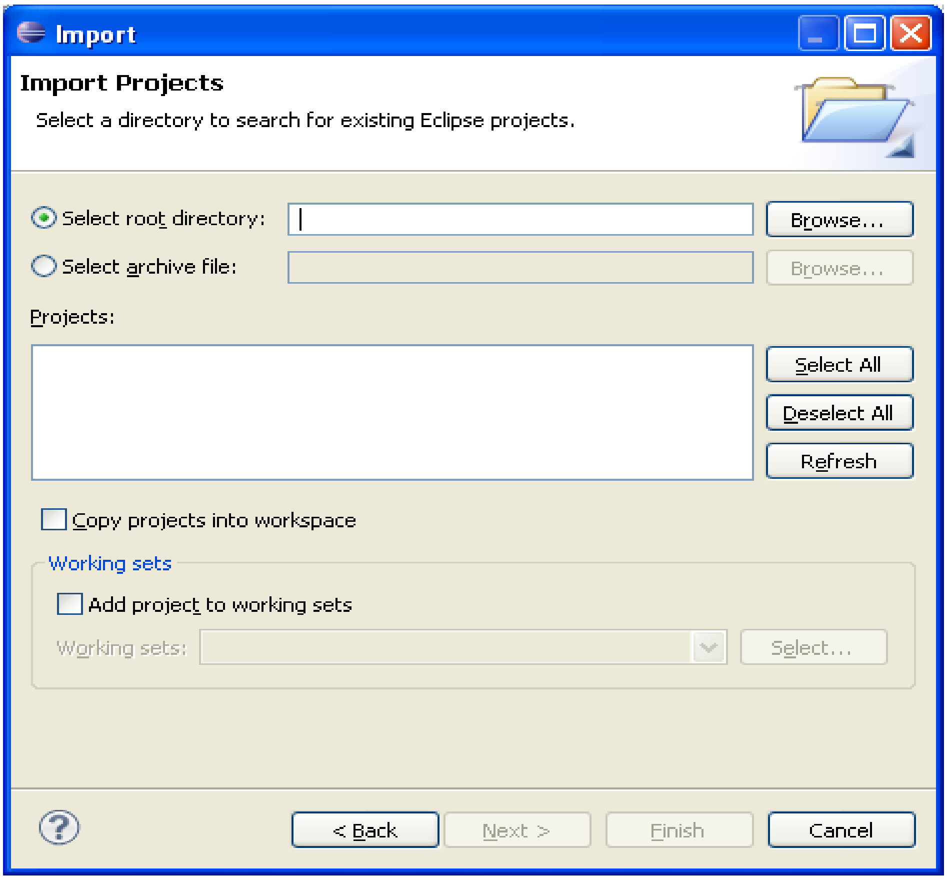
Create new projects from an archive file or directory.

Select an import source:

type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - File System
  - Preferences
- C/C++
- Install
- Run/Debug
- Team

Select **Existing Projects into Worspace** and click **Next**. The next diaglog will appear:



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory:

○ Select archive file:

Projects:

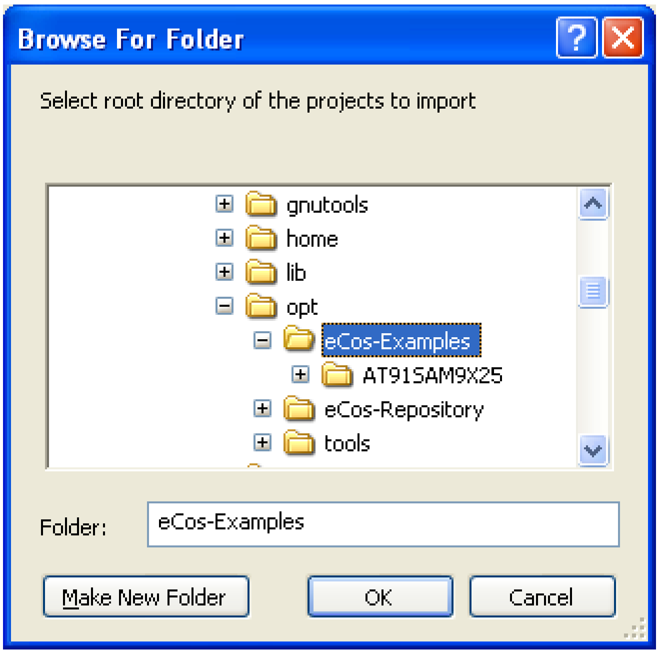
□ Copy projects into workspace

Working sets

□ Add project to working sets

Working sets:

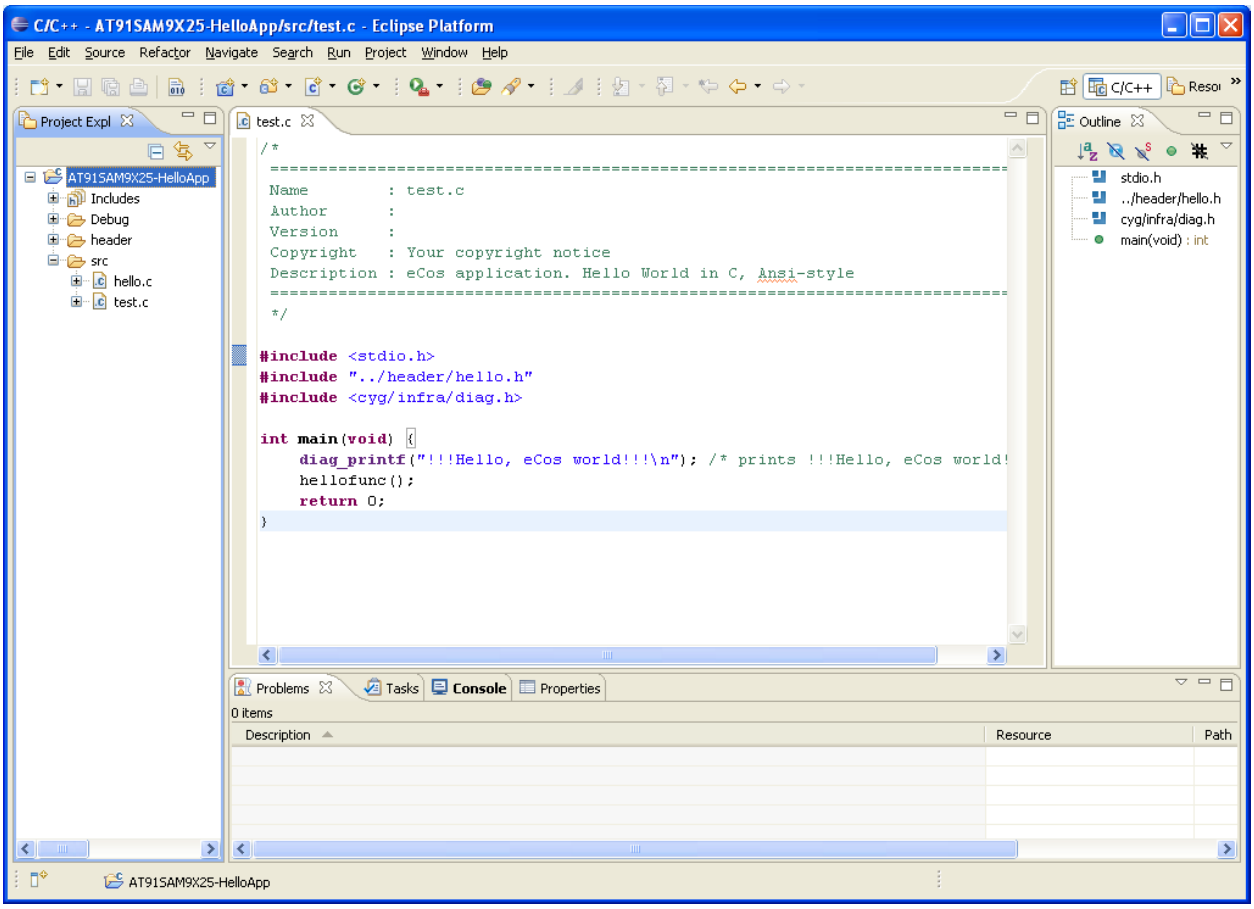
Click **Browse…** and set the path to **…\ Cygwin\opt\eCos-Examples\**



Select root directory of the projects to import

Folder: eCos-Examples

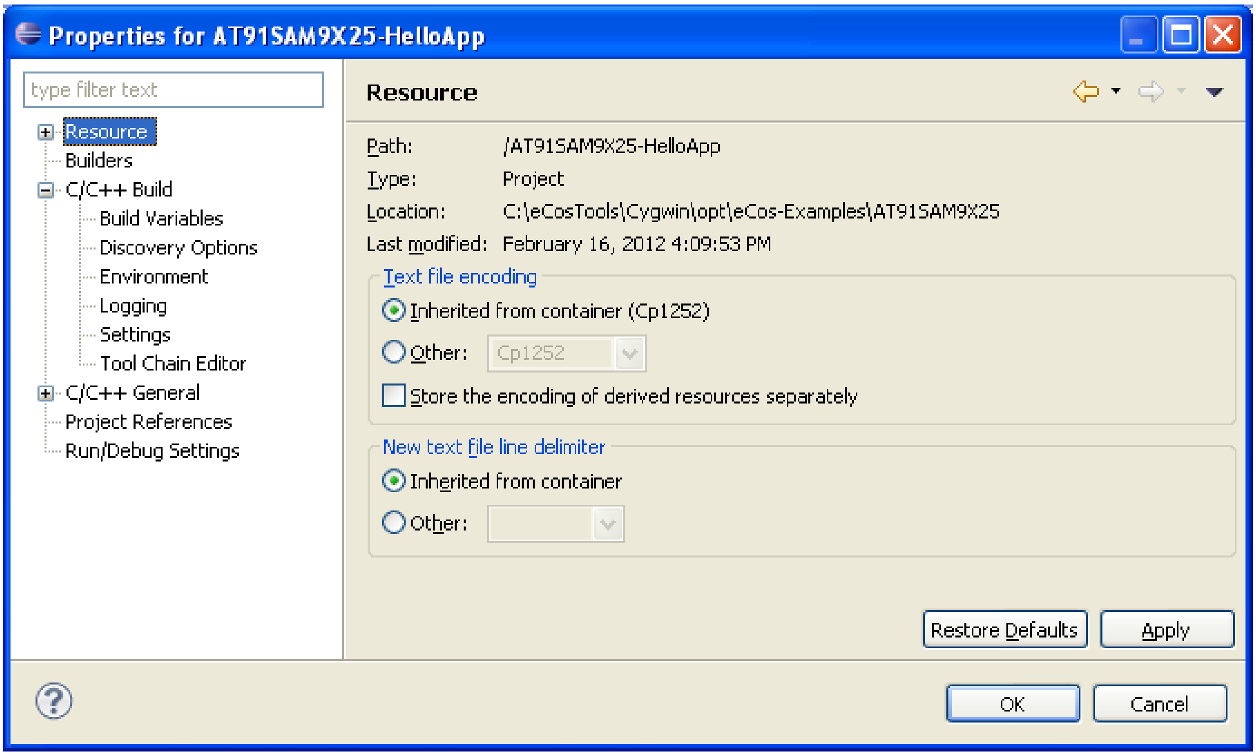
Click **OK** to continue. Then click **Finish** button to import the example projects.



```
/*
 ============================================================================
 Name        : test.c
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : eCos application. Hello World in C, Ansi-style
 ============================================================================
 */

#include <stdio.h>
#include "../header/hello.h"
#include <cyg/infra/diag.h>

int main(void) {
	diag_printf("!!!Hello, eCos world!!!\n"); /* prints !!!Hello, eCos world!
	hellofunc();
	return 0;
}
```

0 items

3. Check settings for the application. In this case we consider **the AT91SAM9X25-HelloApp** project. Select **AT91SAM9X25-HelloApp** in the **Project Explorer**, open **Project>Properties**, a dialog will appear to set up properties for current project:



Properties for AT91SAM9X25-HelloApp

Resource

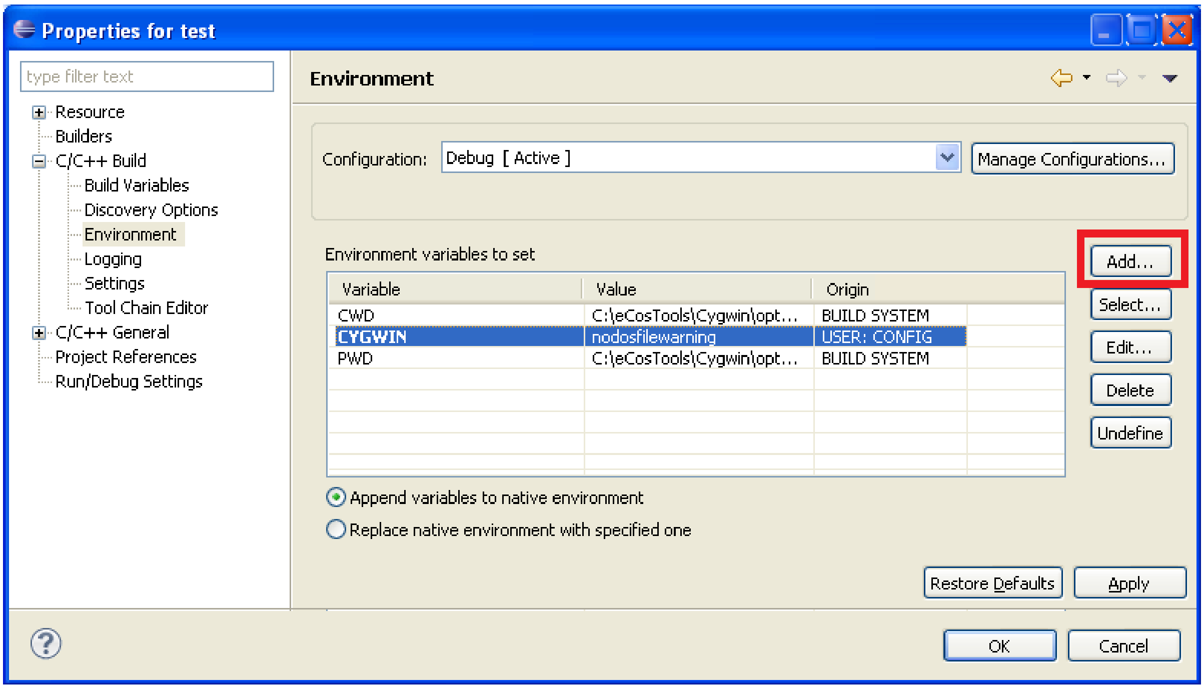
Path: /AT91SAM9X25-HelloApp
Type: Project
Location: C:\eCosTools\Cygwin\opt\eCos-Examples\AT91SAM9X25
Last modified: February 16, 2012 4:09:53 PM

Text file encoding
● Inherited from container (Cp1252)
○ Other: Cp1252
□ Store the encoding of derived resources separately

New text file line delimiter
● Inherited from container
○ Other:

In the sidebar of this dialog, open **"C/C++ Build > Build Variables"** and **“C/C++ Build > Environment”** to be sure the **ECOS_INSTDIR** variable doesn’t exist in the **Build Variables** or **Environment**. If it already exists, just skip this step. Otherwise you have to create it:

In the **“C/C++ Build > Environment”** dialog, click "Add" button to add a variable:



Properties for test

Environment

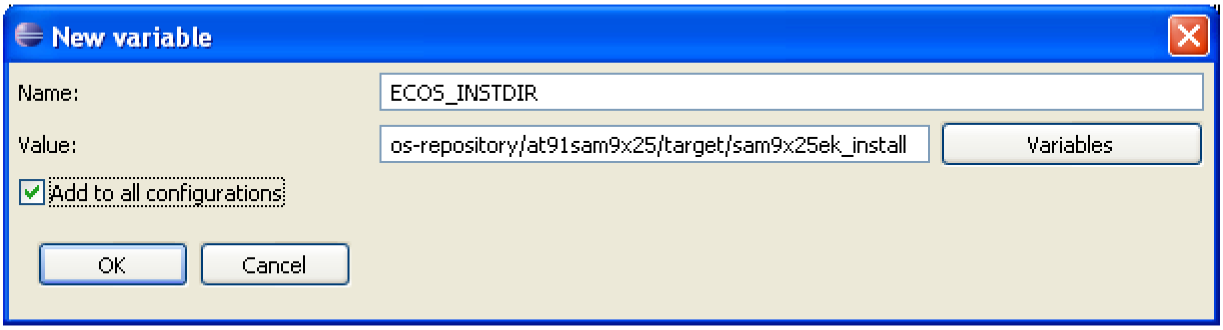
Configuration: Debug [Active]

Environment variables to set

Variable	Value	Origin
CWD	C:\eCosTools\Cygwin\opt...	BUILD SYSTEM
CYGWIN	nodosfilewarning	USER: CONFIG
PWD	C:\eCosTools\Cygwin\opt...	BUILD SYSTEM

● Append variables to native environment
○ Replace native environment with specified one

Create new variable with the name **ECOS_INSTDIR**, Value=**C:/eCosTools/ Cygwin/opt/eCos-Repository/at91sam9x25/target/sam9x25ek_install** (or other value if you use onother eCos installation directory). To avoid error please use back slash symbol (/) in the path. Set the Checkbox **“Add to all configurations”.**



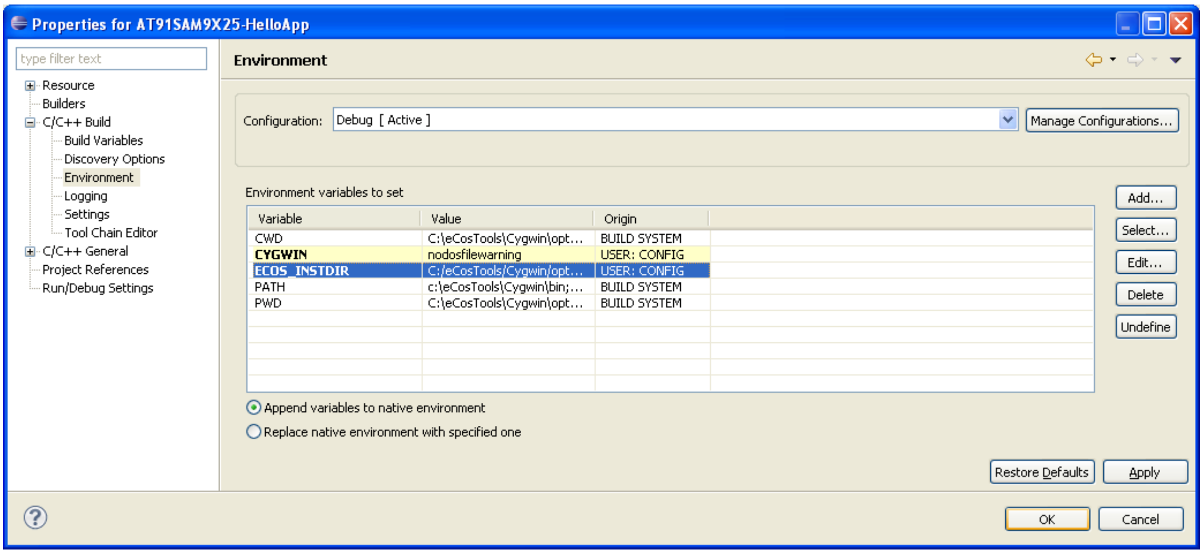
New variable

Name: ECOS_INSTDIR

Value: os-repository/at91sam9x25/target/sam9x25ek_install

☑ Add to all configurations

Click **OK** to add variable.



Properties for AT91SAM9X25-HelloApp

Environment

Configuration: Debug [Active]

Environment variables to set

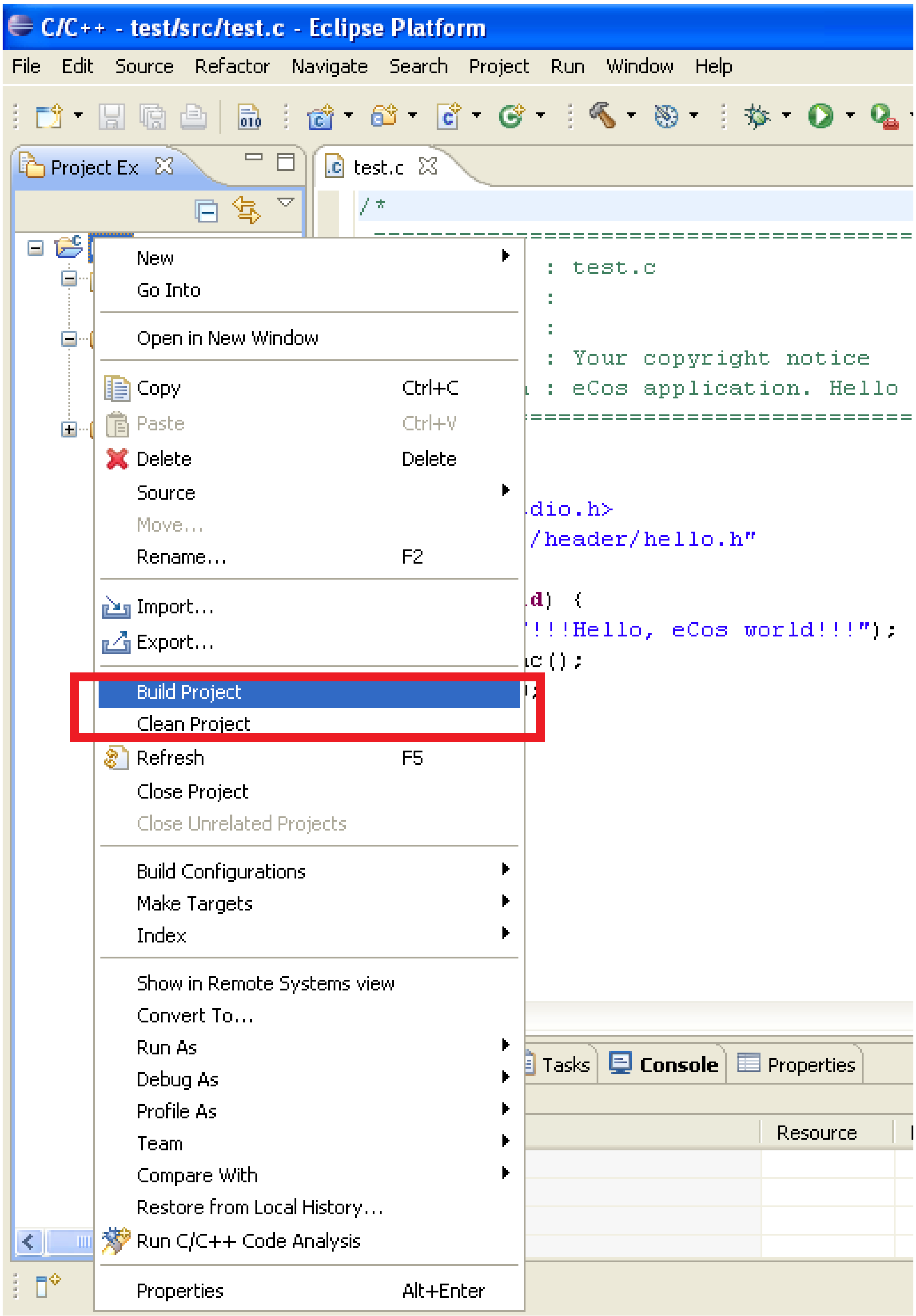
Variable	Value	Origin
CWD	C:\eCosTools\Cygwin\opt...	BUILD SYSTEM
CYGWIN	nodosfilewarning	USER: CONFIG
ECOS_INSTDIR	C:/eCosTools/Cygwin/opt...	USER: CONFIG
PATH	c:\eCosTools\Cygwin\bin;...	BUILD SYSTEM
PWD	C:\eCosTools\Cygwin\opt...	BUILD SYSTEM

● Append variables to native environment
○ Replace native environment with specified one

Now click **Apply** and **OK** to finish.

4. Build project:

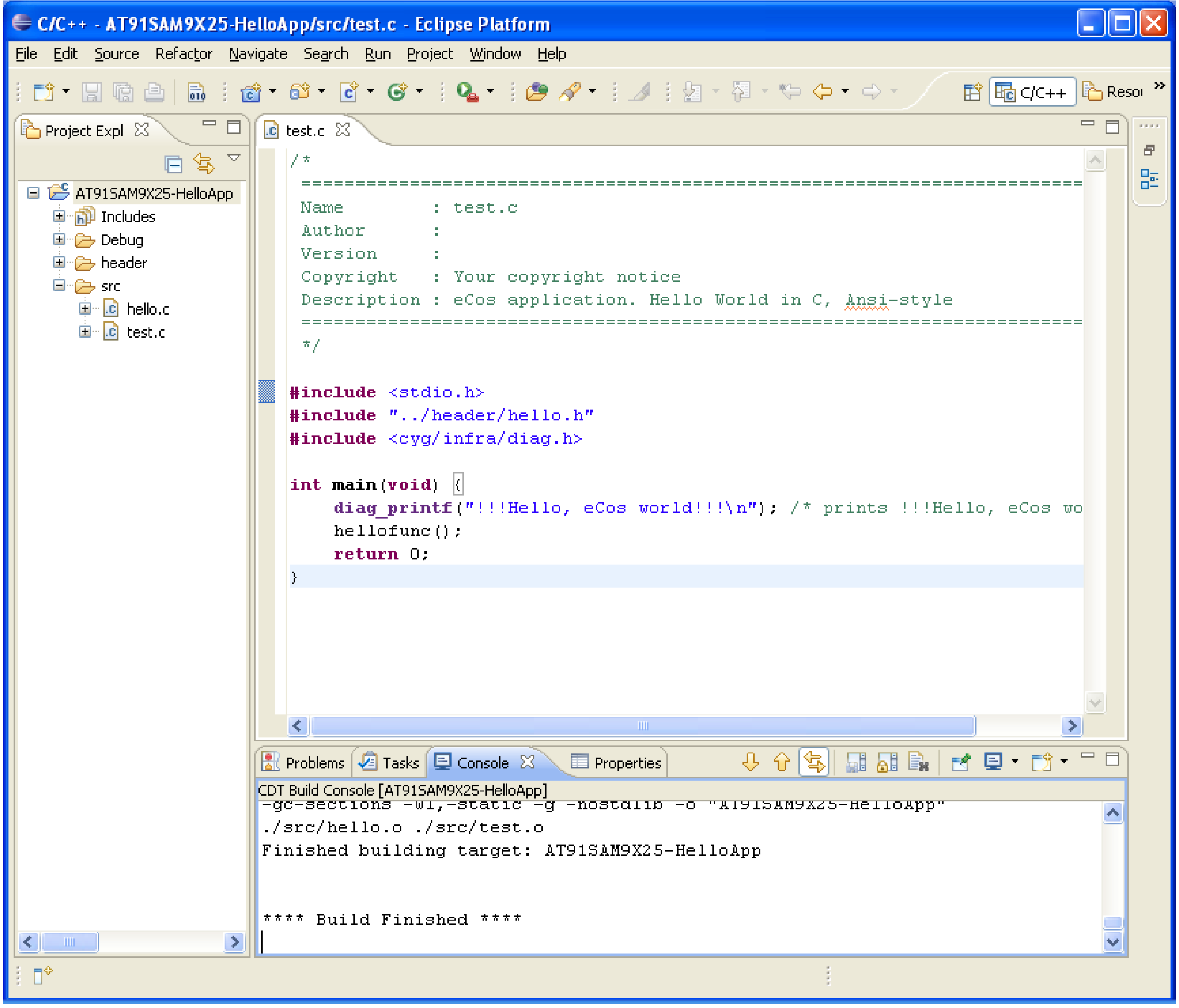
To **build** or **clean** project, use the corresponding command in the project menu or in the popup menu as shown in the next figure:



```
/*
 ========================================
 : test.c
 :
 :
 : Your copyright notice
 : eCos application. Hello
 ============================
dio.h>
/header/hello.h"
d) {
"!!!Hello, eCos world!!!");
c();
```

Remember: the project can be built only after building the **eCos library**. To build the eCos library, follow the steps in part I.

After building, the **Project Explorer** will look like:

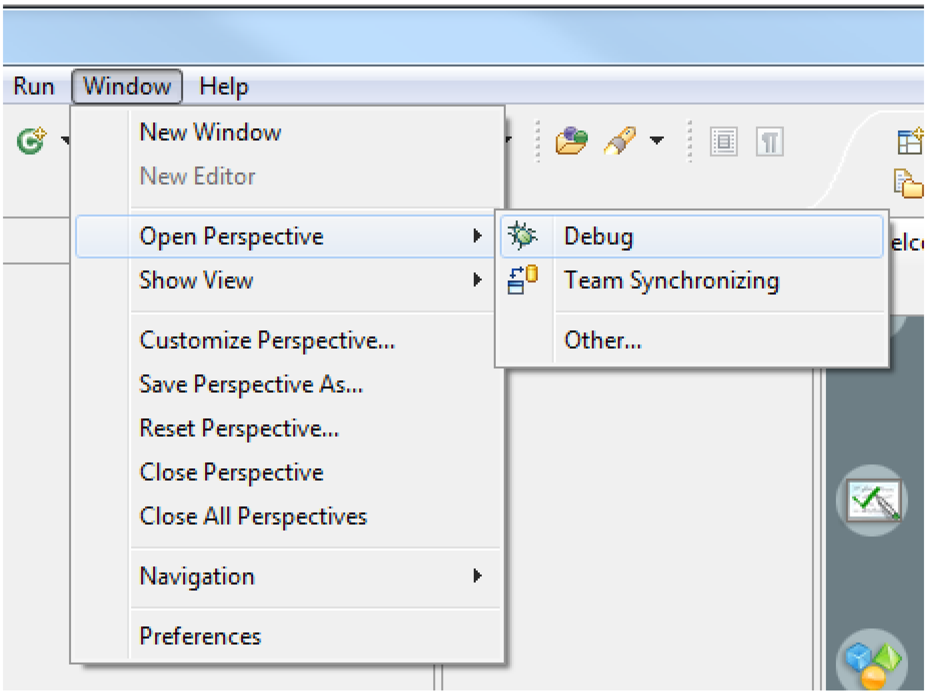

C/C++ - AT91SAM9X25-HelloApp/src/test.c - Eclipse Platform
File Edit Source Refactor Navigate Search Run Project Window Help
C/C++
Resoı
Project Expl
AT91SAM9X25-HelloApp
Includes
Debug
header
src
hello.c
test.c
test.c
Problems
Tasks
Console
Properties
CDT Build Console [AT91SAM9X25-HelloApp]

```
/*
 ==========================================================================
 Name        : test.c
 Author      :
 Version     :
 Copyright   : Your copyright notice
 Description : eCos application. Hello World in C, Ansi-style
 ==========================================================================
 */

#include <stdio.h>
#include "../header/hello.h"
#include <cyg/infra/diag.h>

int main(void) {
    diag_printf("!!!Hello, eCos world!!!\n"); /* prints !!!Hello, eCos wo
    hellofunc();
    return 0;
}
```

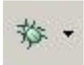
```
-gc-sections -Wl,-static -g -nostdlib -o "AT91SAM9X25-HelloApp"
./src/hello.o ./src/test.o
Finished building target: AT91SAM9X25-HelloApp


**** Build Finished ****
```

5. Run/debug project:

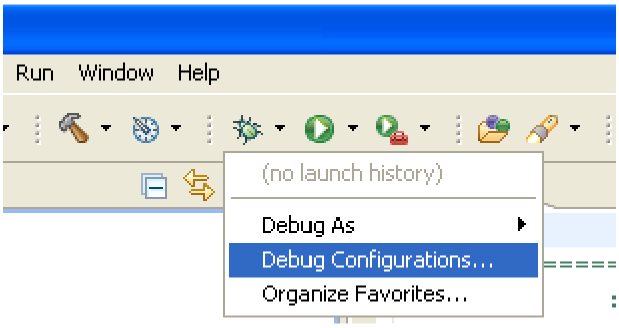
To run/debug **AT91SAM9X25-HelloApp** project, first need to **Configure the debugger.** For the debug process we need to open the **"Debug Perspective"** by using **"Window > Open Perspective > Debug":**



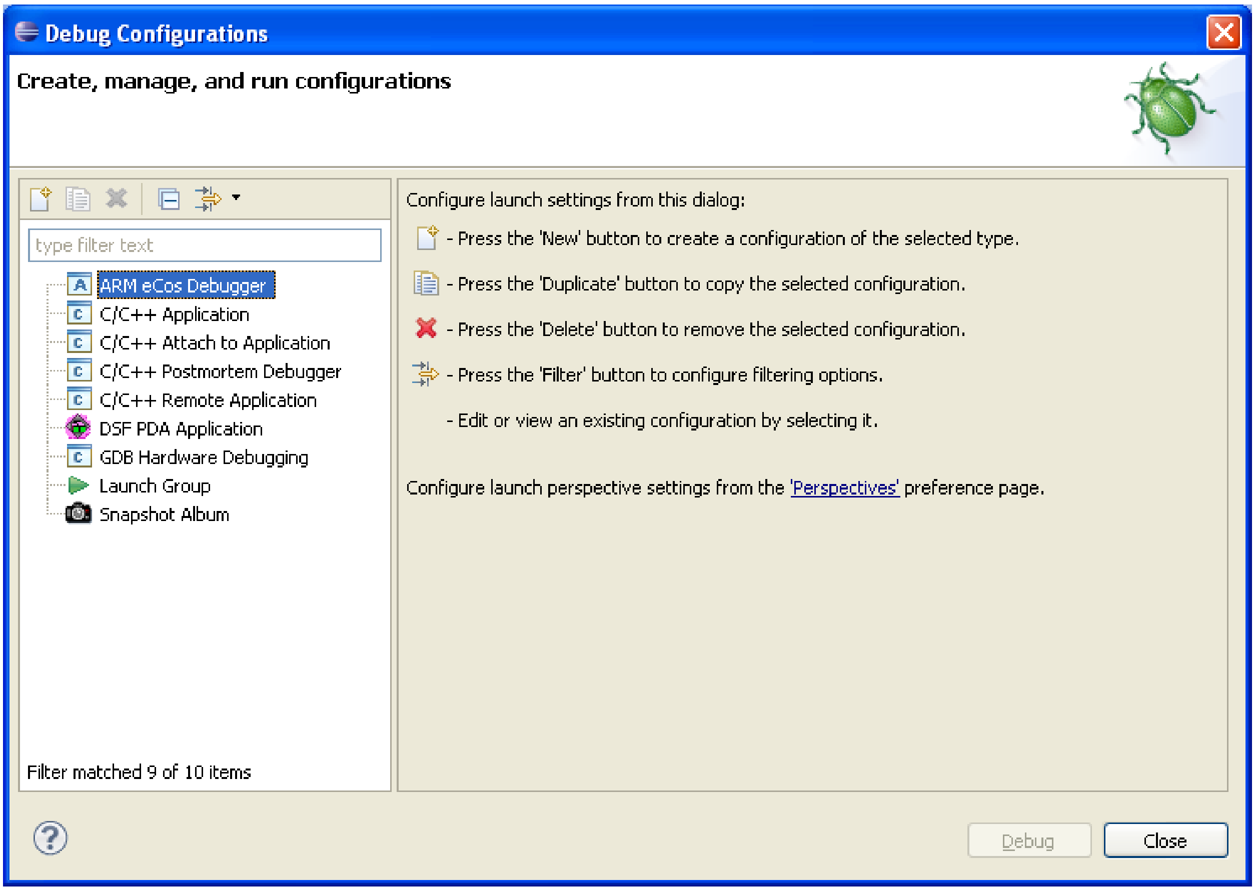
Now we need to configure the debugger. The configuration can be reached by clicking on the down arrow at the **"insect"** button. That brings up the pull-down menu:



And select **"Debug Configuration..."**:



The next window will look like:

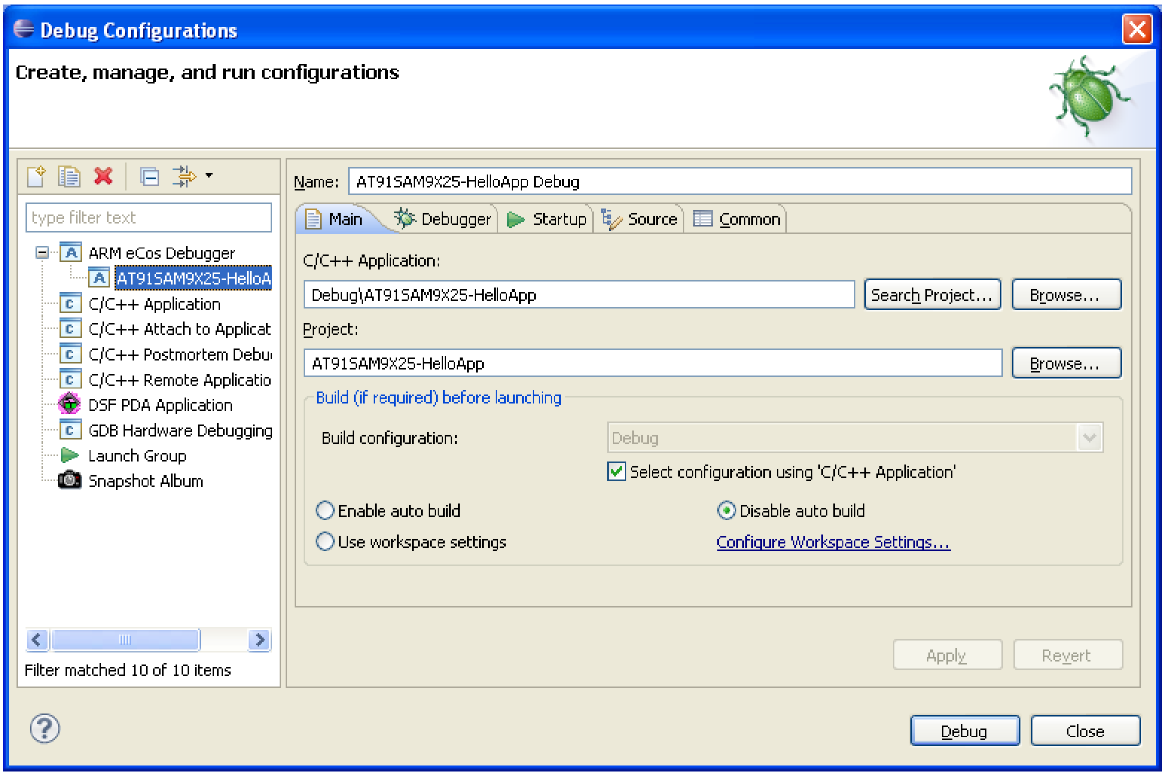


Configure launch settings from this dialog:

- Press the 'New' button to create a configuration of the selected type.
- Press the 'Duplicate' button to copy the selected configuration.
- Press the 'Delete' button to remove the selected configuration.
- Press the 'Filter' button to configure filtering options.
- Edit or view an existing configuration by selecting it.

Configure launch perspective settings from the 'Perspectives' preference page.

Double click on "**ARM eCos Debugger**" to create a configuration of selected type. On the field **“Name”** keep ***“AT91SAM9X25-HelloApp Debug”***. On the **“C/C++ Application”** field keep ***“Debug\AT91SAM9X25-HelloApp”.*** On the **“Project”** field keep ***“AT91SAM9X25-HelloApp”*** to debug the test application**.** Select ***“Disable auto build”*** to improve the launch performance. The **Debug Configuration** dialog will look like:



Name: AT91SAM9X25-HelloApp Debug

C/C++ Application: Debug\AT91SAM9X25-HelloApp

Project: AT91SAM9X25-HelloApp

Build (if required) before launching

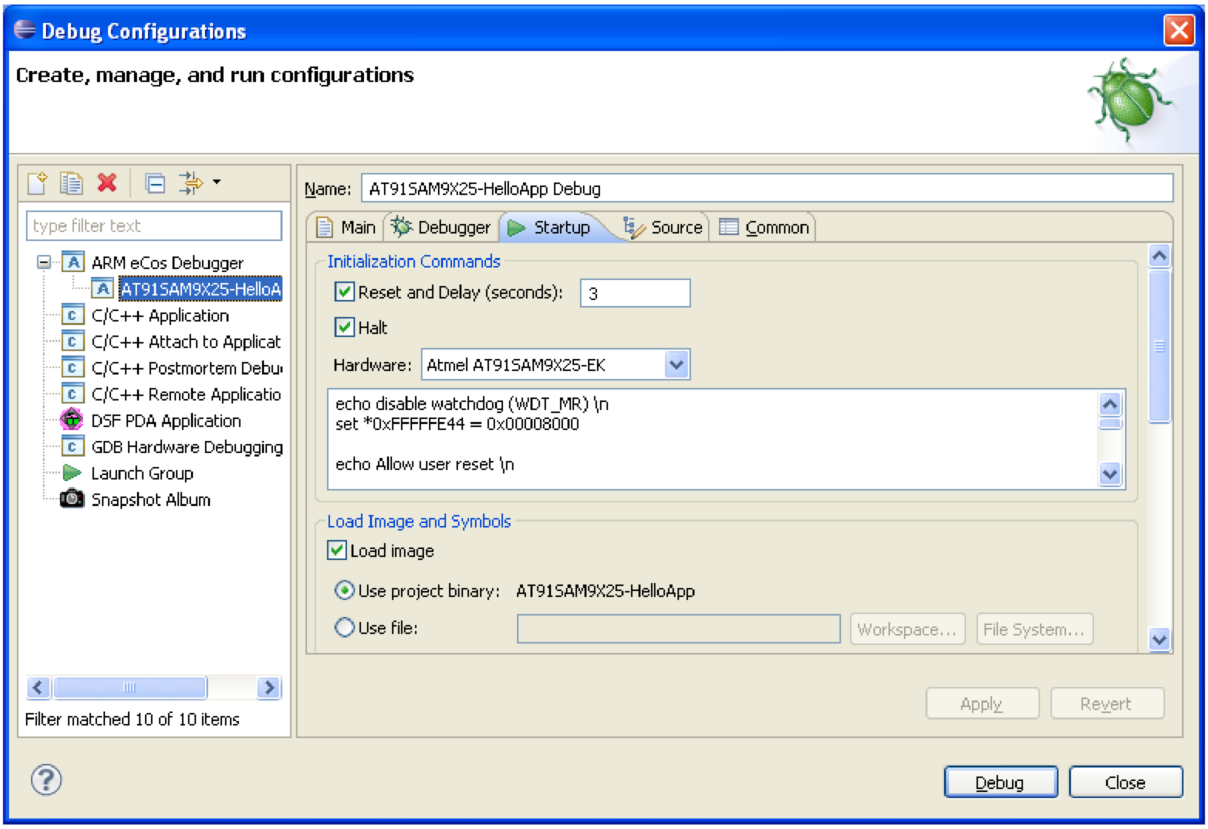
Build configuration: Debug

☑ Select configuration using 'C/C++ Application'

○ Enable auto build ● Disable auto build

○ Use workspace settings

Now move to the **"Startup"** tab. Select **Atmel AT91SAM9X25-EK** in **Hardware** combo box. The InitCommands Text fields will be filled by the startup scripts for **Atmel AT91SAM9X25-EK** automatically. You can edit the scipts if you want.



Name: AT91SAM9X25-HelloApp Debug

Initialization Commands

☑ Reset and Delay (seconds): 3

☑ Halt

Hardware: Atmel AT91SAM9X25-EK

```
echo disable watchdog (WDT_MR) \n
set *0xFFFFFE44 = 0x00008000

echo Allow user reset \n
```

Load Image and Symbols

☑ Load image

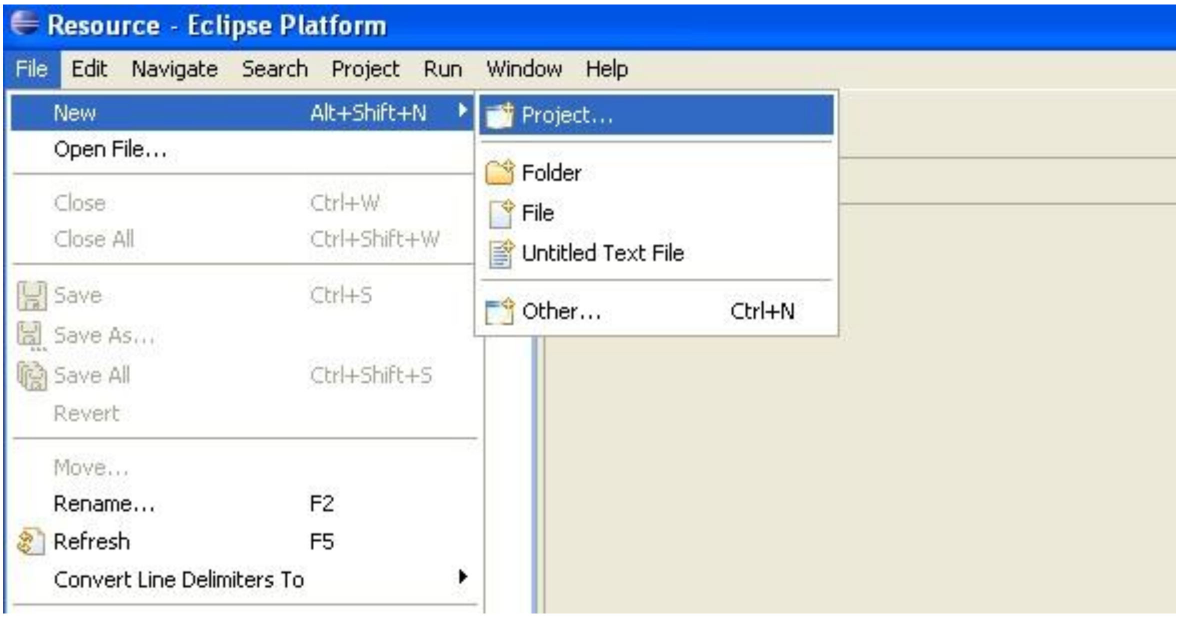
● Use project binary: AT91SAM9X25-HelloApp

○ Use file:

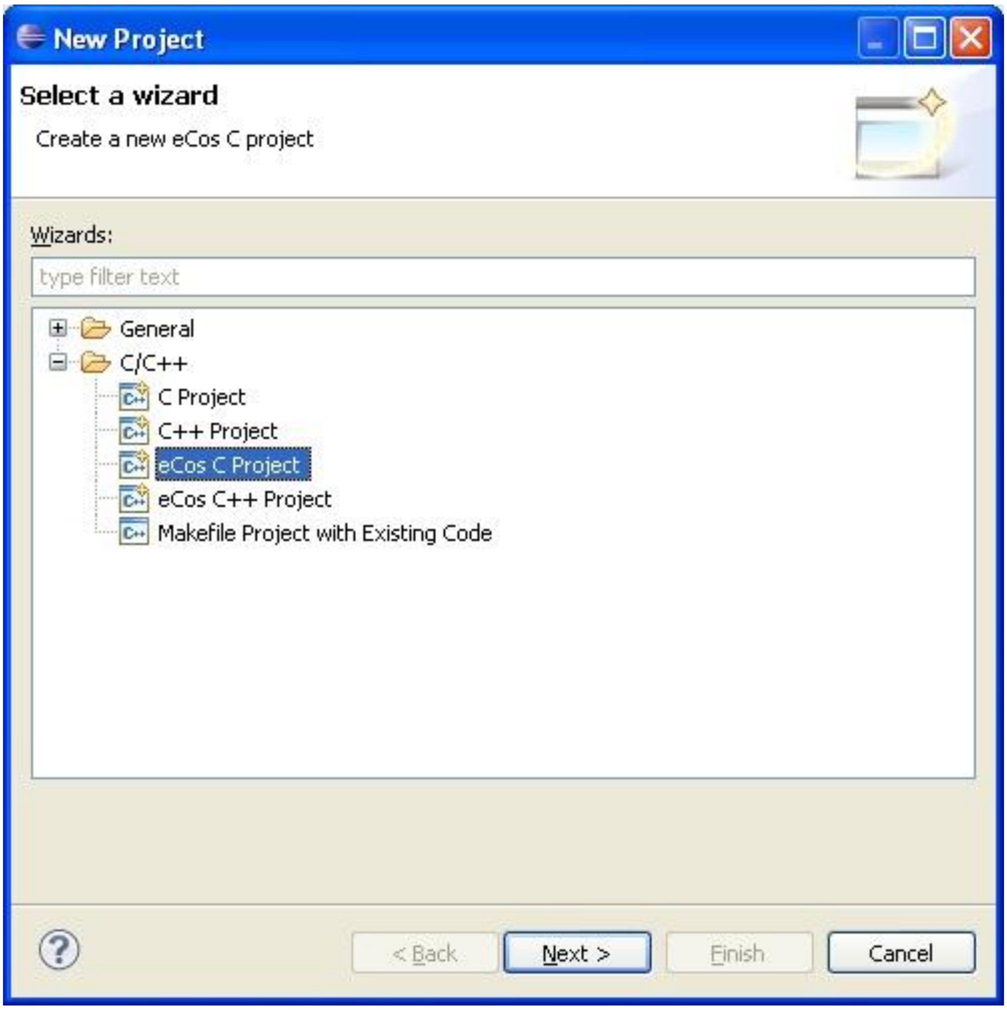
Click **Apply** to save settings and click **Debug** to run the debugger.

Creating a new application

First choose **File>New> Project…** as shown in the picture:



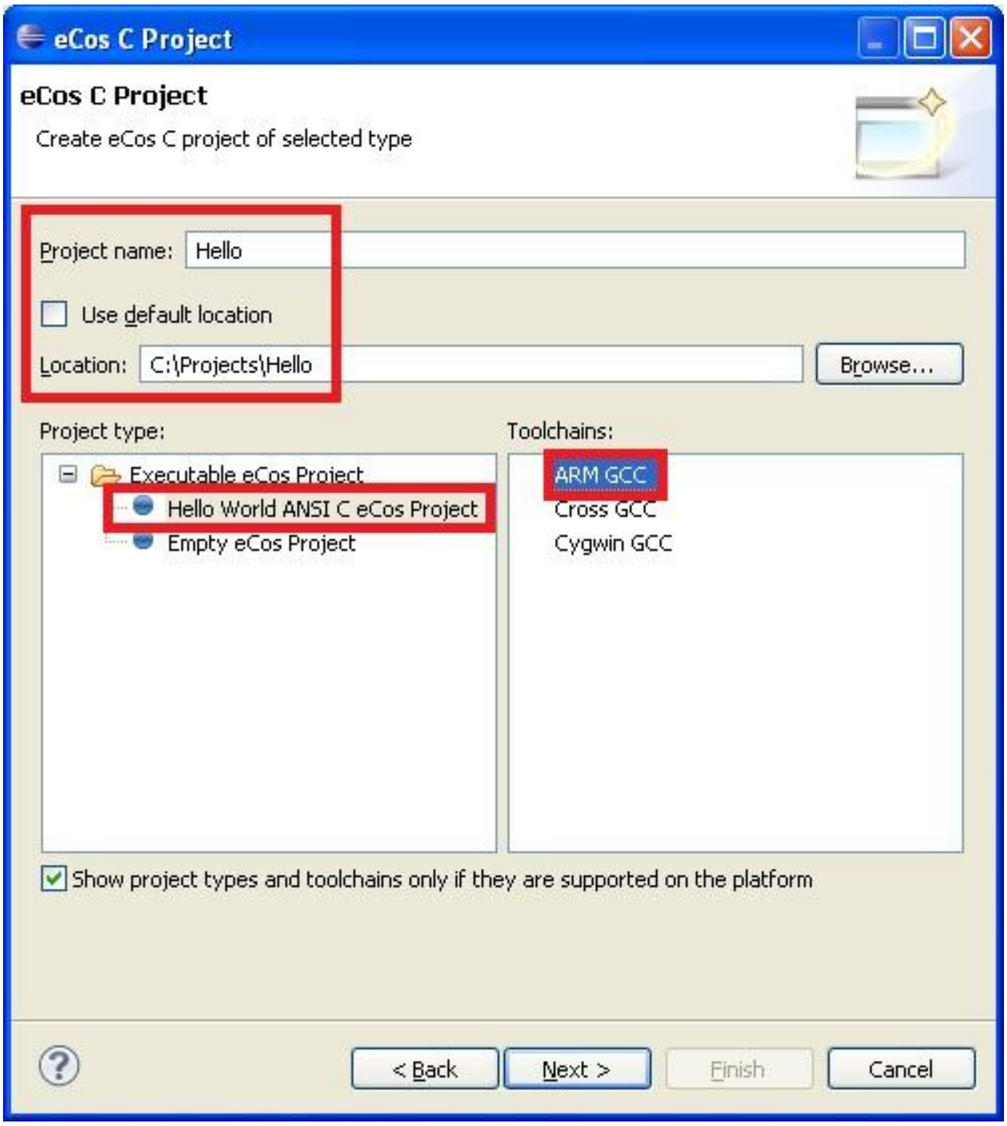
Choose **eCos C Project** in **New Project** wizard:



Select a wizard

Create a new eCos C project

Choose **Hello World ANSI C eCos Project** in the **Project type**. The toolchain we use is **ARM GCC**. Type the name of the project to the **Project name** field and select a location for the project:



eCos C Project

Create eCos C project of selected type

Project name: Hello

□ Use default location

Location: C:\Projects\Hello

Project type:

- Executable eCos Project
  - Hello World ANSI C eCos Project
  - Empty eCos Project

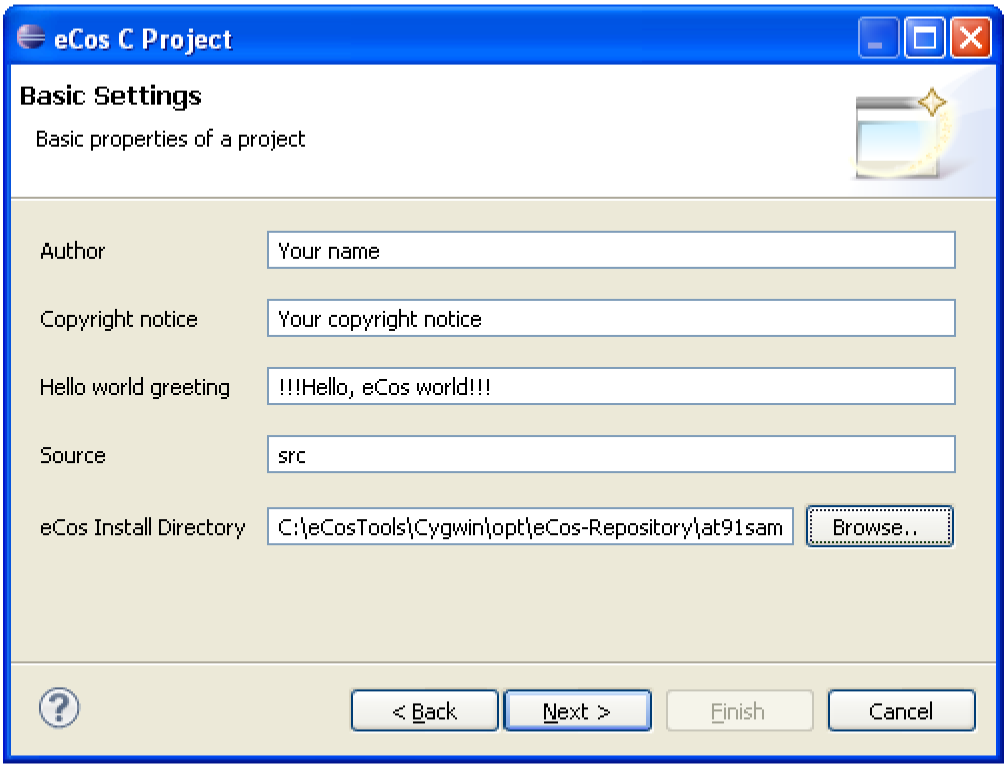
Toolchains:

- ARM GCC
- Cross GCC
- Cygwin GCC

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

Click **Next.**

In **Basic Settings** wizard, click on button **Browse** to find the eCos installation directory:



eCos C Project

Basic Settings

Basic properties of a project

Author: Your name

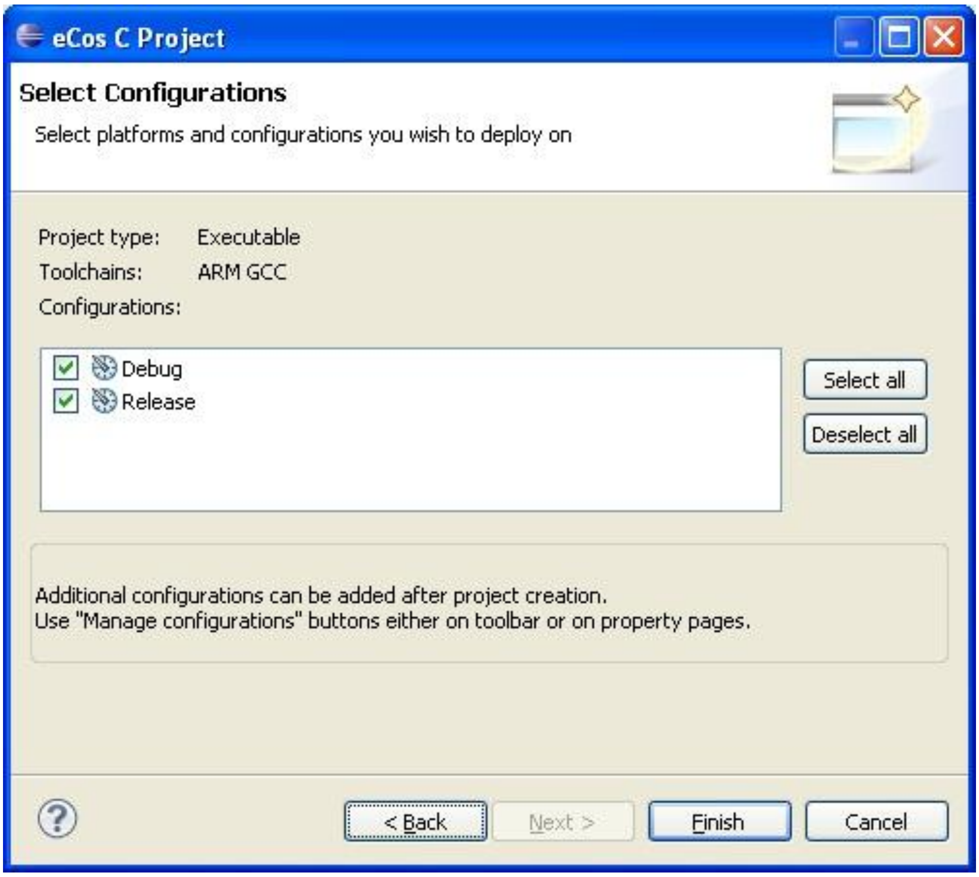
Copyright notice: Your copyright notice

Hello world greeting: !!!Hello, eCos world!!!

Source: src

eCos Install Directory: C:\eCosTools\Cygwin\opt\eCos-Repository\at91sam

Click **Next** then **Finish** to complete creating a new project. Project can be built and debugged completely like in part A.



eCos C Project

Select Configurations

Select platforms and configurations you wish to deploy on

Project type: Executable

Toolchains: ARM GCC

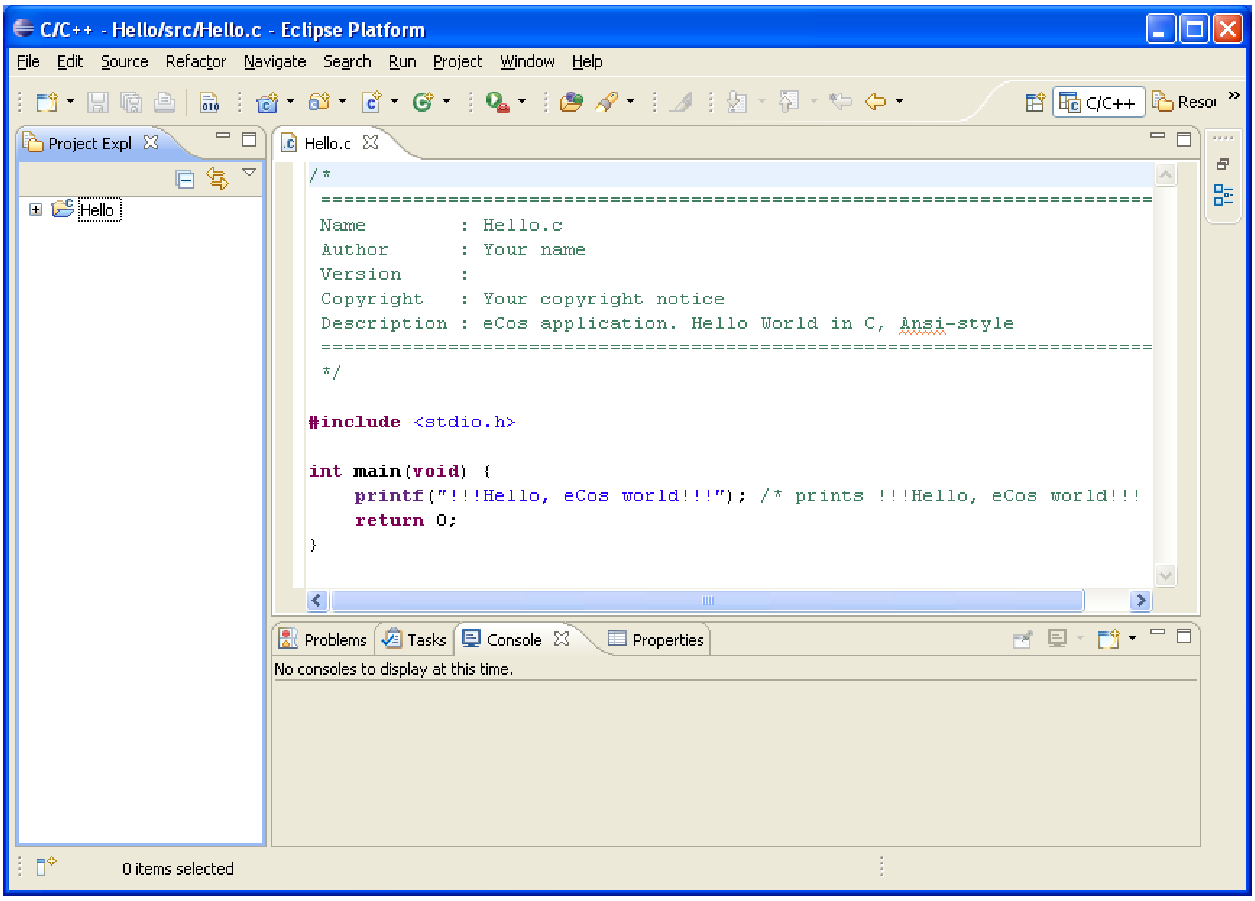
Configurations:

☑ Debug

☑ Release

Additional configurations can be added after project creation.
Use "Manage configurations" buttons either on toolbar or on property pages.

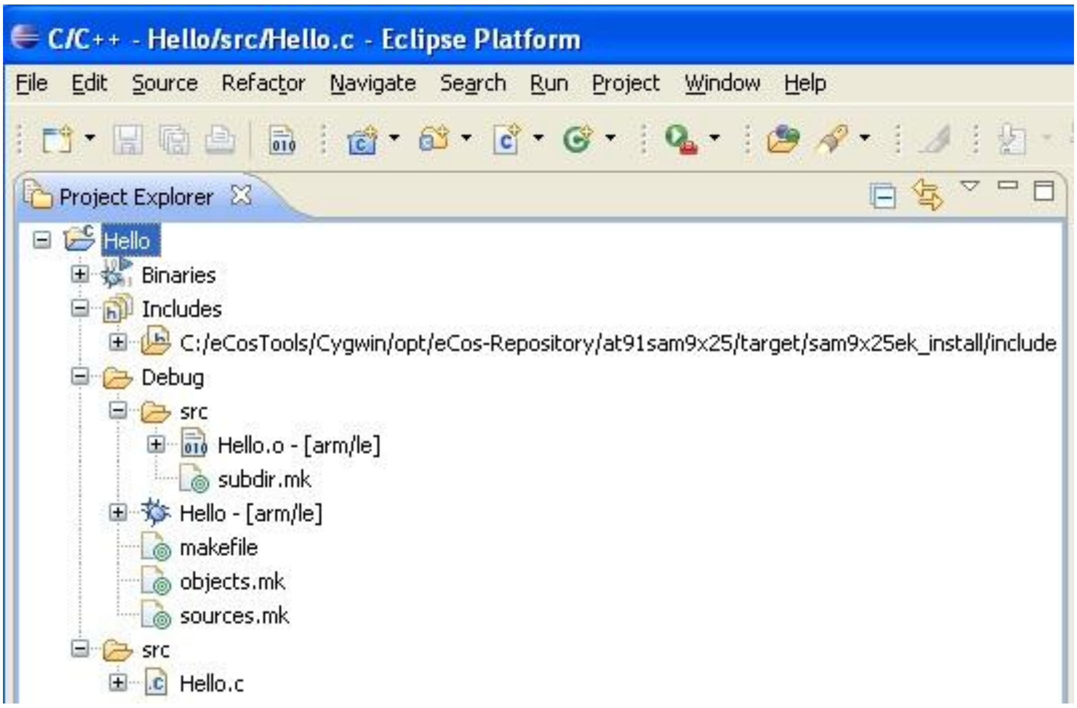
New eCos C project:


C/C++ - Hello/src/Hello.c - Eclipse Platform
File Edit Source Refactor Navigate Search Run Project Window Help
C/C++
Resou
Project Expl
Hello
Hello.c
Problems
Tasks
Console
Properties

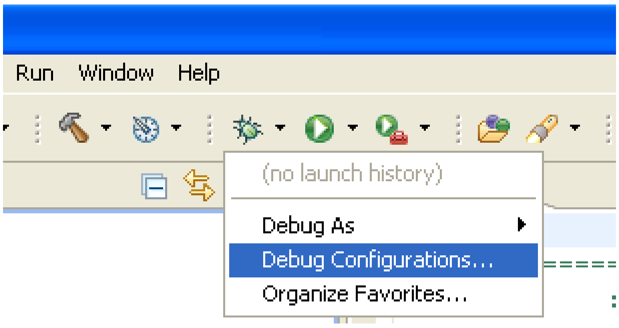
```
/*
 ============================================================================
 Name        : Hello.c
 Author      : Your name
 Version     :
 Copyright   : Your copyright notice
 Description : eCos application. Hello World in C, Ansi-style
 ============================================================================
 */

#include <stdio.h>

int main(void) {
    printf("!!!Hello, eCos world!!!"); /* prints !!!Hello, eCos world!!!
    return 0;
}
```

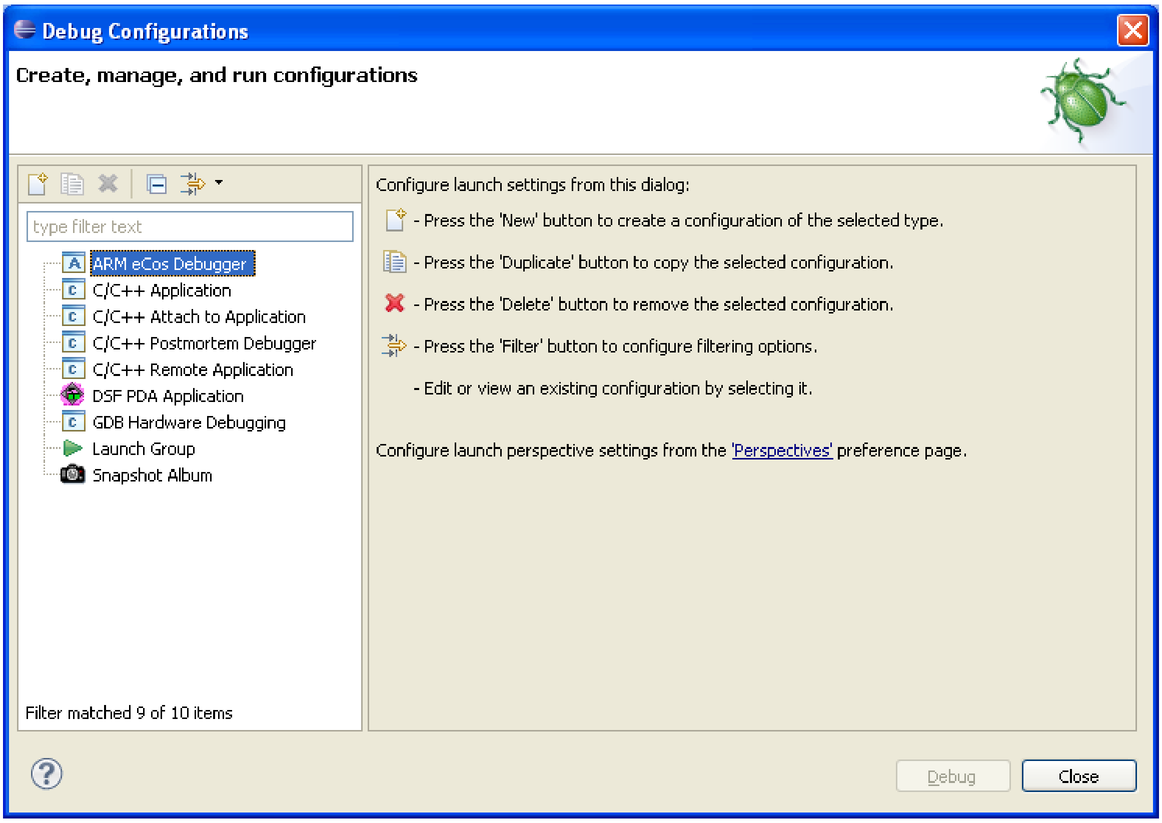
No consoles to display at this time.

0 items selected


C/C++ - Hello/src/Hello.c - Eclipse Platform
File Edit Source Refactor Navigate Search Run Project Window Help
Project Explorer

```
Hello
    Binaries
    Includes
        C:/eCosTools/Cygwin/opt/eCos-Repository/at91sam9x25/target/sam9x25ek_install/include
    Debug
        src
            Hello.o - [arm/le]
            subdir.mk
        Hello - [arm/le]
        makefile
        objects.mk
        sources.mk
    src
        Hello.c
```





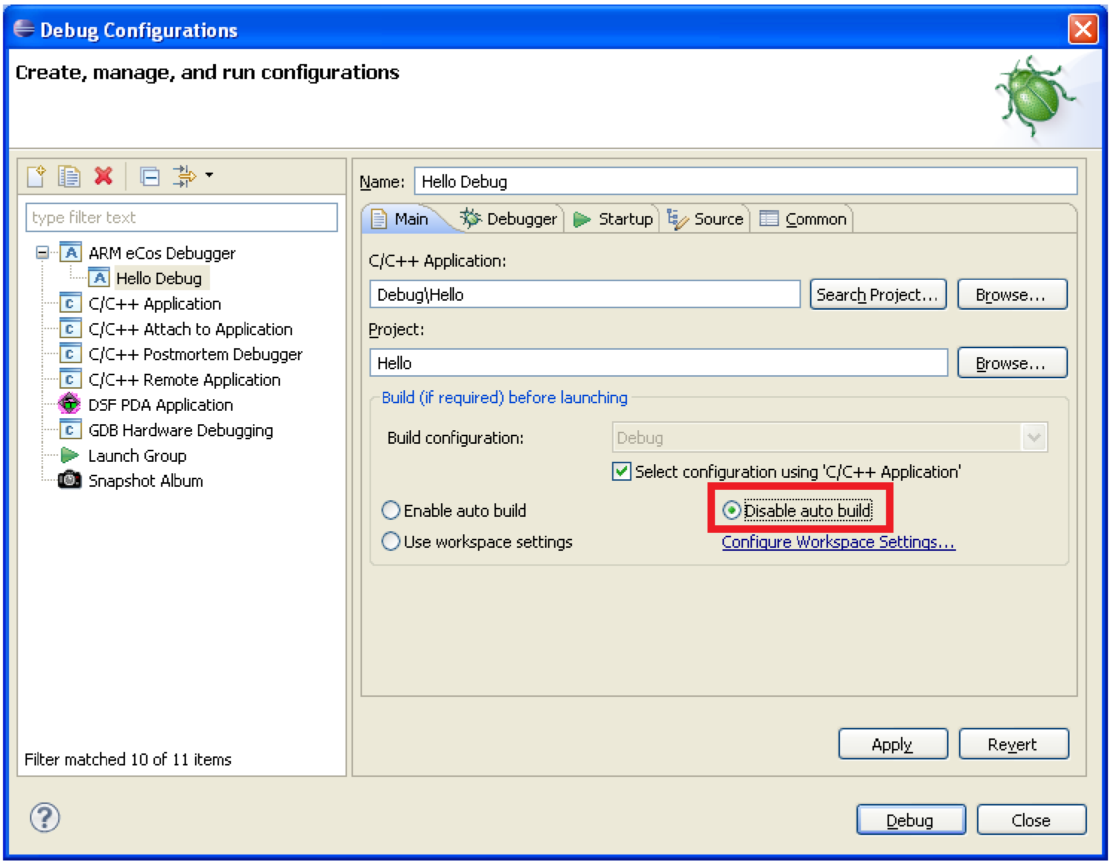
Create, manage, and run configurations

Configure launch settings from this dialog:

- Press the 'New' button to create a configuration of the selected type.
- Press the 'Duplicate' button to copy the selected configuration.
- Press the 'Delete' button to remove the selected configuration.
- Press the 'Filter' button to configure filtering options.
- Edit or view an existing configuration by selecting it.

Configure launch perspective settings from the 'Perspectives' preference page.

Filter matched 9 of 10 items



Debug Configurations

Create, manage, and run configurations

- ARM eCos Debugger
  - Hello Debug
- C/C++ Application
- C/C++ Attach to Application
- C/C++ Postmortem Debugger
- C/C++ Remote Application
- DSF PDA Application
- GDB Hardware Debugging
- Launch Group
- Snapshot Album

Filter matched 10 of 11 items

Name: Hello Debug

C/C++ Application: Debug\Hello

Project: Hello

Build (if required) before launching

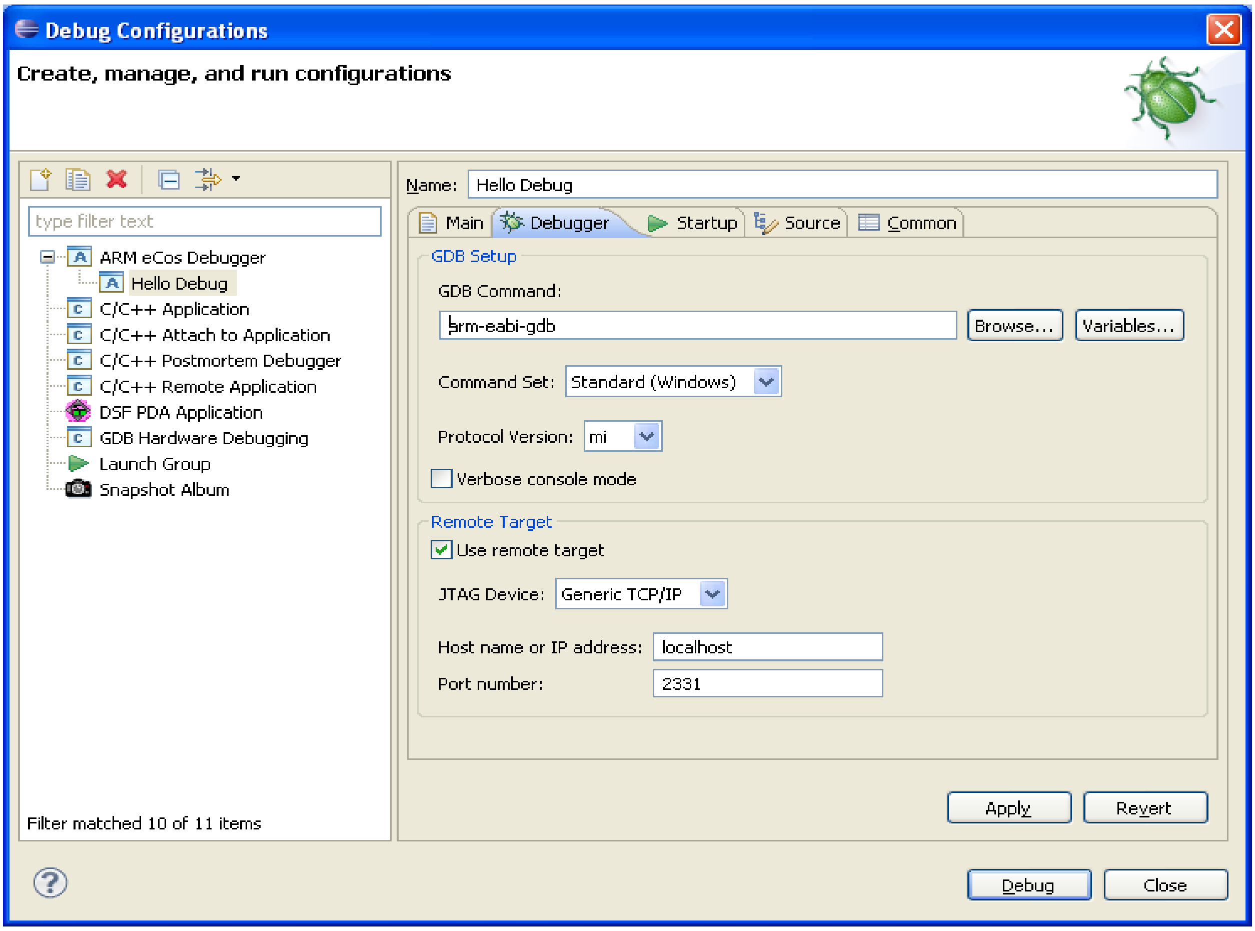
Build configuration: Debug

☑ Select configuration using 'C/C++ Application'

○ Enable auto build ● Disable auto build

○ Use workspace settings

Configure Workspace Settings...



Debug Configurations

Create, manage, and run configurations

- ARM eCos Debugger
  - Hello Debug
- C/C++ Application
- C/C++ Attach to Application
- C/C++ Postmortem Debugger
- C/C++ Remote Application
- DSF PDA Application
- GDB Hardware Debugging
- Launch Group
- Snapshot Album

Filter matched 10 of 11 items

Name: Hello Debug

GDB Setup

GDB Command: arm-eabi-gdb

Command Set: Standard (Windows)

Protocol Version: mi

□ Verbose console mode

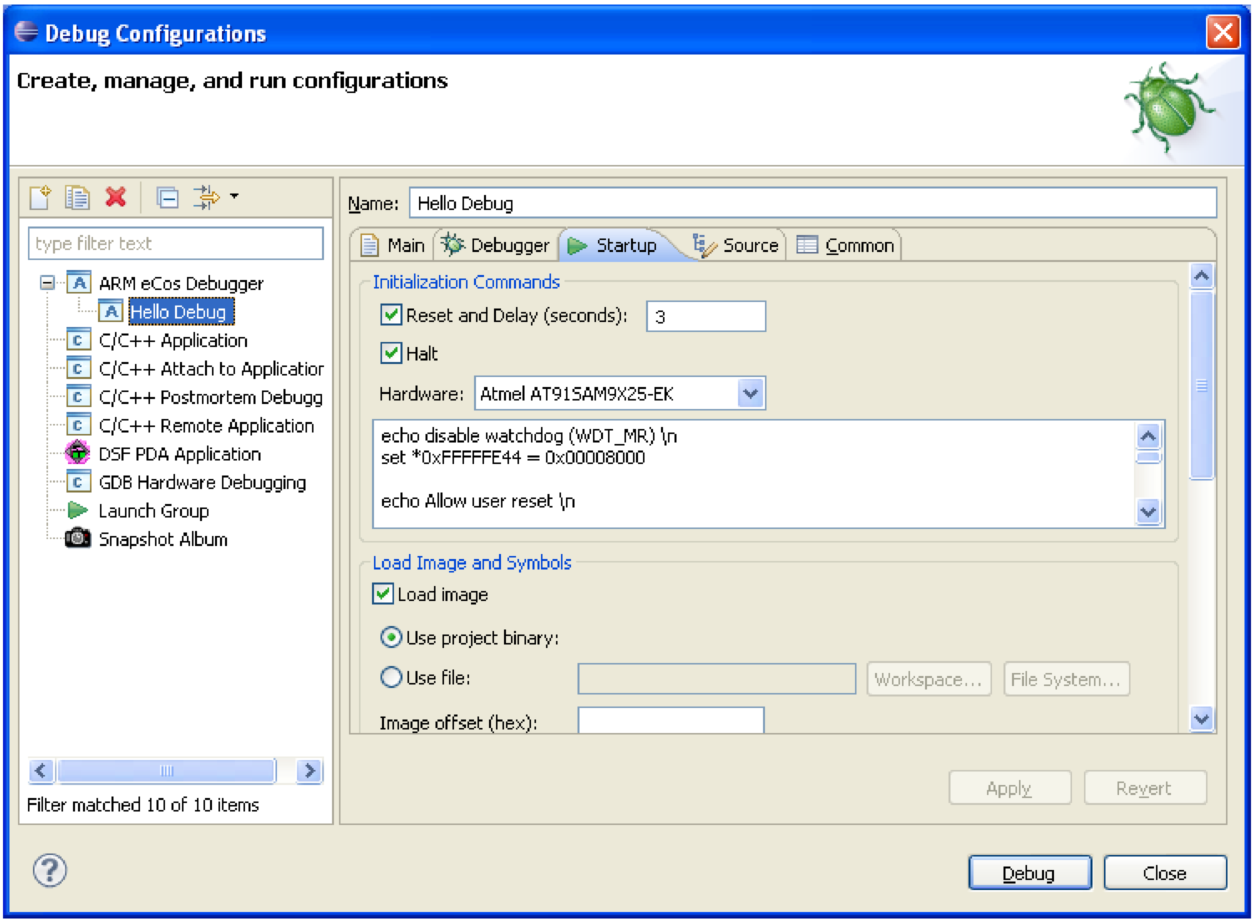
Remote Target

☑ Use remote target

JTAG Device: Generic TCP/IP

Host name or IP address: localhost

Port number: 2331



Debug Configurations

Create, manage, and run configurations

Name: Hello Debug

Initialization Commands

☑ Reset and Delay (seconds): 3

☑ Halt

Hardware: Atmel AT91SAM9X25-EK

```
echo disable watchdog (WDT_MR) \n
set *0xFFFFFE44 = 0x00008000

echo Allow user reset \n
```

Load Image and Symbols

☑ Load image

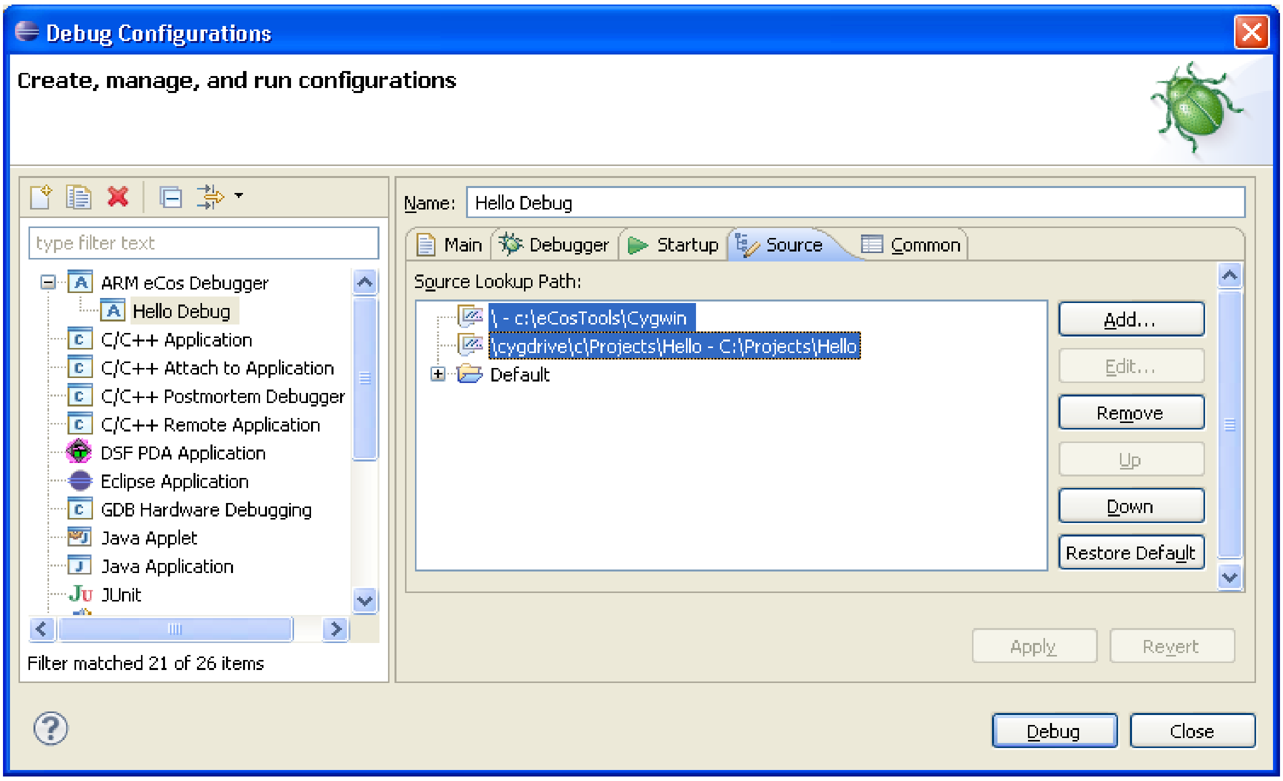
● Use project binary:

○ Use file:

Image offset (hex):

Filter matched 10 of 10 items

In the **Source** tab you can see some path mapping. These lines are added automatically so that the debugger can look up the source code. First line is the path mapping of the Cygwin root (that contains the eCos repository). The second one is the path mapping of the current project.



Debug Configurations

Create, manage, and run configurations

Name: Hello Debug

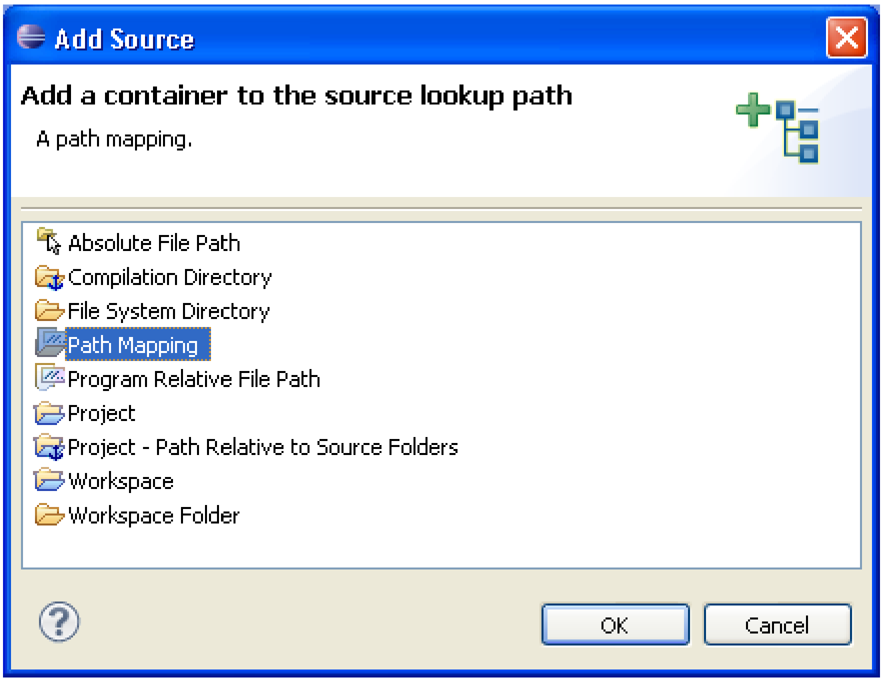
Source Lookup Path:

```
\ - c:\eCosTools\Cygwin
\cygdrive\c\Projects\Hello - C:\Projects\Hello
Default
```

Filter matched 21 of 26 items

If your project contains source files that are located outside these paths, you should add manually Path Mapping for these files, as shown below.

Click **Add** button, the **Add Source** dialog will appear:



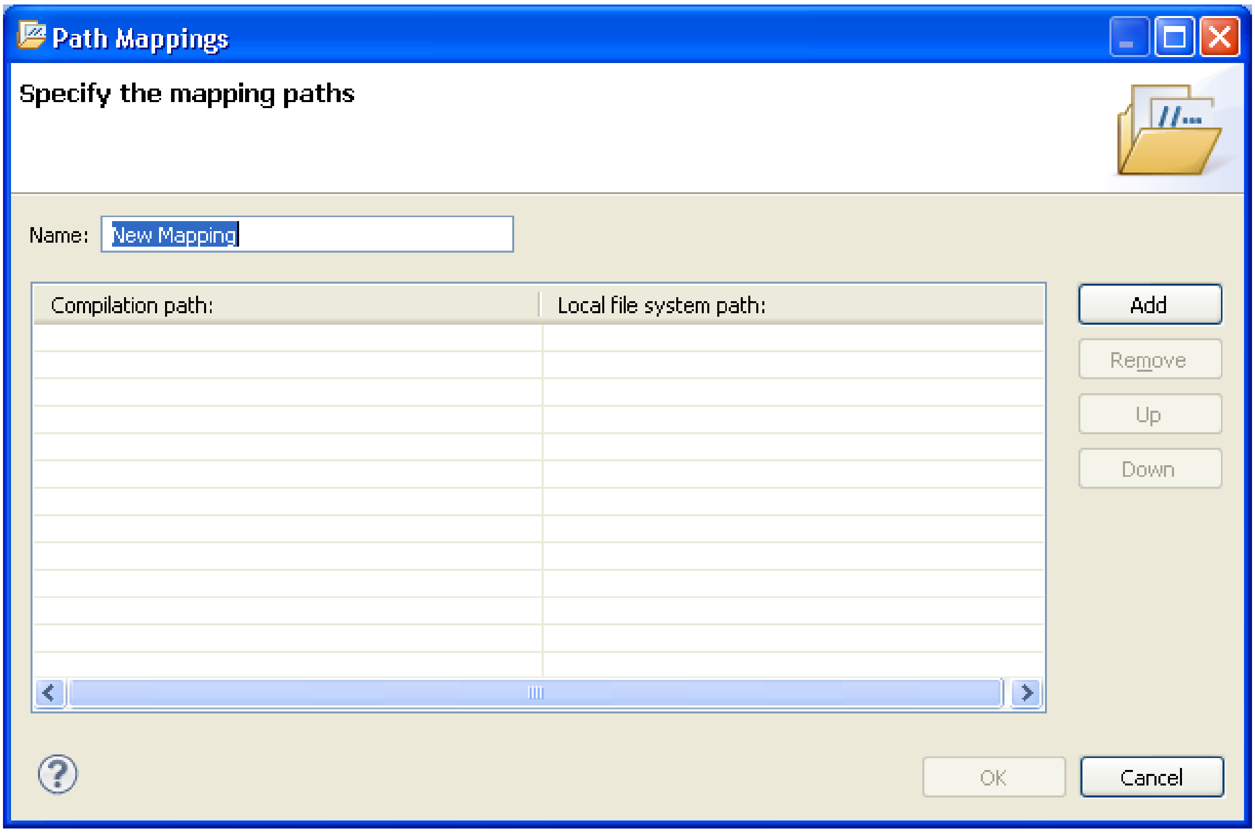
Add Source

Add a container to the source lookup path

A path mapping.

- Absolute File Path
- Compilation Directory
- File System Directory
- Path Mapping
- Program Relative File Path
- Project
- Project - Path Relative to Source Folders
- Workspace
- Workspace Folder

Double click on **Path Mapping** to open a new dialog:



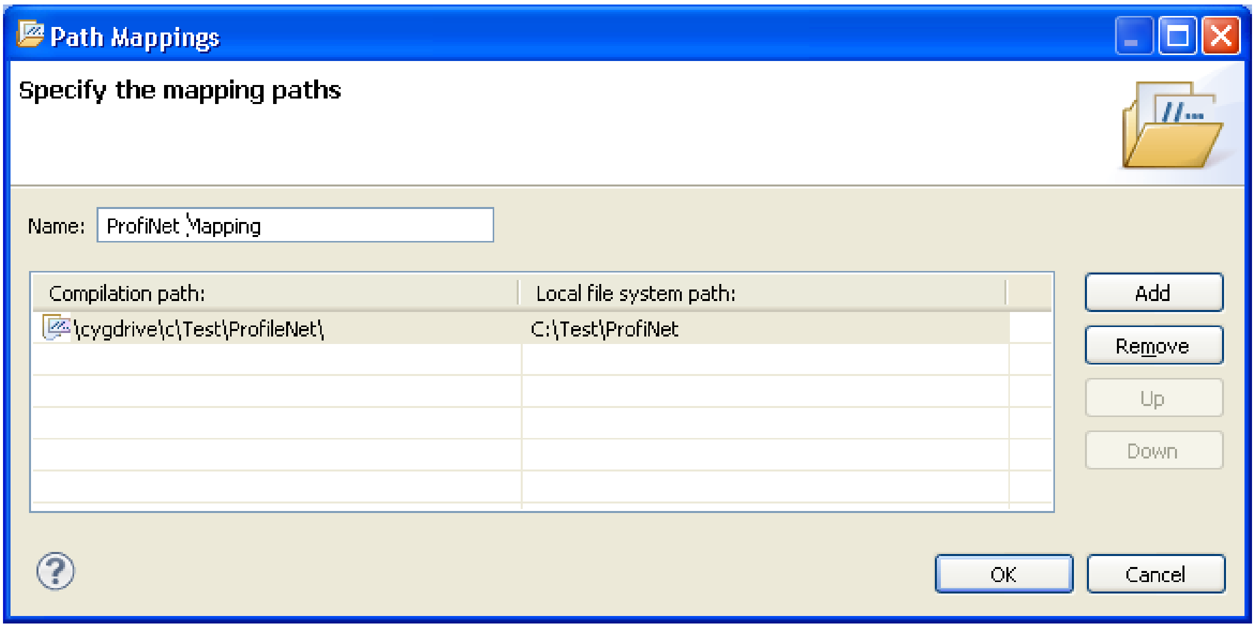
Path Mappings

Specify the mapping paths

Name: New Mapping

Compilation path:	Local file system path:

Click **Add** to begin adding Path Mapping. On the first column (**Compilation path**) put the Cygwin path to your source code, on the second one (**Local file system path**) put the Windows path to your source code. For example:



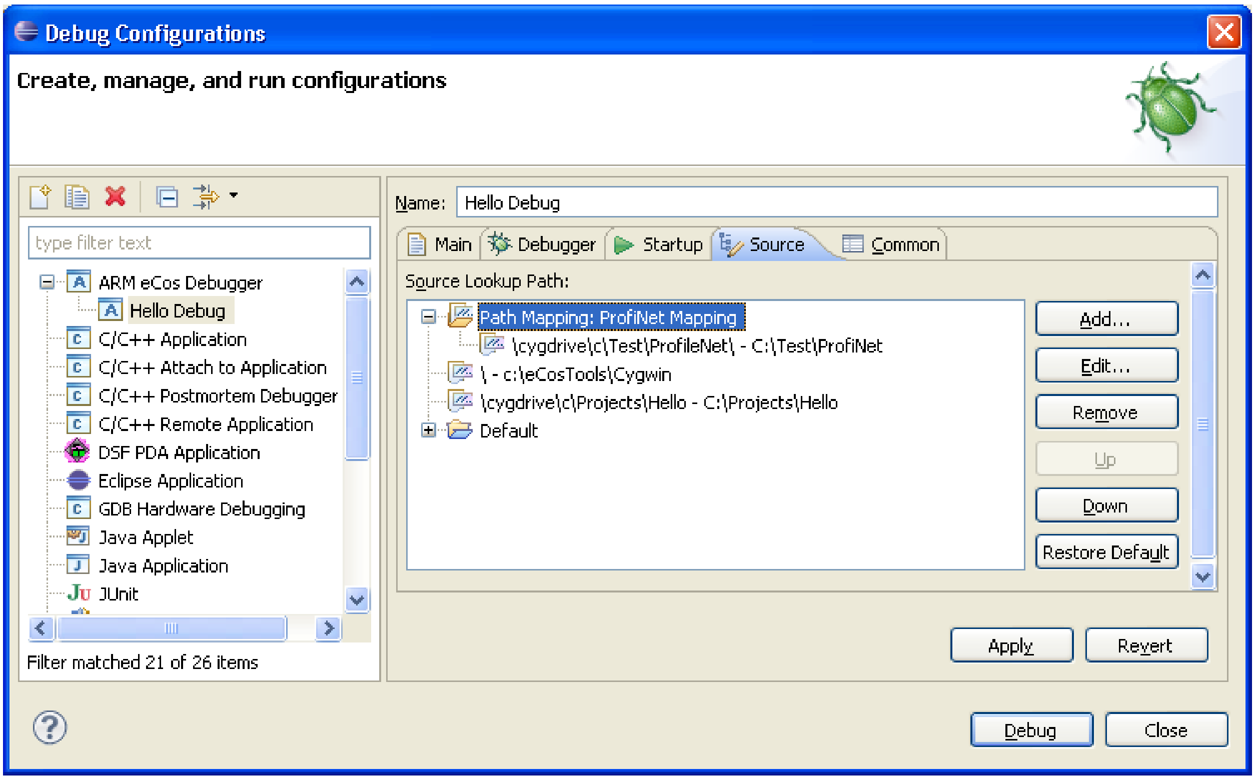
Path Mappings

Specify the mapping paths

Name: ProfiNet Mapping

Compilation path:	Local file system path:
\cygdrive\c\Test\ProfileNet\	C:\Test\ProfiNet

Click **OK** to finish adding Path Mapping:



Debug Configurations

Create, manage, and run configurations

type filter text

- ARM eCos Debugger
  - Hello Debug
- C/C++ Application
- C/C++ Attach to Application
- C/C++ Postmortem Debugger
- C/C++ Remote Application
- DSF PDA Application
- Eclipse Application
- GDB Hardware Debugging
- Java Applet
- Java Application
- JUnit

Filter matched 21 of 26 items

Name: Hello Debug

Source Lookup Path:

- Path Mapping: ProfiNet Mapping
  - \cygdrive\c\Test\ProfileNet\ - C:\Test\ProfiNet
  - \ - c:\eCosTools\Cygwin
  - \cygdrive\c\Projects\Hello - C:\Projects\Hello
- Default

Click **Apply** to save settings and click **Debug** to run the debugger.