Digi Embedded Linux

Building Your First Application

This tutorial introduces the power and features of Digi ESP™ for Embedded Linux as a development environment. It shows how to create a simple Linux application, transfer it to a target development board, debug it, and run it. This tutorial takes about 30 minutes. For more information on concepts and tasks in this tutorial, see the *Digi ESP Online Help*, by selecting **Help > Help Contents** from the Digi ESP menu.

Table of contents

90000853_F

1. Conventions in this tutorial

This tutorial uses these conventions, frames, and symbols to display information:

Convention	**Use**
`Style`	New terms and variables in commands, code, and other input.
`Style`	In examples, to show the contents of files, the output from commands. In text, the C code. Variables to be replaced with actual values are shown in italics.
Style	Items in the Digi ESP interface, such as menu items, dialogs, tabs, and buttons. In examples, to show the text that should be entered literally.
$	A prompt that indicates the action is performed in the host computer.
#	A prompt that indicates the action is performed in the target device.
`X.Y`	Used to indicate a version, for example DigiEL-X.Y can refer to DigiEL-4.0 or DigiEL-4.1 or any other version.

A warning that helps to solve or to avoid common mistakes or problems.

A hint that contains useful information about a topic.

```
$   A host computer session.
$   Bold text indicates what must be input.
```

```
#   A target session.
#   Bold text indicates what must be input.
```

2. Introduction

The instructions in this guide are based on the assumption that hardware has already been prepared and connected as instructed in the *Quick Start Guide*.

If Digi Embedded Linux (Digi EL) has been installed in the host computer, start the computer now.

If not, insert the LiveDVD and boot the host computer from the DVD drive. At the boot menu, select the option **Start or install Digi Embedded Linux LiveDVD**.

These instructions are based on the Kubuntu Linux distribution that comes with the kit. If executing another Linux distribution, menu names and options may differ.

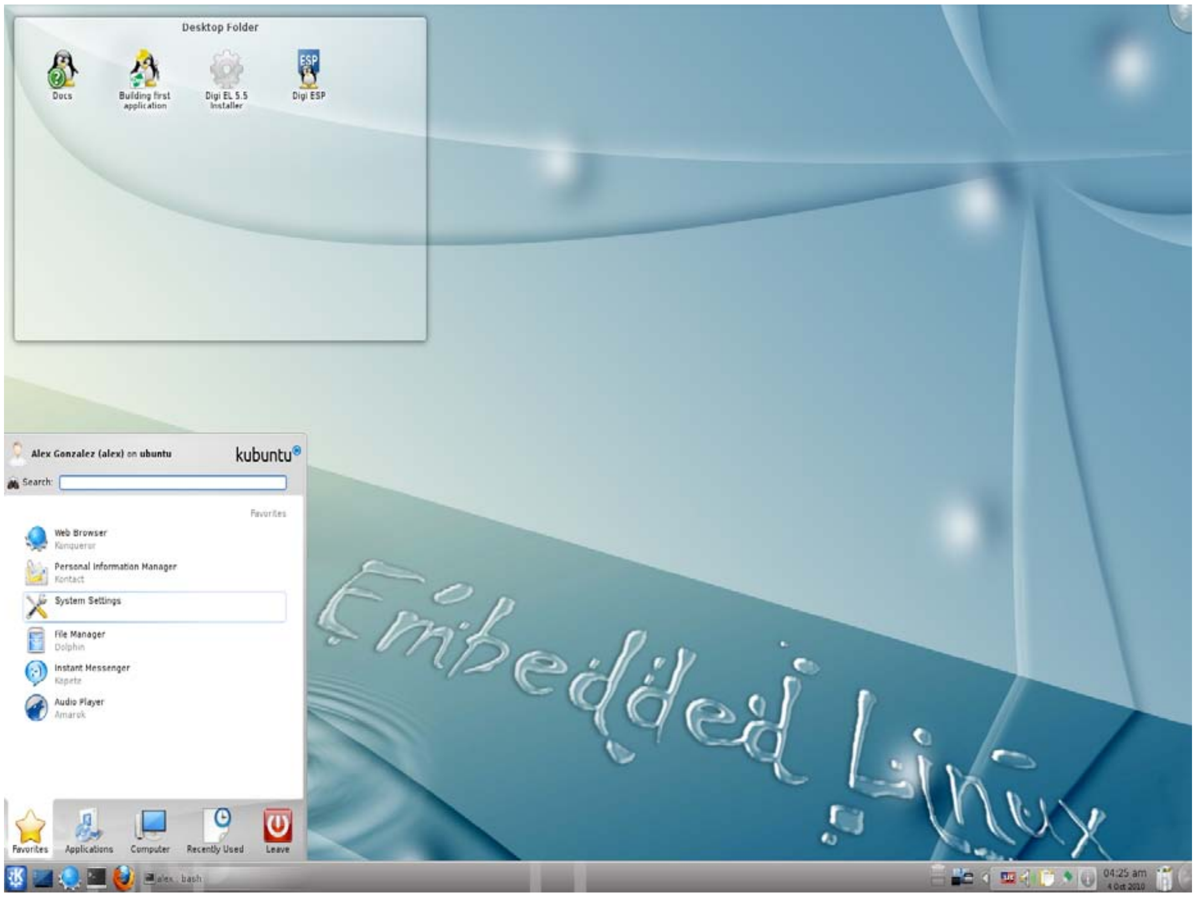
3. Check and change configuration settings as needed

Using Digi ESP^{TM} requires several system configuration settings on the host computer to be checked and changed as needed: display settings, network settings, and keyboard layout.

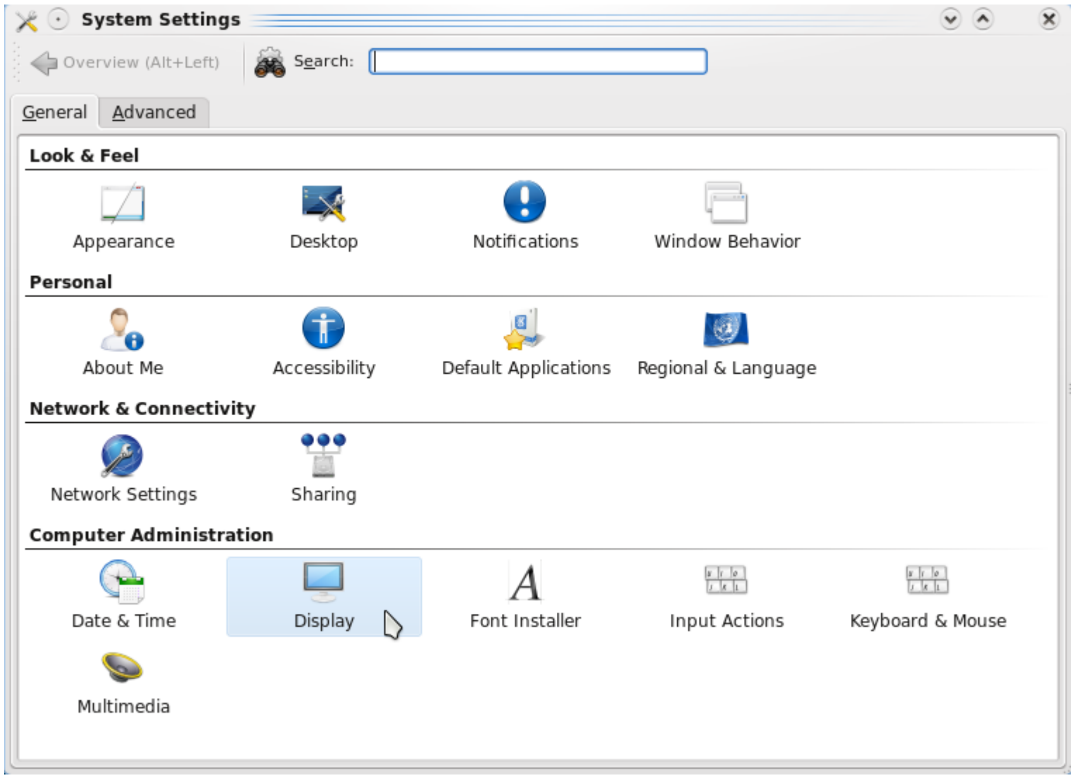
3.1. Display settings

The Digi ESP graphic environment requires a monitor that supports a minimum of 1024 x 768 screen resolution with 16-bit color.

To configure the screen's resolution go to **K-menu > Computer > System Settings**.



In the **Computer Administration** section, click **Monitor & Display**.



3.2. Network settings

To establish the connection between the development computer and the hardware target platform, both devices must be physically connected to the same network and have IP addresses of the same subnet.

To configure the target's network settings, contact your network administrator for this information:

- IP address for the host computer's Ethernet card.

If the hardware target platform has an Ethernet interface:

- IP address for the target's wired Ethernet. This IP address must be in the same subnet as the host computer.
- Subnet mask for the Ethernet interface.

And in case of having a wireless capable module:

- IP address for the target's WLAN interface; that is, a free IP address of the wireless network.
- Subnet mask for the Wireless interface.

If a DHCP server is installed in the network, there is no need to request the wired IP addresses. They are automatically assigned by the DHCP server.

3.2.1. *Configure the host's network settings*

If the Linux host machine has a known and configured IP address, skip this step.

The host computer's IP address may be unknown (if the address was obtained through a DHCP server) or unassigned. To obtain or assign an IP address for the host computer open a console and execute the command **ifconfig**.

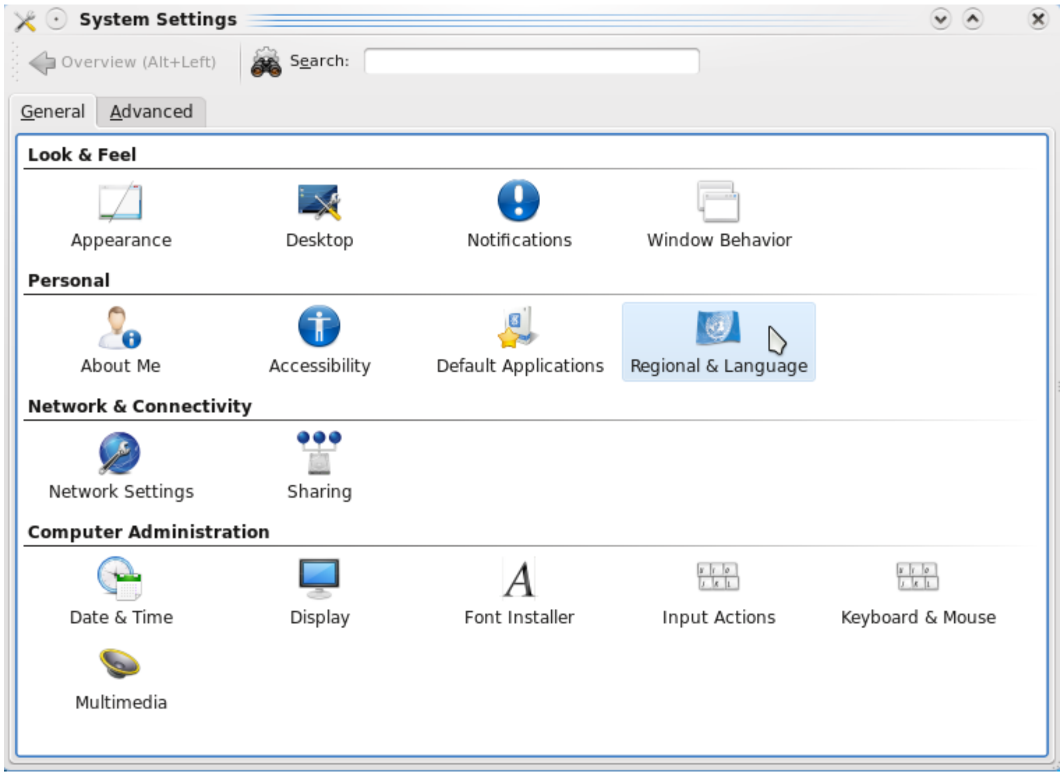
Check whether there is an IP address for the **eth0** network card. If it is, write it down. If not, set one provided by your network administrator by editing the file **/etc/network/interfaces** (it might be a different file in other Linux distributions):

```
    iface eth0 inet static
    address 192.168.42.1
    netmask 255.255.255.0
    gateway 192.168.42.1
```

Configuring the host computer's network settings requires Administrator privileges. If running Digi ESP from the LiveDVD, enter an empty password for the root.

3.3. Keyboard layout

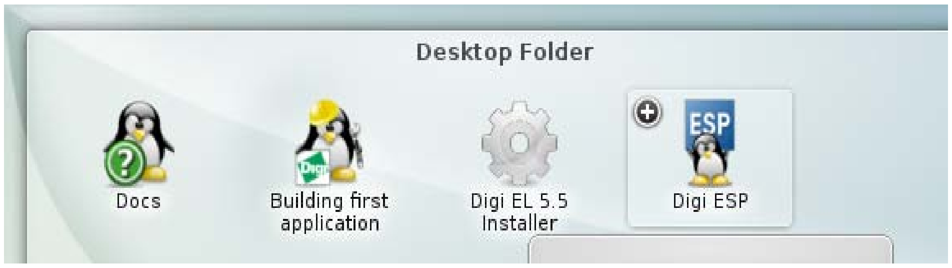
If Digi ESP is running from the LiveDVD or was installed without specifying a keyboard layout, the English keyboard layout is established by default. To change the keyboard layout on the host computer, go again to **System Settings** and on the **Personal** section click **Regional & Language**.



Then click on **Keyboard Layout** and select the appropriate regional keyboard layout from the list.

4. Start Digi ESP

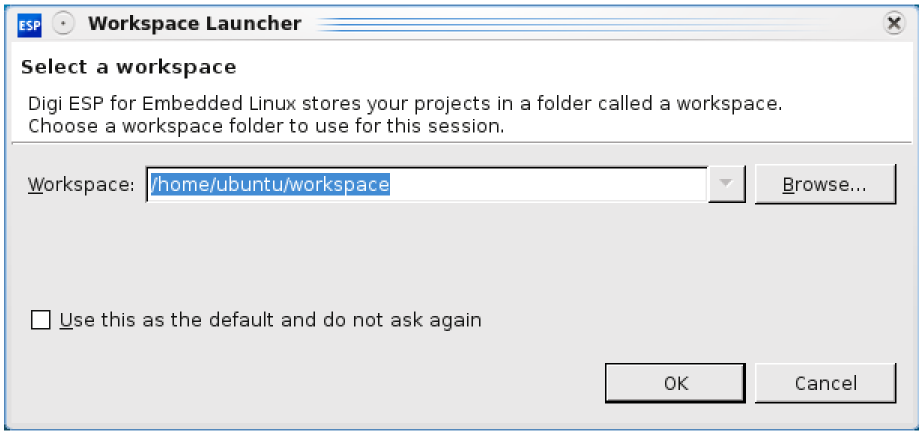
To start Digi ESP, click the Digi ESP icon on the desktop:



If no icon is displayed, run this command from the shell:

```
$   /usr/local/DigiEL-X.Y/digiesp/digiesp
```

When Digi ESP starts, the **Workspace Launcher** dialog is displayed asking for a *workspace* directory to be specified. The workspace is the directory where the Workbench stores the user projects and configurations. The default location for this directory is a subfolder named **workspace** in your user home directory, for example, ***/home/username/*workspace**.


Workspace Launcher

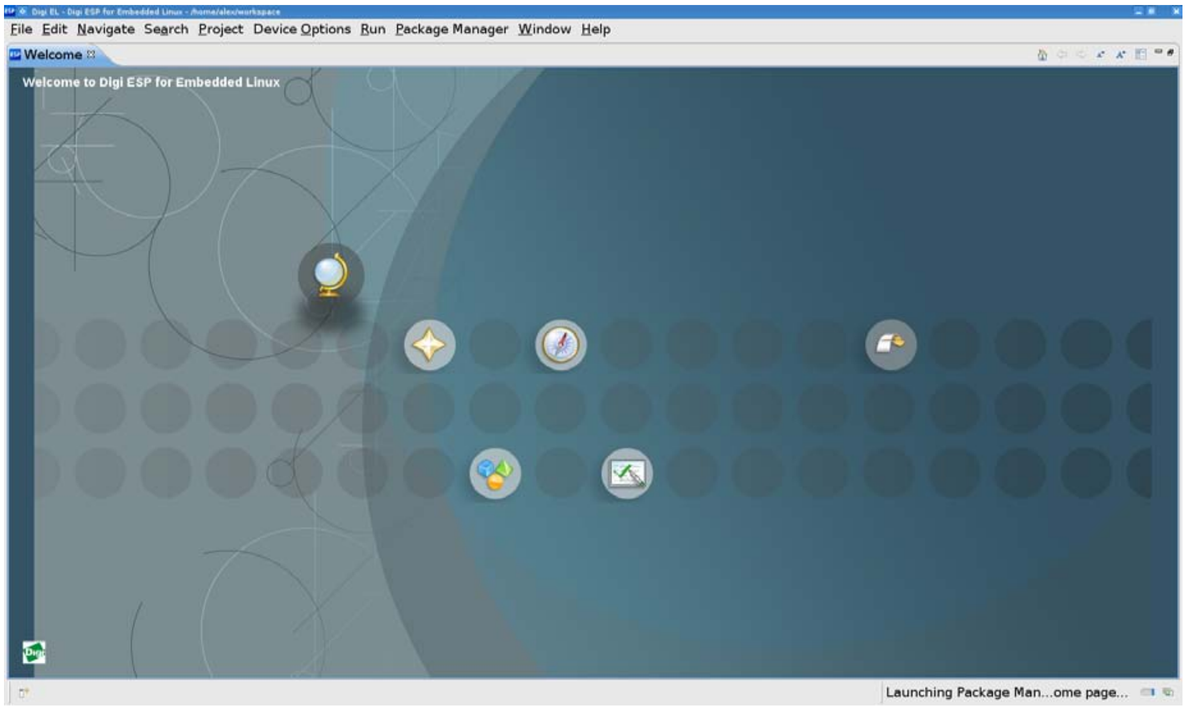
Select a workspace

Digi ESP for Embedded Linux stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: /home/ubuntu/workspace

Enter the location of the workspace, and click **OK**.

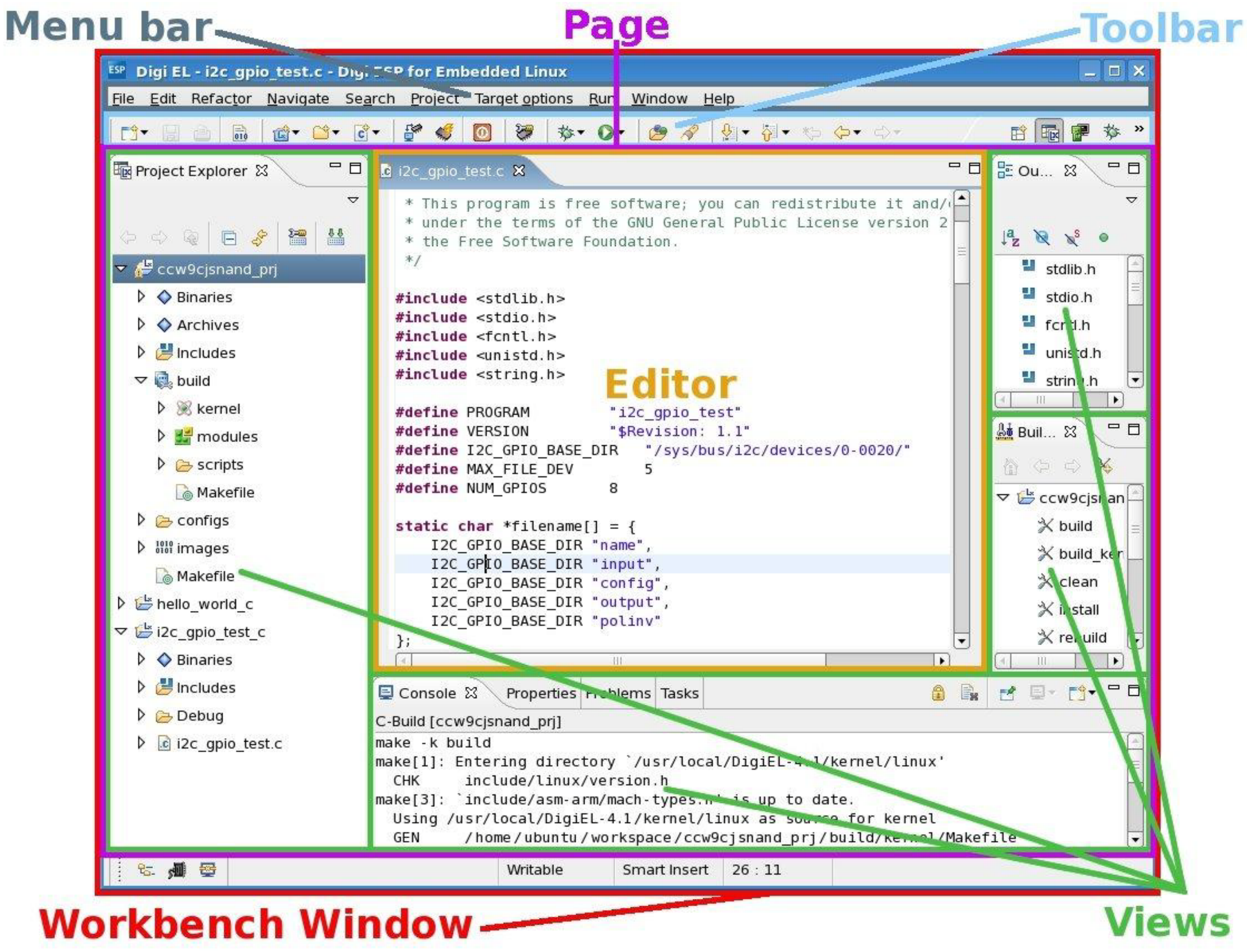
The **Welcome** page is displayed.


Digi EL - Digi ESP for Embedded Linux - /home/alex/workspace
File Edit Navigate Search Project Device Options Run Package Manager Window Help
Welcome
Welcome to Digi ESP for Embedded Linux
Launching Package Man...ome page...

Click on the arrow on the right side of the page to close the Welcome window and return to the Workbench. The Welcome window can be reached later from the menu option **Help > Welcome**.

5. Key components of the Digi ESP user interface

The figure shows key components and terms in the ESP user interface, including toolbars, perspectives, views, and the Editor, that are used throughout the rest of this tutorial. The *Digi ESP Online Help* covers toolbar buttons, views and perspectives in more detail.



```
 * This program is free software; you can redistribute it and/
 * under the terms of the GNU General Public License version 2
 * the Free Software Foundation.
 */

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

#define PROGRAM         "i2c_gpio_test"
#define VERSION         "$Revision: 1.1"
#define I2C_GPIO_BASE_DIR   "/sys/bus/i2c/devices/0-0020/"
#define MAX_FILE_DEV        5
#define NUM_GPIOS       8

static char *filename[] = {
    I2C_GPIO_BASE_DIR "name",
    I2C_GPIO_BASE_DIR "input",
    I2C_GPIO_BASE_DIR "config",
    I2C_GPIO_BASE_DIR "output",
    I2C_GPIO_BASE_DIR "polinv"
};
```

```
C-Build [ccw9cjsnand_prj]
make -k build
make[1]: Entering directory `/usr/local/DigiEL-4.1/kernel/linux'
  CHK     include/linux/version.h
make[3]: `include/asm-arm/mach-types.h' is up to date.
  Using /usr/local/DigiEL-4.1/kernel/linux as source for kernel
  GEN     /home/ubuntu/workspace/ccw9cjsnand_prj/build/kernel/Makefile
```

6. Work with the target

The target prints out messages in one of the serial ports (see your specific *Quick Start Guide* sheet). Working with the target involves these tasks:

- Identify the device node of the host's serial port
- Open and configure the Serial Console view
- Power up the target
- Configure the target's network settings
- Start Linux on the target
- Use common Linux commands and verify network connectivity
- Connect to the target's web server
- Create a Target Remote Configuration

6.1. Discover the device node of the host's serial port

In Linux, serial ports are special files called *device nodes*. They are normally populated as ***/dev/ttySn*** (where ***n*** is an index number).

To determine the available serial ports on your computer, execute the following command from the shell:

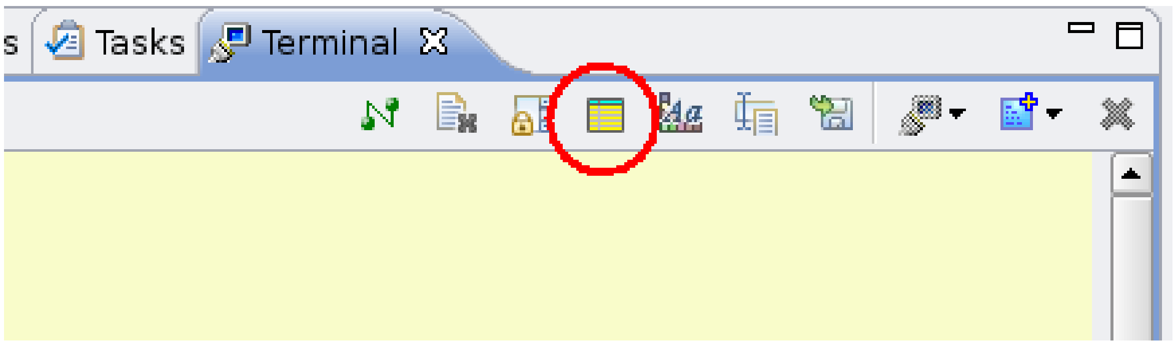
```
$  dmesg | grep ttyS
[17179574.464000] serial8250: ttyS0 at I/O 0x3f8 (irq = 4) is a 16550A
[17179574.468000] 00:08: ttyS0 at I/O 0x3f8 (irq = 4) is a 16550A
[17179574.468000] 0000:00:0b.0: ttyS4 at I/O 0xc800 (irq = 169) is a 16550A
[17179574.468000] 0000:00:0b.0: ttyS5 at I/O 0xc808 (irq = 169) is a 16550A
```

This command lists the serial ports detected by the system. In the example, three serial ports have been detected: **ttyS0**, **ttyS4** and **ttyS5**.

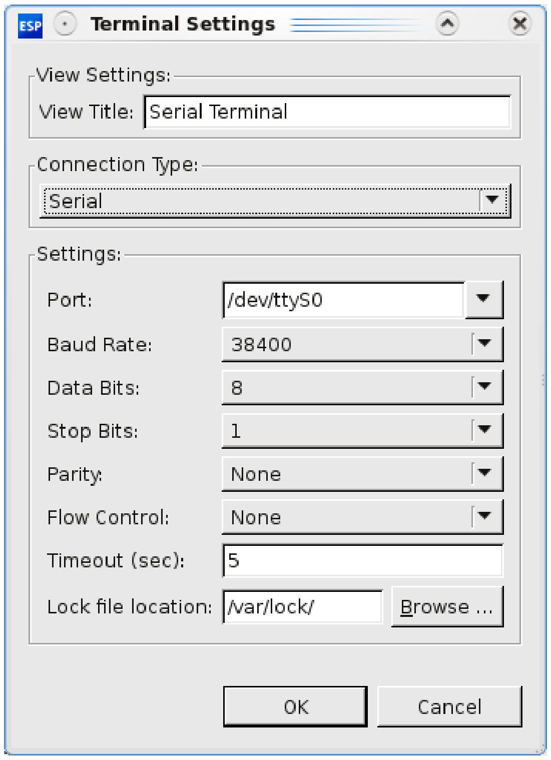
If using a USB to Serial converter, serial ports will normally populate as ***/dev/ttyUSBn****, where* ***n*** *is an index number.*

6.2. Open and configure the Serial Console view

The target board prints out messages on the serial port. To open a **Serial Terminal** in Digi ESP, select **Window > Show View > Terminal**. This view is typically displayed in the bottom part of the Workbench window, next to the **Task** view. Open the **Terminal Settings** dialog by clicking the **Settings** button on the **Terminal** view's toolbar.



In the **View Title**, type *Serial Terminal.* In the **Connection Type** combo, select **Serial.** The **Serial Port** must be configured with the device node into which the serial cable is plugged. Try with each one of the serial ports detected before until the correct one is found. Do not change the other values (38400 baud, 8 data bits, 1 stop bit, no parity). Click **OK** to accept the configuration.



Terminal Settings

View Settings:

View Title: Serial Terminal

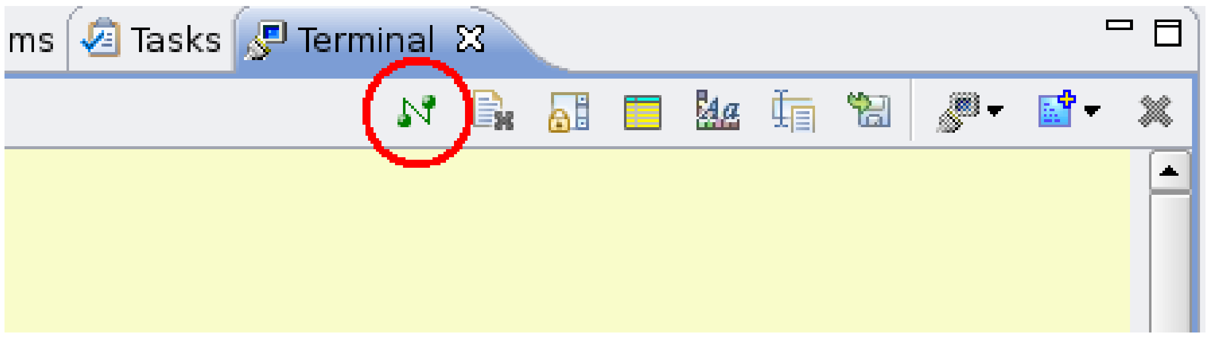
Connection Type:

Serial

Settings:

Port:	/dev/ttyS0
Baud Rate:	38400
Data Bits:	8
Stop Bits:	1
Parity:	None
Flow Control:	None
Timeout (sec):	5
Lock file location:	/var/lock/

Once the serial port is configured, establish the connection by clicking the **Connect/Disconnect** button of the Terminal view's toolbar.



In some Linux distributions, serial ports have restricted access. If the serial port cannot be opened, its access permissions must be changed.

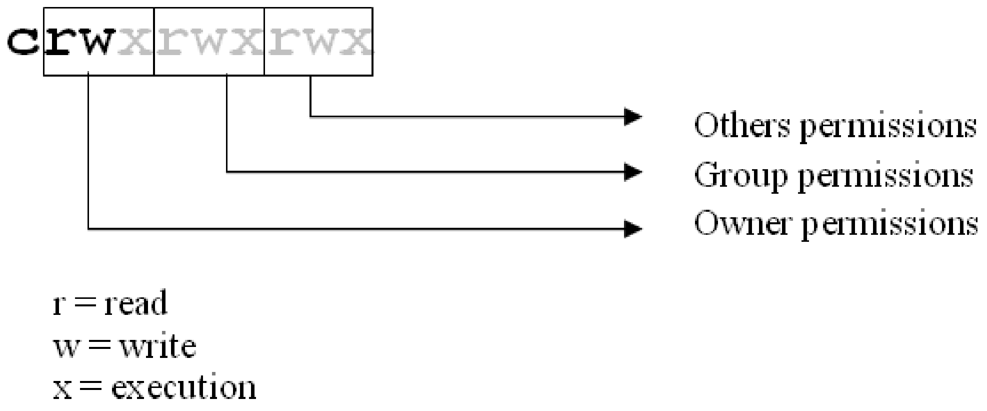
6.2.1. *Granting access to the serial port*

If the serial port cannot be accessed, the most probable reason is that the serial port device node does not have read/write permissions for standard users.

To print the serial port permissions open a console and execute the **ls** command, specifying the actual serial port device node in place of **/dev/ttyS0**.

```
$  ls –l /dev/ttyS0
crw------- 1 root dialout 4, 64 2007-03-26 08:37 /dev/ttyS0
```

The **ls** command lists the owner (**root**), group (**dialout**), and permissions (**crw-------**) of the serial port **/dev/ttyS0**. The serial port permissions are organized as follows:



To make the serial port accessible to every user, add read/write permissions for the Group and Others. Open a console and execute the **chmod** command, as **root**, specifying the actual serial port device node in place of **/dev/ttyS0**.

```
#  chmod 666 /dev/ttyS0
```

Kubuntu distribution does not usually have a 'root' user. To execute commands as root you need to prepend the 'sudo' command to the command you want to execute. Example: 'sudo chmod 666 /dev/ttyS0'.

You can also permanently work with root permissions by executing the command 'sudo –s'.

This command grants read/write access to the serial port for every user. Check the new permissions with the **ls** command:

```
$  ls –l /dev/ttyS0
crw-rw-rw- 1 root dialout 4, 64 2007-03-26 08:37 /dev/ttyS0
```

6.3. Power up the target

Power on the target development board using the main slider power switch on the development board (usually located at one of the corners).

The LEDs on the board light up and boot messages are displayed in the Serial Console. These messages vary depending on the module and development board model. When they appear, press a key within four seconds to stop the auto boot process. If no messages are displayed, try configuring the Serial Console with a different port among the detected ones.

```
U-Boot 1.1.6 (Nov 14 2008 - 01:08:53)
for ConnectCore 9P 9360 on Development Board

DRAM:  64 MB
NAND:  128 MB
Error: no valid bmp image at address 0x62a18
CPU:   NS9360 @ 176.947200MHz
SPI ID:2007/04/26, V1_5, CC9P9360, SDRAM 64MByte, CL2, 7.8us
Hit any key to stop autoboot:  0
#
```

6.4. Configure the target's network settings

The target can be configured to use a dynamic IP address from an available DHCP server, or a fixed IP address.

6.4.1. *Wired Ethernet*

If the hardware target platform has a wired Ethernet interface follow these steps to configure it:

6.4.1.1. *Dynamic IP address*

If a DHCP server is available, to assign a dynamic IP address assigned to the target’s wired Ethernet interface, enter the following from the boot loader shell:

```
#   setenv dhcp yes
#   saveenv
```

6.4.1.2. *Fixed IP address*

To configure a fixed IP address for the target's Ethernet interface do the following from the boot loader shell:

```
#   setenv ipaddr AAA.AAA.AAA.AAA
#   setenv netmask BBB.BBB.BBB.BBB
#   setenv serverip EEE.EEE.EEE.EEE
#   saveenv
```

Where:

- ***AAA.AAA.AAA.AAA*** is the IP address of the target's wired Ethernet (it must be in the same subnet than the host computer's IP).
- ***BBB.BBB.BBB.BBB*** is the target's subnet mask.

- ***EEE.EEE.EEE.EEE*** is the IP address of the host computer.

The last command, **saveenv**, saves the target's network settings in NVRAM.

6.4.2. *Wireless*

If the hardware target platform has a wireless interface follow these steps to configure it:

6.4.2.1. *Dynamic IP address*

If a DHCP server is available, to assign a dynamic IP address assigned to the target’s wireless interface, enter the following from the boot loader shell:

```
#   setenv dhcp_wlan yes
#   saveenv
```

6.4.2.2. *Fixed IP address*

To configure a fixed IP address for the target's wireless interface do the following from the boot loader shell:

```
#   setenv ipaddr_wlan CCC.CCC.CCC.CCC
#   setenv netmask_wlan DDD.DDD.DDD.DDD
#   setenv serverip EEE.EEE.EEE.EEE
#   saveenv
```

Where:

- ***CCC.CCC.CCC.CCC*** is the IP address for the wireless adapter (only for wireless capable modules).
- ***DDD.DDD.DDD.DDD*** is the wireless subnet mask (only for wireless capable modules).
- ***EEE.EEE.EEE.EEE*** is the IP address of the host computer.

The last command, **saveenv**, saves the target's network settings in NVRAM.

6.5. Start Linux

Reset the target but this time, do not stop the auto boot process. After some seconds, the boot loader unpacks and launches the pre-installed Linux kernel from the built-in Flash memory. These output messages are displayed on the Serial Console:

```
...
...
[LINUX KERNEL BOOT MESSAGES]
...
...
Starting dropbear sshd: OK
Starting vsftpd server: OK
Starting httpd webserver: OK


BusyBox v1.18.4 (2011-06-20 03:54:39 CEST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

/ #
```

6.6. Use common Linux commands

After Linux starts successfully, a basic shell is available in the Serial Console. Try entering some Linux commands. To list files, use the **ls** command. To navigate through directories, use the **cd** command.

6.7. Check network connectivity

For development, it is important that the hardware target platform is able to communicate through the network with the host computer.

6.7.1. *Finding the IP address of the target*

If the target is configured to obtain its IP address through DHCP, use this command to find the assigned IP address:

```
#   ifconfig eth0
```

Or, if the target platform has a wireless interface and no wired Ethernet, use this command:

```
#   ifconfig wlan0
```

6.7.2. *Ping the IP address of the host computer*

To verify the network connectivity, ping the IP of the host computer and check that it responds. For example, if the server IP is 192.168.42.1 do:

```
#   ping 192.168.42.1
PING 192.168.42.1 (192.168.42.1) 56(84) bytes of data.
64 bytes from 192.168.42.1: icmp_seq=1 ttl=128 time=0.361 ms
64 bytes from 192.168.42.1: icmp_seq=2 ttl=128 time=0.253 ms
64 bytes from 192.168.42.1: icmp_seq=3 ttl=128 time=0.384 ms
64 bytes from 192.168.42.1: icmp_seq=4 ttl=128 time=0.255 ms
```

If it doesn't respond, check that both the target and host are in the same subnet. If the target is a wireless only platform, you may need to configure it to connect to your wireless access point.

6.7.3. *Configuring the wireless interface*

If the hardware target platform doesn't have a wired Ethernet and it only has a wireless interface you must correctly configure it to connect to your wireless access point.

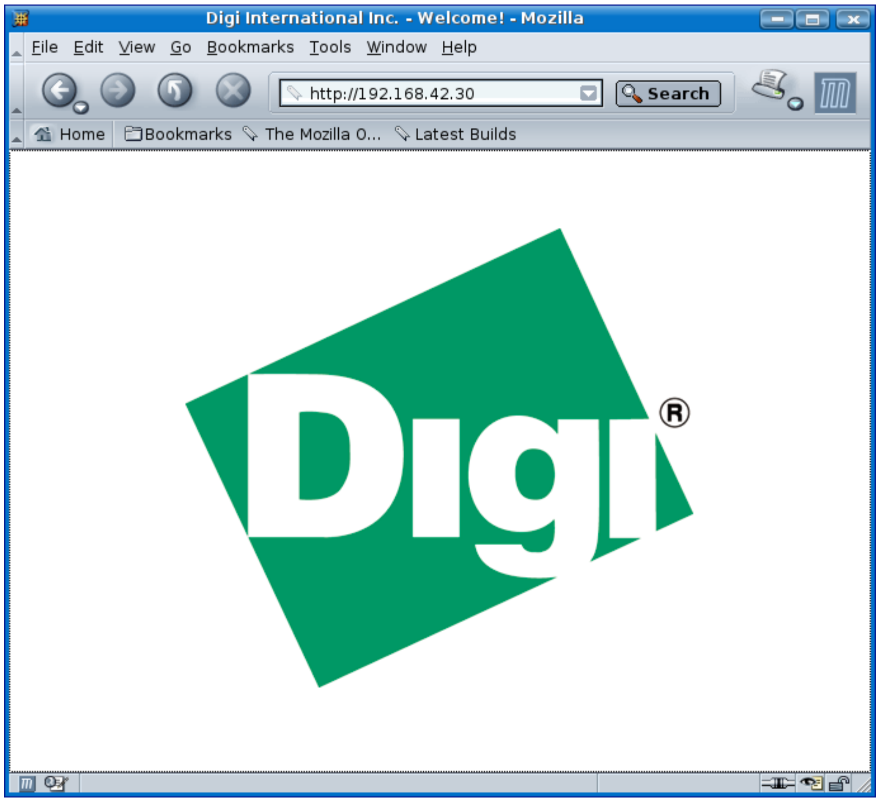
By default, the target will try to connect to the nearest open wireless access point in range so the easiest way to create the connection is to configure the access point to be open (no authentication or encryption).

Creating the connection to an encrypted access point requires further configuration steps using the Linux wireless tools. Please refer to *Digi ESP Online Help*, accessible from menu **Help > Help Contents > Digi ESP for Embedded Linux** and specifically to chapter **WLAN adapter guide > Wireless connection examples** to see some examples to help you through.

6.8. Connect to the target's web server

A web server is included and started by default. The web server serves a default web page that resides in the target filesystem. To access the web server:

1. Open a web browser.
2. In the **Address** input box, enter the IP address of the target. This page is displayed:


Digi International Inc. - Welcome! - Mozilla
File Edit View Go Bookmarks Tools Window Help
http://192.168.42.30
Search
Home
Bookmarks
The Mozilla O...
Latest Builds
Digi®

6.9. Create a Target Remote Configuration

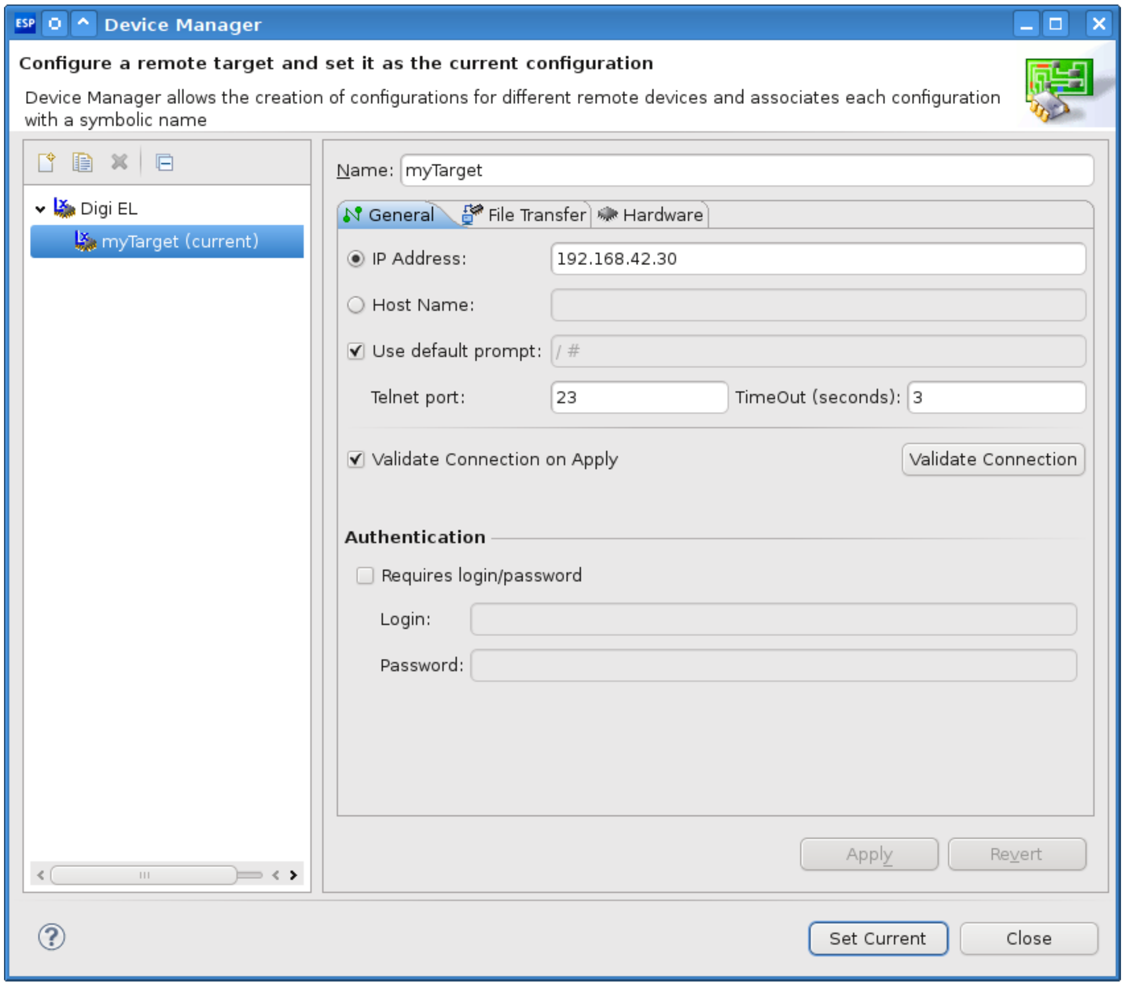
For Digi ESP to communicate with the target a new *remote configuration* –a set of configuration options for a specific target– is needed.

To create a new remote configuration, select menu **Device Options > Device Manager**.

In the **Remote Configurations** tree, select **DigiEL** and click **New**. In the **Name** edit box, enter the name of the remote configuration, for example, **myTarget**.

Remote configuration settings are organized on three tabs:

On the **General** tab, in the **IP Address** field, enter the IP address of the target; for example, **192.168.42.30**.


Device Manager

Configure a remote target and set it as the current configuration

Device Manager allows the creation of configurations for different remote devices and associates each configuration with a symbolic name

- Digi EL
  - myTarget (current)

Name: myTarget

General | File Transfer | Hardware

(●) IP Address: 192.168.42.30

( ) Host Name:

[✓] Use default prompt: / #

Telnet port: 23 TimeOut (seconds): 3

[✓] Validate Connection on Apply

Authentication

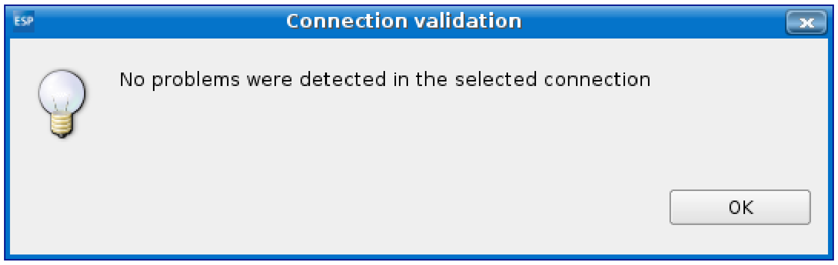
[ ] Requires login/password

Login:

Password:

Click the **Validate Connection** button to test this connection. This action validates bidirectional connectivity via telnet.

If Digi ESP can reach the target's IP address, this message is displayed:

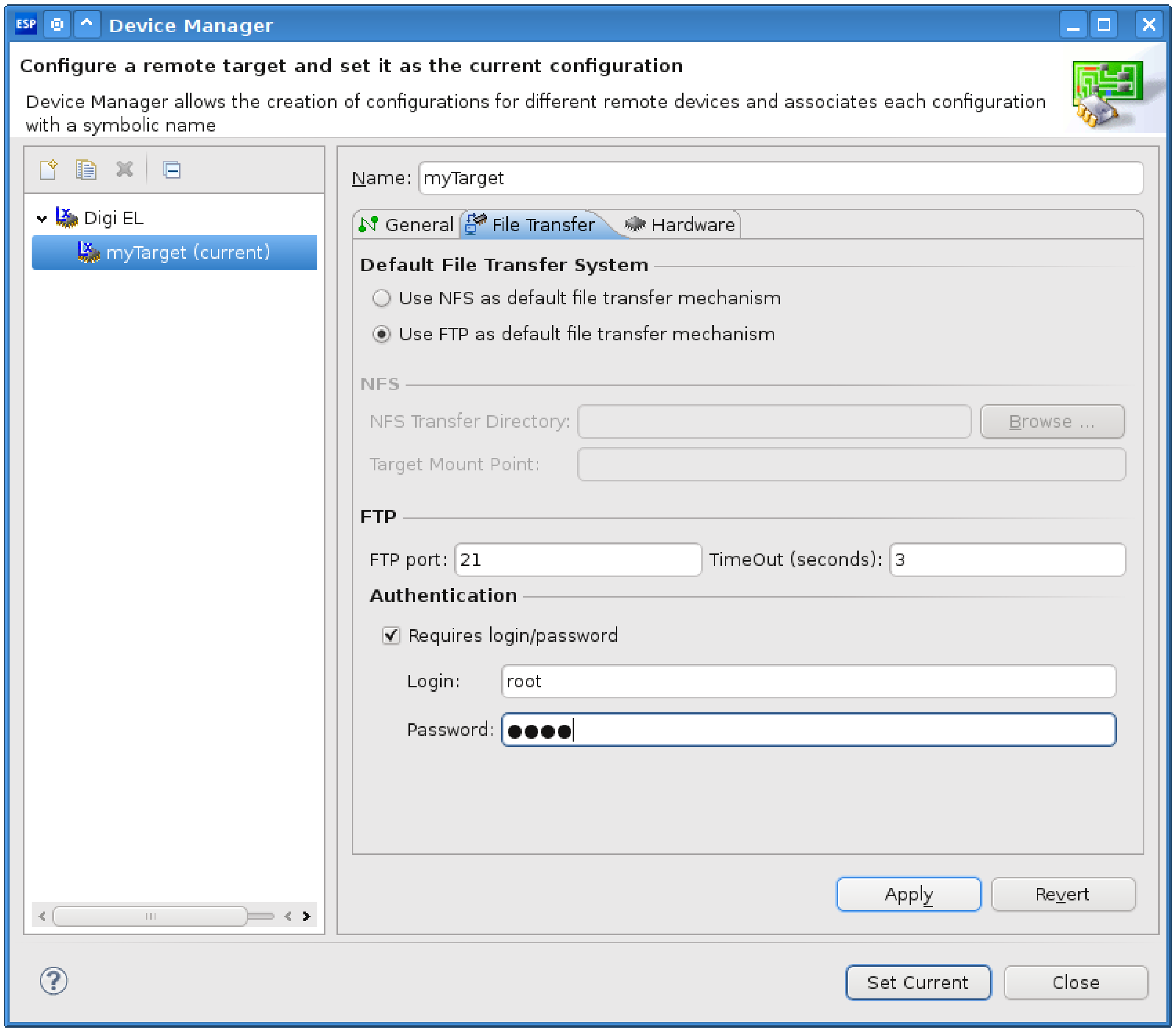


Connection validation

No problems were detected in the selected connection

If it is not displayed, verify that the network settings of the host and the target are in the same subnet and that they are physically connected to the same network.

Click on the **File Transfer** tab to configure the file transfer mechanism between Digi ESP and the embedded system. Select **Use FTP as default file transfer mechanism**. In the **Authentication** group, select (enable) the **Requires login/password** checkbox. Enter user login information: for **Login** enter **root**, for **Password** enter **root**.



Device Manager

Configure a remote target and set it as the current configuration

Device Manager allows the creation of configurations for different remote devices and associates each configuration with a symbolic name

Name: myTarget

Default File Transfer System

( ) Use NFS as default file transfer mechanism

(●) Use FTP as default file transfer mechanism

NFS

NFS Transfer Directory:

Target Mount Point:

FTP

FTP port: 21 TimeOut (seconds): 3

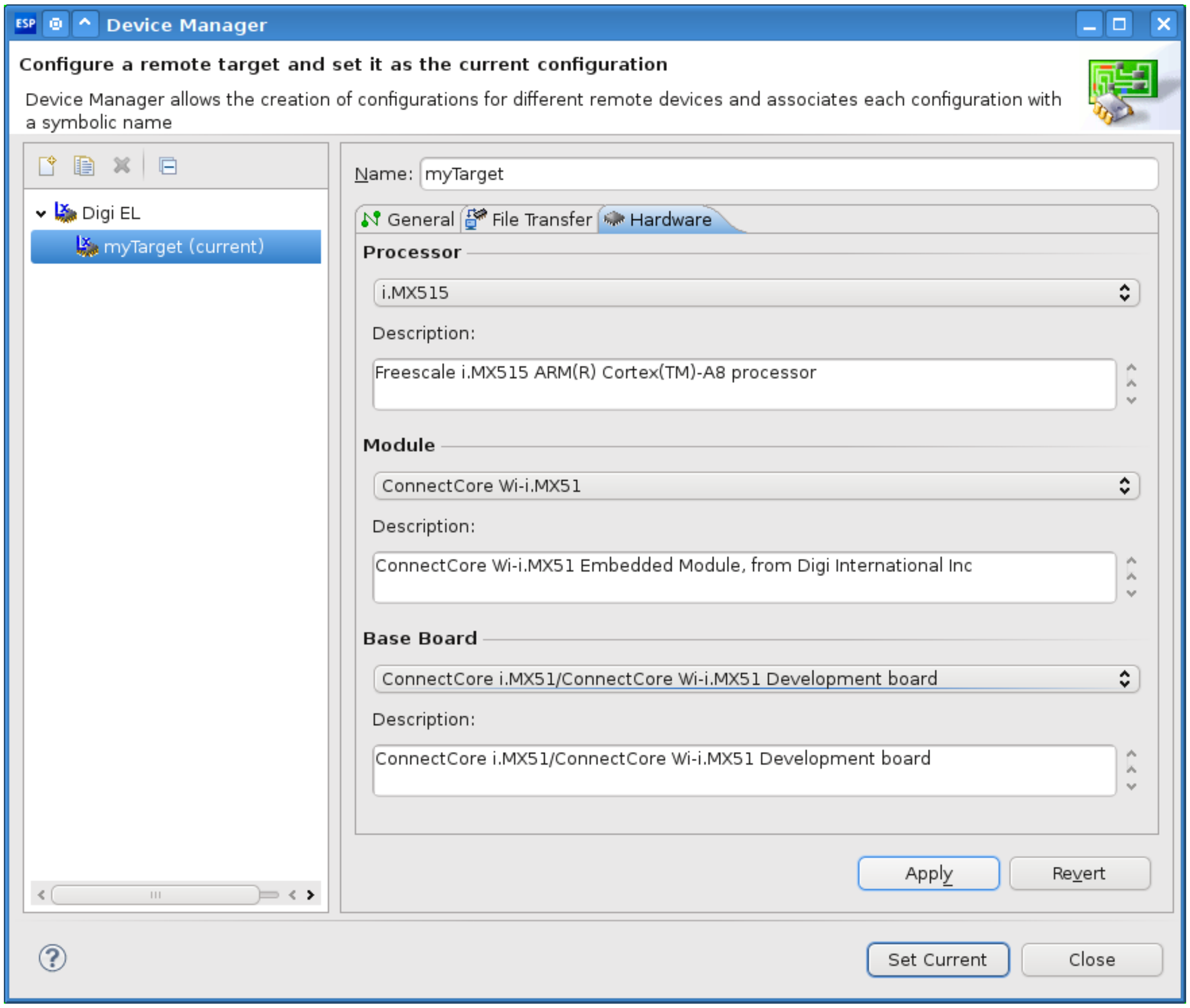
Authentication

[✓] Requires login/password

Login: root

Password: ●●●●

Click on the **Hardware** tab to identify the hardware components of the target device. Depending on the model type of the development board, select the appropriate values for **Processor**, **Module**, and **Base Board**.



Configure a remote target and set it as the current configuration

Device Manager allows the creation of configurations for different remote devices and associates each configuration with a symbolic name

Name: myTarget

Processor

i.MX515

Description:

Freescale i.MX515 ARM(R) Cortex(TM)-A8 processor

Module

ConnectCore Wi-i.MX51

Description:

ConnectCore Wi-i.MX51 Embedded Module, from Digi International Inc

Base Board

ConnectCore i.MX51/ConnectCore Wi-i.MX51 Development board

Description:

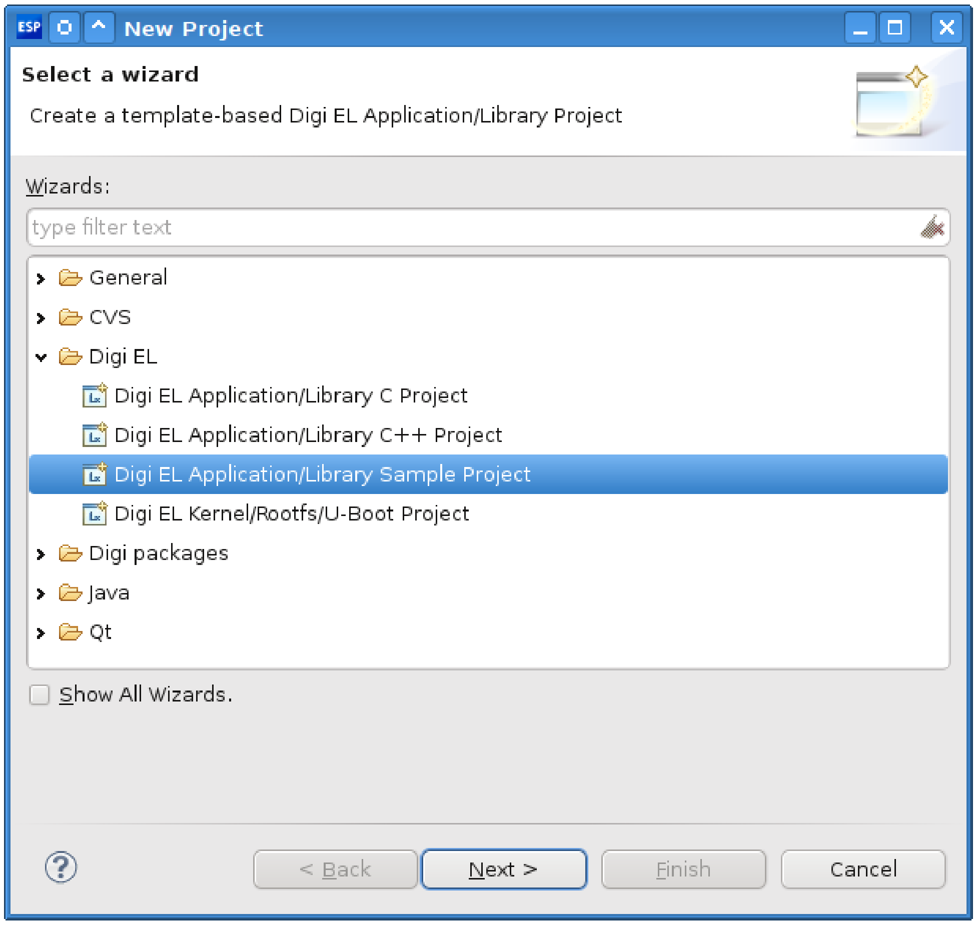
ConnectCore i.MX51/ConnectCore Wi-i.MX51 Development board

Click the **Set Current** button to make this remote configuration the current configuration used by Digi ESP.

7. Create a template-based managed make project

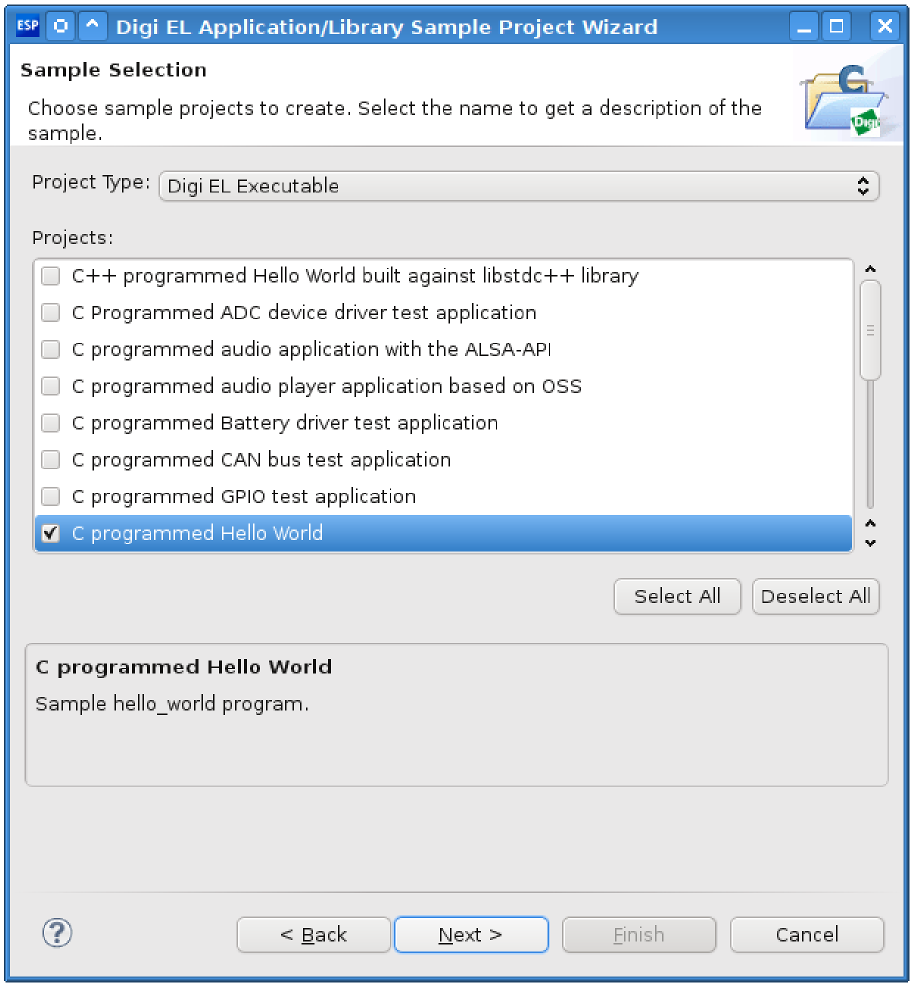
Next, create a new application project.

Select **File > New > Project....** The **New Project** wizard is displayed. From the list of projects, select **Digi EL > Digi EL Application/Library Sample Project**. This selection creates a template-based application. Click **Next.**


New Project
Select a wizard
Create a template-based Digi EL Application/Library Project
Wizards:
type filter text
General
CVS
Digi EL
Digi EL Application/Library C Project
Digi EL Application/Library C++ Project
Digi EL Application/Library Sample Project
Digi EL Kernel/Rootfs/U-Boot Project
Digi packages
Java
Qt
Show All Wizards.
< Back
Next >
Finish
Cancel

The **Sample Selection** page is displayed. The **Project Type** combo selects which template-based project to create. Select **Digi EL Executable**, which indicates that an application project, instead of a library, will be created.

From the **Projects** list, select **C programmed Hello World** and click **Next**.



Choose sample projects to create. Select the name to get a description of the sample.

Project Type: Digi EL Executable

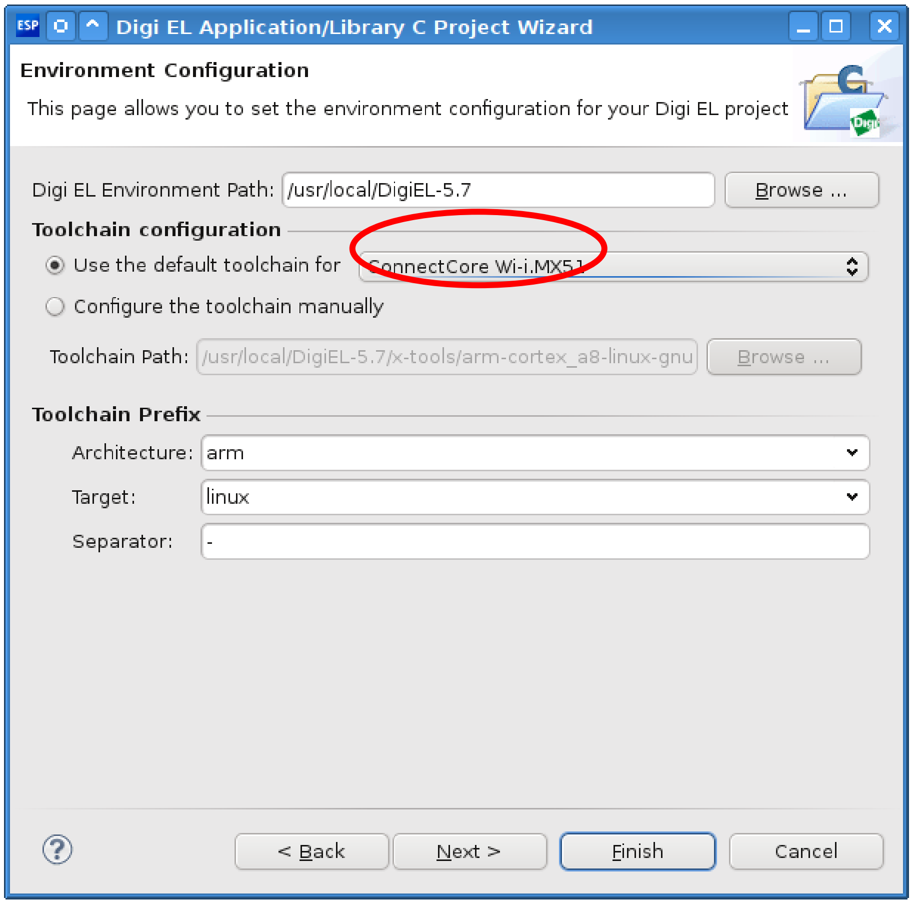
Projects:

- [ ] C++ programmed Hello World built against libstdc++ library
- [ ] C Programmed ADC device driver test application
- [ ] C programmed audio application with the ALSA-API
- [ ] C programmed audio player application based on OSS
- [ ] C programmed Battery driver test application
- [ ] C programmed CAN bus test application
- [ ] C programmed GPIO test application
- [✓] C programmed Hello World

C programmed Hello World

Sample hello_world program.

The next dialog lets you choose the Digi EL environment and toolchain to build the project with. Select the target hardware platform to let the wizard select the default toolchain for it, or manually select a toolchain of your choice.



Digi EL Application/Library C Project Wizard

Environment Configuration

This page allows you to set the environment configuration for your Digi EL project

Digi EL Environment Path: /usr/local/DigiEL-5.7

Toolchain configuration

(●) Use the default toolchain for: ConnectCore Wi-i.MX51

( ) Configure the toolchain manually

Toolchain Path: /usr/local/DigiEL-5.7/x-tools/arm-cortex_a8-linux-gnu

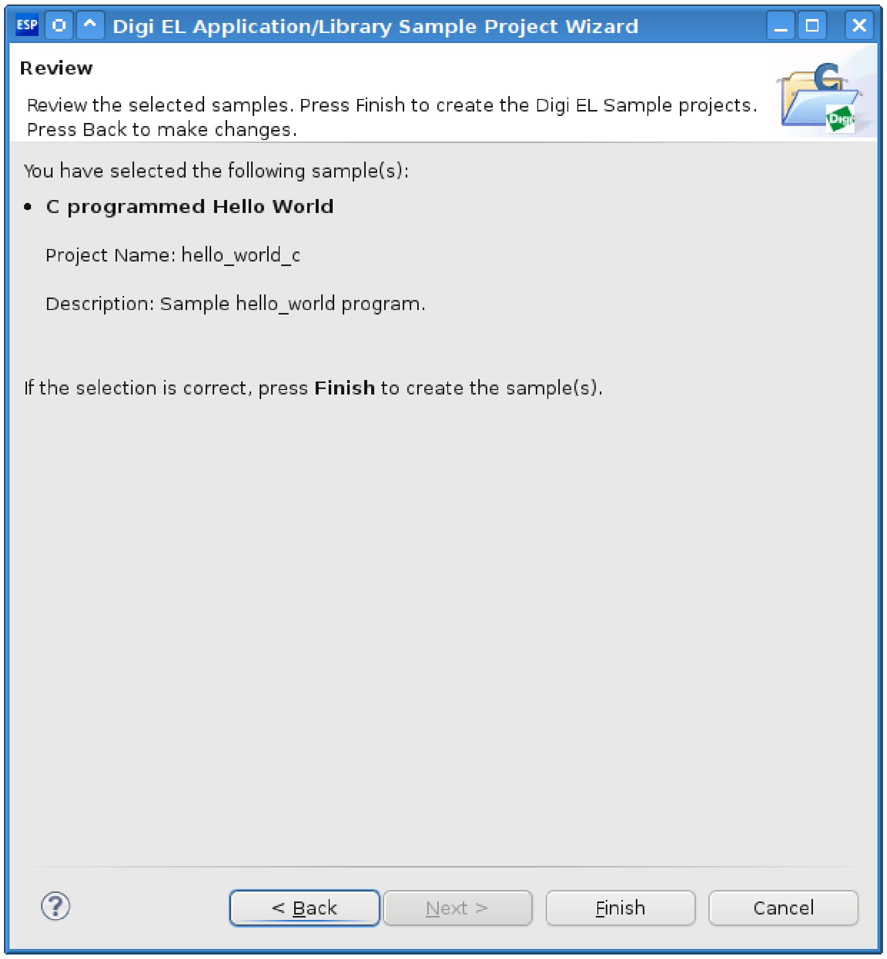
Toolchain Prefix

Architecture: arm

Target: linux

Separator: -

The last page of the wizard shows a summary of the selections. Click **Finish** to create the application project.



Review

Review the selected samples. Press Finish to create the Digi EL Sample projects. Press Back to make changes.

You have selected the following sample(s):

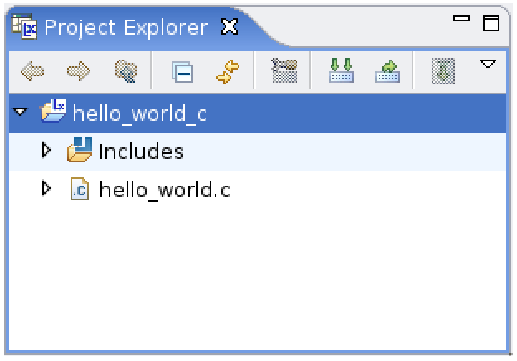
- **C programmed Hello World**

  Project Name: hello_world_c

  Description: Sample hello_world program.

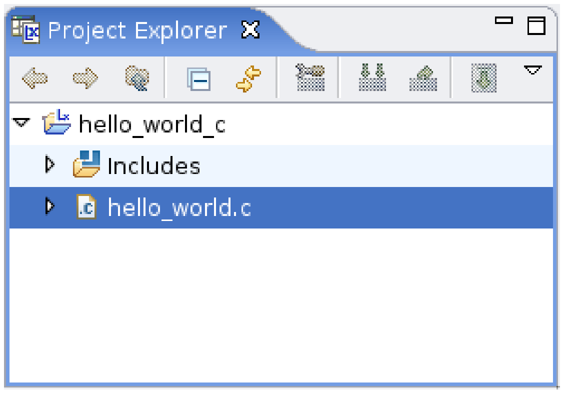
If the selection is correct, press **Finish** to create the sample(s).

The newly created project appears in the **Project Explorer** view (on the left side).

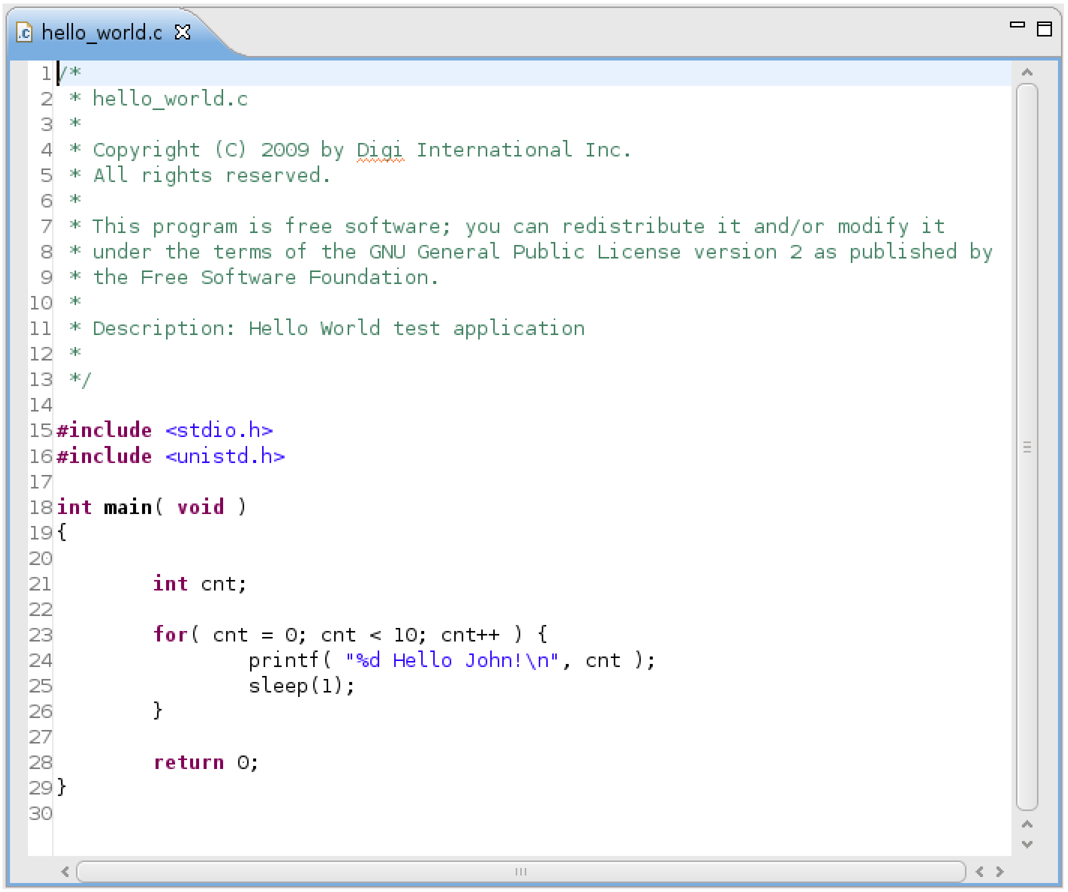


8. Open the sample application source code

In the **Project Explorer** view, select the application project **hello_world_c** and expand it to see its contents. There is a source file called **hello_world.c**; double-click the file.



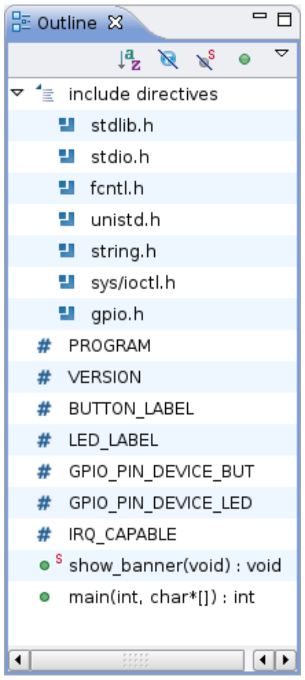
The **C/C++ Editor** is opened with the application's code.



```
/*
 * hello_world.c
 *
 * Copyright (C) 2009 by Digi International Inc.
 * All rights reserved.
 *
 * This program is free software; you can redistribute it and/or modify it
 * under the terms of the GNU General Public License version 2 as published by
 * the Free Software Foundation.
 *
 * Description: Hello World test application
 *
 */

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

int main( void )
{

	int cnt;

	for( cnt = 0; cnt < 10; cnt++ ) {
		printf( "%d Hello John!\n", cnt );
		sleep(1);
	}

	return 0;
}

```

Use the **Outline** view to inspect the structure of the file currently open in the editor area, and navigate through the source code. In the **Outline** view, double-click **main**. The editor highlights the **main** function.

This is a very simple example; the **Outline** view becomes much more useful when working with larger source files. This is how it would look like in a different bigger source file:



Edit the source file to modify the output message of the application. In the **for** loop, change this statement:

```
        printf( "%d Hello John!\n", cnt );
```

To print your name instead:

```
        printf( "%d Hello your_name!\n", cnt );
```

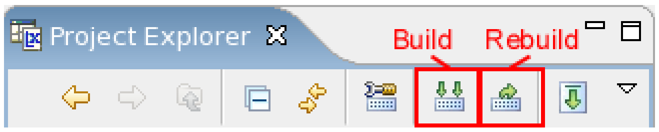
Add a new statement before the **return** line:

```
        printf( "Bye your_name!\n");
```

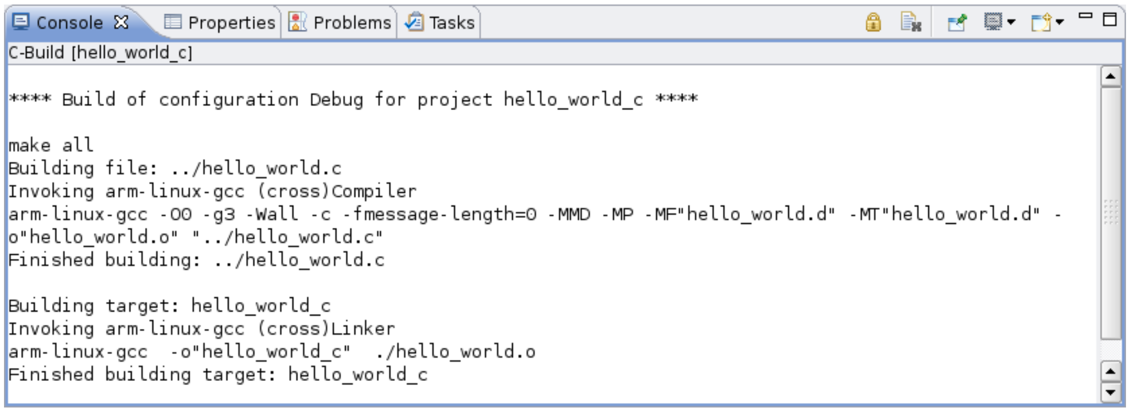
To save these changes to the source file select **File > Save**, or press the shortcut key **Ctrl+S**.

9. Build the sample application

To build the sample application, in the **Project Explorer** view, select the project from the **Projects** tree and click the **Build Project** button on the view's toolbar.



When the build process is launched, the **Console** view displays output messages of the building tools.

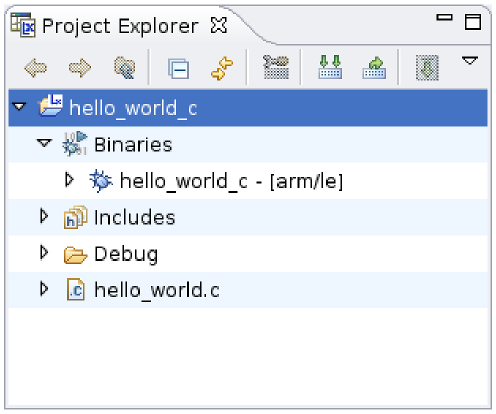


```
**** Build of configuration Debug for project hello_world_c ****

make all
Building file: ../hello_world.c
Invoking arm-linux-gcc (cross)Compiler
arm-linux-gcc -O0 -g3 -Wall -c -fmessage-length=0 -MMD -MP -MF"hello_world.d" -MT"hello_world.d" -
o"hello_world.o" "../hello_world.c"
Finished building: ../hello_world.c

Building target: hello_world_c
Invoking arm-linux-gcc (cross)Linker
arm-linux-gcc  -o"hello_world_c"  ./hello_world.o
Finished building target: hello_world_c
```

The **Problems** view lists any warnings and errors that occurred during the build process. In such case, the **C/C++ Editor** marks the lines where the problems were detected. The sample application project should build successfully, indicated by the message:

```
Finished building target: hello_world.c
```

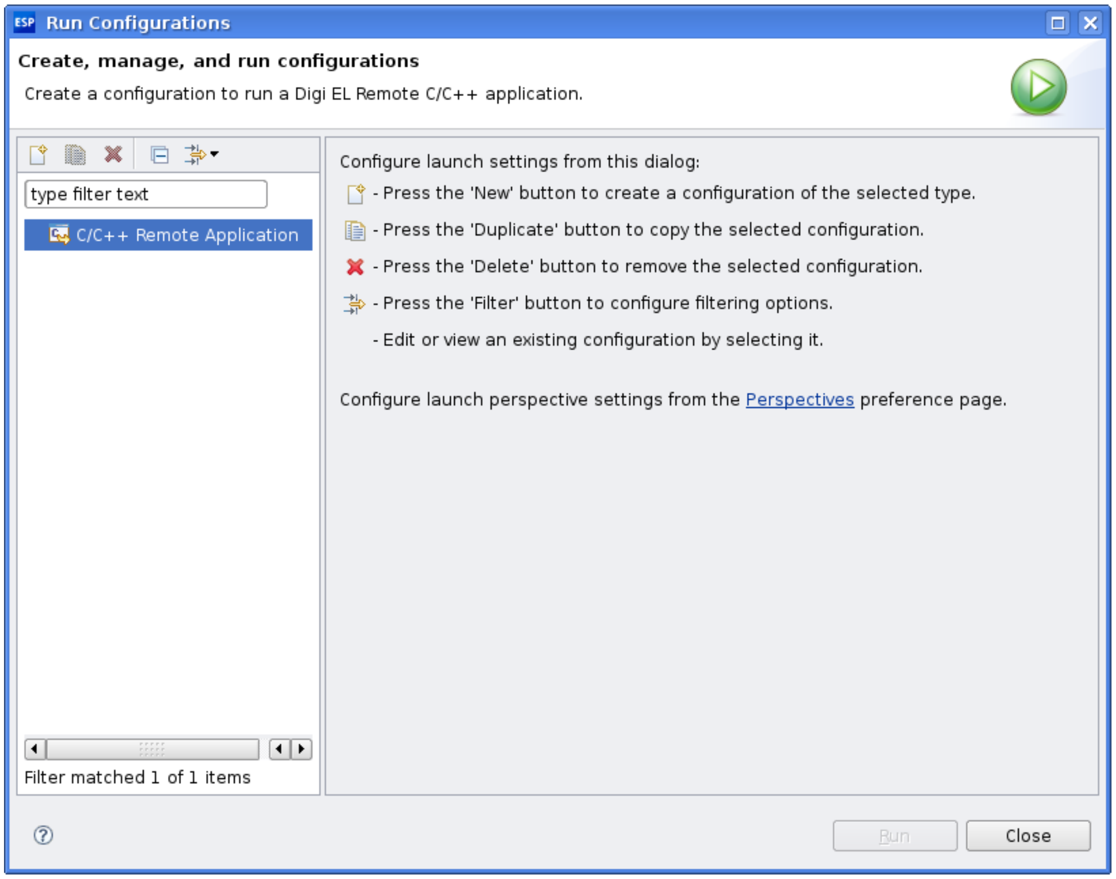
Expand the **Binary** object in the **Project Explorer** view. The executable image file is displayed.



10.Run the sample application in connection with the target

Once the sample project is built without any problem, launch it remotely. To do so, create a *launch configuration* to set the launch options of the application.

Select **Run > Run Configurations…**; the **Run** dialog is displayed.



Create, manage, and run configurations

Create a configuration to run a Digi EL Remote C/C++ application.

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 1 of 1 items

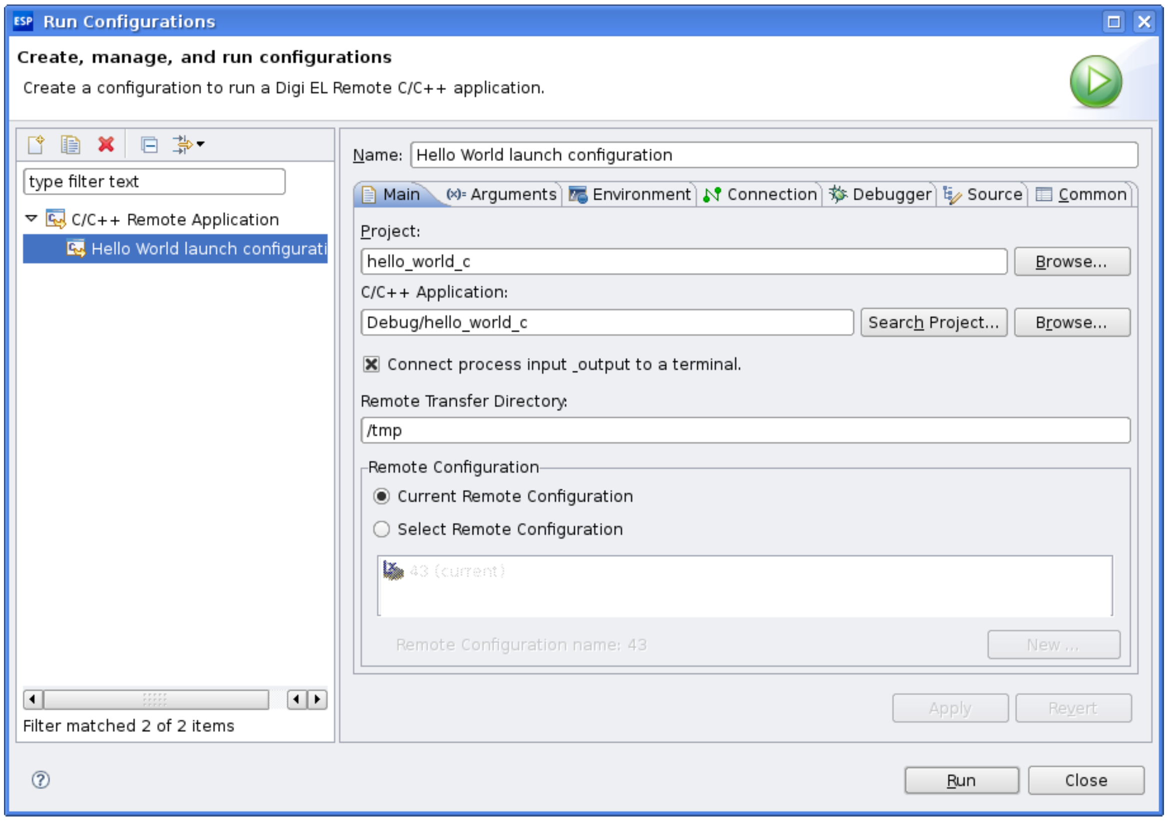
Double-click **C/C++ Remote Application** to create a new empty configuration.

In the **Name** text box, type the name of the launch configuration. For example, **Hello World launch configuration**.

Below the **Name** field, the launch configuration parameters are organized on several tabs.

From the **Main** tab select the name of the project to run, **hello_world_c**, and the relative path of the executable to launch, **Debug/hello_world_c**. In the **Remote Configuration** section, select **Current Remote Configuration**.

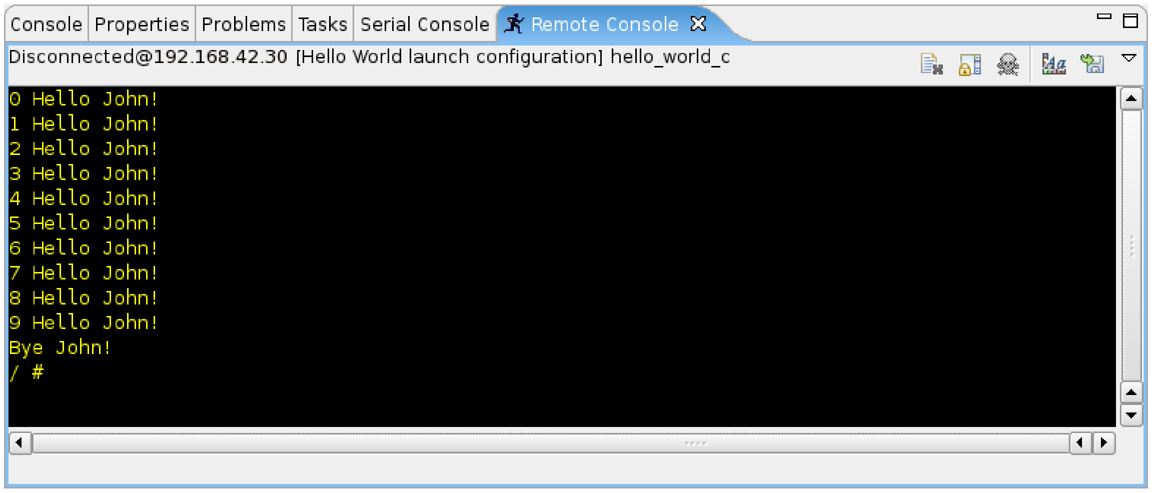
Use the default values for the rest of the parameters.


Run Configurations
Create, manage, and run configurations
Create a configuration to run a Digi EL Remote C/C++ application.
type filter text
C/C++ Remote Application
Hello World launch configurati
Filter matched 2 of 2 items
Name: Hello World launch configuration
Main
Arguments
Environment
Connection
Debugger
Source
Common
Project:
hello_world_c
Browse...
C/C++ Application:
Debug/hello_world_c
Search Project...
Browse...
Connect process input _output to a terminal.
Remote Transfer Directory:
/tmp
Remote Configuration
Current Remote Configuration
Select Remote Configuration
43 (current)
Remote Configuration name: 43
New ...
Apply
Revert
Run
Close

Click **Run** to launch the application.

The application is automatically transferred to the target using FTP, because FTP is the transfer mechanism selected on the current remote configuration.

The **Remote Console** view is opened to display the input/output of the application running on the target device.



Disconnected@192.168.42.30 [Hello World launch configuration] hello_world_c

```
0 Hello John!
1 Hello John!
2 Hello John!
3 Hello John!
4 Hello John!
5 Hello John!
6 Hello John!
7 Hello John!
8 Hello John!
9 Hello John!
Bye John!
/ #
```

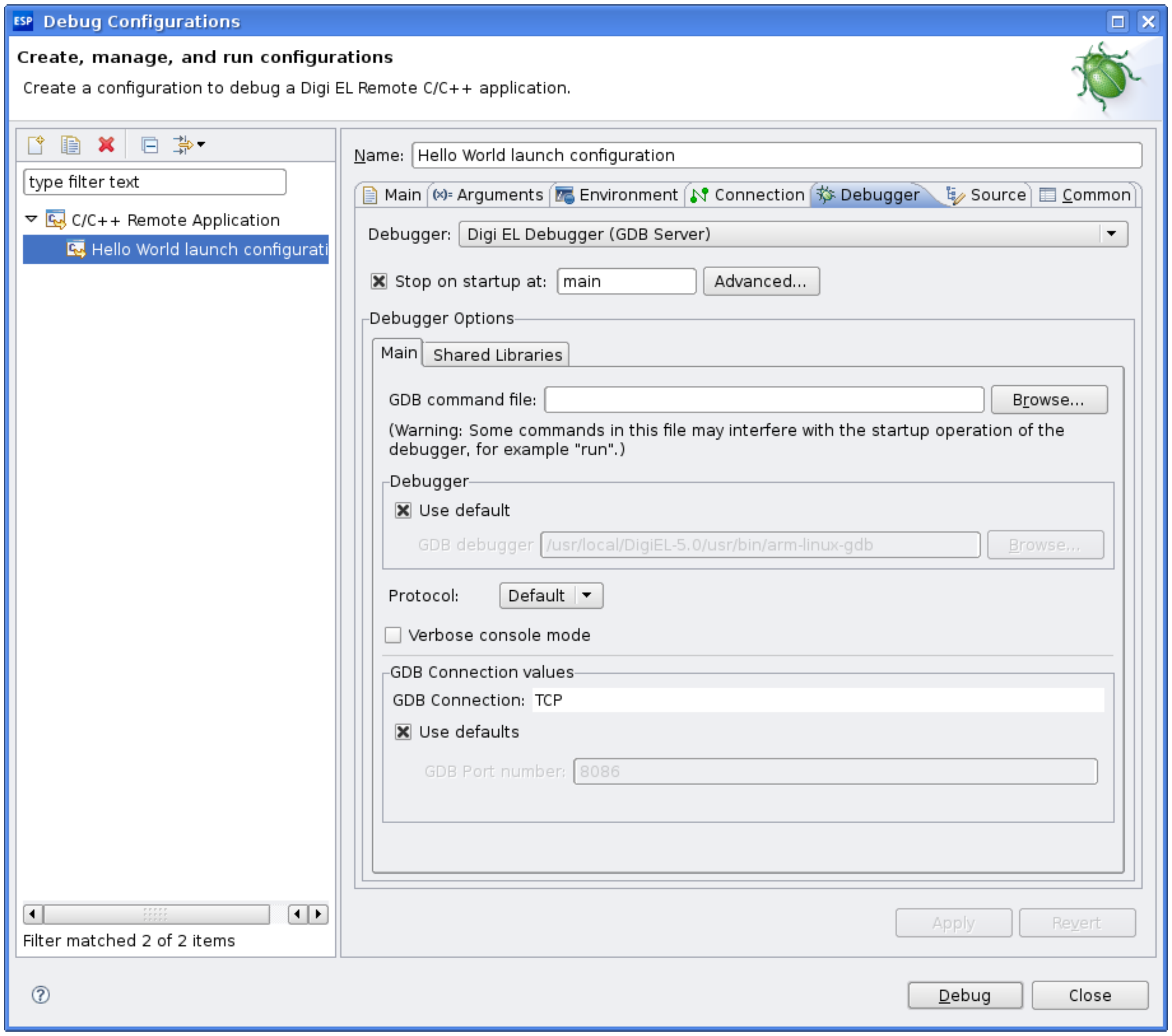
If the Remote Console view is closed while the application is running on the target, the connection is lost and the application is terminated.

11.Debug the application

Debugging using Digi ESP is very easy. Debugging the application requires a launch configuration. Select menu **Run > Debug Configurations...**.

The Debug dialog is displayed. Select the **C/C++ Remote Application** category, and the **Hello World launch configuration** that was created in the previous section.

From the **Debugger** tab, make sure that **Digi EL Debugger (GDB Server)** is selected and that **Stop on startup** is checked to force the debugger to stop execution at the main function when the application is launched.


Debug Configurations

Create, manage, and run configurations

Create a configuration to debug a Digi EL Remote C/C++ application.

type filter text

C/C++ Remote Application

Hello World launch configurati

Filter matched 2 of 2 items

Name: Hello World launch configuration

Main | Arguments | Environment | Connection | Debugger | Source | Common

Debugger: Digi EL Debugger (GDB Server)

[✓] Stop on startup at: main Advanced...

Debugger Options

Main | Shared Libraries

GDB command file: Browse...

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

Debugger

[✓] Use default

GDB debugger /usr/local/DigiEL-5.0/usr/bin/arm-linux-gdb Browse...

Protocol: Default

[ ] Verbose console mode

GDB Connection values

GDB Connection: TCP

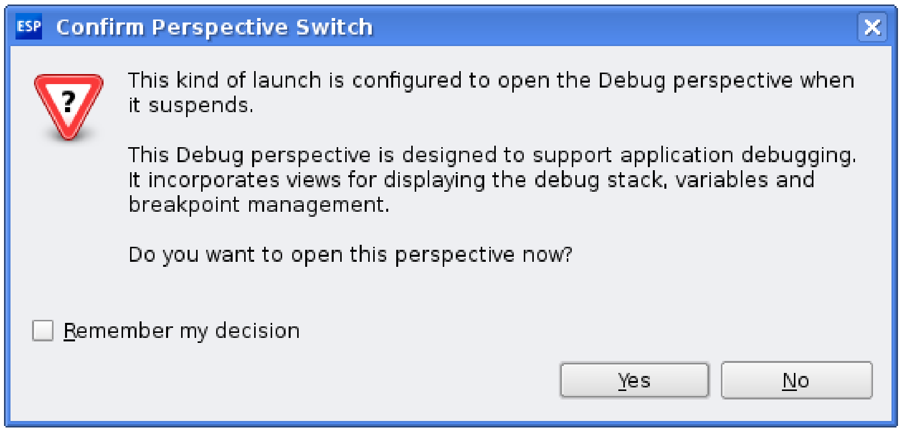
[✓] Use defaults

GDB Port number: 8086

Apply Revert

Debug Close

Click **Debug** to start the debug process. A prompt is displayed asking whether to switch to the **Debug** perspective.


Confirm Perspective Switch

This kind of launch is configured to open the Debug perspective when it suspends.

This Debug perspective is designed to support application debugging. It incorporates views for displaying the debug stack, variables and breakpoint management.

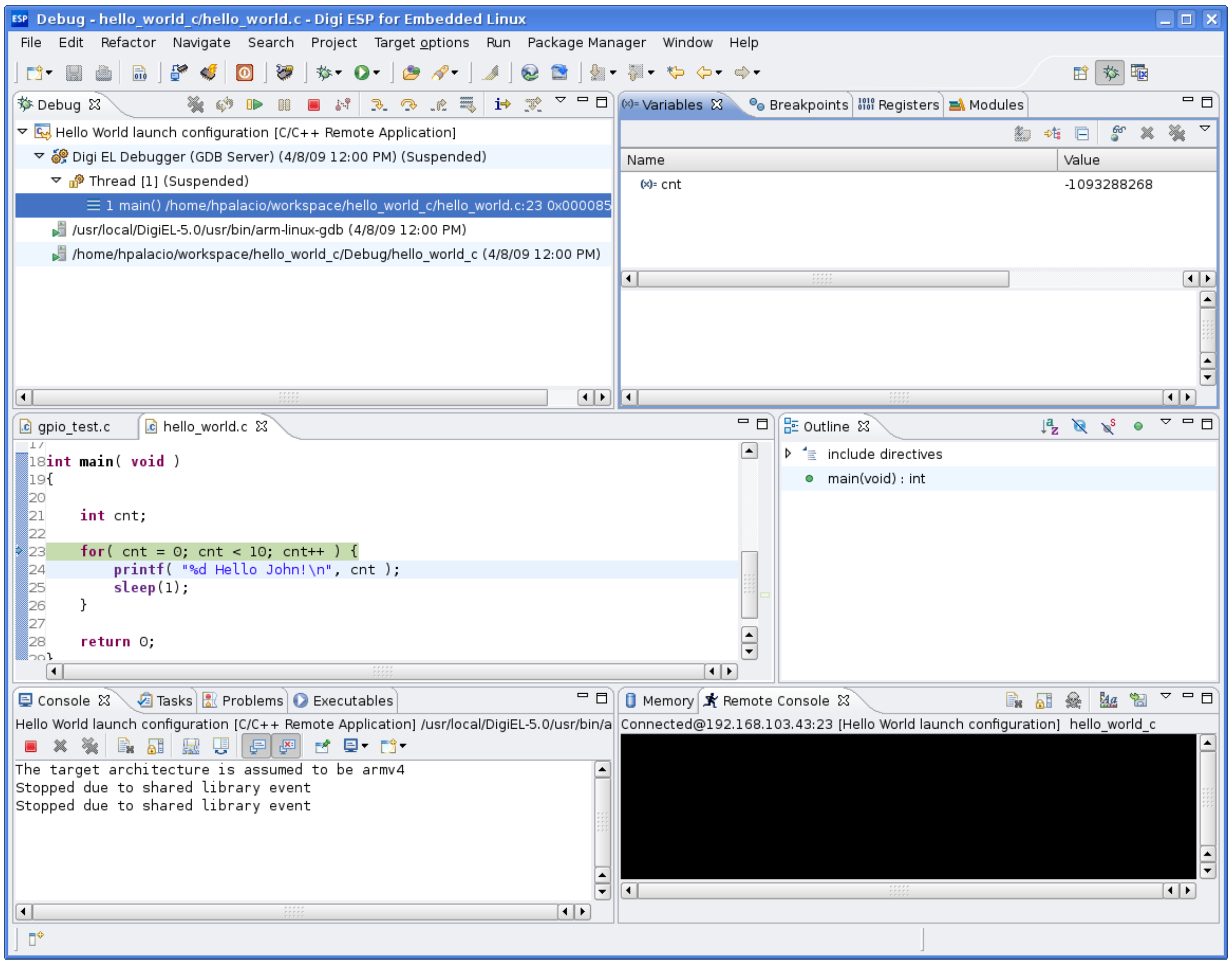
Do you want to open this perspective now?

[ ] Remember my decision

Yes No

Click **Yes**.

The **Debug** perspective has a layout oriented to debugging.


Debug - hello_world_c/hello_world.c - Digi ESP for Embedded Linux
File Edit Refactor Navigate Search Project Target options Run Package Manager Window Help
Debug
Variables
Breakpoints
Registers
Modules
gpio_test.c
hello_world.c
Outline
Console
Tasks
Problems
Executables
Memory
Remote Console

```
Hello World launch configuration [C/C++ Remote Application]
  Digi EL Debugger (GDB Server) (4/8/09 12:00 PM) (Suspended)
    Thread [1] (Suspended)
      1 main() /home/hpalacio/workspace/hello_world_c/hello_world.c:23 0x000085
  /usr/local/DigiEL-5.0/usr/bin/arm-linux-gdb (4/8/09 12:00 PM)
  /home/hpalacio/workspace/hello_world_c/Debug/hello_world_c (4/8/09 12:00 PM)
```

Name	Value
cnt	-1093288268

```
int main( void )
{

    int cnt;

    for( cnt = 0; cnt < 10; cnt++ ) {
        printf( "%d Hello John!\n", cnt );
        sleep(1);
    }

    return 0;
}
```

```
include directives
main(void) : int
```

```
Hello World launch configuration [C/C++ Remote Application] /usr/local/DigiEL-5.0/usr/bin/a
The target architecture is assumed to be armv4
Stopped due to shared library event
Stopped due to shared library event
```

```
Connected@192.168.103.43:23 [Hello World launch configuration] hello_world_c
```

The application is remotely launched and stops at the main function's first line of code.

To step through the instructions, press the **F6** key.

To insert a breakpoint, double-click on the left border of any code line on the Editor. To remove a breakpoint, double-click on it again.

Other debugging tasks include inspecting variables and expressions, displaying memory and registers, and showing the disassembled code interleaved with the source code. These tasks are covered in the *Digi ESP Online Help*.

12.What's next?

Congratulations!

You have created, built, transferred, run, and debugged your first application for the target device.

Now you can use Digi ESP to develop the Linux kernel, the rootfs, complex applications, and libraries to create a powerful system for your target device.

To get the most of Digi ESP for Embedded Linux, read the *Digi ESP Online Help*, accessible from menu **Help > Help Contents > Digi ESP for Embedded Linux**.