This document provides an exercise using Digi JumpStart for Windows Embedded CE 6.0. This document shows how to develop, run, and debug a simple application on your target hardware platform. This tutorial takes about **30 minutes**. For detailed information, see *Digi JumpStart for Windows Embedded CE 6.0 User's Guide* integrated in Visual Studio **Help > Contents**.

Complete all the tasks in this guide in the order presented.

Conventions in this tutorial

This document 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	For menu items, dialogs, tabs, buttons, and other controls.
	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.
Menu name > option	A menu followed by one or more options; for example, **File > New**.

This manual also uses these frames and symbols:

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.
```

1. Introduction

1.1. Software and hardware requirements before beginning this tutorial

The instructions in this guide assume that the Digi JumpStart for Windows Embedded CE software is installed on the host computer, and the host computer is connected to the development board. If the software is not installed or the hardware connected, do both now.

See your network administrator for this information needed to configure the target's network settings:

- IP address of the development computer
- IP address to use for the target
- Netmask for the Ethernet interface
- Netmask for the wireless interface (ConnectCore Wi-9C and ConnectCore Wi-9M 2443 only)
- IP address to use for the target's WLAN interface, ask the network administrator for a free IP address for the wireless connection (ConnectCore Wi-9C and ConnectCore Wi-9M 2443 only).

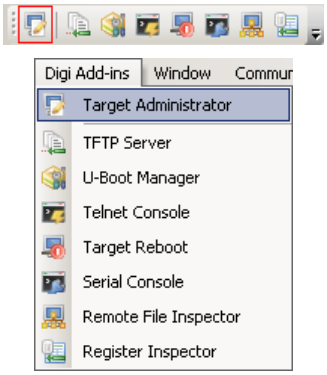
2. Configure the IP addresses and network settings

This task opens and configures the Target Administrator, powers up the target, and configures the target's network settings.

2.1. Open Target Administrator

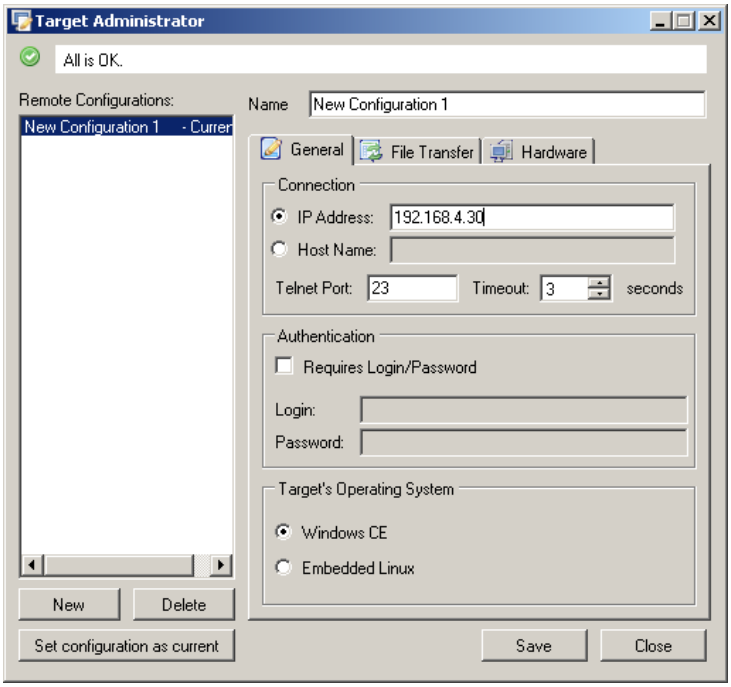
For Visual Studio to communicate with the target a new remote configuration–a set of configuration options for a specific target–is needed.

To create a new remote configuration, select **Digi Addins > Target Administrator.** Or click in the **Target Administrator** icon:



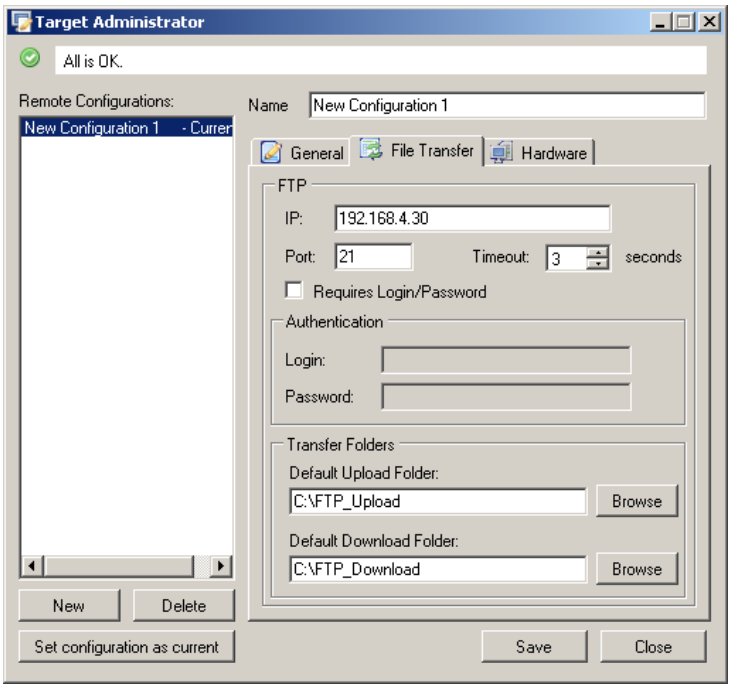
Remote configuration settings are organized on three tabs:

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

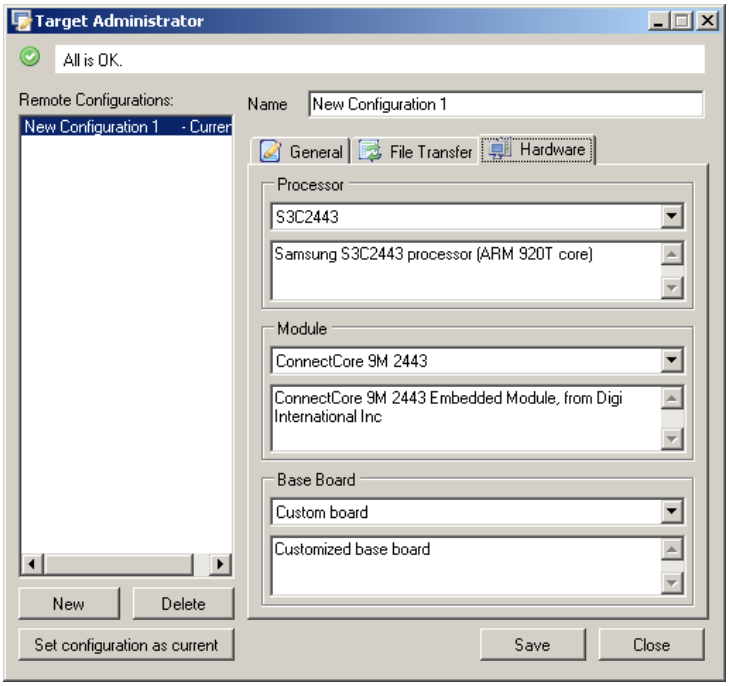


Click on the **File Transfer** tab to configure the file transfer mechanism between Visual Studio and the embedded system. If you check **Requires Login/Password** checkbox you must file Login and Password fields, if not, values will set to "anonymous".

You can also select your local FTP Upload and Download folders, using by Remote File Inspector.



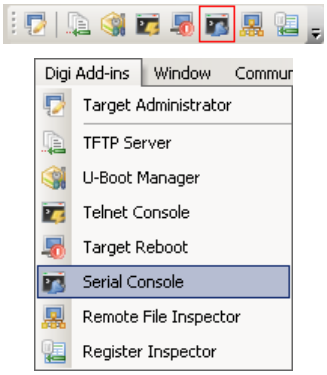
Click on the **Hardware** tab to identify the hardware components of the target device. Depending on the model type of the development board, select the appropriate values for Processor, Module, and Base Board



Click the Save **button** to save the configurations made and then click the **Set configuration as current** button to make this remote configuration the current configuration used by Visual Studio.

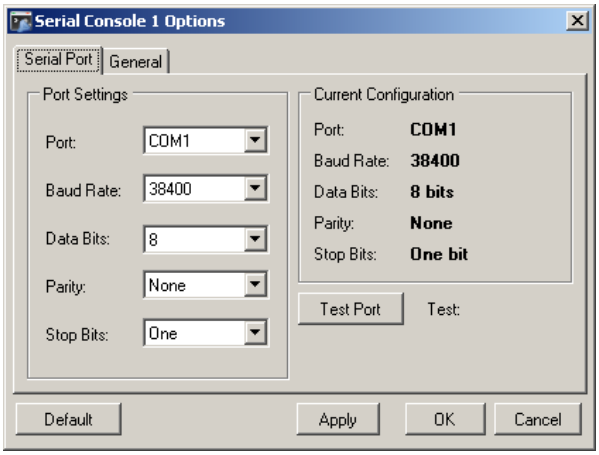
2.2. Open and configure the Serial Console view

The target board prints out messages on the serial port. To open the **Serial Console** view in Visual Studio, select **Digi Addins > Serial Console** or click in the **Serial Console** icon:



Open the **Serial Console Options** dialog by clicking the **Serial Console Options** button on the **Serial Console** view's toolbar.

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.



Once the serial port is configured, establish the connection by clicking the **Open Port** button of the **Serial Console** view's toolbar.

2.3. Power up the target

Power up the target using the main power on the development board.

Turn on the switch **SW5** on the ConnectCore 9C/Wi-9C development board, or turn on the switch **S2** on the ConnectCore 9P 9360 development board or on the ConnectCore 9M/Wi-9M 2443 development board .

For more information about the position of the switch, see the Development Board chapter of the ConnectCore_9C_Wi-9C_Hardware_Reference.pdf, ConnectCore_9P_9360_Hardware_Reference.pdf or ConnectCore_9M_Wi-9M_2443_Hardware_Reference.pdf, *depending the platform being used.*

After power-up, the LEDs on the development board light up. After a few seconds the system will print boot messages in the Serial Console window.

Press any key within four seconds of receiving messages to stop the auto boot process.

2.4. Configure IP addresses and network settings

1. From the boot loader shell, enter these commands:

```
#   setenv ipaddr aaa.aaa.aaa.aaa
#   setenv netmask bbb.bbb.bbb.bbb
#   setenv ipaddr_wlan ccc.ccc.ccc.ccc
#   setenv netmask_wlan ddd.ddd.ddd.ddd
#   setenv serverip eee.eee.eee.eee
#   saveenv
```

In the commands, replace the variables with these actual values:

- ***aaa.aaa.aaa.aaa*** with the IP address to assign to the target.
- ***bbb.bbb.bbb.bbb*** with the target's network mask.
- ***ccc.ccc.ccc.ccc*** with the IP address for the WLAN adapter (ConnectCore Wi-9C and ConnectCore Wi-9M 2443 only).
- ***ddd.ddd.ddd.ddd*** with the wireless network mask (ConnectCore Wi-9C and ConnectCore Wi-9M 2443 only).
- ***eee.eee.eee.eee*** with the IP address of the development computer.

Do not use the equal sign (=) with `setenv`

2. Reset the target. Depending on the development board, press the reset button: on the ConnectCore 9C/Wi-9C development board, press switch **SW6**, on the ConnectCore 9P 9360 and ConnectCore 9M/Wi-9M 2443 development boards, press switch **S1**.

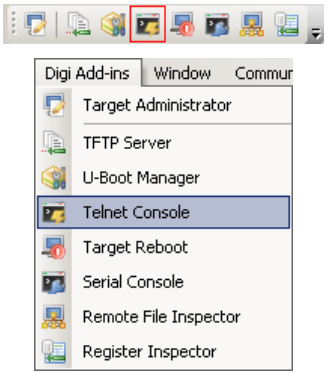
For more information about the position of the switch, see the Development Board chapter of the ConnectCore_9C_Wi-9C_Hardware_Reference.pdf, ConnectCore_9P_9360_Hardware_Reference.pdf or ConnectCore_9M_Wi-9M_2443_Hardware_Reference.pdf, *depending on the platform being used.*

3. Explore networking capabilities

3.1. Open a Telnet Console

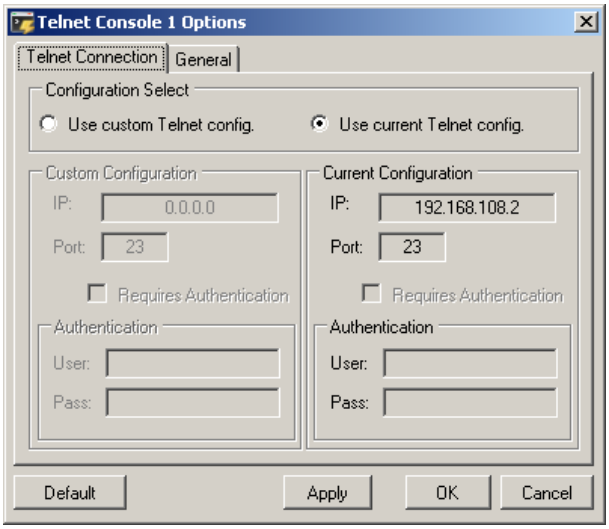
A Telnet client server is included in the Digi Addins, making it possible to open a Telnet session from the development computer.

To open a Telnet console view in Visual Studio, select **Digi Addins > Telnet Console** or click in the **Telnet Console** icon:



Open the **Telnet Console Options** dialog by clicking the **Telnet Console Options** button on the **Telnet Console** view's toolbar.

Select **Use current Telnet configuration** to use the current configuration.



Once the Telnet Console is configured, establish the connection by clicking the **Connect** button of the **Telnet Console** view's toolbar.

To see a list of all available commands, enter:

```
help
```

This command list is displayed:

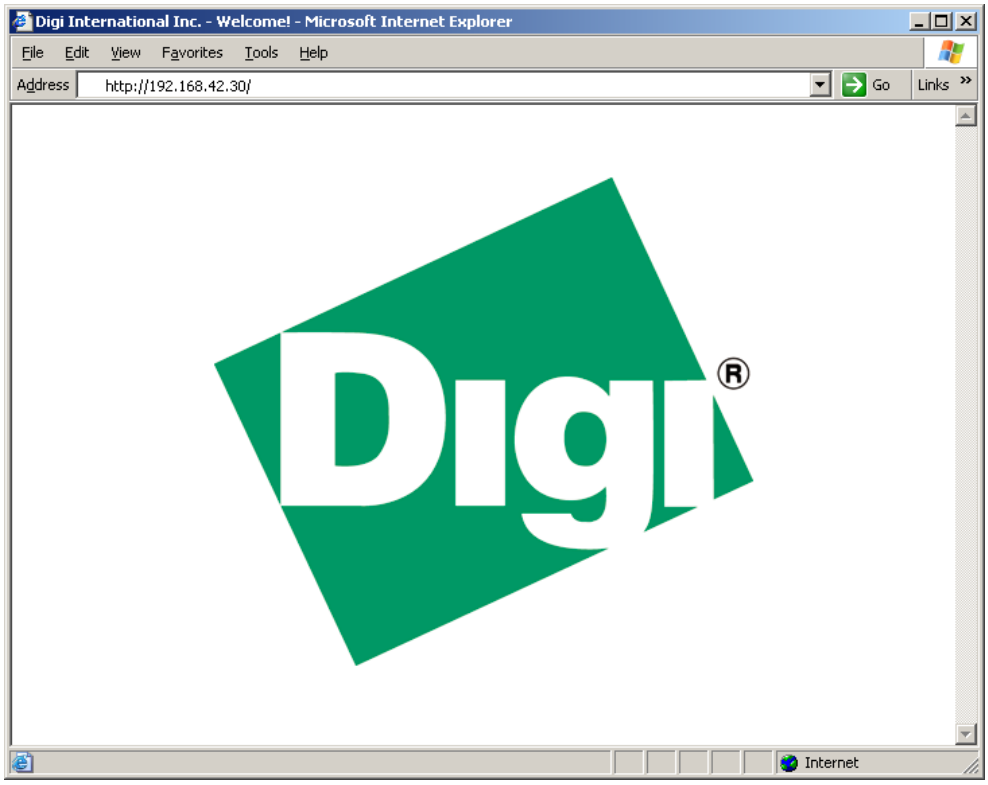
```
The following commands are available:
        ATTRIB  Set/display file attributes.
        CALL    Call batch script.
        CD      Change directory.
        CHDIR   Same as CD.
        CLS     Clear the screen.
        COPY    Copy files.
        DATE    Display/set system date.
        DEL     Delete a file.
        DIR     Print contents of a directory.
        ECHO    Echo output on the screen or change echoing parameters.
        ERASE   Same as DEL.
        EXIT    Exit command interpreter.
        HELP    Print help for command interpreter or individual commands.
        GOTO    Transfer control to a label in batch processing.
        IF      Conditionally execute a command.
        MD      Create a directory.
        MKDIR   Same as MD.
        MOVE    Move/rename files.
        PATH    Alias for SET PATH.
        PAUSE   Suspend execution of a batch file.
        PROMPT  Reconfigure system prompt.
        PWD     Print current working directory.
        RD      Remove directory.
        REM     Record comments in batch file.
        REN     Change file name.
        RENAME  Same as REN.
        RMDIR   Same as RD.
        SET     Set or list environment variables.
        SHIFT   Shift arguments of a batch file.
        START   Start detached process.
        TIME    Display/change system time.
        TITLE   Set the window title for a CMD.EXE session.
        TYPE    Output contents of a file or files to the screen.
Use HELP [command name] to display extended help for given command, or
HELP CMD to display help on general topics such as
command input options, I/O redirection or CMD parameters.
```

3.2. Connect to Web Server

A Web server is included and started by default, and serves a default Windows CE Web page that resides in the target image. To connect to the Web server:

1. Open a browser.
2. In the **Address** box, enter the IP address of the target.

   The default Web page opens:

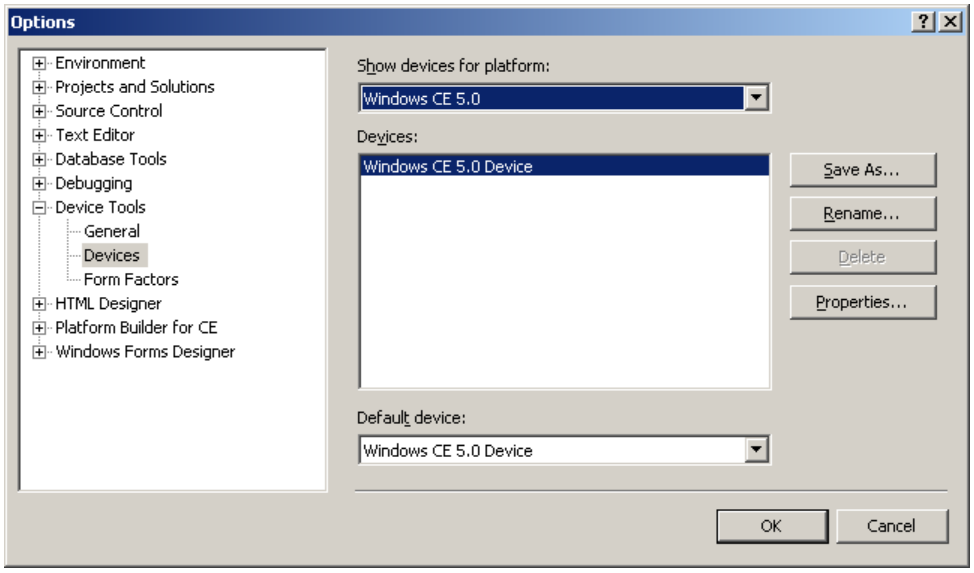


4. Configure device transport

To transfer an application to the target and debug it, a *device transport configuration* must be created. To create a device transport configuration:

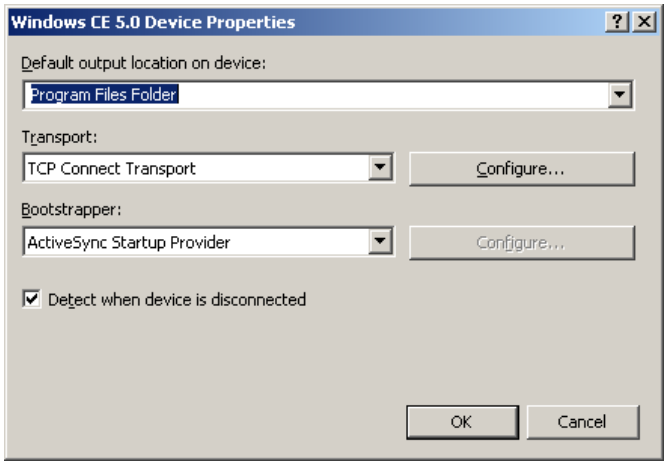
1. Start Visual Studio® 2005.
2. Select **Tools > Options**.

   The **Options** dialog opens.
3. Select **Device Tools > Devices**.
4. From the Show devices for platform pulldown menu, select **Windows CE 5.0.** (This selection is also valid for Windows Embedded CE 6.0.)
5. In the **Devices** list box, select **Windows CE 5.0 Device**. Under **Default device**, select **Windows CE 5.0 Device**. (This selection is also valid for Windows Embedded CE 6.0.)



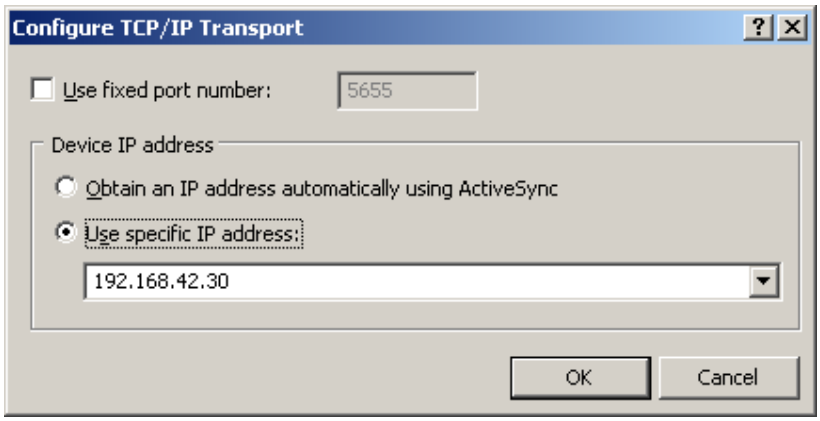
6. Click **Properties**.

   The **Windows CE 5.0 Device Properties** dialog opens:



7. Select these values:
   - **Default output location**: **Program Files Folder**
   - **Transport**: **TCP Connect Transport**
   - **Bootstrapper**: **ActiveSync Startup Provider**
8. At the right of the **Transport** method pulldown menu, click **Configure**.

   The **Configure TCP/IP Transport** dialog opens:



9. Click **Use specific IP address**, enter the target's IP address, and click **OK**.
10. To accept the device transportation configuration, click **OK** in the remaining open dialogs.

Visual Studio 2005 is now ready to transfer and debug any Windows Embedded CE application to the target device.

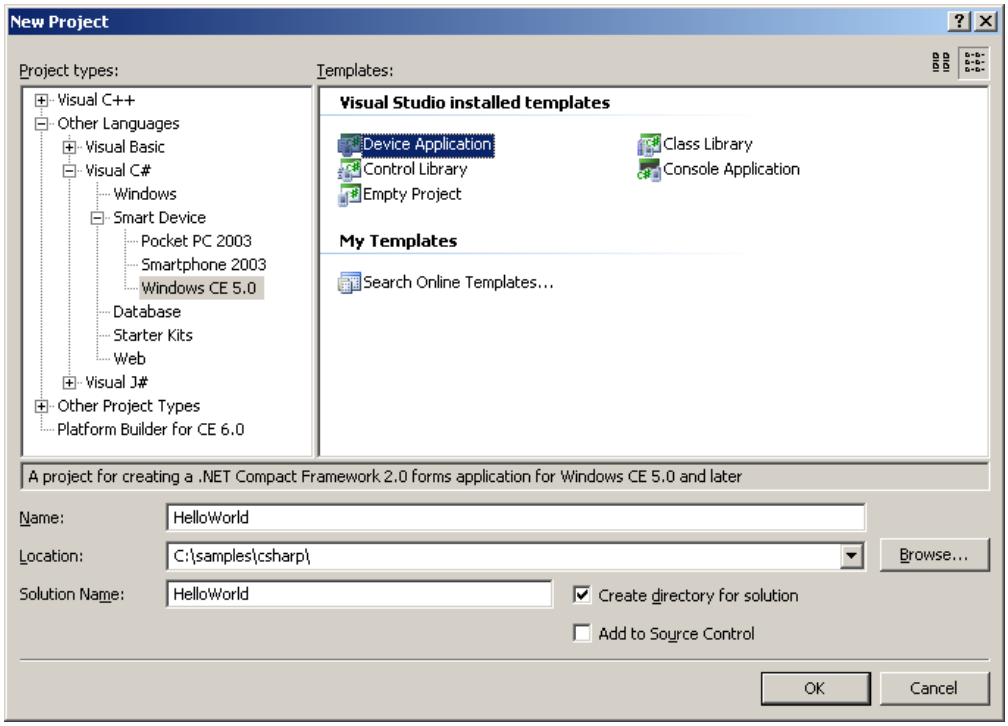
5. Develop an application with Visual Studio 2005

This task creates and builds a simple **HelloWorld** program using Visual C#®. The project will have an empty form and basic source files.

5.1. Create the project

1. Select **File > New > Project**.

   The **New Project** dialog box opens:



2. Do these steps:

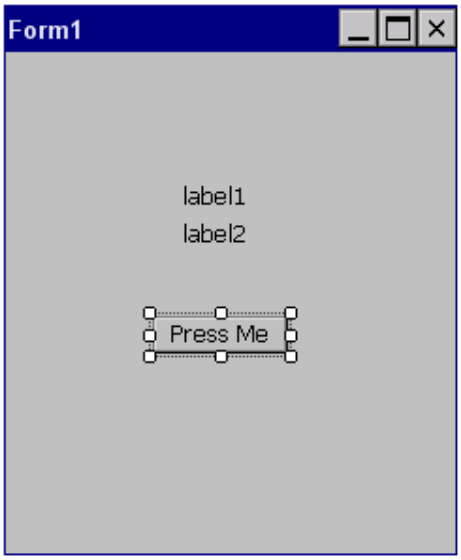
   - In the **Project Types** tree select **Visual C#** (if Visual C# is not your default language in Visual Studio, expand **Other Languages > Visual C#**) select **SmartDevice**, and **Windows CE 5.0** (This selection is also valid for Windows Embedded CE 6.0.)
   - In the **Templates** section, select **Device Application**.
   - In the bottom part of the dialog box, in the **Name** input box, enter **HelloWorld** for project name.
   - In the **Location** input box, enter the location for the source files.

   When everything is entered, click **OK**.

5.2. Generate the interface

Now add some information in the form to generate the interface:

1. To open the toolbox, select **View > Toolbox**.
2. Drag and drop a button into the empty form.
3. Drag and drop two labels into the form so the interface looks like this:



4. Right-click the button and select **Properties**. Then, in the **Appearance > Text** field, enter **Press Me** as the new text. Leave the default text for the labels as they are.

5.3. Generate the source code

Next, add some code in the button's click method.

1. To open the button's click method source code, double-click the button on the form. Code similar to this is displayed:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace HelloWorld
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {

        }
    }
}
```

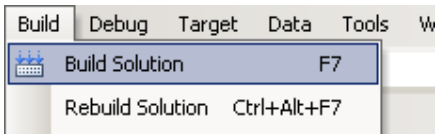
2. Set **label1** to display **Hello World!** and **label2** to display a counter's value that increases with each click of the button. Enter:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace HelloWorld
{
    public partial class Form1 : Form
    {
        int counter = 1;
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            label1.Text = "Hello World!";
            label2.Text = counter.ToString();
            counter++;
        }
    }
}
```

3. Save the file.

5.4. Build the sample application

To build the sample application, select **Build > Build Solution**:



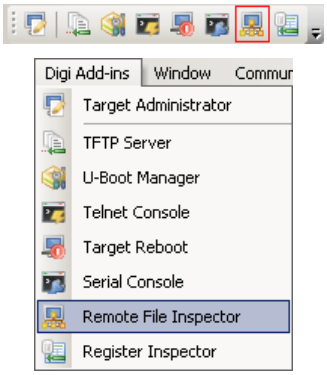
When the build completes, messages in the output window (**View > Output**) indicate:

- That the build process was successful
- The location of the executable image

6. Transfer the application to the target using Remote File Inspector

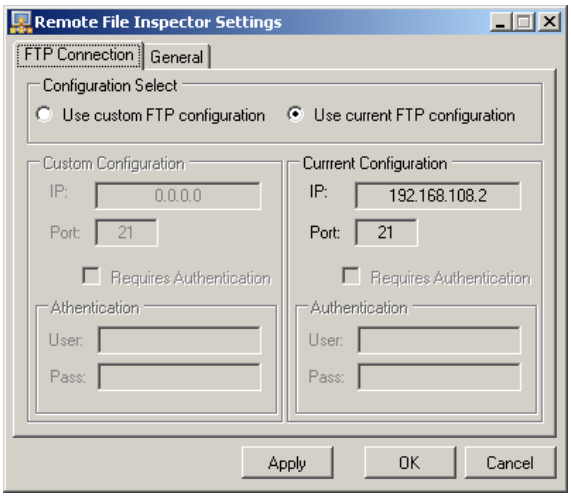
Now use the Remote File Inspector to transfer the application executable into the embedded platform's RAM:

1. Open the Remote File Inspector, select Digi Addins > Remote File Inspector or click in the Remote File Inspector icon:



2. Open the **Remote File Inspector** settings dialog by clicking the **Remote File Inspector** settings button on the Telnet Console view's toolbar.

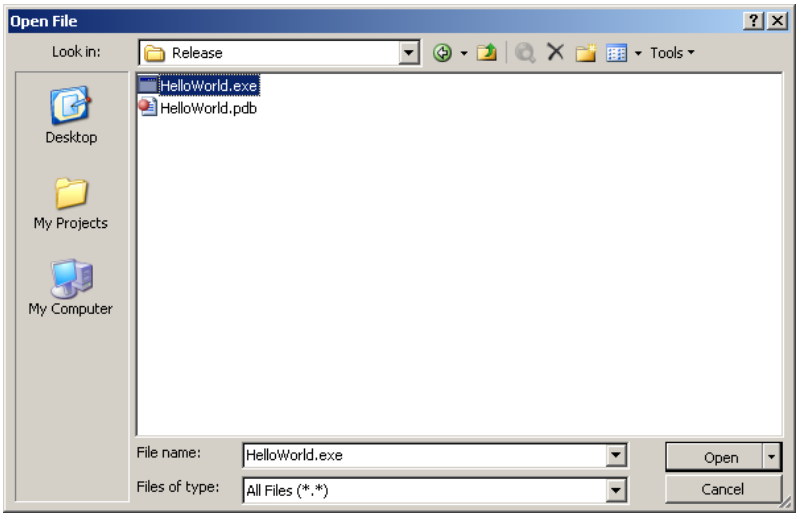
Select **Use current FTP configuration** to use the current configuration.



3. Connect the Remote File Inspector by clicking the **Connect** button

4. Upload the file by clicking the **Upload** button.

5. Select the application binary surfing through the browser.



6. Click **Open** and the file will be uploaded into the embedded platform.

7. Run the application

Now go to the target device and run the application.

1. On the target's desktop, double-click **My Device**, and the double-click **HelloWorld**.
2. Click the **Press Me** button several times.

   label1 displays **Hello World!**, and **label2** displays the number of times the **Press Me** button was pressed:

3. Close the application by clicking the **X** on the application's title bar.

8. Debug the application

8.1. Establish the debug connection

1. In Visual Studio 2005, select **Tools > Connect to Device**.
2. Select **Windows CE 5.0 Device** (also valid for Windows Embedded CE 6.0), and click **Connect**. A dialog opens and reports whether the connection was successful.
3. Close the dialog.

8.2. Begin to deploy and debug the application

Select **Debug > Start Debugging**, and select **Windows CE 5.0 Device** again.

(To keep this dialog from opening, uncheck **Show me this dialog every time I deploy the application.**)

Then click **Deploy**.

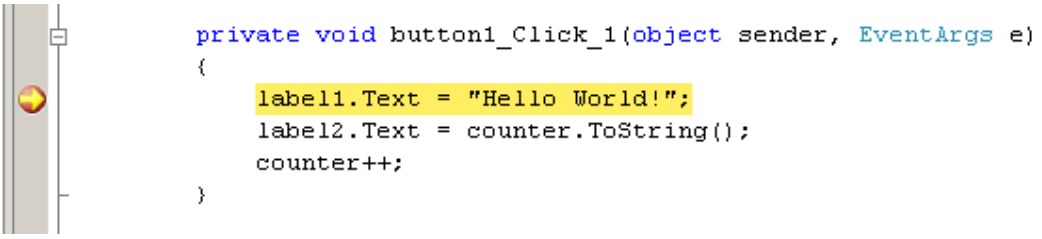
This step transfers the application to the target and runs it for debugging. The Visual Studio perspective changes to Debug mode.

8.3. Add/remove breakpoints

To add a breakpoint, in any line of code, either click the left border of the editor or press F9. A red circle appears where the code was clicked.

To remove a breakpoint, click again.

Try adding a breakpoint to any line in the button's click method. Then, in the target, click the button. The code stops at the breakpoint; the yellow arrow indicates the line where the program counter is.



8.4. Other debugging controls

Other debugging controls and options accessible from the **Debug** menu include watching and modifying variables, stepping over or into the code, viewing and editing memory locations. For information about these and other debugging controls, see the *Digi JumpStart for Windows Embedded CE 6.0 User's Guide*.

8.5. What's next?

Congratulations – you have created, built, run, and debugged your first application for the target device. Now you are ready to do more real development of the Windows CE kernel and complex applications to create a powerful system for your device.

To learn more about Windows Embedded CE 6.0, see the **Digi JumpStart for Windows Embedded CE 6.0 User's Guide** integrated in Visual Studio **Help > Contents**.

For documentation on your module please see the Digi Web site at *www.digiembedded.com*. For Windows CE specific documentation please see the Visual Studio 2005 integrated help. To

navigate to the integrated help start Visual Studio 2005 then go to 'Help -> Contents'. Next, under the 'Filtered by:' drop down menu, select '(unfiltered)' then see the section entitled 'Digi Jumpstart for Windows Embedded CE 6.0 User's Guide'.