Appendix D

Tutorial 3 — Physical Implementation in a Programmable Logic Device

In this tutorial we focus on the physical implementation of a design project in a target device. We show how to manually choose which pins on a device package are used for the input and output signals in a circuit, and we describe how to use the Quartus II Programmer module to transfer a compiled design project into the selected PLD chip.

D.1 Making Pin Assignments

In the examples given in Tutorial 2, the assignment of signals to device pins was done automatically by the Compiler. In some cases the designer needs to be able to manually specify which pins to use for some of the signals in a circuit. For example, the circuit board that contains the chip(s) being used may have hardwired connections from some of the device pins to other components, such as switches or LEDs. To make use of the hardwired connections, the designer has to be able to specify which device pins signals should be assigned to.

In section C.1.4 we described how to examine the compilation results by using the Floorplan Editor tool. Figure C.8 presented the top view of the chip package and showed the assignments of signals $x3$ and f to pins 4 and 12, respectively. In section D.1.3 we will show how the pin assignments can be changed by using the Floorplan Editor. Quartus II provides several ways of making pin assignments, and we will first describe a method that uses the Assignments dialogue.

To assign pins manually, it is necessary to specify which chip to use. This was already done in section C.1.1, when we selected the EPM7128SLC84-7 as shown in Figure C.2. Open again the project *example_verilog*, which was done in section C.1. Select Assignments | Assign Pins to open the window in Figure D.1. The box labeled Available Pins & Existing Assignments lists all of the device package pins and it displays assignments after they have been made. As an example, we will assign the inputs $x1$, $x2$, and $x3$ to pins 9, 10, and 11. As indicated in Figure D.1, scroll down until these pins are visible and click on pin 9 to highlight it. In the Assignment box click on the browse... button next to the Pin name box. This operation opens the Node Finder window shown in Figure D.2.

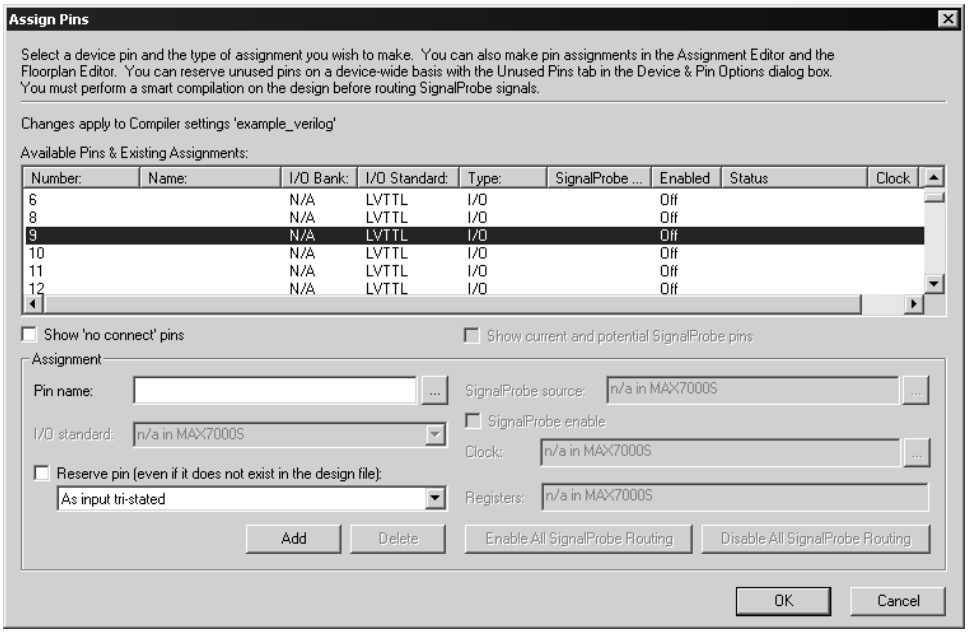


Figure D.1. The Assign Pins dialogue.

Set the Filter to Pins: all and click List to search for pins. Click on the $x1$ input pin and then click on the $>$ button to move this pin to the Selected Nodes box. Click OK to return to the Assign Pins dialogue. In the window in Figure D.1 click on the Add button. This action causes $x1$ to appear in the Name column beside pin 9, and substitutes a Change button in place of the Add button, as depicted in Figure D.3.

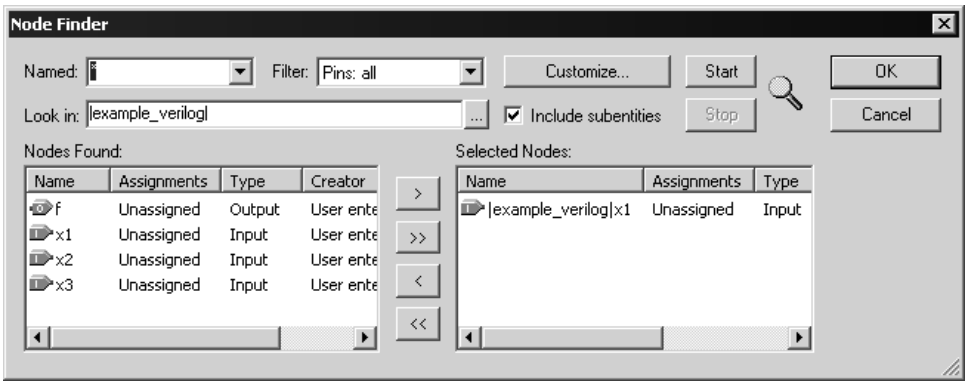


Figure D.2. The Node Finder Window.

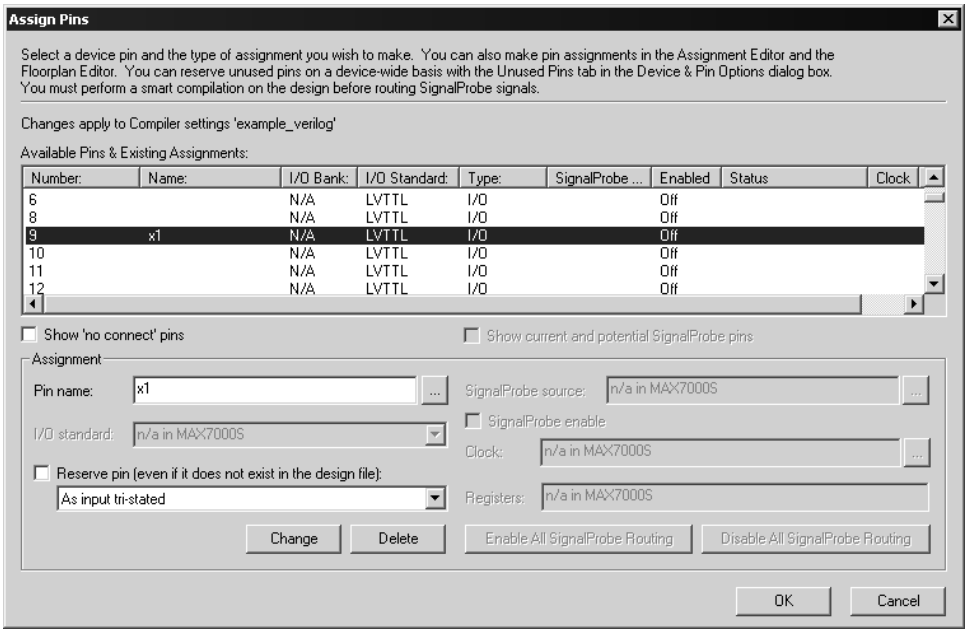


Figure D.3. The assignment of input $x1$ to pin 9.

Following the same procedure, click on pin 10 and assign the $x2$ input, then click on pin 11 and assign the $x3$ input. The Available Pins & Existing Assignments box should now show the assignments given in Figure D.4. Click OK to close the Assign Pins window. At this point the Assignments dialogue is the active window in the Quartus II display. Although we did not open this dialogue directly, it was opened implicitly when we selected the Assign Pins command. To complete the pin assignment procedure you have to select OK to close the Assignments window. Clicking Cancel to close the Assignments window would discard the pin assignments.

Number:	Name:	I/O Bank:	I/O Standard:	Type:	SignalProbe ...	Enabled	Status
6		N/A	LVTTL	I/O		Off	
8		N/A	LVTTL	I/O		Off	
9	x1	N/A	LVTTL	I/O		Off	
10	x2	N/A	LVTTL	I/O		Off	
11	x3	N/A	LVTTL	I/O		Off	
12		N/A	LVTTL	I/O		Off	

Figure D.4. The pin assignments for inputs $x1$, $x2$, and $x3$.

Since we have not recompiled the *example_verilog* project, the compilation results have not yet been affected by our pin assignments. At this point, the assignments are stored internally by Quartus II. When the project is closed the assignments are permanently stored in a file with extension .qsf, which stands for *Quartus settings file*. Select File | Save Project to cause Quartus II to update this file, and then use File | Open to examine the *example_verilog.qsf* file in the Text Editor. Scroll through this file, or use Edit | Find, to locate the three pin assignments, which have the form

```
set_location_assignment Pin_9   -to x1
set_location_assignment Pin_10 -to x2
set_location_assignment Pin_11 -to x3
```

You can modify, add, or delete pin assignments by editing this file, but this is not recommended because it is easy to make an error in the required syntax.

D.1.1 Examining Pin Assignments with the Floorplan Editor

As we mentioned before, it is possible to see the pin assignments that result after compilation by using the Floorplan Editor tool. Quartus II also provides a floorplan tool that can display both the pin assignments produced from the last compilation and the user's pin assignments that have not yet been compiled. Select Assignments | Timing Closure Floorplan to open the Floorplan Editor tool, leading to the window in Figure D.5. If you do not have the device package view selected, click on View | Package Top.

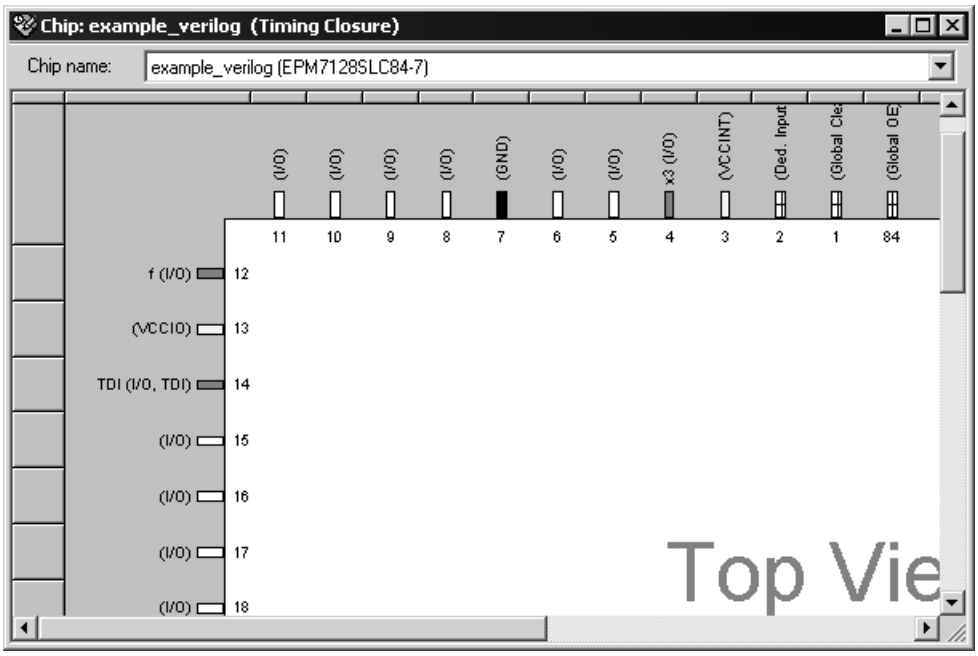
Under the View | Assignments menu there are two settings: Show User Assignments, and Show Fitter Placements. If the first setting is active, then the user's assignments, such as the pin assignments made in section D.1 are displayed; if the second setting is active, then the last compilation results are displayed. Since the two settings are independent you can choose to see both types of assignments, one at a time, or neither. In of Figure D.5a we activated only the setting View | Assignments | Show Fitter Placements, which gives the same picture displayed in Figure C.8. Figure D.5b uses the setting View | Assignments | Show User Assignments, so that the pin assignments we made for inputs $x1$, $x2$, and $x3$ are shown. Each type of assignment can also be shown in the device view, instead of a package view. Select View | Interior Cells to experiment with this feature.

D.1.2 Recompiling the Project with Pin Assignments

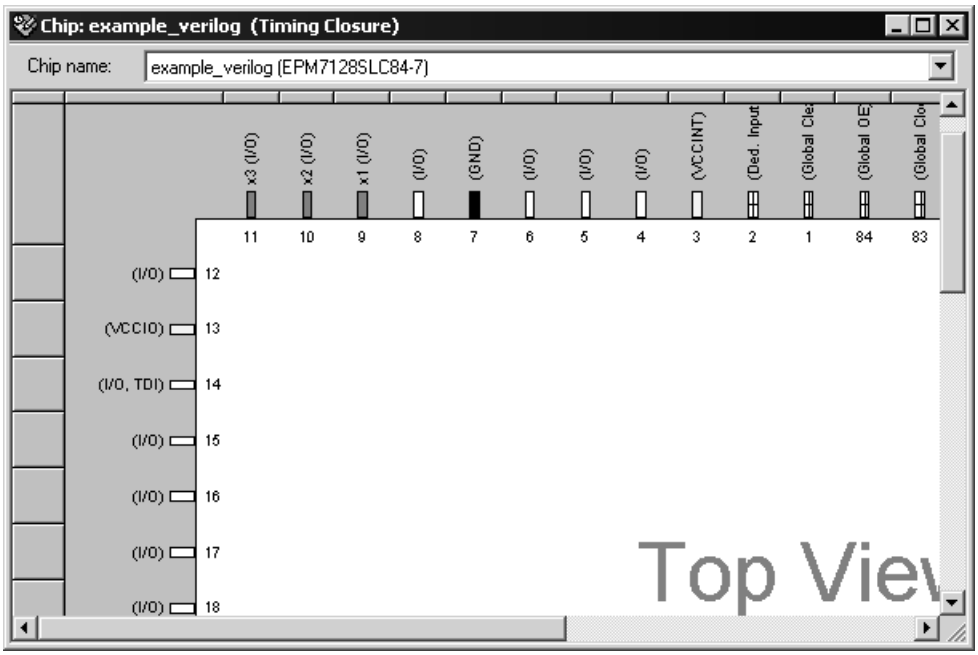
To change the compilation results using our pin assignments, recompile the project. During the compilation process the Fitter uses the pin assignments for the signals that have been specified manually and makes automatic pin assignments for other signals. The Floorplan Editor tool should now display the new pin assignments when View | Assignments | Show Fitter Placements is selected.

D.1.3 Changing Pin Assignments by using the Floorplan Editor

We explained at the beginning of section D.1 how to make pin assignments by using the Assignments | Assign Pins dialogue. Another way to create, or change, pin assignments is to use the Floorplan Editor. If not already done, select Assignments | Timing Closure Floorplan and click on View | Package Top. To see both the compilation results and user assignments at the same time, activate both View | Assignments | Show Fitter Placements and View | Assignments | Show User Assignments. The floorplan display should look similar to the one in Figure D.6. Click on the pin for the output f and drag this pin with the mouse and drop it on pin 15. This operation creates a pin assignment for f. You can select File | Save Project, and then use the Text Editor to see the changed pin assignment in the file *example_verilog.qsf*.



(a) Viewing the Fitter Placements.



(b) Viewing the User Assignments.

Figure D.5. Viewing Pin Assignments in the Floorplan Editor.

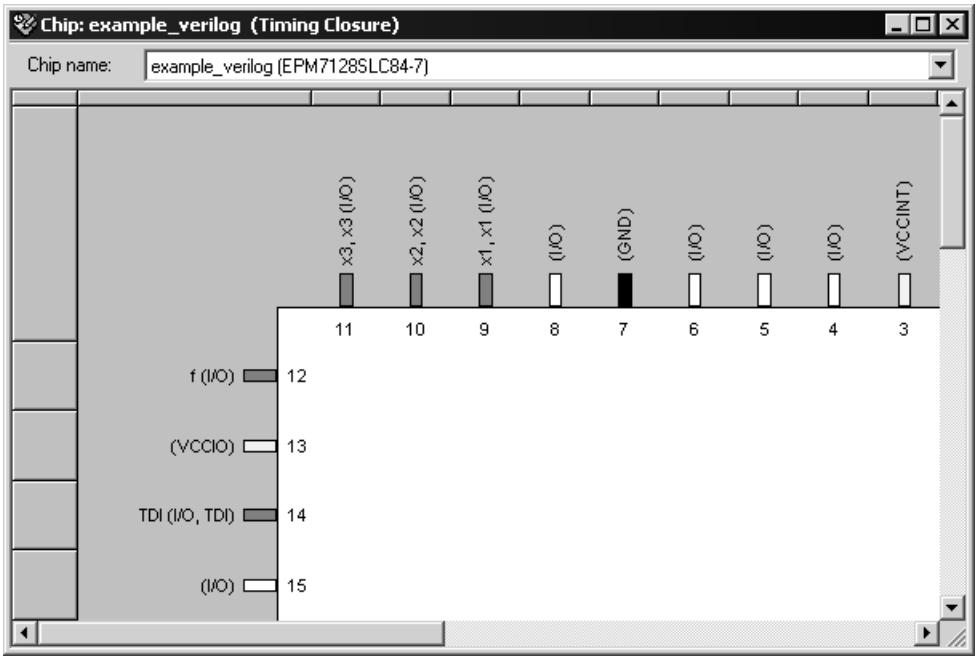


Figure D.6. Using the Floorplan Editor to make pin assignments.

Any pin assignment can be changed in this manner by using the Floorplan Editor. Another variant for making pin assignments is to open the Node Finder tool to search for pins, and then use the mouse to drag-and-drop the pin names from Node Finder onto package pins in the Floorplan Editor. New assignments that are created only affect the compilation results when the project is recompiled.

Pin assignments can be deleted from a project with the same tools employed to create the assignments. In the Assign Pins dialogue, click to highlight an existing pin assignment and then click on the Delete button (see Figure D.3). In the Floorplan Editor, click on a pin that has a pin assignment and select Edit | Delete to remove the assignment. Again, these changes only affect the compilation results when the project is recompiled.

D.2 Downloading a Circuit into a Device

Once a circuit has been compiled, it can be downloaded into the selected device. Downloading involves programming the appropriate switches in the chip to implement the desired circuit. To illustrate the steps involved, we will describe how a circuit can be downloaded into a laboratory development board that is available from Altera Corporation. The board is called the UP-1 Education Board and includes both a MAX 7000 CPLD and an FPGA. The UP−1 board can be obtained by following the instructions in the University Program section of Altera's Web site at http://www.altera.com.

We will describe how the *example_verilog* project that we implemented in a MAX 7000 CPLD can be downloaded into the UP−1 board, assuming that it is connected to the reader's computer. A reader who does not have access to the UP−1 board will not be able to download the circuit, but the steps involved are still easy to follow. The UP−1 board is connected to the computer using a type of cable that is available from Altera. For purposes of this discussion we will assume that a ByteBlaster cable is used, which provides a connection to a port on the computer.

The UP−1 board contains an EPM7128SLC84-7 chip. There is a socket that connects this chip to the ByteBlaster cable. Plug the ByteBlaster cable into this socket and plug the other end of the cable into the parallel port on the computer. Ensure that the UP−1 board is plugged into a power supply and that the green "power LED" is lit.

Use File | Open Project to open the *example_verilog* project. Select Tools | Programmer to open the Programmer module window shown in Figure D.7. The programming file for the *example_verilog* project, which is called *example_verilog.pof*, should be listed in the Programmer window. If this file is not shown click Edit | Add File and type the file name.

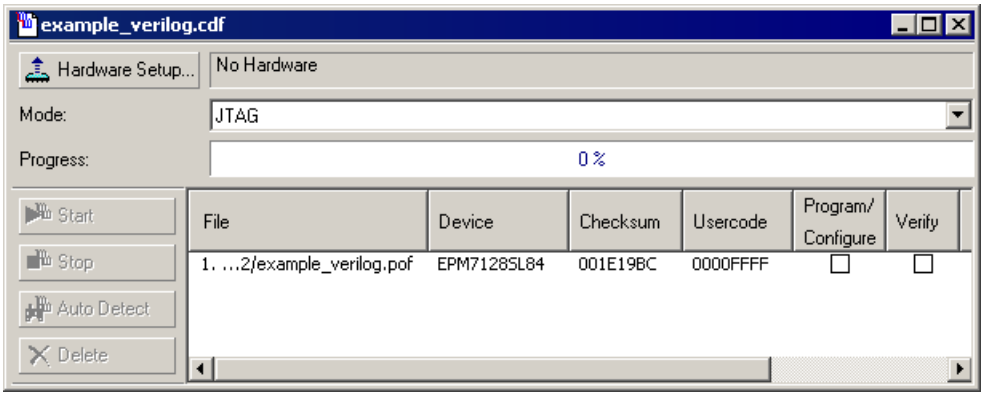
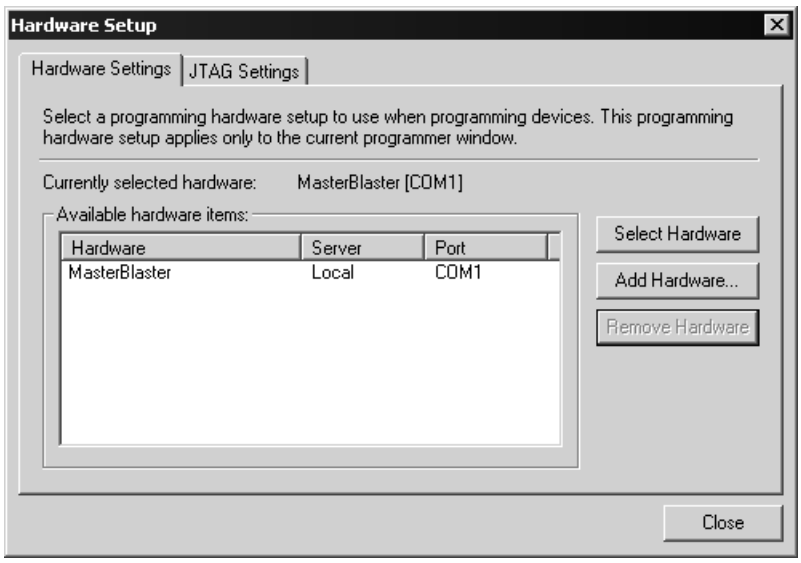
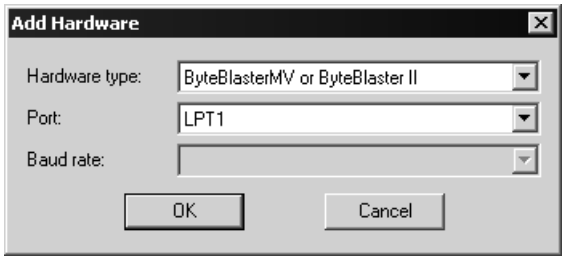


Figure D.7. The Programmer module window.

To specify that the ByteBlaster is to be used as the programming hardware, browse on the Hardware setup button, which opens the window in Figure D.8*a*. If the ByteBlaster is not listed under Available hardware items click on the Add Hardware button. This action opens the window in Figure D.8*b*. Open the drop-down list next to Hardware type and select the item ByteBlasterMV or ByteBlaster II. Click OK to return to the Hardware Setup window. We should note that the driver software for the ByteBlaster has to be installed on the computer being used before the above procedure will work. If the ByteBlaster cable does not appear in the drop-down list in Figure D.8*b*, then run the command *bblpt /i* from a Windows command prompt. This command is available in the directory *C:\quartus\drivers\i386*, assuming that the Quartus II software is installed in the directory *C:\quartus*.



(a) The Hardware Setup window.



(b) The Add Hardware dialogue.

Figure D.8. Adding the ByteBlaster hardware.

The ByteBlaster should now appear in the Available hardware items box. Click on this item to highlight it, and then click the Select Hardware button. Close the Hardware Setup window to return to the Programmer window in Figure D.9. Notice that the ByteBlaster is now shown to the right of the Hardware button, meaning that this cable is now selected. As indicated in the figure, click on the two checkmark boxes under Program/Configure and Verify associated with the *example_verilog.pof* file.

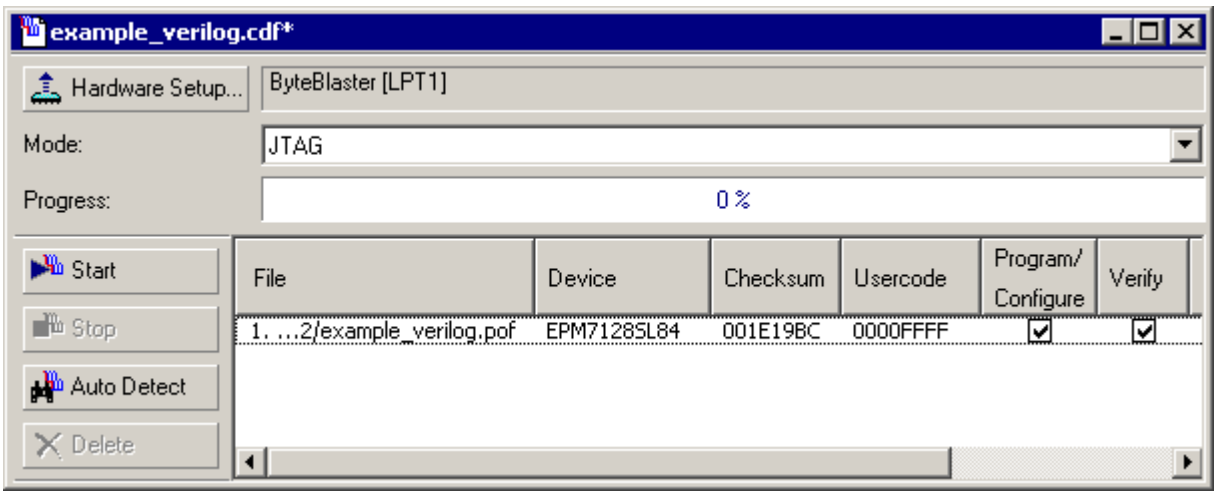


Figure D.9. The final Programmer module window.

To configure the EPM7128SLC84-7 chip, select Processing | Start Programming. The Programmer module automatically downloads the *example_verilog.pof* file through the ByteBlaster cable into the device and then verifies that the programming has been performed correctly. The Programmer module can now be closed. The designer can test the circuit implemented in the chip by using appropriate test equipment.

The UP−1 board also contains an FPGA chip. The procedure used to download a circuit into this chip is similar to the one described for the MAX 7000 device, but a few extra steps are needed. The reader who tries using the FPGA chip should refer to the documentation that accompanies the UP−1 board for detailed instructions.

D.3 Concluding Remarks

In Tutorials 1, 2, and 3, we have introduced many of the most important features of Quartus II. However, many other features are available. The reader can learn about the more advanced capabilities of the CAD system by exploring the various commands and on-line help provided in each application.