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.

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. As an example, we will assign the inputs $x1$, $x2$, and $x3$ to pins 9, 10, and 11, and we will assign the output f to pin 12.

Select Assignments | Pins to open the Assignment Editor window in Figure D.1. Under Category select Pin. Double-click on the entry <<new>> highlighted in blue in the column labeled To, which brings the drop-down menu in Figure D.2. Click on $x1$ as the first pin to be assigned. This will enter $x1$ in the Assignment Editor window. Next, double-click on the box to the right of this new $x1$ entry, in the column labeled Location. Now, the drop-down menu in Figure D.3 appears. Scroll down and select Pin_9. Instead of scrolling down the menu to find the desired pin, you can type the name of the pin in the Location box. Use the same procedure to assign inputs $x2$ and $x3$ to pins 10 and 11, as well as the output f to pin 12, which produces the display in Figure D.4. You can change an existing pin assignment by double-clicking on the given pin and following the procedure above. To save the assignments made, use File > Save. You can also simply close the Assignment Editor window, in which case a pop-up box will ask if you want to save the changes to assignments; click Yes.

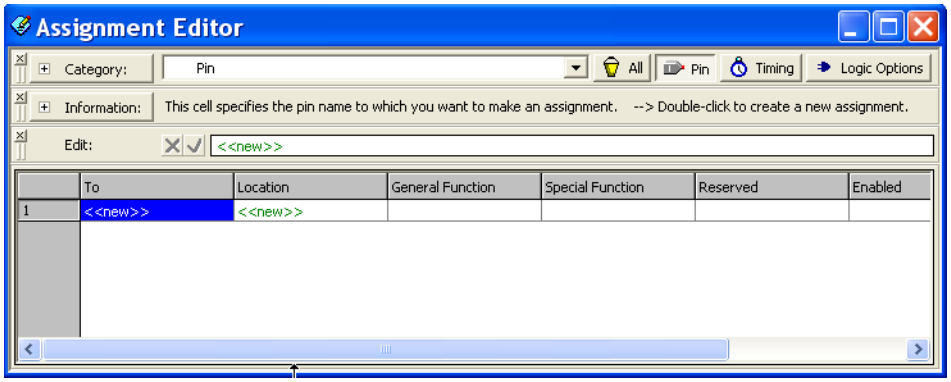


Figure D.1. The Assignment Editor window.

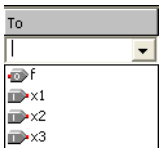


Figure D.2. Input and output names are presented in the drop-down menu.

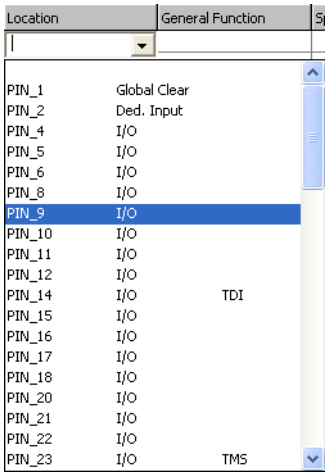


Figure D.3. Pins that are available.

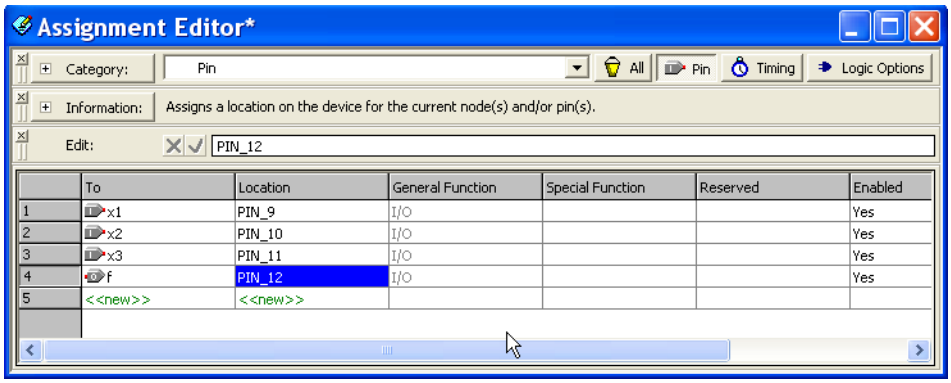


Figure D.4. The pin assignments for inputs $x1$, $x2$, and $x3$.

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 four 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
set_location_assignment Pin_12 -to f
```

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 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.

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.

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.5. 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.

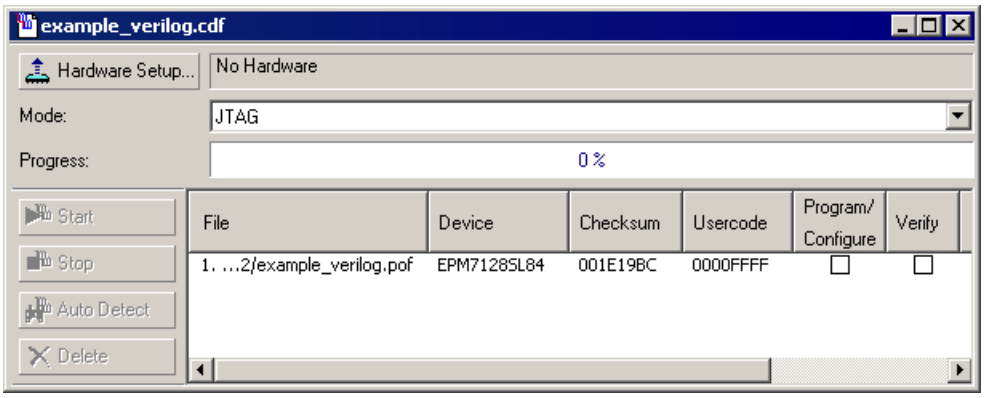
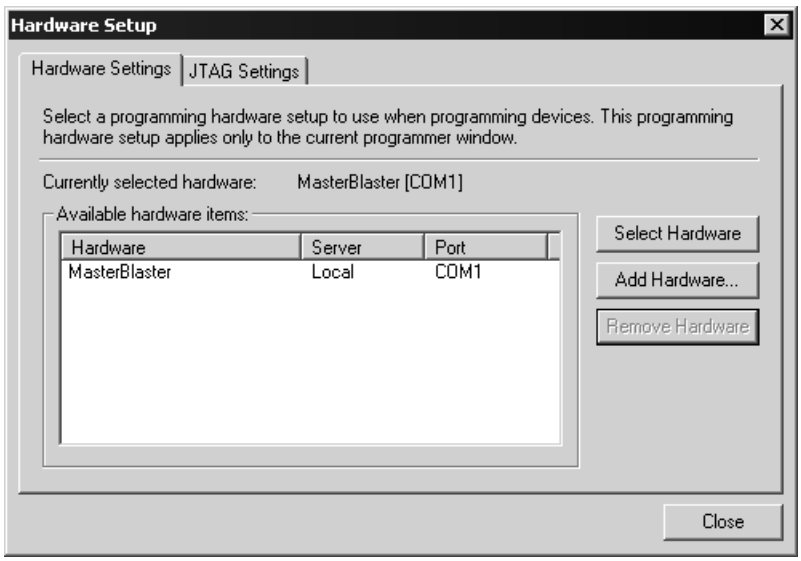
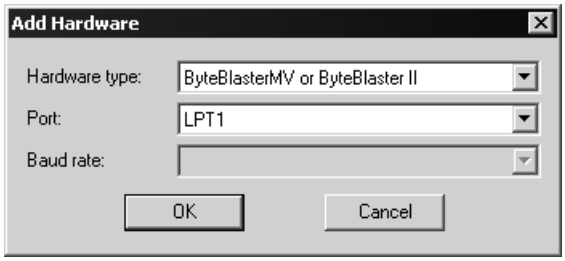


Figure D.5. The Programmer module window.

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.6*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.6*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.6*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.6. 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 FigureD.7. 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.

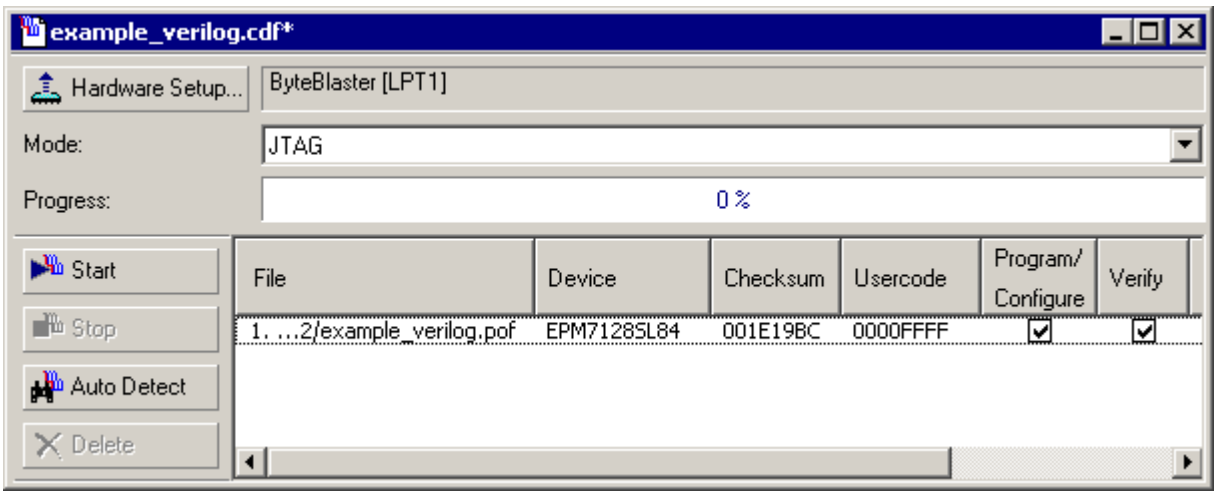


Figure D.7. The final Programmer module window.

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.