Chapter 1

Watch Design — Implementation Tools Tutorial (2.1i)

This chapter contains the following sections.

Installing the Tutorial Files

This tutorial demonstrates the steps in the Alliance Series Design Implementation Tools flow for the Watch design. Before beginning this tutorial, you should have already entered the watch design into the design entry tool of your choice. For a listing of EDA Interface tutorials, please reference the Xilinx Support web site at: http://support.xilinx.com/support/techsup/ tutorials/index.htm

Depending on your design entry tool, the input design is described by either a Xilinx netlist file (*.xnf, *.sxnf) or an EDIF file (*.sedif, *.edf, *.edn). This tutorial passes an input netlist from the front end tool to the back-end Alliance Series Design Implementation Tools, and incorporates placement constraints through a User Constraints File (UCF). Timing constraints will be added on later through the Constraints Editor.

The tutorial design **(WATCH)** is designed to perform like a track coach's stopwatch. There are two inputs to the system (RESET and SRTSTP). The configuration clock on the device is used as a ten-hertz clock signal. Three seven-bit outputs are generated by this system for output to three seven-segment LED displays.

Before proceeding to Step 1 in the tutorial, create a working directory with the tutorial files as follows.

1. Create an empty working directory named *Watch*.
2. Copy the following files created with your design entry tool into the Watch directory:

File Name	Description
watch.edn	Input netlist file (EDIF)
tenths.ngc	LogiBLOX implementation file
watch.ucf	User constraints file

Step 1: Creating an Implementation Project

The design implementation tools are organized under a single program called the Design Manager. The Design Manager helps you manage the design flow process by keeping track of design versions and the implementation revisions within each version. The Design Manager also provides access to the entire suite of Xilinx implementation tools needed to complete a design.

While the Design Manager manages your Xilinx design, the Flow Engine actually implements it. The Flow Engine is closely integrated with the Design Manager and shares many of the same menus and dialog boxes.

To begin, use the following steps to create an implementation project.

1. On a workstation, enter the following to start the Design Manager.

   `xilinx &`

   On a PC, select the following to start the Design Manager.

   `Start` → **`Programs`** → **`Xilinx Alliance Series 2.1i`** → **`Design Manager`**

   When you open the Design Manager for the first time, you must create a new project for your design. A project includes all design versions, implementation revisions, reports, and any other Xilinx data created while you work with a design.

   The Design Manager graphically displays information about these items in the project view. When you create a new project, you specify a design to open and a directory for the project. You can create as many projects as you want, but you can only work with one at a time.

2. Select **`File`** → **`New Project`** from the Design Manager menu to create a new implementation project for the tutorial design. The New Project dialog box appears as shown below:

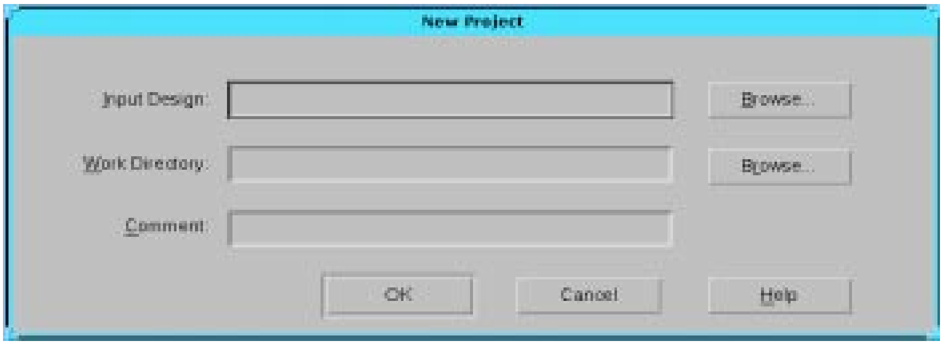


Figure 1-1 New Project Dialog Box

This dialog box contains the following fields.

Table 1-1 New Project Dialog Box Fields

Field	Description
Input Design	Top level netlist file containing the design definition
Work Directory	Directory used to store the implementation data created as the design is compiled
Comment	Enter any optional notation for the design in this field

3. Click **Browse** to the right of the Input Design field to specify the input design. The Browse dialog box appears as shown in the following figure.

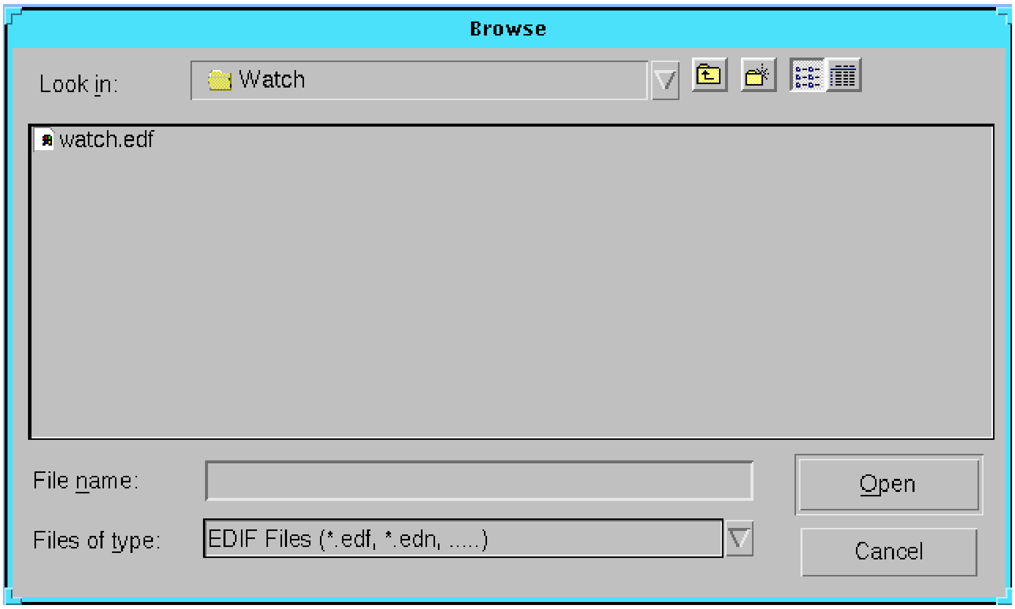


Figure 1-2 Browse Dialog Box

4. Select the appropriate file type from the drop-down list in the Files of Type field. For this tutorial design, EDIF is selected.

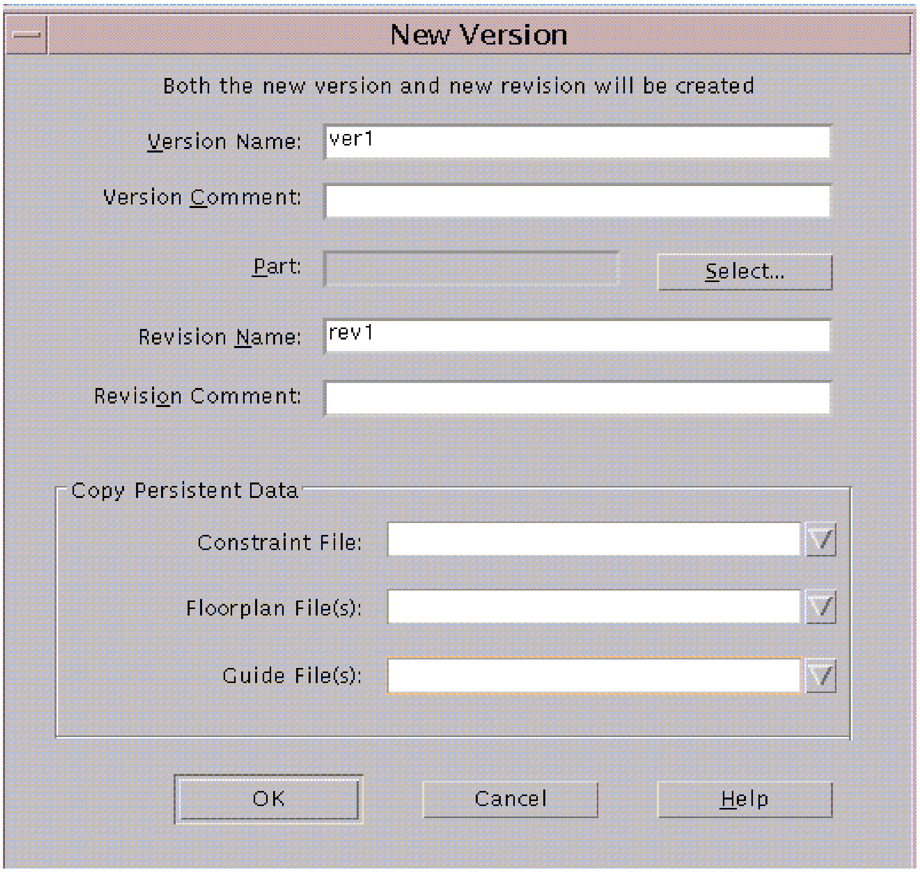
5. Select the **Watch** design file. The file name appears in the File Name field. Click **Open**.

The Browse dialog box closes and the New Project dialog box is updated to include the specified input netlist. By default, the Work Directory field is set to the directory containing the input design. If preferred, you can set this to another directory. Because the files were previously copied to the Watch directory, this directory is used for the implementation project and resulting output files.

6. In the Comment field, enter the following.

   `-tutorial`

7. Click **OK** to close the New Project dialog box.



Both the new version and new revision will be created

Version Name: ver1

Version Comment:

Part:

Revision Name: rev1

Revision Comment:

Copy Persistent Data

Constraint File:

Floorplan File(s):

Guide File(s):

Figure 1-3 New Version Dialog Box

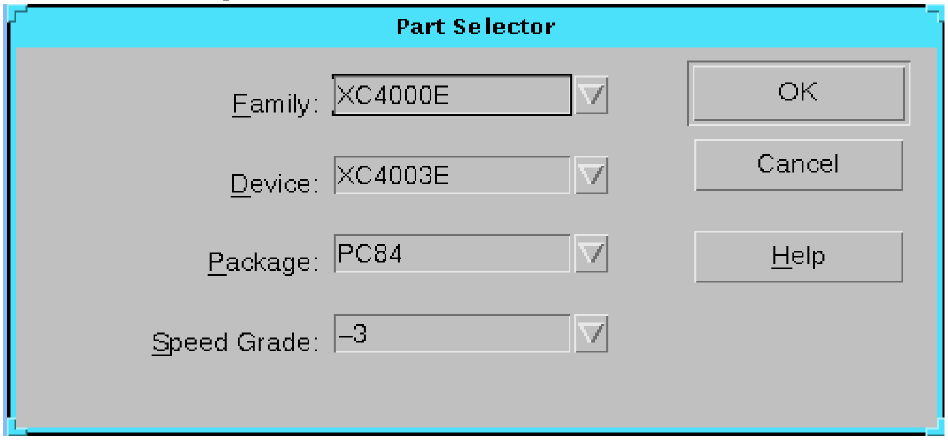
At this point, the New Version dialog box appears, as shown in the preceding figure.

When initially creating a project, the New Version dialog automatically appears to allow you to enter the information necessary to define the new design version. In addition, any time that your input netlist changes due to a change made within your front-end tool, you are prompted by the Design Manager to generate a new design version.

Once a design version is created, you can try different implementation strategies on your design. The data associated with each of these implementation strategies is called an implementation revision. Because a new implementation revision is automatically created when you create a new version, you will see both of these fields already defined in the New Version dialog.

8. By default, the **Version Name** field shows **ver1** as the default version, and the **Revision Name** field shows **rev1** as the default revision. Comments to note options and strategies can be entered in the Version and Revision Comment fields.

9. Click **`Select`** to display the Part Selector dialog box. The Part field will automatically contain a part number if you specified the target device in your design entry tool. If this field is empty, please define it.



Part Selector

Family: XC4000E

Device: XC4003E

Package: PC84

Speed Grade: -3

Figure 1-4 Part Selector Dialog Box

10. Use the drop-down lists for the fields in the Part Selector dialog box to enter the Family, Device, Package, and Speed Grade for the

design. This design targets an **XC4003E-3-PC84**. Click **OK**. The part number appears in the Part field in the New Version dialog box.

11. The Copy Persistent Data heading allows you to specify the copying of constraint, guide, or floorplan data to the new revision that is about to be created. You can choose to copy data from a previous revision or a custom file or choose None if you do not want to copy data. For this tutorial, we will keep all 3 drop-down boxes defined as **None**.

    By default, the Design Manager copies floorplan and constraints file data from the “last” revision. The “last” revision is the bottommost revision in the Design Manager project view.

Note: A special case exists for the first Version and Revision created in a project. In this situation, if a <design_name>.ucf exists in the same directory as the design netlist, then it will be copied into the revision directory even if the “Constraints File” field is set to “None”.

12. Click **OK** in the New Version dialog box.

    The Design Manager loads your design and displays a new design version and implementation revision icon in the project view, as shown in the following figure.

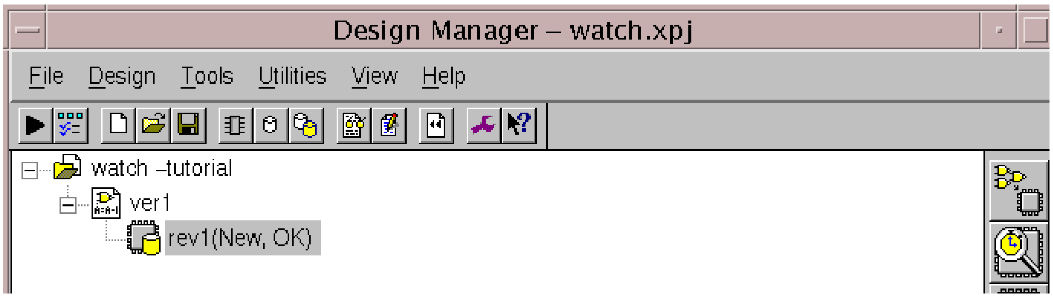


Figure 1-5 Watch Project in Design Manager

Design Manager Status Bar

At the bottom of the Design Manager window is the status bar. The status bar lists the current project, target device, and currently selected version / revision pair. The left-hand portion of the status bar provides help on what is currently selected by your cursor.

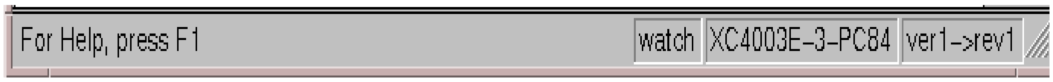

For Help, press F1 | watch | XC4003E-3-PC84 | ver1->rev1

Figure 1-6 Design Manager Status Bar

Design Manager Toolbox

The toolbox, located on the right side of the Design Manager, becomes active when a revision is selected. Icons in the toolbox (as shown in the following figure) represent the Flow Engine, Timing Analyzer, Floorplanner, PROM File Formatter, Hardware Debugger, FPGA Editor, and JTAG Programmer.

Note: The toolbar has drag and drop capability.

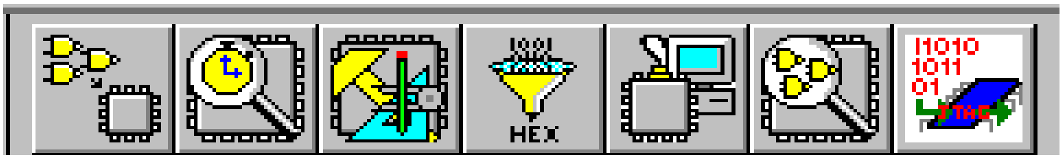


Figure 1-7 Design Manager Toolbox

Step 2: Specifying Options

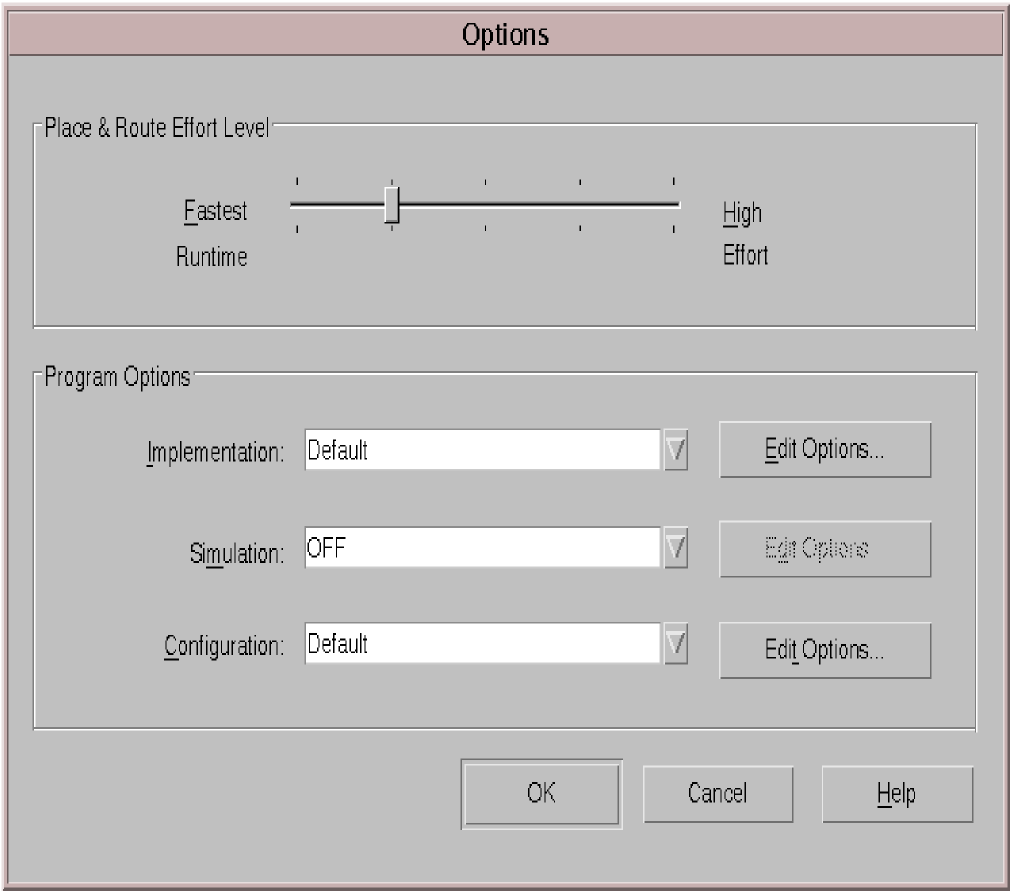
An implementation revision contains data files and reports that are created based on a specific set of implementation strategies. Implementation strategies are defined by specifying a set of options. You can specify options that control how the Flow Engine implements a design, creates timing simulation data, creates netlist files, generates reports, and creates configuration data. The options available depend on the target device family.

You can use the tools to create as many implementation revisions as you want for a design version. For example, if you want to try various implementation strategies on a netlist, several revisions can be created for a single design version. *By default, however, the Design Manager recompiles within the current revision.*

Within the Design Manager, notice how the project view displays rev1 under the initial version of the watch project. The status of the revision is noted as (New, OK). *New* refers to the state of the design and is updated throughout the tutorial as the different compilation stages are completed. *OK* is the status of the current state and indicates no errors in the design processing.

Use the following steps to specify options for this design.

1. Select **`Design`** → **`Options`** to open the Options dialog box as shown in the following figure



Implementation: Default
Simulation: OFF
Configuration: Default

Figure 1-8 Options Dialog Box

This dialog allows you to set options used in the implementation, simulation, and configuration flow. Changes made in this dialog box apply to the selected implementation revision. The dialog box above appears if you are targeting an FPGA. A slightly different Options dialog would appear if you were targeting a CPLD.

Select the **HELP** button to read through the information regarding this entire dialog box.

2. Select **`Edit Options`** next to the Implementation Program Option.

   The XC4000 Implementation Options dialog box is displayed. The implementation options control how the software maps, places, routes, and optimizes a design.

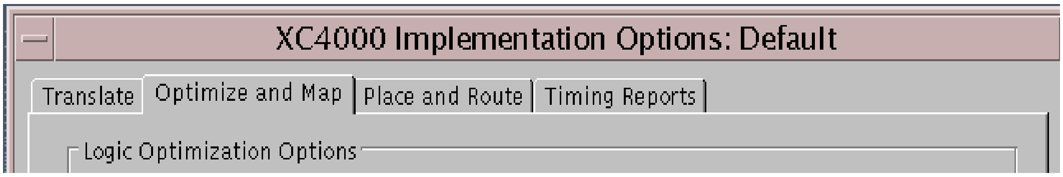

XC4000 Implementation Options: Default
Translate
Optimize and Map
Place and Route
Timing Reports
Logic Optimization Options

Figure 1-9 XC4000 Implementation Options Dialog Box

3. Select the **`Timing Reports`** tab.
4. Select **`Produce Logic Level Timing Report`**. The option to produce a **Post Layout Timing Report** should already be selected by default. For both reports, select Report Paths in Timing Constraints.
5. The timing reports we just enabled are useful for evaluating design performance. They will be analyzed in detail later in this tutorial.
6. Click **`OK`** to save the Implementation options and return to the Options dialog.
7. Select **OFF** from the drop down list next to the Configuration Program Options. This disables the generation of a bitstream for our design. We will revisit this option later once we have completed evaluating the performance of our design.
8. Click **`OK`** to exit the Options dialog box.

As previously mentioned, an implementation revision is created based on a specific set of implementation strategies. In addition to the program options we just set, an implementation strategy is defined by the constraints applied onto the design.

For this design, you were initially asked to copy over a User Constraints File (UCF) into your design directory. Since the Design Manager by default copies over your constraints information into the new revision created, you should be able to open up the **watch.ucf** file found under your newly created **rev1** directory. With a text editor, view the location constraints that are specified for this design.

The User Constraints File (UCF) provides a mechanism for constraining a logical design without returning to the design entry tools. However, it requires the user to understand the exact syntax needed to define constraints. On the other hand, the **Constraints Editor** is a graphical tool in the Xilinx Development System that allows you to enter timing and pin location constraints. We will take advantage of this tool to not only view the constraints specified currently in the *watch.ucf* file, but to also add in some timing constraints of our own.

9. From within Design Manager, select **Utilities->Constraints Editor.** You will be immediately prompted with a dialog that informs you that the Translate step needs to be run before launching this utility.

In order to gain a better understanding of what will take place once you provide an answer to this question, please continue onto the next step of the tutorial *before* clicking YES or NO.

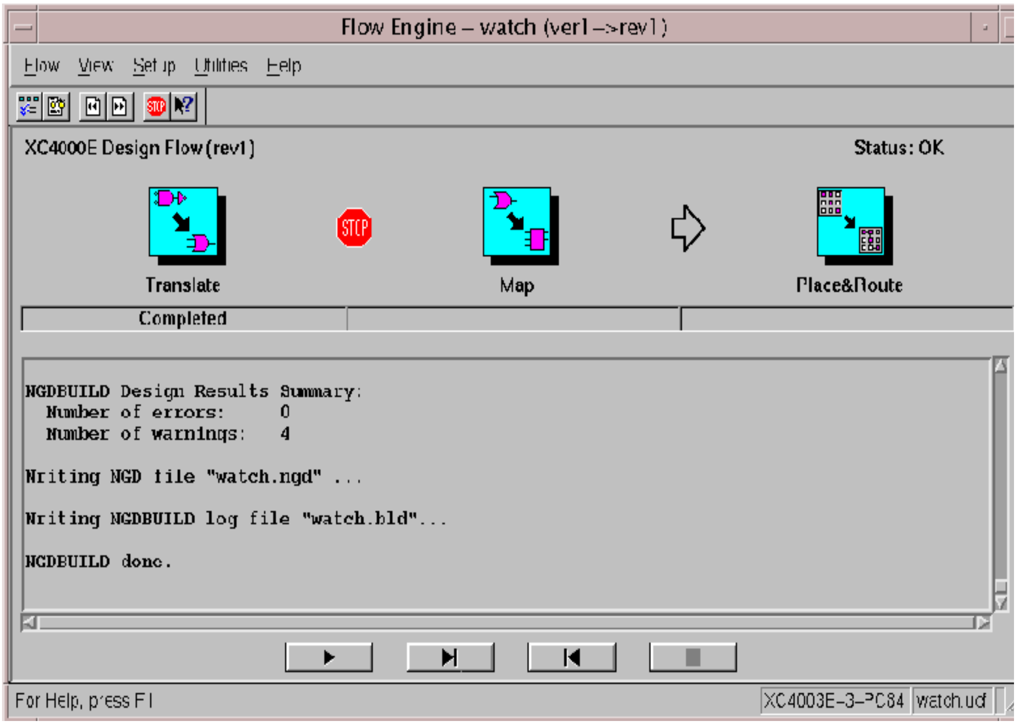
Step 3: Translating the Design

The Design Manager manages the files created during the implementation process while the Flow Engine controls the implementation process itself. The programs run by the Flow Engine use the settings supplied by the user in the options dialog box. The Flow Engine gives you complete control over how a design is processed. Typically, one sets all their options first and then runs through the entire flow by selecting “Implement” from the Design Menu.

In this tutorial, you are attempting to further define the design by setting constraints after having defined options. As you were just informed, in order to invoke the Constraints Editor, the Translate step must first be run.

- Select **YES** to continue with the flow

The Flow Engine is invoked for the first time. The steps in the design flow are graphically represented in the upper half of the Flow Engine window. The status of each stage is also shown. Refer to the "Translating Design" figure below:


Flow Engine – watch (ver1 –>rev1)
Flow View Setup Utilities Help
XC4000E Design Flow (rev1)
Status: OK
STOP
Translate
Map
Place&Route
Completed
For Help, press F1
XC4003E-3-PC84
watch.ucf

```
NGDBUILD Design Results Summary:
  Number of errors:     0
  Number of warnings:   4

Writing NGD file "watch.ngd" ...

Writing NGDBUILD log file "watch.bld"...

NGDBUILD done.
```

Figure 1-10 Translating Design

Notice the "STOP" sign placed between the Translate and MAP steps. This breakpoint has been automatically set in this situation to instruct the Flow Engine to stop after the Translate step is complete.

During translation, the program **NGDBuild** is executed, and performs the following functions:

- Converts input design netlists and writes results to a single merged NGD netlist. The merged netlist describes the logic in the design as well as any location and timing constraints.
- Performs timing specification and logical design rule checks
- Adds the User Constraints File (UCF) to the merged netlist

Once complete, the Flow Engine shuts down and the Constraints Editor is invoked.

Step 4: Using the Constraints Editor

The Constraints Editor is a utility that allow you to edit constraints previously defined (through a UCF file), as well as add new constraints to your design. Input files to the Constraints Editor are:

- NGD (Native Generic Database) file. This file also serves as the input to the mapper, which then outputs the physical design database, an NCD (Native Circuit Description) file.
- Corresponding UCF (User Constraint File).

By default, when the NGD file is opened, the UCF file with the same base name as the NGD file (found in current revision directory), is loaded. This insures that changes made to the constraints are stored within the revision.

Upon successful completion, the Constraints Editor writes out a valid UCF file. NGDBuild uses the UCF file, along with design source netlists to produce a newer NGD file that incorporates the changes made. The NGD is then read by the MAP program (the next step in the design flow). In our design, the **watch.ngd** file and **watch.ucf** file are automatically read into the Constraints Editor.

The Global Tab appears in the foreground of the Constraints Editor window. This window automatically displays all the clock nets in your design, and allows you to define the associated period, pad to setup, and/or clock to pad values.

1. Select the Period cell on the row associated with the clock net **oscout**. Double-click your left mouse button. This invokes the **Clock Period** dialog box.

   Within the Clock Signal Definition, keep the default **(Specific Time)** selected to define an explicit period for the clock rather designate a period which is relative to another timing specification.

2. Enter a value of **20** in the Time text box. Verify that **ns** is selected from the Units pull-down list. Click **OK**.

   Notice that the period cell is updated with the global clock period constraint we just defined (with a default 50% duty cycle)

Note: For the purpose of this tutorial, we invoked a secondary dialog by double-clicking on a cell to specify our constraint values. A new feature to the Constraints Editor in 2.1i allows for the direct entry of constraints into cells by simply clicking once.

3. Select the **Ports** tab from the Constraints Editor's main window.

   The left hand side displays a listing of all the current ports as defined the user. Notice that certain cells in the Location column are pre-populated with device pins locking down ports to actual pins on the target device. This information was obtained by the Constraints Editor by way of the watch.ucf file it read in.

4. Select **File->Save**.

   The change made within the Constraints Editor is now saved into the watch.ucf file in your current revision directory.

5. You will prompted with a reminder to rerun the Translate step. Click **OK.**

6. Select **File->Exit**

IMPORTANT: Read this Before Mapping the Design — How to Stop the Design Processing

Before we continue implementing our design in the Flow Engine, review the following procedure for stopping the processing of the design after the MAP step. Because the steps for the tutorial design can often finish quickly, you should be familiar with this procedure before you start the Flow Engine.

Setting a break point anywhere in the design process is useful when you want to stop and evaluate your performance before going forward. For example, setting a breakpoint after the Translate step is useful when you want to perform a functional simulation of a design and copy the resulting *design*.ngd file to your working directory. After copying the *design*.ngd file, you can run the appropriate NGD2XXX program on the file to create functional simulation data. For more information on the NGD2XXX programs, see the appropriate chapter in the *Development System Reference Guide.*

Note: This procedure can be utilized at any time in the Flow Engine to stop after any of the steps in the design flow.

1. In this tutorial, you want to stop processing the design after the MAP step. To do this, you must set a break point to stop the Flow Engine. To stop after the MAP step from within the Flow Engine, click the stop sign toolbar icon while MAP is running.
2. The Stop After dialog box is displayed with the default setting of Configure as shown in the following figure. The list box displays the break points appropriate for the current state of the design. Because the design has not completed processing at this point, all possible break points are listed.

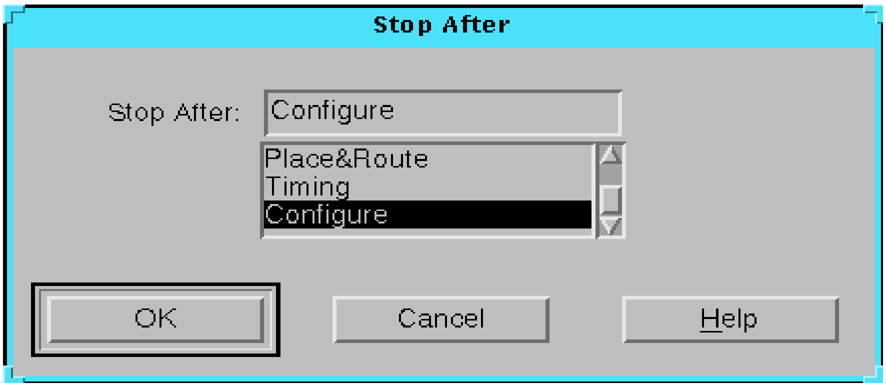
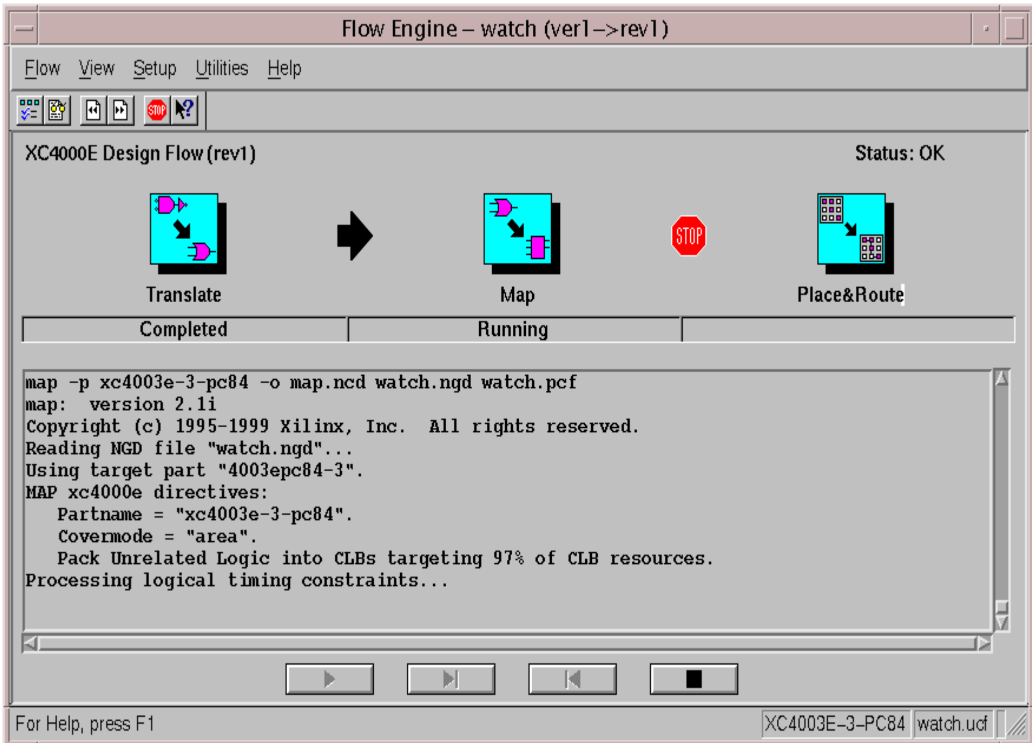


Figure 1-11 Stop After Dialog Box

3. Select MAP in the list box and click **OK**. The stop sign is added to the design flow between the Map and Place and Route steps as shown in the following figure.

Note: The status bar at the bottom of the Flow Engine window will be updated with the specified user constraints file (watch.ucf).



```
map -p xc4003e-3-pc84 -o map.ncd watch.ngd watch.pcf
map:  version 2.1i
Copyright (c) 1995-1999 Xilinx, Inc.  All rights reserved.
Reading NGD file "watch.ngd"...
Using target part "4003epc84-3".
MAP xc4000e directives:
   Partname = "xc4003e-3-pc84".
   Covermode = "area".
   Pack Unrelated Logic into CLBs targeting 97% of CLB resources.
Processing logical timing constraints...
```

Figure 1-12 Mapping Design

Starting the Flow Engine and Translating/Mapping your Design

Now that all implementation strategies have been defined (options and constraints), let's continue with the implementation of our design.

1. Select **Design->Implement** from the Design Manager.
2. The Flow Engine automatically detects that changes were made to your constraints file, which requires the Translate step to be re-run. In order for the changes we just made to the Constraints Editor to take affect, select **YES.**
3. Perform the procedure previously described in the "IMPORTANT: Read this Before Mapping the Design — How to Stop the Design Processing" section to stop the processing of the design.

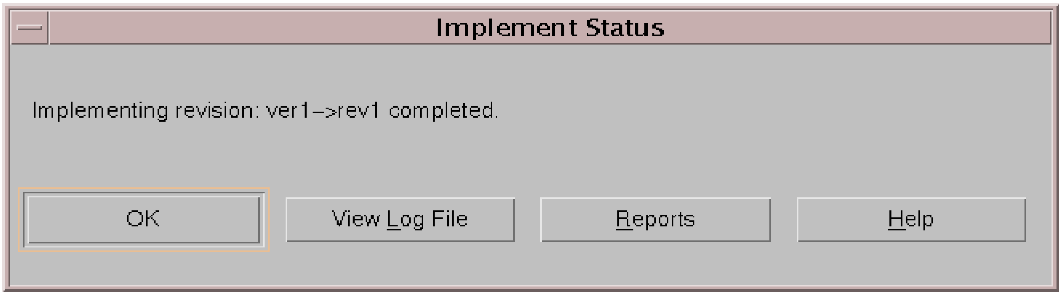
Step 5: Mapping the Design

At this point, the input netlist is being translated (once again), and merged into a single design file. Next, the design will be mapped into CLBs and IOBs. After mapping, the design will be placed and routed. The final step in the design flow is the Configure step in which a configuration bitstream is created for downloading to a target device or for formatting into a PROM programming file.

Map performs the following functions:

- Allocates CLB and IOB resources for all basic logic elements in the design
- Processes all location and timing constraints, performs target device optimizations, and runs a design rule check on the resulting mapped netlist.

After the MAP step is done, the Flow Engine shuts down and the Implement Status dialog box appears:



Implementing revision: ver1–>rev1 completed.

Figure 1-13 Implement Status Dialog Box

1. Select **Reports** to invoke the Report Browser window. The Translation Report appears as the first report generated. The Map Report and Logic Level Timing Report files are created as a result of the Map stage completing. New reports that have not been read are denoted with a gold star in the upper left corner of the file icon.

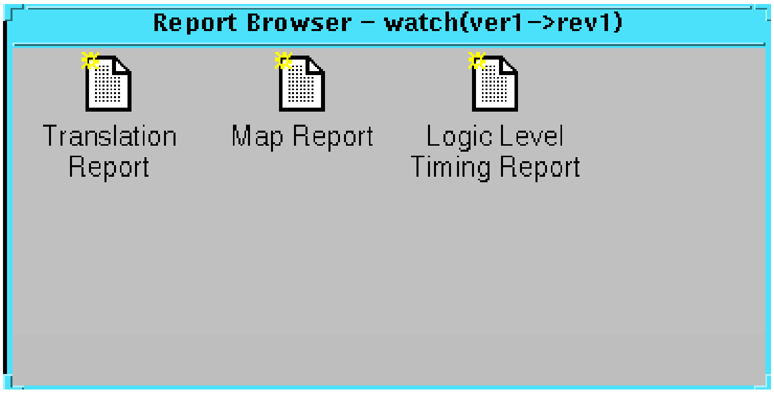


Figure 1-14 Report Browser after Running Map

2. Double-click on a report to review its output. The following table lists the types of reports and descriptions.

Table 1-2 Report Browser Reports

Report	Description
Translation Report	Includes warning and error messages from the translation process.
Map Report	Includes information on how the target device resources are allocated, references to trimmed logic, and device utilization. For detailed information on the Map report, refer to the *Development System Reference Guide*.
Logic Level Timing Report	Provides a summary analysis of your timing constraints based on block delays and estimates of route delays. This report is produced after Map and prior to PAR (Place And Route).

3. Select **OK** to close the Implement Status dialog. Keep the Report Browser open for now. We will be evaluating some of these reports in further detail in the next section.

Notice that the Design Manager project view displays the status of the revision as (Mapped, OK). *Mapped* refers to the state of the design and is updated throughout the tutorial as the different compilation stages are completed. *OK* is the status of the current state and indicates no errors in the design processing.

The design has now been mapped to the target architecture. The next step involves checking the design paths for block delays.

Step 6: Using Timing Analysis to Evaluate Block Delays After Mapping

After the design is mapped, you can use the Logic Level Timing Report to evaluate the logical paths in the design. Because the design is not placed and routed yet, actual routing delay information is not yet available. The timing report describes the logical block delays and estimated routing delays. The net delays that are provided are based on an optimal distance between blocks (also referred to as *unplaced floors*).

Estimating Timing Goals with 50/50 Rule

You can get a preliminary idea of how realistic your timing goals are by evaluating a design after the map stage. A rough guideline (known as the 50 / 50 rule) specifies that the block delays in any single path make up approximately 50% of the total path delay after the design is routed. For example, a path with 10ns of block delay should meet a 20ns timing constraint after it is placed and routed. If your design is extremely dense, or if you are using an architecture with fewer routing resources (i.e - 4025E vs. a 4028XL), your net delays can be more than 50% of the total path delay.

Report Paths in Timing Constraints Option

Because timing constraints were defined for this tutorial design, the Report Paths in Timing Constraints option was selected. This option forces the Logic Level Timing Report to provide a period and path analysis on the constraints specified. Taking a look at the report, the period timing constraint is listed on top, as is the minimum period obtained by the tools after mapping. Because we limited our report to one path per timing constraint, we see a breakdown of a single path that contains 4 levels of logic. Notice the percentage of block (logic) delay versus routing delay for this calculation. The unplaced floors listed are estimates (indicated by the letter “e” next to the net delay) based on optimal placement of blocks.

If you do not generate a Logical Level Timing Report, PAR still processes a design based on the relationship between the block delays, floors, and timing specifications for the design. For example, if a PERIOD constraint of 8 ns is specified for a path, and there are block delays of 7 ns and unplaced floor net delays of 3 ns, PAR stops and generates an error message. In this example, PAR fails because it determines that the total delay (10 ns) is greater than the constraint placed on the design (8 ns).

Use the Logic Level Timing Report to determine timing violations that may occur prior to running PAR.

Step 7: Placing and Routing the Design

After the mapped design is evaluated to verify that block delays are reasonable given the design specifications, the design can be placed and routed. The Flow Engine can perform the following place and route algorithms.

- Timing Driven — run PAR with timing constraints specified from within the input netlist or from a constraints file
- Non-Timing Driven — run PAR and ignore all timing constraints

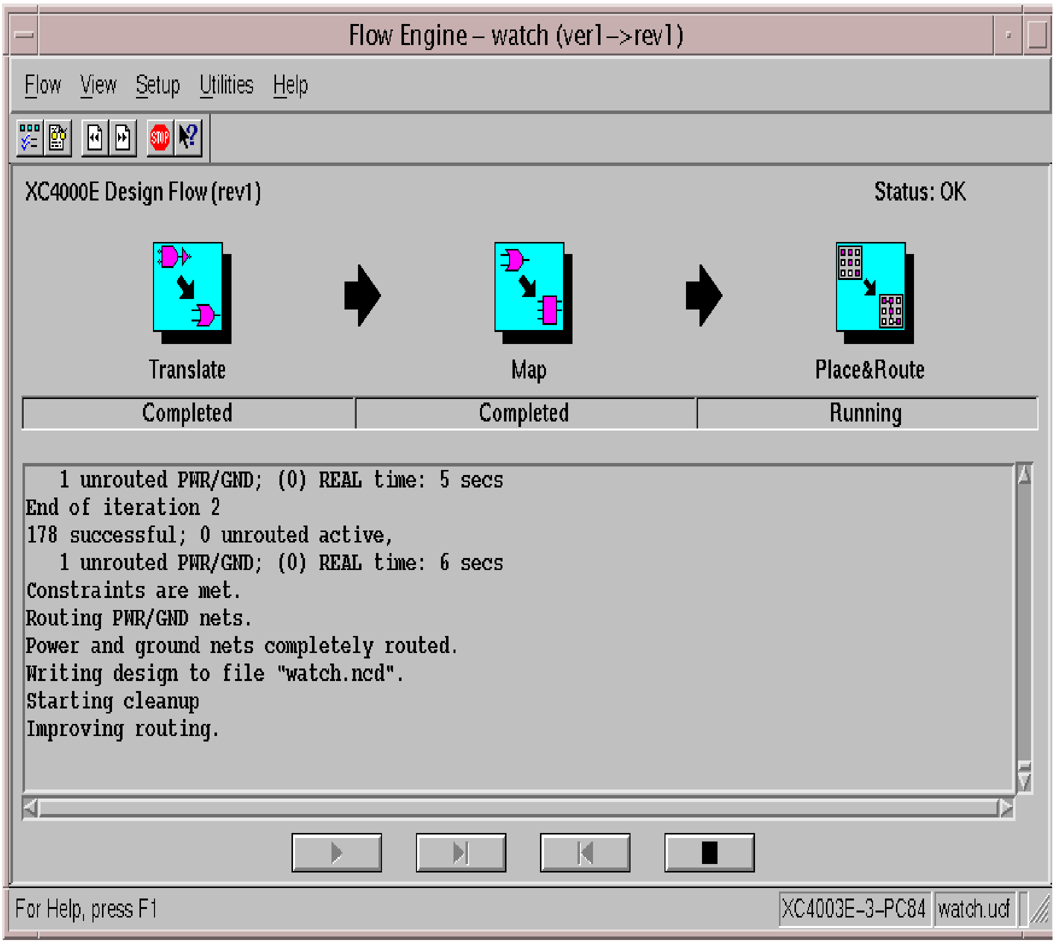
In this tutorial, timing driven placement and timing driven routing are automatically performed by PAR because timing constraints are specified for this design.

Close out the Report Browser and any open reports.

To place and route the design, perform the following step:

1. In the Design Manager window, select **`Design`**→ **`Implement`** to continue running the implementation flow.

   The Flow Engine will once again be invoked. The Status:OK message in the upper right corner indicates that no errors are generated by PAR at this point. Refer to the following figure:



```
   1 unrouted PWR/GND; (0) REAL time: 5 secs
End of iteration 2
178 successful; 0 unrouted active,
   1 unrouted PWR/GND; (0) REAL time: 6 secs
Constraints are met.
Routing PWR/GND nets.
Power and ground nets completely routed.
Writing design to file "watch.ncd".
Starting cleanup
Improving routing.
```

Figure 1-15 Placing and Routing Design

2. Review the reports generated to make sure the place and route process finished as expected.

   The four new reports created in the Report Browser are the Place and Route Report, the Pad Report, the Asynchronous Delay Report, and the Post-Layout Timing Report, as shown in the following figure and described in the following table.

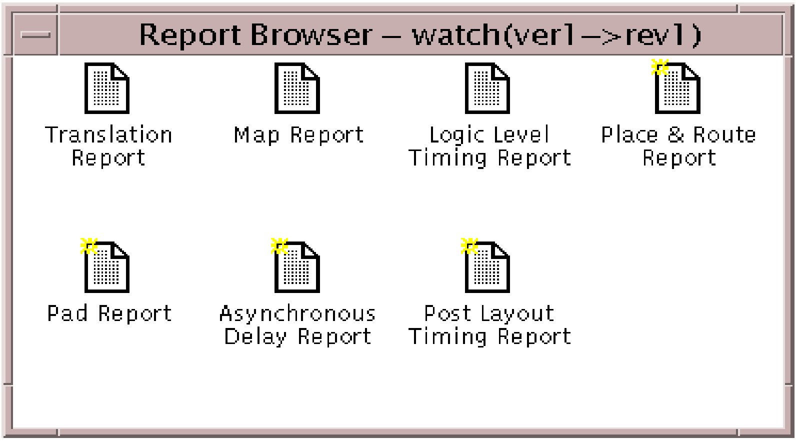

Report Browser – watch(ver1 –>rev1)
Translation Report
Map Report
Logic Level Timing Report
Place & Route Report
Pad Report
Asynchronous Delay Report
Post Layout Timing Report

Figure 1-16 Reports Available After Place & Route

Table 1-3 Description of Reports Available After Place & Route

Report	Description
Place & Route Report	Provides a device utilization and delay summary. Use this report to verify that the design successfully routed and that all timing constraints were met.
Pad Report	Contains a report of the location of the device pins. Use this report to verify that pins locked down were placed in the correct location.
Asynchronous Delay Report	Lists all nets in the design and the delays of all loads on the net.
Post-Layout Timing Report	Incorporates both the logic and routing delays to generate an evaluation of the design's timing constraints, clock frequencies, and path delays.

Note: In the Design Manager window, the status of the current version / revision is now *(Routed, OK).*

Step 8: Evaluating Post-Layout Timing

After the design is placed and routed, a Post Layout Timing Report is generated by default to verify that the design meets your specified timing goals. This report evaluates the logical block delays and the routing delays. The net delays are now reported as actual routing delays after the place and route process (indicated by the letter “R” next to the net delay).

Double-click on the Post Layout Timing Report to open it. The following is a summary of this report.

- The minimum period value increased due to the actual routing delays.
- After the Map step, logic delay contributed to about 80% of the minimum period attained. The post-layout report indicates that the logical delay value decreased somewhat. The total unplaced floors estimate changed as well. Routing delay after PAR now equals about 31% of the period; a true report of net delays after the place and route step.
- The post-layout result does not necessarily follow the 50/50 rule previously described because the worst case path includes primarily component delays. After the design was mapped, block delays constituted about 80% of the period. After place and route, the majority of the worst case path is still made up of logic delay. Since total routing delay makes up only a small percentage of the total path delay, spread out across three nets, expecting this to be reduced any further is unrealistic. In general, you can reduce excessive block delays and improve design performance by decreasing the number of logic levels in the design.

Step 9: Creating Timing Simulation Data

After the design is placed and routed and the timing is statically verified, the next step is to create timing simulation data. To create timing simulation data, perform the following steps in the Design Manager.

1. Select **`Design`**→ **`Options`** to open the Options dialog box.
2. Select the simulator that corresponds to your design entry tool from the Simulation drop-down list in the Program Options section of the dialog box.

3. Click **OK** to close the Options dialog box.
4. Select **Design**→ **Implement** from the Design Manager

   Within the Flow Engine, you will now notice a new stage appear directly after Place & Route. This new stage, called Timing(Sim), is solely dedicated to producing timing simulation data. In the tutorial, this stage did not appear originally because the Program Option for Simulation was not selected to a specific simulator in the initial pass. By default, this option is set to OFF. For all designs, you have the choice of selecting all options at the beginning of the design processing, or coming back to set them later.

   During the Timing(Sim) step, the Flow Engine runs the NGDAnno program to create a back-annotated NGD file. The NGD file is then used as input to one of the NGD2XXX programs to produce the preferred simulation file format. By default, the files created are named *time_sim*. To make it easy to find the output files for your third-party simulation environment, the files are automatically copied to your working directory.

Step 10: Creating Configuration Data

The next step is creating configuration data. This step includes creating a bitstream for the target device by running the configure step, as follows:

1. Select **Design**→ **Options** to open the Options dialog box.
2. Select **Default** from the drop-down list for Configuration Program Options.
3. Click on the **Edit Options** button corresponding to Configuration, which just became enabled with our selection of Default.

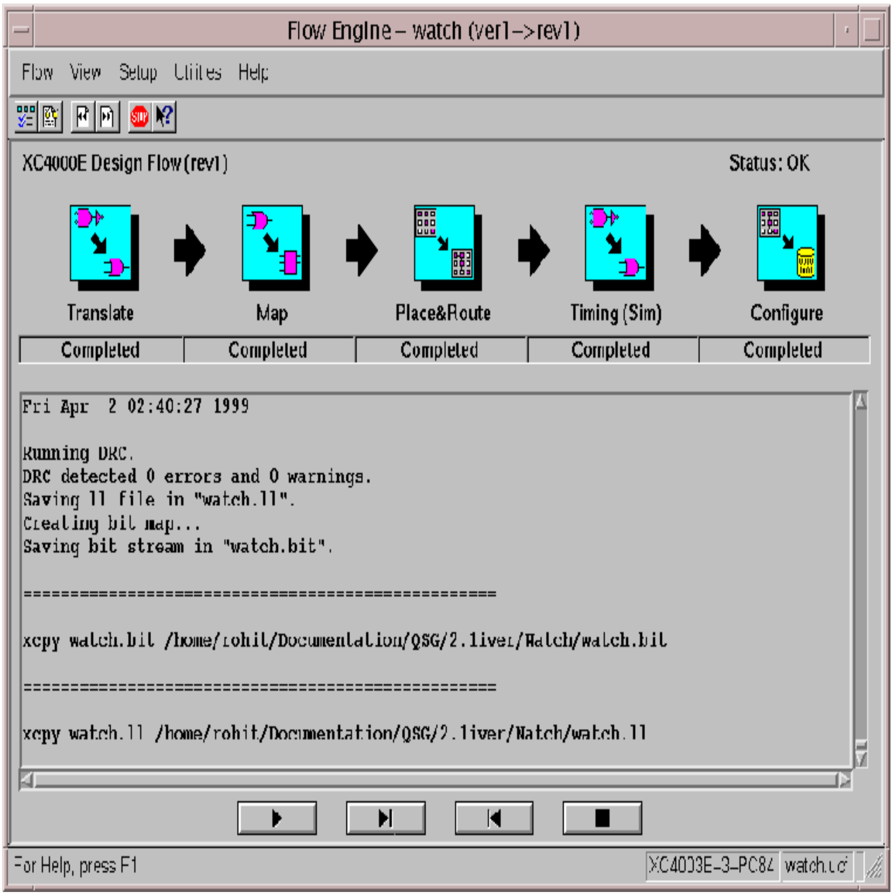
   The XC4000 Configuration Options dialog box appears. The configuration templates set options that define the initial configuration parameters, start-up sequence, readback capabilities, and other advanced features. In this tutorial, a configuration file is created that can be used for programming, verifying, and debugging XC4000E designs.
4. In the Configuration tab, verify that **PullUp** is selected next to the Done pin, and that the **Perform CRC During Configuration** option is selected.

5. Select the Readback tab, and verify that **CCLK** is selected as the readback clock.
6. Click **`OK`** to close the XC4000 Configuration Options dialog box.
7. Click **`OK`** to close the Options dialog box.
8. Select **`Design`**→ **Implement** from the Design Manager

   The Flow Engine comes up running the BitGen program in the newly added Configure stage. BitGen creates the *design_name*.bit and *design_name*.ll files (in this tutorial, the watch.bit and watch.ll files). The *design_name*.bit file is the actual configuration data. The *design_name*.ll file is the logical allocation file that is used during hardware debugging to determine the location of the probable points in the design. These files are automatically copied to your working directory. Verify that they are in this directory.

   The *design_name*.11 file is used to perform device readback with the Hardware Debugger tool. For more information on device readback, please refer to the latest version of the *Watch Design-Hardware Verification Tutorial (http://support.xilinx.com/support/techsup/tutorials/index.htm)*

   The following figure shows the Flow Engine window after the configure step is finished.



```
Fri Apr  2 02:40:27 1999

Running DRC.
DRC detected 0 errors and 0 warnings.
Saving ll file in "watch.ll".
Creating bit map...
Saving bit stream in "watch.bit".

==================================================

xcpy watch.bit /home/rohit/Documentation/QSG/2.1iver/Watch/watch.bit

==================================================

xcpy watch.ll /home/rohit/Documentation/QSG/2.1iver/Watch/watch.ll
```

Figure 1-17 Configuring Design

9. The Flow Engine saves the configuration options in the BitGen Report. Review the report using the Report Browser. Verify that the specified options were used when creating the configuration data.

Step 11: Using the PROM File Formatter

If you are going to program a single device using the Hardware Debugger, all you need is a *design*.bit file. If you are going to program several devices in a daisy chain configuration, or program your devices using a PROM, you must use the PROM File Formatter (PFF) to create a PROM file. The PROM File Formatter accepts any number of bitstreams and creates one or more PROM files containing one or more daisy chain configurations.

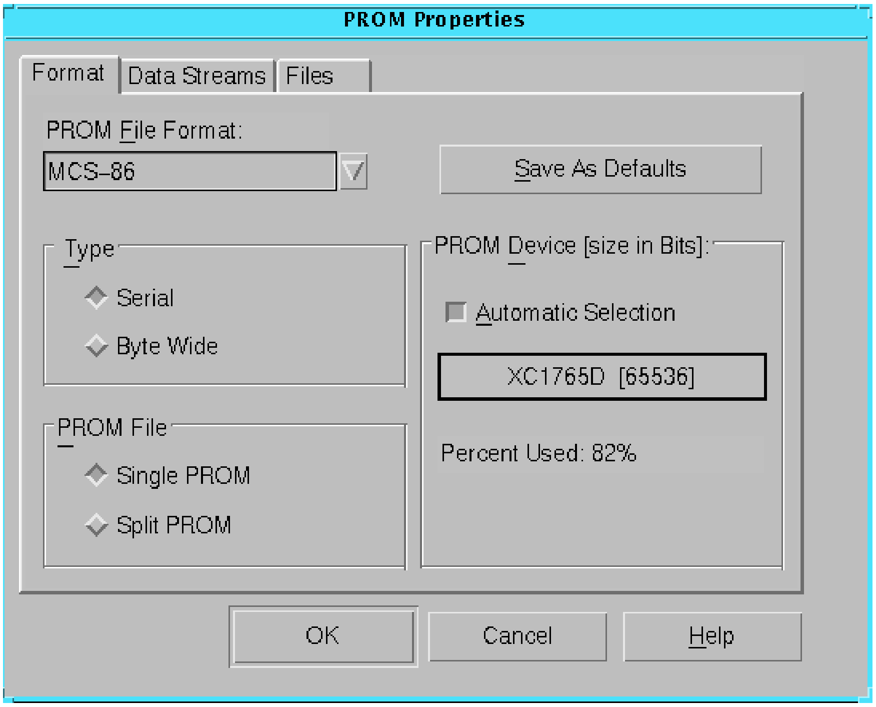
1. To start the PROM File Formatter, click the PROM File Formatter icon in the toolbox in the Design Manager.

   The PFF starts with a default PROM that matches the currently selected (configured) revision. At this point, you can add additional bitstreams to the daisy chain; create additional daisy chains; remove the current bitstream and start over; or immediately save the current PROM file configuration.

   The status bar at the bottom of the PFF window displays the PROM format, data format, current PROM size, and percentage of the selected PROM used by the current PROM configuration. The currently selected PROM is an XC1765D. 53,984 bits of data are required to hold the configuration bitstream for the XC4003E target device for this tutorial. The PFF determined that an XC1765D is the correct PROM because it can hold up to 65,536 configuration bits (or 82% full).

   The right half of the PFF window is a directory structure used for locating bitstreams. Only files with a .BIT extension are shown in the list. For detailed information on using the PROM File Formatter to create daisy chains or complex PROM configurations, see the *PROM File Formatter Reference/User Guide*. This tutorial describes how to save the default PROM file.

2. Select **`File`** $\rightarrow$ **`PROM Properties`** to open the PROM Properties dialog box, shown in the following figure.



PROM File Format: MCS–86

Type
- Serial
- Byte Wide

PROM File
- Single PROM
- Split PROM

PROM Device [size in Bits]:
- Automatic Selection
- XC1765D [65536]

Percent Used: 82%

Figure 1-18 PROM Properties Dialog Box with Single PROM

3. Select the following options in this dialog box.
   - PROM File Format from the drop-down list
   - PROM Type
   - Number of PROMS used to hold the data

     If you have more data than space available in the PROM, you must split the data into several individual PROMs with the Split PROM option. In this case, only a single PROM is needed. Click **OK** to accept the PROM Properties.
4. Select **File** $\rightarrow$ **Save** to save the PROM file.
5. Specify your working directory as the area where the PROM Description File will be saved.

   The PROM File Formatter saves both the PROM file (watch.mcs) and a PROM Description File (watch.pdr). The PDR file can be reopened if any changes are required. Verify that the files exist in your directory.

6. Select **`File`** → **`Exit`** to close the PROM File Formatter.

   This completes the tutorial. For more information on the Alliance design flow and implementation methodologies (especially some of the tools/programs that were not covered as part of this tutorial), please reference the online version of the Software Manuals at http://support.xilinx.com