Performance Lab

Performance Lab

Introduction

This lab shows how using location constraints along with proper primitive selection, can obtain peak performance from a pipelined design.

The SLOPIPE design needs to be implemented in an XC4005E-3 device. After implementing SLOPIPE, consider some ways to improve its performance, including timing constraints and location constraints. Finally, implement the PIPE project to see how much location constraints can improve the project performance.

About the SLOPIPE and PIPE Projects

The intention of this lab is to show that using location constraints can sometimes help improve the performance of a project. Although it is often unnecessary to use location constraints in a project, in certain designs they are very effective. FPGAs utilize pipelined applications very well because of their register rich architecture. Typically, applications that have a great deal of arithmetic, such as DSP designs, get terrific performance when pipelined. Although most designs do not usually perform better with location constraints, in this case, knowing that pipelining is being applied, gives experienced designers confidence in placing logic.

The SLOPIPE project (see figure 1), is a very basic pipelined designed that can easily be improved upon with a little experience. The PIPE project is the improved SLOPIPE project that gets significantly better performance by making some design changes and using location constraints in a UCF file (see figure2).

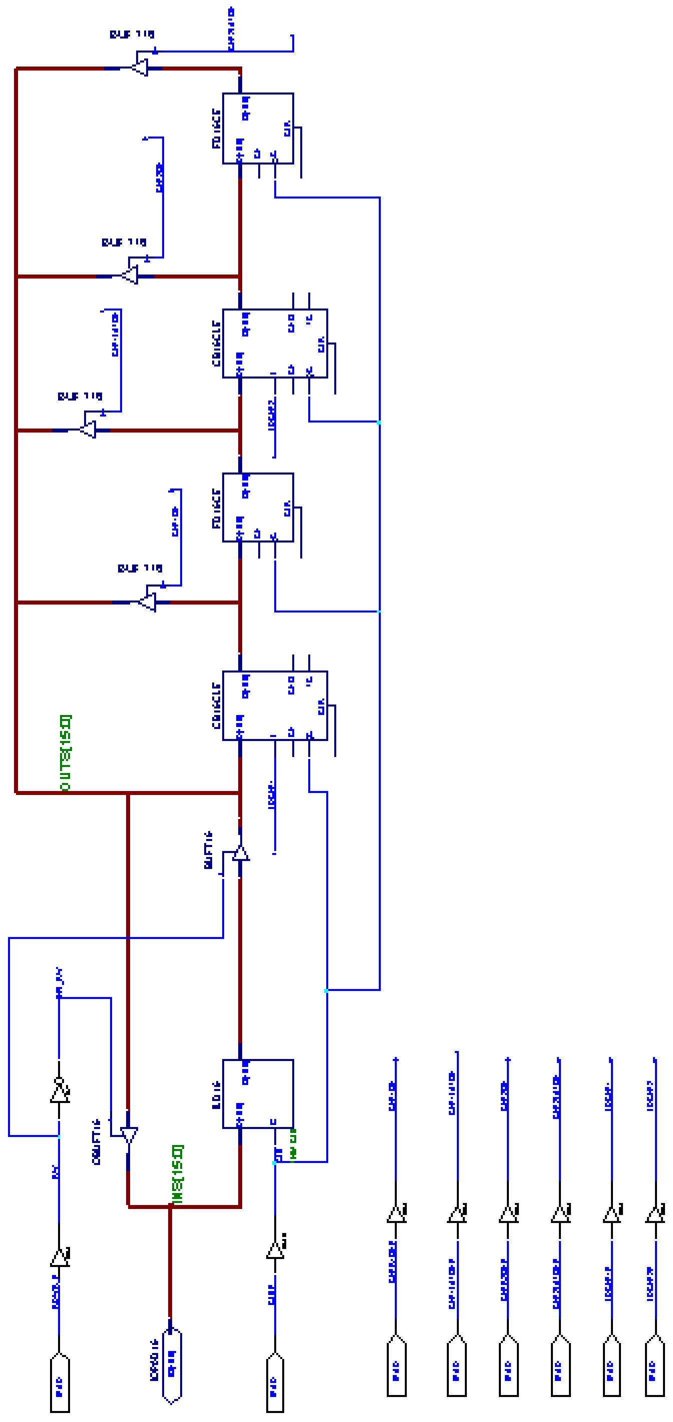


Figure 1. The SLOPIPE Project.

<u>About Attributes and Constraints</u>

Some of the most commonly used attributes are:

TNM	TNM=JOE

The TNM attribute tags specific flip-flops, RAMs, pads, and input latches as members of a group to simplify the application of timing specifications to the group. This allows making very path specific timing constraints.

TS	TS01=FROM:MOE:TO:LARRY=50NS

The value of the TS attribute corresponds to a specific timing specification that can then be applied to paths in a design.

Some of the most commonly used constraints are:

LOC	LOC=CLB_R3C5

The LOC constraint defines where a symbol's logic can be placed within an FPGA. It specifies the absolute placement of logic on the FPGA die. It can be a single location, a range of locations, or a list of locations. However, use this constraint sparingly, since its overuse can actually prevent the M1 software from finding a solution.

RLOC	RLOC=R1C2

Relative LOCation constraints (RLOC) group logic elements into discrete sets and permits the definition of the location of any element within the set relative to other elements in the set. This gives the M1 software more flexibility in placing the logic, since it is giving the tool some choice of placement.

RLOC_ORIGIN	RLOC_ORIGIN=R2C1

An RLOC_ORIGIN constraint fixes the members of a set at exact die locations. This constraint must specify a single location, not a range. This in effect makes an RLOC constraint into a LOC constraint, and gives the M1 software no control over the logic placement.

For more detailed information about attributes and constraints, refer to the "**Libraries Guide**" in the DynaText Browser.

Improving the performance of PIPE with a UCF file

The PIPE.UCF file has already been included in the project directory **`c:\F15_labs\pipe`**. The constraints entered into this file include the RLOC_ORIGIN and LOC constraints (see Figure 2).

```
INST $I42 : RLOC_ORIGIN = R1C1;
INST $I43 : RLOC_ORIGIN = R1C3;
INST $I21 : LOC=CLB_R*C2;
INST $I22 : LOC=CLB_R*C4;
```

Figure 2. The PIPE.UCF for the PIPE project.

Procedure

Implementing the SLOPIPE project

1) Open the "**Foundation Project Manager**", by clicking on **Start→Programs→Foundation Series→Foundation Project Manager**.

2) In the "**Foundation Project Manager**" window, click on **File→Open Project.**

3) Go to the **`c:\F15_labs`** directory within Foundation and select the "**SLOPIPE**" project, and then click on the "**Open**" button.

4) In the "**Foundation Project Manager**" window, open the "**Schematic Editor**" by clicking on its icon.

5) Note what the design looks like. It should appear exactly like Figure 1.

6) While in the "**Foundation Schematic Editor**", click on the "**M1**" button to generate an EDIF netlist and to automatically create a new project in the M1 software.

7) After the translation is complete, M1 will show the design as the current project in the "**M1 Design Manager**" window.

8) In the "**M1 Design Manager**" window, click on the left most button on the horizontal scroll bar, or click on **Design→Implement**. The Design Manager will prompt for the correct device by bringing up the "**Implement**" window. Make sure that the **XC4005E-3-PC84** is selected.

9) Select <**Options…**> and then checkmark "**Product Post Layout Timing Report**".

10) Click on “**OK**”, and click on the “**Run**” button within the “**Implement**” window. This will start the implementation process and bring up the “**Flow Engine**” window.

11) While the design is being processed, the Flow Engine window is brought up. Monitor the design's progress through the **Flow Engine’s Message** window.

Using the Timing Analyzer

12) To determine if the implementation has met the timing specifications, the short and simple way is to read the “**Post Layout Timing Report**” from the Report Browser. This report is only generated if you followed step (8), above.

13) Otherwise, you can use the Timing Analyzer to determine the performance results: Click on **Tools→Timing Analyzer**. Clicking on the “**Timing Analyzer**” icon on the Vertical Tool Bar will also start it.

14) In the “**Timing Analyzer**” window, click on the **Analyze→Design Performance**.

15) After reviewing this report, complete the chart and answer the question in the **Questions** portion of the lab. When this is completed exit the **M1 Design Manager**.

Implementing the PIPE project with the PIPE.UCF file

16) Open the “**Foundation Project Manager**”, by clicking on **Start→Programs→Foundation Series→Foundation Project Manager**.

17) In the “**Foundation Project Manager**” window, click on **File→Open Project.**

18) Go to the `c:\F15_labs` directory within Foundation and select the “**PIPE**” project. Then click on the “**Open**” button.

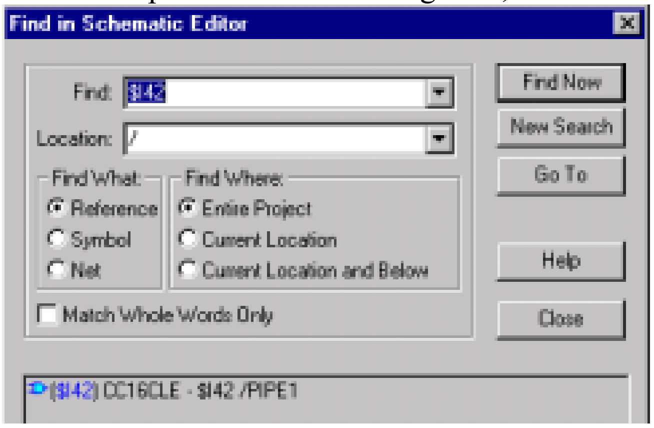
19) In the “**Foundation Project Manager**” window, open the “**Schematic Editor**” by clicking on its icon.

20) Double-click on the **PIPE.UCF** file in the upper left window to view the UCF file. Note these lines:

```
INST $I42 : RLOC_ORIGIN = R1C1;
INST $I43 : RLOC_ORIGIN = R1C3;
INST $I21 : LOC=CLB_R*C2;
INST $I22 : LOC=CLB_R*C4;
```

21) Go back to the **Foundation Project Manager** and select **Document→Find Object…**

22) Under **Find**, <u>type</u> "**`$I42`**", which is the reference name for one of the objects in the schematic. <u>Select</u> the other options as shown in Figure 3, and <u>click</u> <**Find Now**>.



($I42) CC16CLE - $I42 /PIPE1

Figure 3

23) <u>Highlight</u> reference **$I42/PIPE1** at the bottom, and <u>click</u> <**Go To**>. The Schematic Editor will appear with the chosen schematic selected in red. (You may need to select the "Full page zoom" icon once you get into the schematic editor.)

24) Similarly, you could find the other instances using the **Find Object…** command. However, to save time, just note that each successive flip-flop and counter block is given a location constraint indicated in the **UCF file**.

25) Compare the **PIPE** project to the **SLOPIPE** project in Figure 1 and note the differences between the two projects. Alternatively, you can <u>open</u> both schematics simultaneously in the schematic editor:
   A) In the schematic editor, <u>select</u> **File → Open**.
   B) Then <u>click</u> **Browse** >>. <u>Double-click</u> the directories until the **directories** location is **`c:\F15_labs\slopipe`**. Then <u>click</u> file **`PIPE1.SCH`** and <**OK**>.
   C) Now you can <u>select</u> Window 1 and compare it to Window 2 (the latest opened window).

26) While in the "**Foundation Project Manager**", <u>click</u> on the "**M1**" button to generate an EDIF netlist and to automatically create a new project in the M1 software.

27) After the translation is complete, M1 will show the design as the current project in the "**M1 Design Manager**" window.

28) In the "**M1 Design Manager**" window, <u>click</u> on the left most button on the horizontal scroll bar, or <u>click</u> on **Design→Implement**. The Design Manager will prompt for the correct device by bringing up the "**Implement**" window. Make sure that the **XC4005E-3-PC84** is selected.

29) The **`PIPE.UCF`** file needs to be specified as the constraints file within M1. <u>Click</u> on the "**Options**" button within the "**Implement**" window that was just brought up. At the top of the window, a box to enter the "**User Control File**" name will be seen. Browse for the

UCF file that was created. When it is found, highlight it and click on the "**Open**" button. The file will now appear in the User window. Also, make sure to check the box to "**Produce Post Layout Timing Report**".

30) While the design is being processed, the Flow Engine window is brought up. Monitor the design's progress through the **Flow Engine's Message** window.

31) After the implementation has been completed, either view the **Post Layout Timing Report**, or enter the Timing Analyzer to determine the maximum clock frequency and complete the chart in the **Questions** section of the lab.

32) After completing the timing analysis, exit the **M1 Design Manager**.

Questions

Project	Maximum Clock Frequency
SLOPIPE	
PIPE	

1) After reviewing the SLOPIPE design in figure 1, what possible recommendations could be made for improving the design's performance?

2) Compare the layouts of the PIPE and SLOPIPE projects by viewing the results of each implementation in the EPIC Design Editor.

Conclusion

This lab has shown that another way to meet the performance objectives is to enter a Location Constraint into a UCF file and control the Flow Engine's placement of the design. However, the M1 Development System is much more likely to meet performance objectives if location constraints are not placed on the design. Although this tool is capable of utilizing placement constraints, location constraints can prevent a fit from being found if the design is over constrained or has a high utilization.

Performance Lab Answers

Project	Maximum Clock Frequency
SLOPIPE	~ 30 Mhz
PIPE w/UCF	~ 40 Mhz

1) The following is a list of design changes that could improve the designs performance:

a) Use the global buffer resources of the device. Since the XC4000 family has eight global buffers designed to improve the performance of high fan-out control signals, routing the load and output enable signals will decrease the routing delay and save routing resources that might otherwise be useful in this project. By inserting BUFGs before the LDCNT1, LDCNT2, and RW signals, the performance will improve greatly. Since the other control signals on the tri-state buffers were not mentioned as being especially long, the remaining global buffers were not used for them.

b) Use the CC16CLE macros instead of the CB16CLE macros. This will improve the register-to-register delay within these counters, since they will use the carry logic feature of the XC4000 family. This also makes the design more compact and reduces the number of CLBs used.

c) Switch to Fast Slew Rate on the bidirectional bus. This will decrease the output time and hence improve the pin-to-pin performance. However, if the sixteen signals are all placed next to each other, ground bounce could become a problem if all the signals switch at one time and if good high-speed board design was not used. To reduce the risk of having ground bounce problems, make certain the board is made with proper high-speed board design skills.

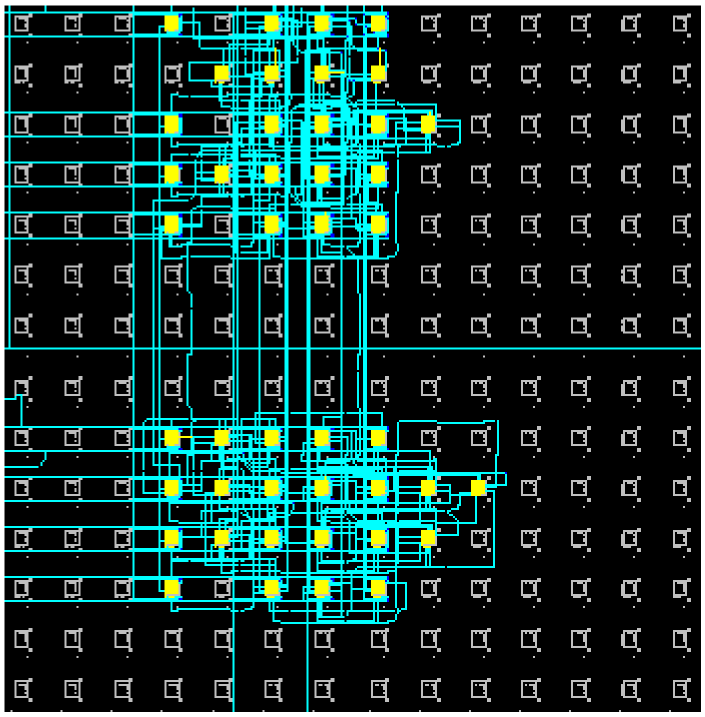
d) In general, if the logic can be placed very close together, it will reduce the net delays. The caveat to this is that the logic becomes very inflexible and the compiler may have trouble utilizing a great deal of the device. This of course always depends on the size of the device chosen and the design being implemented.

e) Choose a faster speed grade of device. Of course this will be more costly, but it will solve the problem.

f) Let the compiler continue trying to place and route the design without any timing specifications. This is usually chosen as a last resort, but will sometimes produce surprising results. Select placement effort = 5 in the implementation options dialog box, and select a number of delay-based cleanup passes to be made by the router.

g) Try using the Multi-Pass Place and Route or the Re-entrant Routing tools.

2) EPIC provided the following layout for the SLOPIPE project (note that the logic is dispersed). Your results may vary.



EPIC provided the following layout for the PIPE project with a UCF file (note that the logic was placed with the location constraints from the UCF file). Your results may vary.