ASIC Design Laboratory
Lab 11 : IC Layout Process Analysis and Critical Timing Tuning
Lab Manual

Fall 2020

The purpose of this lab is to provide you with exposure to layout-based timing analysis, timing tuning, and the details of the layout process you executed in the previous lab.

In this lab, you will perform the following tasks:

- Evaluate what happens during of the various steps for a standard-cell place-and-route flow with Innovus.
- Generate and review Innovus Timing Analysis Reporting of the provided design's Layout.
- Adjust the IO positioning in Layout of the a design in order to tune it's critical path and general timing characteristics.

1 Lab Setup

In a UNIX terminal window, issue the following commands, to setup your Lab 11 workspace:

mkdir –p ~/ece337/Lab11
cd ~/ece337/Lab11
dirset
setup11

The setup11 command is an alias to a script file that will check your Lab 11 directory structure and give you file needed for starting the lab. If you have trouble with this step please ask for assistance from your TA.

IMPORTANT: Make sure to add this new workspace into your 337 Repository, like you did in Lab1.

This way, you will always have the original copy in storage.

Before moving on to exploring the layout process, we need to first synthesize the mapped form of the provided design and add a pad-frame to it like in the previous lab. To do so, execute the following commands from your Lab11 directory:

make mapped_layout_lab_design

Followed by:

pads layout_lab_design

2 Analysis of the Layout Process

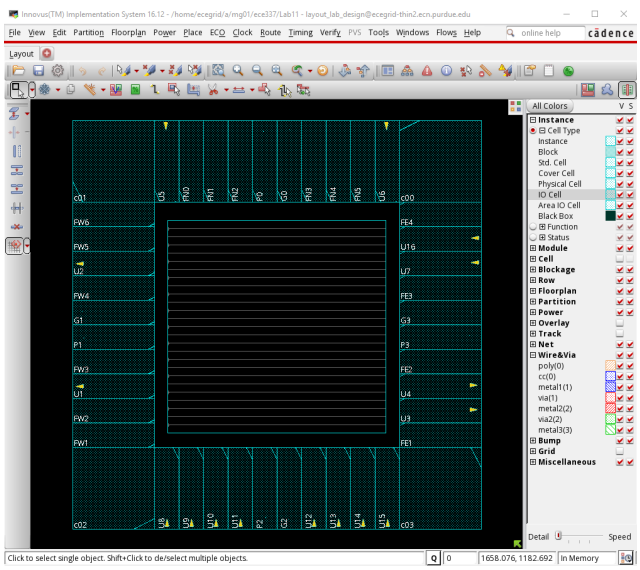
As you witnessed during the prior lab, standard cell place-and-route (or layout) of designs is a near fully automated process that is controlled via predefined scripts or 'flows' of tool commands. These various commands are used to both step the tool and provide it with designer insights to help with the challenging problem of of effectively placing and routing (connecting) all of the components for a design. Optimal design layout, even with the help of functional cells with standardized geometries is an problem that's incredibly difficult to solve in reasonable time. In fact, it is part of the set of problems that are considered NP-Complete (which will be covered in the algorithms course if you are not already familiar with the term). Therefore, layout approaches have to be design based on various heuristics and often use a semi-randomized placement followed by various optimization passes to have a chance at finishing in a reasonable time-span.

2.1 Innovus Configuration for a Design Run

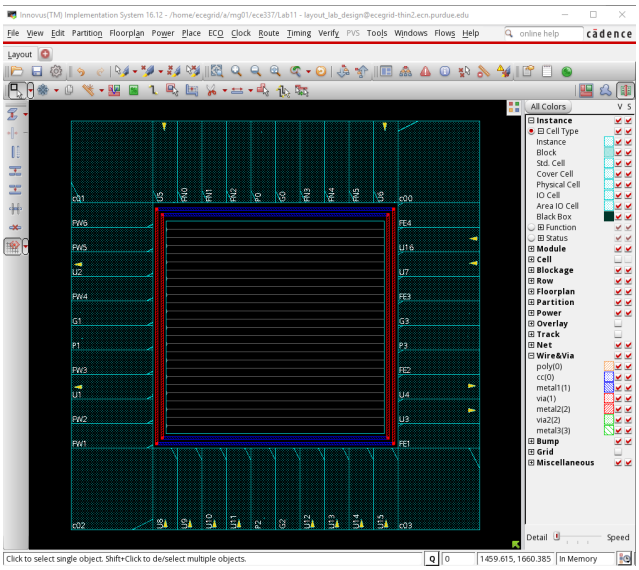
Before we can really deal with the design's contents, we first need to configure the tool with a variety of settings and information about the design's external connections, shape, size, constraint files, and reference libraries. This has been packaged for you in the 'setup.tcl' script and it's supporting script 'init.tcl'. Please look over those two scripts. When you are ready to proceed issue the following commands in the Innovus terminal.

source setup.tcl

Once it finishes, your Innovus should look similar to Figure 1(a). At this step our layout has just the basic scaffolding including a minimally sized IO pad-frame and an initial core box with row spacing/slots for standard cells to placed later.



(a) Innovus after initial setup



(b) Innovus after adding supply rings

Figure 1: Initial Skeleton for Design Layout

Next we need to establish a starting placement of the cells for the design and their needed power and ground supplies. First we are going to add the power and ground supply rings that will be used to connect the power and ground pads with the core while allowing for both to be easily and uniformly connected to the grid of cells that will be formed in the core section. To do this run the 'supply_rings.tcl' script with the following command in the Innovus terminal.

source supply_rings.tcl

Your Innovus should now look like Figure 1(b). Investigate it and answer the following questions on your evaluation sheet.

Question: Which which layers are used for the sections of the supply rings?

Question: What are the widths of the supply ring bars and their spacing?

Question: What is the spacing between the inner supply ring and the core area, and which prior command controls this?

2.2 Initial Cell placement and Power connections

Now that we have a skeleton in place for the design, it's time to add the design's functional cells. The 'place_cells.tcl' contains the commands and settings for handling the somewhat random initial starting placement of the cells. This staring placement has the cells slightly clustered based on their relative connections in the mapped file and will be tuned via the various optimization passes later in this flow. To place the cells, run the following command in the Innovus terminal.

source place_cells.tcl

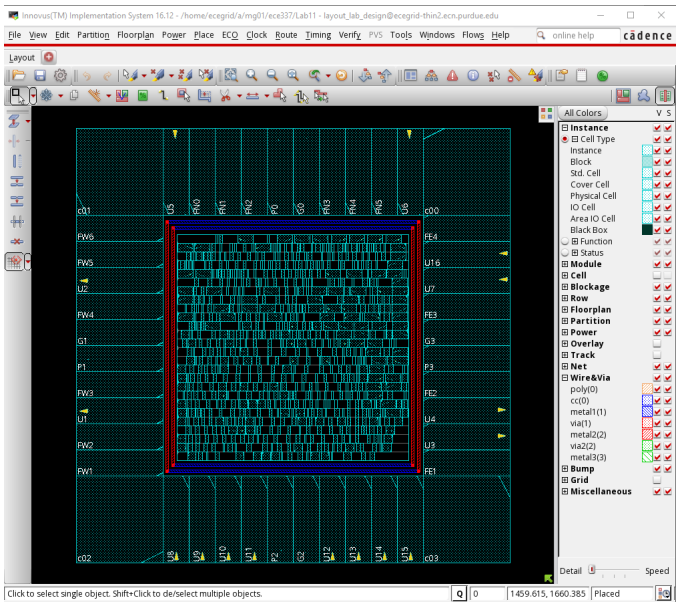


Figure 2: Innovus after initial cell placement

Your Innovus should now look similar to Figure 2. Investigate the current layout and answer the following questions on your evaluation sheet.

Question: How did the layout change after running the 'place_cells.tcl' script?

Question: What is a rough estimate of the core utilization based on the cells?

Next we need to connect the cells together and to the power and ground supplies. First we'll route the power connections since they follow a planned pattern and we need to reserve their space in the various metal layers. To do run the following command in the Innovus terminal.

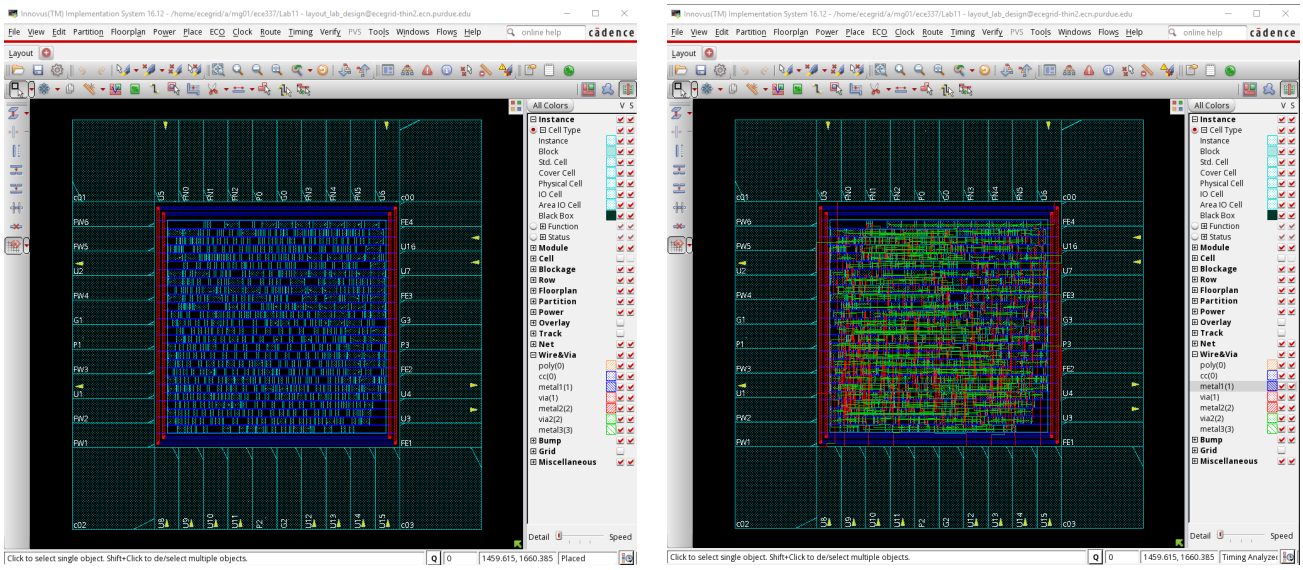
sroute

Your Innovus should now look similar to Figure 3(a). Investigate the current layout and answer the following question on your evaluation sheet.

Question: How did the layout change after running the 'sroute' command?

With power connected to the cells, we are now going to do an initial routing of connecting wires for the logical connections to the cells. The set of commands for doing this are listed in the 'pre_cts_route.tcl' script. The name for this script designates that this will be the routing done before the clock tree synthesis work would be done. Look over the script and then run it using the following command in your Innovus terminal.

source pre_cts_route.tcl



(a) After routing of power and ground connections

(b) After routing logic connections

Figure 3: Innovus at different stages of initial routing

Your Innovus should now look like Figure 3(b). Investigate the updated layout and answer the following question on your evaluation sheet.

Question: In what ways are the three metal layers typically used in the routing?

NOTE: As a hint, think about similarities in how they are drawn.

Now that we have everything connected, we can start optimizing the design after accounting for the various wire delays and loading effects that routed connections have. However, since clock-tree synthesis (CTS) hasn't been performed yet the clock pin is currently connected in an unbalanced way so clock skew should be filtered out from consideration during this stage of optimization. Look over the 'pre_cts_opt.tcl' script and then run it via the following command in your Innovus terminal.

source pre_cts_opt.tcl

In order for large sequential designs to work well, the clock connections need to be converted from raw wire connections to a balanced tree of buffers and wire connections. This is needed for two primary reasons and is referred to as clock-tree synthesis (CTS). Firstly, it is paramount that the clock signal's edges arrive at every flip-flop as close to the same point in time as possible, as any difference in arrival times (skew) can result in correct values being missed or intermediate values being grabbed. The other reason is to manage the fan-out (how many cells are driven from the same source) of the clock signal's IO pad buffer/driver and the buffers at each subsequent level of the synthesized tree. Unfortunately, the current version of Innovus's CTS commands no longer correctly work the standard cell library used by this class, so we are going to have to skip this step for this lab. While this does not prevent the clock signal from being connect, it does mean that the design will likely have fairly bad clock skew in this layout and the library would have to be updated in order to fabricate a realistic IC.

The next step is to do another routing and optimization pass after the clock-tree synthesis has been done (or would have been). This round of optimization will attempt to improve the overall balance of design timing after now accounting for clock skew being present, as well as adjusting the tree's buffer placement and connection routing if needed. Look over the 'post_cts_opt.tcl' script and then run it via the following command in your Innovus terminal.

source post_cts_opt.tcl

Now we are ready to finalize the design and perform one last round of optimization after everything is in place to do more thorough analysis of circuit timing as well as various fabrication focused design rules, such as filler cells and the metal fill you saw in the prior lab. For this lab we aren't going to worry about metal fill as it will just occlude any other changes that you see happening during this step. Look over the 'final_route.tcl' script and then run it via the following command in your Innovus terminal.

source final_route.tcl

Your Innovus should now look similar to Figure 4. Investigate it and answer the following question on your evaluation sheet.

Question: What percentage of the core appears to be filled with cells after running 'final_route.tcl'? And which command do you think primarily resulted in the change?

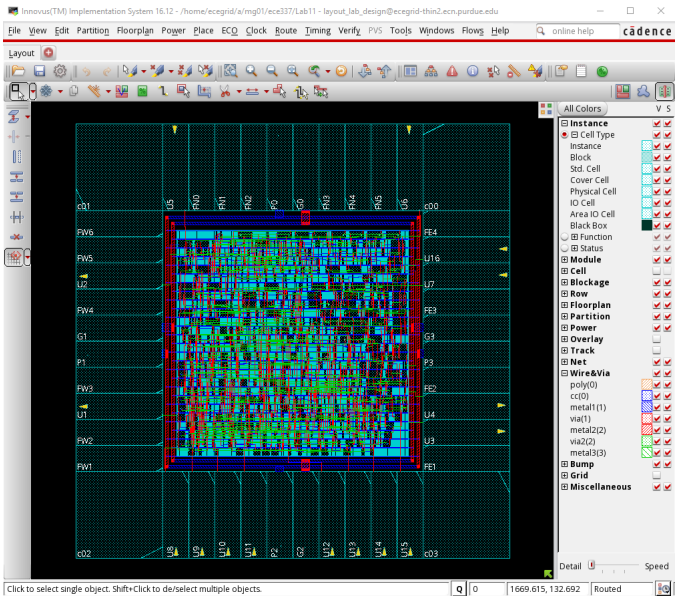


Figure 4: Innovus after running 'final_route.tcl'

Now that filler cells have been place and the full view of the design and fabrication constraints can be analyzed, we need to last round of design optimization. Look over the 'final_opt.tcl' script and then run it via the following command in your Innovus terminal.

source final_opt.tcl

We now have an optimized design (or at least as optimized as the tool could make it) and would now be concerned with making sure it meets it's various constraints, from both timing and fabrication requirements. You should already be somewhat familiar with verifying fabrication design rules, such as geometry and connectivity requirements, from the prior lab. After timing and design rules have been met the next step would be to have Innovus export at lower level (mask level) representation of the design for more detailed analysis in a tool (like Virtuoso) which works at the transistor detailed masking layer level. If you are curious about that step then look over the provided 'export.tcl' script.

For this lab the next major step to consider is how to both verify and potentially tune chip-wide circuit timing.

3 Performing Timing Analysis on a Layout

To run the timing analysis, you will need to specify the timing constraints. Innovus can generate its own timing constraints based on common .sdc formats. The setup11 script provided you with both .sdc constraint files needed for this analysis. Examine their format. In a full flow run through timing analysis would be automatically run and reported as part of the flow. However, as a designer you also need to get familiar with running timing reports and debugging them in order to identify and fix potential problems with a design layout. To get started generate one for the post routing stage of the design.

Timing → Report Timing

This will bring up another window as shown in Figure 5.

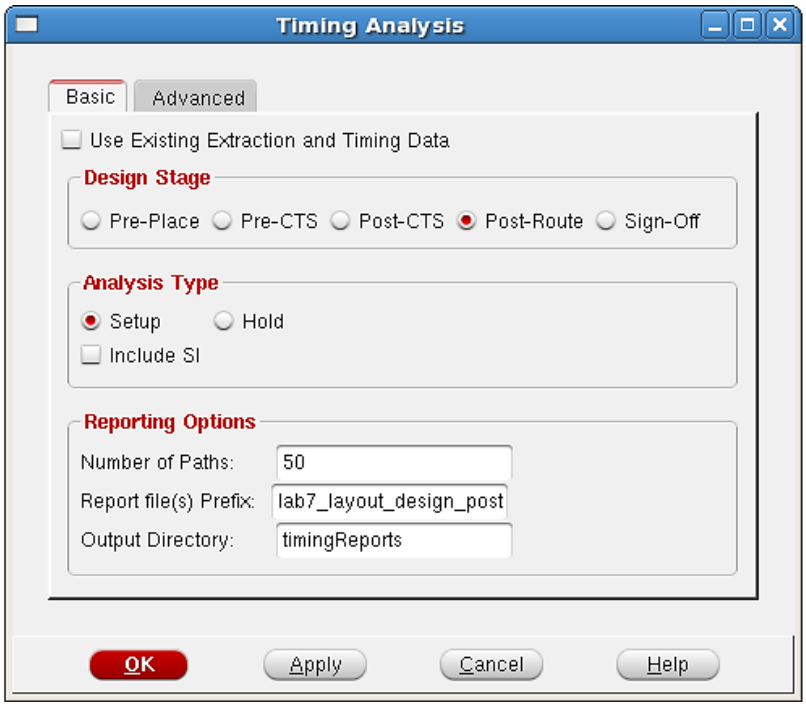


Figure 5: Timing Analysis Configuration

Set the settings appropriately for your design and click OK.

Now that the reports have been generated, one can either directly look at the report files (saved as ASCII text files in the 'timingReports' folder. We can also use the built in 'Timing Debug' tool as long as there is at least one path with negative slack. For this lab, the current setup is guaranteed to yield a critical path violation/timing issue, so we'll use the 'Timing Debug' tool.

Timing → Debug Timing

You should now see a window like Figure 6

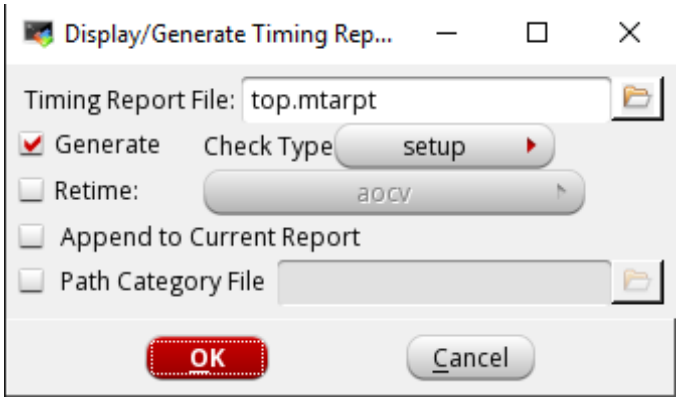


Figure 6: Debug Timing Configuration

Leave the settings as default and click 'OK'.

This should open the window for the Timing Debugger, which should look similar to Figure 7 and shows a summary of the observed critical paths with both a histogram of their respective slack amounts and lists of each involved path below.

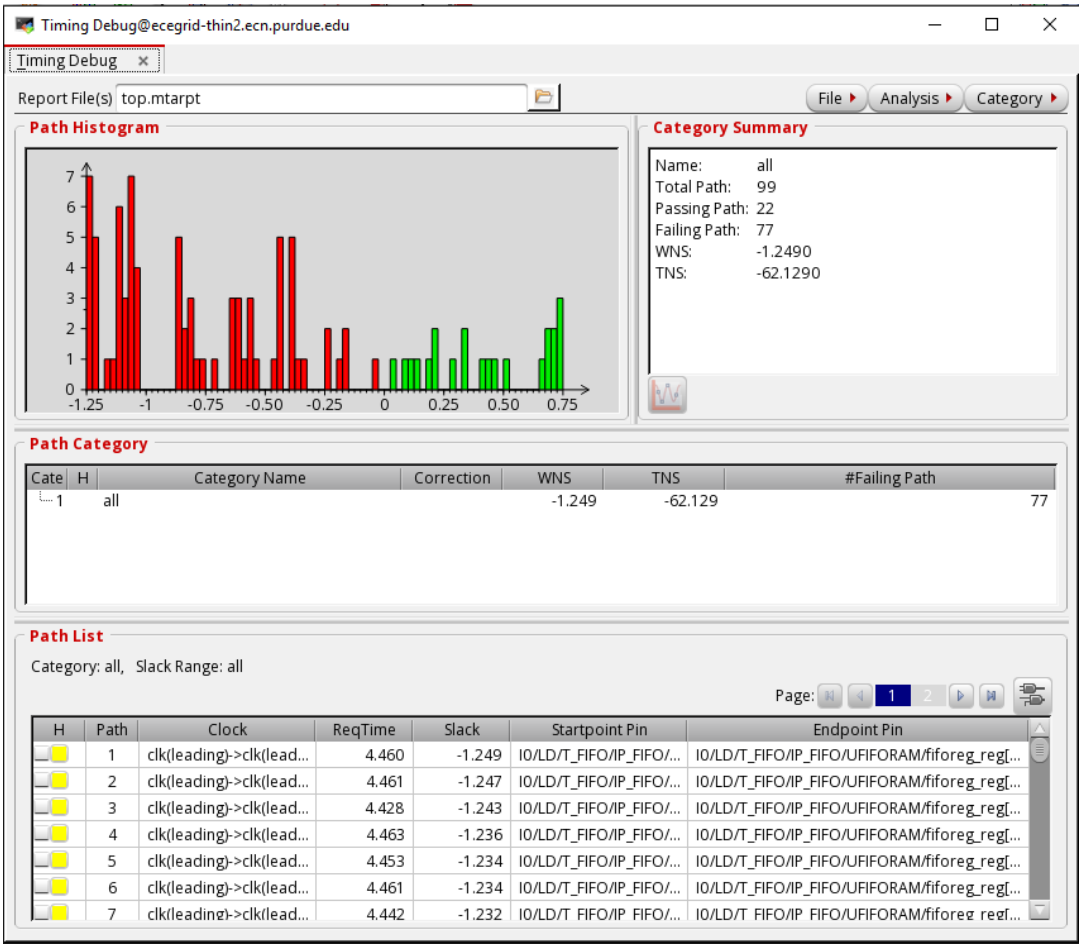


Figure 7: Timing Debugger Window

From the list, you can select a path by double clicking on the path and the selected path will be highlighted in your layout. It will also create a window focused on the timing analysis of that path (similar to Figure 8, which has a useful feature that shows the path in schematic mode if you select the Schematic tab. This timing path browser will be very important for diagnosing timing issues and discerning if synthesis and layout optimizations should be strong enough to help or if the design is going to need architectural changes in order to realistically meet the timing constraints. In general, if the browser shows a large percentage of the paths violating the timing requirements (highlighted by red bars on the histogram) or there are paths with very large timing violations, mostly like there will need to be design changes, in order to meet timing. Explore the different subsections of the timing browser by selecting one of the paths from the table and activating that different info/viewing modes for it.

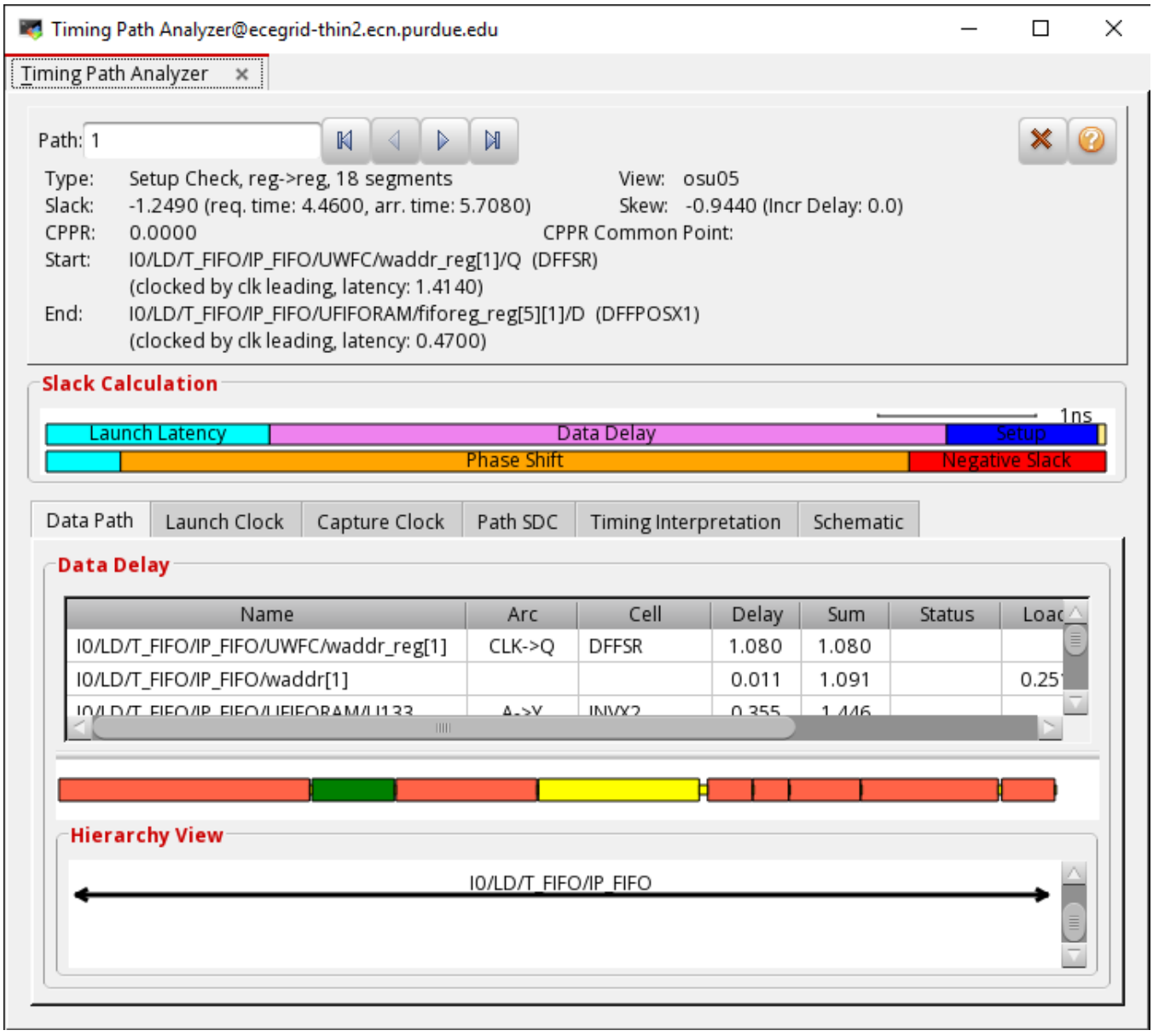


Figure 8: Timing analysis window for a selected critical path

After opening the Timing Path Analyzer for the most critical path (the first one in the list) answer the following questions on your evaluation sheet.

Question: What are the starting and ending gate/cell level components of the critical path?

Question: How many combinational logic cells are in between?

Question: What are the starting and ending blocks (design modules) of the critical path?

Question: Is the synthesis critical path similar to the layout one, and what are the their respective delays?

For the synthesis critical path, you will need to look the design's report in the 'reports' folder. It is possible for you to modify the synthesis script so it will show more paths and a few more options in the report. To do this, you must find out the options you want by opening the manual for "report_timing" command in dc_shell-t and then modifying your makefile accordingly. Once you written down the answers to the questions above, have a TA check off your work up to this point.

4 Tuning Critical Path Timings

In general tuning the timing of the whole chip in a couple of ways depending on how bad the current timing is. When timing is close, you may be able to improve it by simply rerunning the synthesis and layout flow with tighter constraints. However, there are limits to this and sometimes overly tight constraints will result in the tool having too much error to between actual and required such that it effectively 'just gives up'.

Another approach is is that sometimes the position of the IO pads can be negatively affecting the layout of the design from either overcrowding a region or spreading out cells that could be better clustered. For this lab we are going to focus on this aspect. The IO pad-frame configuration provided seems like a fairly logical one in terms of grouping, but it is actually results in a sub-par placement of design cells. Generally logic cells will be placed near where they get their inputs and where they might be used. Your next task is to make changes to only the pad-frame configuration, while keeping the same size and shape with 10 pads per side, such that you reduce the amount of negative slack in the design. To help you in this is the following list of design insights:

- The 'd_plus' and 'd_minus' pads/toplevel ports are driven by the internal USB focused portions of the design including a shift register, which are controlled at a high level by the 'transmit' port.
- The 'write_data', 'write_enable', 'fifo_empty', and 'fifo_full' ports/pads all directly either come from or control a FIFO module inside the design, which is where data is housed before being loading in the USB-side's shift register.

Once you have improved the design's timing, complete the two related questions on evaulation sheet and have it sign-off by a TA.

5 Closing Remarks

- Turn in your check-off sheet at the beginning of Lab during the first week of the Cooperative Design Lab.