Understanding ACE Timing Reports (AN024)

Achronix
Data Acceleration

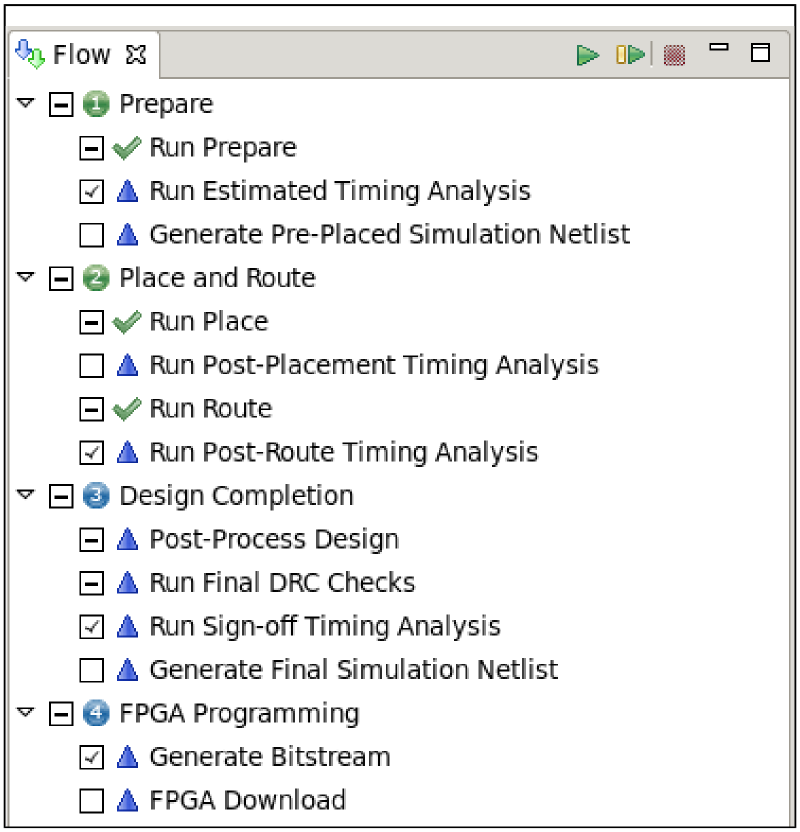
January 29, 2021 **Application Note**

Accurate timing constraints and proper understanding of timing analysis reports are critical to successful FPGA design projects. This application note introduces ACE users to the structure of ACE timing reports generated during an ACE place-and-route run. It shows how basic timing constraints are used to drive static timing analysis (STA), and how these constraints are represented in timing reports.The following sections describe the timing reports generated by the ACE flow, as well as how they are presented in the ACE GUI.

Timing analysis is driven by SDC, with the minimum constraint to any design being the specification of a clock via a `create_clock` constraint. For details of all SDC timing commands supported by ACE, refer to the section, SDC Commands, in the *ACE User Guide* (UG070) .

Configuration of the Default Timing Reports

The ACE place and route flow is integrated with STA, which is run throughout the flow. Timing reports are generated if the user enables the various timing analysis steps as shown here, in the **Project View** → **Flow** tab:



Flow

- ⊟ 1 Prepare
  - ⊟ ✓ Run Prepare
  - ☑ Run Estimated Timing Analysis
  - □ Generate Pre-Placed Simulation Netlist
- ⊟ 2 Place and Route
  - ⊟ ✓ Run Place
  - □ Run Post-Placement Timing Analysis
  - ⊟ ✓ Run Route
  - ☑ Run Post-Route Timing Analysis
- ⊟ 3 Design Completion
  - ⊟ Post-Process Design
  - ⊟ Run Final DRC Checks
  - ☑ Run Sign-off Timing Analysis
  - □ Generate Final Simulation Netlist
- ⊟ 4 FPGA Programming
  - ☑ Generate Bitstream
  - □ FPGA Download

Figure 1: *Selected Timing Analysis Steps in Flow*

To enable specific timing reports during the flow, ensure that the box to the left of the report step to be run is checked. These include:

- Run Estimated Timing Analysis
- Run Post-Placement Timing Analysis
- Run Post-Route Timing Analysis
- Run Sign-off Timing Analysis

The reports above are generated during the flow, and the user can explore these reports in the ACE GUI while the flow is running. All of the reports generated are located in the active implementation directory (default: `impl_1`) `reports/` sub-directory and come in three formats:

- HTML which is the version available in the ACE GUI
- CSV (comma delimited)
- TXT, which is the raw ASCII output

Running Timing Analyses

Run Estimated Timing Analysis

Prior to running placement, you can run an estimated timing analysis. For a new design it is recommended to run timing at this stage, and to look at the results in order to get a sense of the timing paths, as well as to observe how the timing constraints are being applied by the timer.

At the Estimated Timing Analysis stage, instance and net delays are estimates that can change after placement, and again after routing. However, some of the timing numbers in the reported timing paths are static and come from the SDC inputs to ACE. These define the arrival times of the clocks input ports (i.e., clock insertion delays), as well as the data arrival times and required times at the IPINs and OPINs. These arrival and required times appear in all timing reports. The most basic timing constraints that govern this come in the form of SDC commands.

> **Note**
>
> Clock insertion delays in these reports are estimated as are the data path delays. For flop-to-flop timing paths in the same clock domain, the clock skew will be zero.

Run Post-Placement Timing Analysis

After running placement the user can run timing analysis reports again by clicking on the box next to the Run Post-Placement Timing Analysis step in the **Project Perspective → Flow tab.** By default, the placed timing reports are not enabled.

The timing constraints shown in these reports will be identical to the constraints show in the Estimated Timing Analysis reports. However, now the cells in the path have been placed and the delay through the cell instances and their respective nets will have changed. Therefore, the resultant timing paths reported at this stage will be different from the paths in the Estimated Timing Analysis reports.

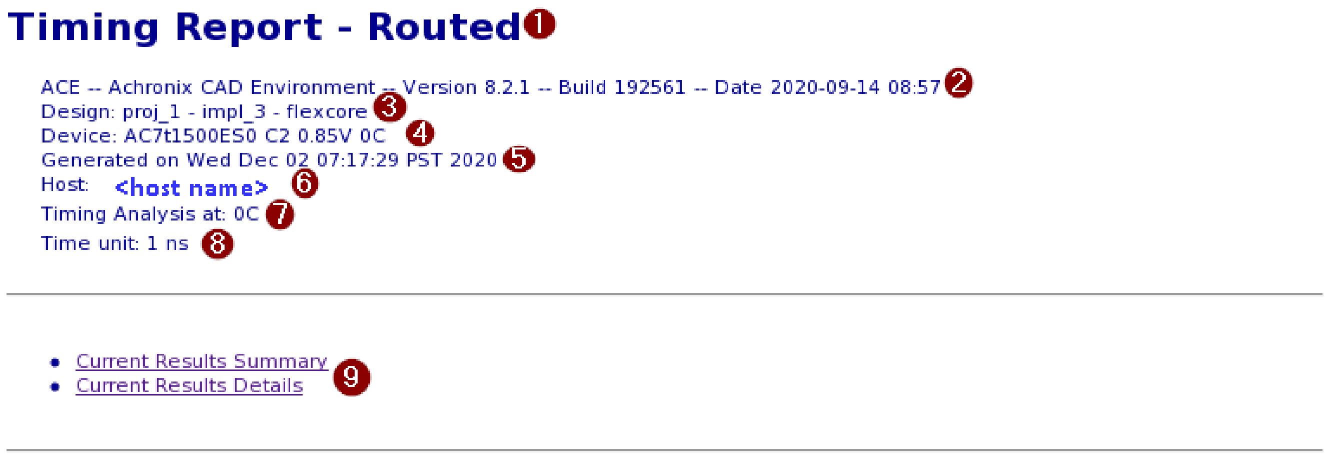
Run Post-Route Timing Analysis

After routing the design a user can run timing analysis. These reports use the same timing corner as the previous reports and have exactly the same results as the final timing report generated in the Run Sign-off Timing Analysis flow step. These reports include instance location for all instances reported in the timing path. The instance sites are listed to the right of each of the instance input pins.

Run Sign-off Timing Analysis

At the end of the full flow, the user can run the sign-off timing analysis. All of the timing slacks reported must be positive (highlighted in green) in order to ensure that the design works in silicon.

Timing Report Sections



Timing Report - Routed

ACE -- Achronix CAD Environment -- Version 8.2.1 -- Build 192561 -- Date 2020-09-14 08:57
Design: proj_1 - impl_3 - flexcore
Device: AC7t1500ES0 C2 0.85V 0C
Generated on Wed Dec 02 07:17:29 PST 2020
Host: <host name>
Timing Analysis at: 0C
Time unit: 1 ns

- Current Results Summary
- Current Results Details

Figure 2: ***Timing Report Header***

Below are the annotations for the figure above.

1. The first line in the report indicates to which step in the flow the timing report is related: Timing Report - <flow step>.
2. The second line indicates the ACE build version.
3. The third line indicates the <active project> - <active impl> - <design name>.
4. The fourth line contains the name of the device, as well as the timing corner (speed grade, core voltage, junction temperature) specific to the report. The timing corner will be the same as the corner defined in the Project Perspective → Design Preparation tab.
5. The fifth line indicates the date and time that the report was generated.
6. The sixth line indicates the host name where the ACE run is running when the report was generated.
7. The seventh line indicates the temperature corner. It is the same value as the junction temperature for Speedster FPGAs, but for Speedcore eFPGAs it can have different values depending on the temperature corner.

8. The eight and last line is the time units that are used in all of the timing report delay and slack related entries.
9. The timing reports are divided into two main sections: **Summary** and **Details**.

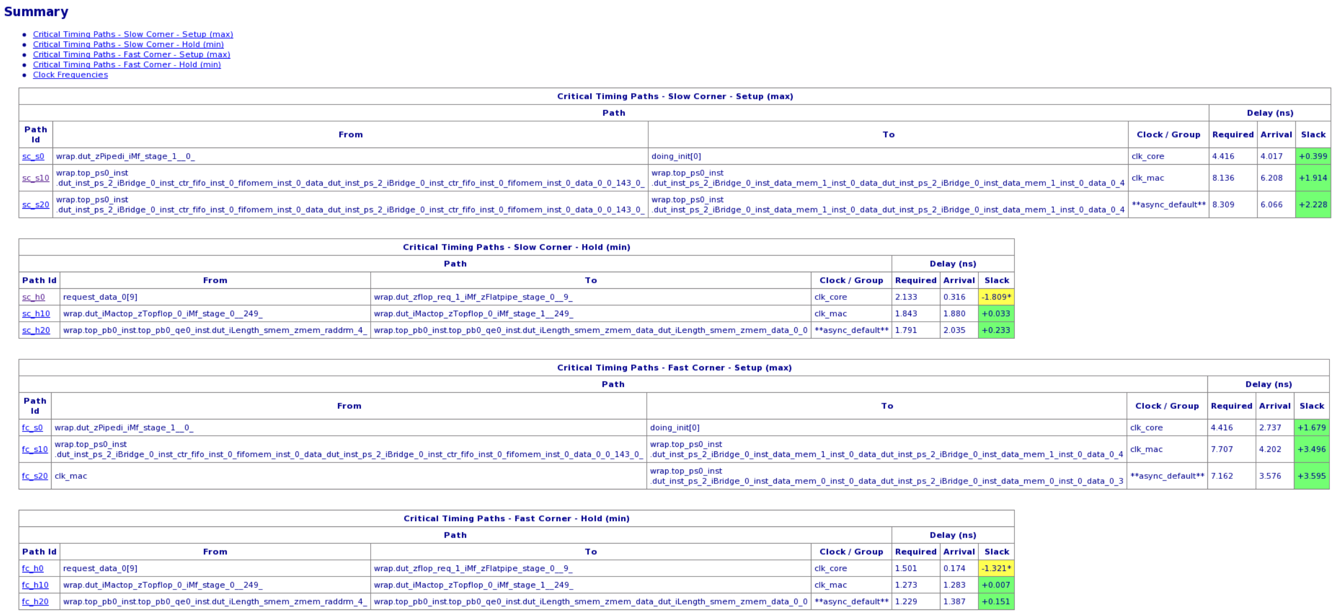
Note

It is recommended to run the ACE flow using the lowest temperature corner as this typically produces worst-case cycle times. This corner is design dependent and, therefore, it is worthwhile running the ACE flow at all temperatures to see which one achieves the best final timing. Also, regardless of which temperature ACE is configured in the **Design Preparation** → **Junction Temperature** setting, all temperatures are measured by the timer and timing issues resolved by the flow.

The HTML version of the file enables the user to navigate the report file using HTML hypertext buttons indicated with **blue** text in the GUI at any time.

Current Results Summary

The timing slack numbers are highlighted to help draw attention to them. In the following report there are negative slack numbers (highlighted in yellow) , because the slack value reported is a violation. A positive slack is highlighted in green.



Summary

- Critical Timing Paths - Slow Corner - Setup (max)
- Critical Timing Paths - Slow Corner - Hold (min)
- Critical Timing Paths - Fast Corner - Setup (max)
- Critical Timing Paths - Fast Corner - Hold (min)
- Clock Frequencies

Critical Timing Paths - Slow Corner - Setup (max)						
Path				Delay (ns)		
Path Id	From	To	Clock / Group	Required	Arrival	Slack
sc_s0	wrap.dut_zPipedi_iMf_stage_1__0_	doing_init[0]	clk_core	4.416	4.017	+0.399
sc_s10	wrap.top_ps0_inst .dut_inst_ps_2_iBridge_0_inst_ctr_fifo_inst_0_fifomem_inst_0_data_dut_inst_ps_2_iBridge_0_inst_ctr_fifo_inst_0_fifomem_inst_0_data_0_0_143_0_	wrap.top_ps0_inst .dut_inst_ps_2_iBridge_0_inst_data_mem_1_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_1_inst_0_data_0_4	clk_mac	8.136	6.208	+1.914
sc_s20	wrap.top_ps0_inst .dut_inst_ps_2_iBridge_0_inst_ctr_fifo_inst_0_fifomem_inst_0_data_dut_inst_ps_2_iBridge_0_inst_ctr_fifo_inst_0_fifomem_inst_0_data_0_0_143_0_	wrap.top_ps0_inst .dut_inst_ps_2_iBridge_0_inst_data_mem_1_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_1_inst_0_data_0_4	**async_default**	8.309	6.066	+2.228

Critical Timing Paths - Slow Corner - Hold (min)						
Path				Delay (ns)		
Path Id	From	To	Clock / Group	Required	Arrival	Slack
sc_h0	request_data_0[9]	wrap.dut_zflop_req_1_iMf_zFlatpipe_stage_0__9_	clk_core	2.133	0.316	-1.809*
sc_h10	wrap.dut_iMactop_zTopflop_0_iMf_stage_0__249_	wrap.dut_iMactop_zTopflop_0_iMf_stage_1__249_	clk_mac	1.843	1.880	+0.033
sc_h20	wrap.top_pb0_inst.top_pb0_qe0_inst.dut_iLength_smem_zmem_raddrm_4_	wrap.top_pb0_inst.top_pb0_qe0_inst.dut_iLength_smem_zmem_data_dut_iLength_smem_zmem_data_0_0	**async_default**	1.791	2.035	+0.233

Critical Timing Paths - Fast Corner - Setup (max)						
Path				Delay (ns)		
Path Id	From	To	Clock / Group	Required	Arrival	Slack
fc_s0	wrap.dut_zPipedi_iMf_stage_1__0_	doing_init[0]	clk_core	4.416	2.737	+1.679
fc_s10	wrap.top_ps0_inst .dut_inst_ps_2_iBridge_0_inst_ctr_fifo_inst_0_fifomem_inst_0_data_dut_inst_ps_2_iBridge_0_inst_ctr_fifo_inst_0_fifomem_inst_0_data_0_0_143_0_	wrap.top_ps0_inst .dut_inst_ps_2_iBridge_0_inst_data_mem_1_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_1_inst_0_data_0_4	clk_mac	7.707	4.202	+3.496
fc_s20	clk_mac	wrap.top_ps0_inst .dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_0_3	**async_default**	7.162	3.576	+3.595

Critical Timing Paths - Fast Corner - Hold (min)						
Path				Delay (ns)		
Path Id	From	To	Clock / Group	Required	Arrival	Slack
fc_h0	request_data_0[9]	wrap.dut_zflop_req_1_iMf_zFlatpipe_stage_0__9_	clk_core	1.501	0.174	-1.321*
fc_h10	wrap.dut_iMactop_zTopflop_0_iMf_stage_0__249_	wrap.dut_iMactop_zTopflop_0_iMf_stage_1__249_	clk_mac	1.273	1.283	+0.007
fc_h20	wrap.top_pb0_inst.top_pb0_qe0_inst.dut_iLength_smem_zmem_raddrm_4_	wrap.top_pb0_inst.top_pb0_qe0_inst.dut_iLength_smem_zmem_data_dut_iLength_smem_zmem_data_0_0	**async_default**	1.229	1.387	+0.151

Figure 3: ***Current Results Summary***

Each of the Summary and Details sections are divided into five timing sections, which are all specific to the timing corner defined in the header. In the Summary section, the Critical Timing Paths show the worst-case timing (timing that has the least amount of slack) for each section. In the Details section, there are the top ten paths (10 worst-case paths) reported for each section. Each of these sections report on each of the path groups defined in the different clock group sub-sections.

Note

By default, each defined clock creates a path group automatically, reporting unique sub-sections for each clock. Therefore, there are three setup/hold clock groups in the summary section above. The path ID for the 10 worst-case paths in the first path group is numbered from sc_s0 to sc_s9, for the second path group it is numbered from sc_s10 to sc_s19 and for the third path group it is numbered from sc_s20 to sc_s29.

Critical Timing Paths - Slow Corner - Setup (max)

This section has a path ID of sc_s0, and is the worst-case setup timing, using all of the max and late delays. This path ID is typically the worst-case for setup timing for the given timing corner.

Critical Timing Paths - Slow Corner - Hold (min)

This section has a path ID of sc_h0, and contains worst-case hold timing, using all of the max and late delays. This path ID can have the worst-case hold timing if the source of the hold timing violation is clock skew on a short data path.

Critical Timing Paths - Fast Corner - Setup (max)

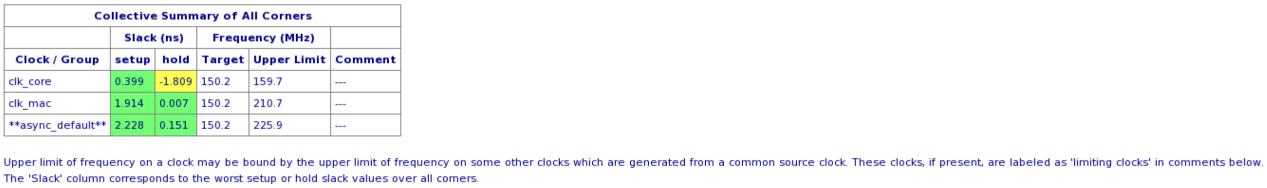
This has a path ID of fc_s0, and is the worst-case setup timing, using all of the min and early delays. This path ID typically will not contain the worst-case timing, and if there are violations in this corner, the timing might be difficult to close.

Critical Timing Paths - Fast Corner - Hold (min)

This has a path ID of fc_h0, and contains worst-case hold timing, using all of the min and early delays. This path ID can contain the worst-case hold timing for short paths if clock skew is minimal.

Clock Frequencies

This section contains a summary of the setup-and-hold timing on a per clock group basis. In the following example of routed summary report, the negative setup slack of −1.809 is highlighted in yellow:



Collective Summary of All Corners

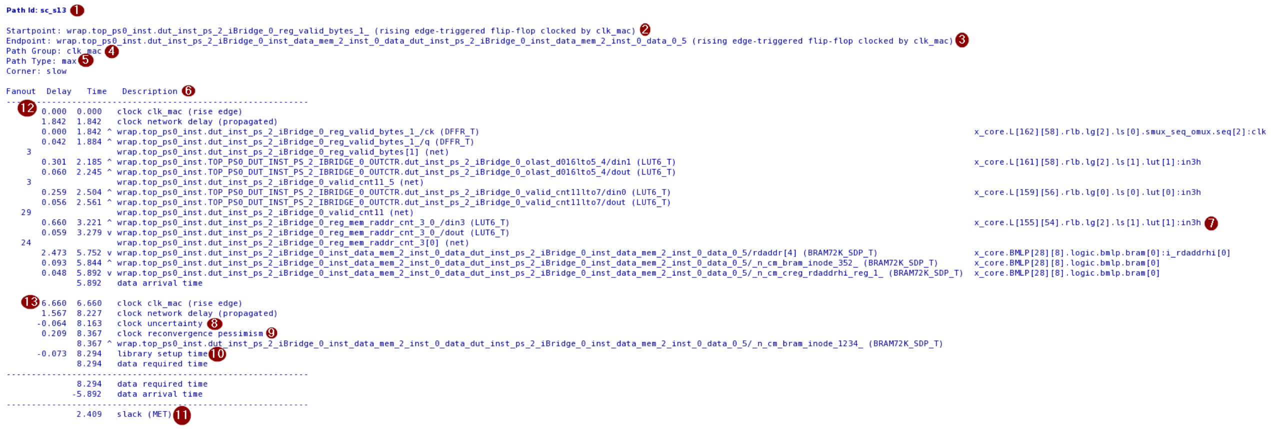
	Slack (ns)		Frequency (MHz)		
Clock / Group	setup	hold	Target	Upper Limit	Comment
clk_core	0.399	-1.809	150.2	159.7	---
clk_mac	1.914	0.007	150.2	210.7	---
async_default	2.228	0.151	150.2	225.9	---

Upper limit of frequency on a clock may be bound by the upper limit of frequency on some other clocks which are generated from a common source clock. These clocks, if present, are labeled as 'limiting clocks' in comments below.
The 'Slack' column corresponds to the worst setup or hold slack values over all corners.

Figure 4: ***Clock Frequency Summary***

Example Setup Timing Path, Flop to Flop

In the following example, the timing path is between two flops in the same clock domain (clk_mac). The following timing path labels each of the lines in a typical flop-to-flop one cycle, same clock, setup timing path, and indicates which of the timing values come from the SDC. The numbers highlighted in the timing report are explained in the information box below. Reports are configured to include both input pins and output pins, as well as nets. For each of the nets, the "Fanout" column indicates how many downstream input pins are connected to the output pin of that instance.



```
Path Id: sc_s13

Startpoint: wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_valid_bytes_1_ (rising edge-triggered flip-flop clocked by clk_mac)
Endpoint: wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_0_5 (rising edge-triggered flip-flop clocked by clk_mac)
Path Group: clk_mac
Path Type: max
Corner: slow

Fanout  Delay   Time   Description
------------------------------------------------------------
        0.000  0.000   clock clk_mac (rise edge)
        1.842  1.842   clock network delay (propagated)
        0.000  1.842 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_valid_bytes_1_/ck (DFFR_T)                                                                  x_core.L[162][58].rlb.lg[2].ls[0].smux_seq_omux.seq[2]:clk
        0.042  1.884 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_valid_bytes_1_/q (DFFR_T)
   3                   wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_valid_bytes[1] (net)
        0.301  2.185 ^ wrap.top_ps0_inst.TOP_PS0_DUT_INST_PS_2_IBRIDGE_0_OUTCTR.dut_inst_ps_2_iBridge_0_olast_d016lto5_4/din1 (LUT6_T)                             x_core.L[161][58].rlb.lg[2].ls[1].lut[1]:in3h
        0.060  2.245 ^ wrap.top_ps0_inst.TOP_PS0_DUT_INST_PS_2_IBRIDGE_0_OUTCTR.dut_inst_ps_2_iBridge_0_olast_d016lto5_4/dout (LUT6_T)
   3                   wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_valid_cntl1_5 (net)
        0.259  2.504 ^ wrap.top_ps0_inst.TOP_PS0_DUT_INST_PS_2_IBRIDGE_0_OUTCTR.dut_inst_ps_2_iBridge_0_valid_cntl1lto7/din0 (LUT6_T)                              x_core.L[159][56].rlb.lg[0].ls[0].lut[0]:in3h
        0.056  2.561 ^ wrap.top_ps0_inst.TOP_PS0_DUT_INST_PS_2_IBRIDGE_0_OUTCTR.dut_inst_ps_2_iBridge_0_valid_cntl1lto7/dout (LUT6_T)
  29                   wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_valid_cntl1 (net)
        0.660  3.221 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_mem_raddr_cnt_3_0_/din3 (LUT6_T)                                                            x_core.L[155][54].rlb.lg[2].ls[1].lut[1]:in3h
        0.059  3.279 v wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_mem_raddr_cnt_3_0_/dout (LUT6_T)
  24                   wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_mem_raddr_cnt_3[0] (net)
        2.473  5.752 v wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_0_5/rdaddr[4] (BRAM72K_SDP_T)                 x_core.BMLP[28][8].logic.bmlp.bram[0]:i_rdaddrhi[0]
        0.093  5.844 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_0_5/_n_cm_bram_inode_352_ (BRAM72K_SDP_T)       x_core.BMLP[28][8].logic.bmlp.bram[0]
        0.048  5.892 v wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_0_5/_n_cm_creg_rdaddrhi_reg_1_ (BRAM72K_SDP_T)  x_core.BMLP[28][8].logic.bmlp.bram[0]
               5.892   data arrival time

        6.660  6.660   clock clk_mac (rise edge)
        1.567  8.227   clock network delay (propagated)
       -0.064  8.163   clock uncertainty
        0.209  8.367   clock reconvergence pessimism
               8.367 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_2_inst_0_data_0_5/_n_cm_bram_inode_1234_ (BRAM72K_SDP_T)
       -0.073  8.294   library setup time
               8.294   data required time
------------------------------------------------------------
               8.294   data required time
              -5.892   data arrival time
------------------------------------------------------------
               2.409   slack (MET)
```

Figure 5: ***Flop-to-flop Setup Timing Path***

Below are the annotations for setup path info (the figure above):

1. Path ID. sc_s13 indicates it is a setup path in slow corner.
2. Source flop.
3. Capture flop.
4. Path group clk_mac identified by capture clock.
5. Path Type max indicates Setup.
6. Timing report columns:
    a. Fanout column indicates net load.
    b. Delay column indicates incremental net/cell delay.
    c. Time column indicates cumulative delay.
    d. The symbol ^/v in the description column imply rising and falling edge respectively.
7. Placement location
8. Clock uncertainty applies only to capture clock. There are two possible sources of clock setup uncertainty:
    a. User-supplied SDC which contains the `set_clock_uncertainty -setup` constraint
    b. The ACE default `set_clock_uncertainty -setup` value. These two sources are additive to each other, so if the user supplies uncertainty it is based on effects outside the ACE design.

    > **Note**
    >
    > The clock uncertainty value used for setup timing path analysis can be, and usually is, different than the value used for hold timing path analysis.

9. CRPR. Clock re-convergence pessimism removal to remove timing inaccuracy caused by different min /max delay applied to a shared segment.
10. Setup time. The minimum time before clock edge that the data needs to be stable at the input.
11. Slack (data required time less data arrival time. A positive slack implies MET and a negative slack implies VIOLATED.

12. Clock arrival time at the top-level pin:
    a. The number displayed in the "Delay" column for the "Clock arrival time at the top level pin" comes directly from the SDC. It is equal to the applicable clock edge of the `create_clock` definition for the respective clock. The rise value is used in the above example, as the clock edge is listed as "^" or rise edge. By default, if these values are not explicitly set in the SDC syntax, the value here is set to time zero (0.000).
    b. There are many ways to define a clock with the `create_clock` syntax. By default, the first asserted edge is a rising edge at time zero (0.000). This setting can be modified by using the `create_clock -waveform` option.
13. Clock cycle time (second edge). By default, the value for the clock's second edge comes from the `create_clock -period` value. With the first edge rising at time 0.000, and the second edge rising at the value of the period specified. Typically data paths have one cycle to get from the source register to the capture register. This time can vary if both edges of the clock are used, creating timing paths where there are different edges between the source clock edge and the capture clock edge. If the source clock edge is rising and the capture clock edge is falling, this value would either be 50% of the `create_clock -period` or the second value of the `create_clock -waveform` values.

Cell Instance Site Naming Convention

There is a naming convention for the cell instance sites that provides insight into the physical distance between an output pin and the next input pin. In general the instance names start with a location such as x_core.L[0][0].rlb.lg[0].ls[0].lut. In the GUI's "Time Report - Routed" report, this is represented as:

```
x_core.L[0][0].rlb.lg[0].ls[0]
```

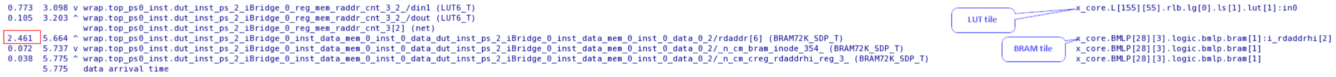
This instance is the first instance in the lower left-hand corner. The next instance to the right of it is named:

```
x_core.L[1][0].rlb.lg[0].ls[0]
```

The instance above the first instance is named:

```
x_core.L[0][0].rlb.lg[0].ls[1]
```

The naming convention increments up for cell instances in both the X and Y axis. Therefore, if there are two cell instances that are connected by a net, which have cell instance names that are separated by a large delta in the respective site names, the placement of these instances can contribute to large net delays, as is shown in the following example a snippet of the routed timing report having large delay between a LUT and a BRAM tile.



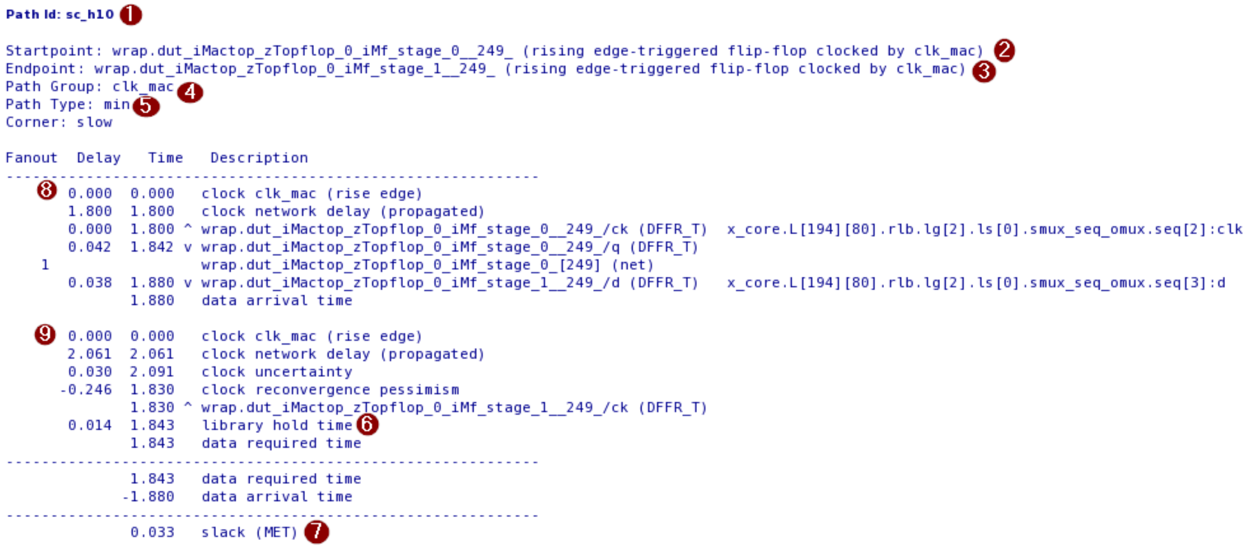
```
0.773  3.098 v wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_mem_raddr_cnt_3_2_/dinl (LUT6_T)                                                                    x_core.L[155][55].rlb.lg[0].ls[1].lut[1]:in0
0.105  3.203 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_mem_raddr_cnt_3_2_/dout (LUT6_T)
               wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_reg_mem_raddr_cnt_3[2] (net)
2.461  5.664 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_0_2/rdaddr[6] (BRAM72K_SDP_T)                  x_core.BMLP[28][3].logic.bmlp.bram[1]:i_rdaddrhi[2]
0.072  5.737 v wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_0_2/_n_cm_bram_inode_354_ (BRAM72K_SDP_T)      x_core.BMLP[28][3].logic.bmlp.bram[1]
0.038  5.775 ^ wrap.top_ps0_inst.dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_dut_inst_ps_2_iBridge_0_inst_data_mem_0_inst_0_data_0_2/_n_cm_creg_rdaddrhi_reg_3_ (BRAM72K_SDP_T)  x_core.BMLP[28][3].logic.bmlp.bram[1]
       5.775   data arrival time
```

Figure 6: ***Large Net Delay Due to Instance Placement***

The net's stage delay is very large due to the distance between the instance x_core.L[155][55]... and x_core.BMLP[28][3]... There can be a number of reasons to cause this delay, and looking at the placement in the context of the timing path can also shed more light. See Example Setup Timing Path Flop to OPIN (see page) to see one of the reasons that net delay might be long.

Example Hold Timing Path, Flop to Flop

The following timing path labels each of the lines in a typical flop-to-flop zero cycle, same clock, hold timing path.



```
Path Id: sc_h10

Startpoint: wrap.dut_iMactop_zTopflop_0_iMf_stage_0__249_ (rising edge-triggered flip-flop clocked by clk_mac)
Endpoint: wrap.dut_iMactop_zTopflop_0_iMf_stage_1__249_ (rising edge-triggered flip-flop clocked by clk_mac)
Path Group: clk_mac
Path Type: min
Corner: slow

Fanout  Delay   Time   Description
------------------------------------------------------------
         0.000  0.000   clock clk_mac (rise edge)
         1.800  1.800   clock network delay (propagated)
         0.000  1.800 ^ wrap.dut_iMactop_zTopflop_0_iMf_stage_0__249_/ck (DFFR_T)  x_core.L[194][80].rlb.lg[2].ls[0].smux_seq_omux.seq[2]:clk
         0.042  1.842 v wrap.dut_iMactop_zTopflop_0_iMf_stage_0__249_/q (DFFR_T)
    1                   wrap.dut_iMactop_zTopflop_0_iMf_stage_0_[249] (net)
         0.038  1.880 v wrap.dut_iMactop_zTopflop_0_iMf_stage_1__249_/d (DFFR_T)   x_core.L[194][80].rlb.lg[2].ls[0].smux_seq_omux.seq[3]:d
                1.880   data arrival time

         0.000  0.000   clock clk_mac (rise edge)
         2.061  2.061   clock network delay (propagated)
         0.030  2.091   clock uncertainty
        -0.246  1.830   clock reconvergence pessimism
                1.830 ^ wrap.dut_iMactop_zTopflop_0_iMf_stage_1__249_/ck (DFFR_T)
         0.014  1.843   library hold time
                1.843   data required time
------------------------------------------------------------
                1.843   data required time
               -1.880   data arrival time
------------------------------------------------------------
                0.033   slack (MET)
```

Figure 7: ***Flop-to-flop Hold Timing Path***

Below are the annotations for hold path info (the figure above):

1. Path ID. sc_h10 indicates it is a hold path in slow corner.
2. Source flop.
3. Capture flop.
4. Path group clk_mac identified by capture clock.
5. Path Type min indicates hold.
6. Hold time, minimum time after the clock edge that the data will need to propagate to output.
7. Slack (data arrival time less data required time). A positive slack implies MET and a negative slack implies VIOLATED.
8. Clock arrival time at the top-level pin:
    a. The number displayed in the "Delay" column for the "Clock arrival time at the top level pin" comes directly from the SDC. It is equal to the applicable clock edge of the `create_clock` definition for the respective clock. The rise value is used in the above example, as the clock edge is listed as "^" or rise edge. By default, if these values are not explicitly set in the SDC syntax, the value here is set to time zero (0.000).
    b. There are many ways to define a clock with the `create_clock` syntax. By default, the first asserted edge is a rising edge at time zero (0.000). This setting can be modified by using the `create_clock -waveform` option.
9. This edge is same as clock arrival edge for hold check.

I/O Timing Paths

Timing paths that include timing constraints that drive input ports to registers, as well as registers to output ports have additional timing constraints. The basic interactions are described in the following timing paths.

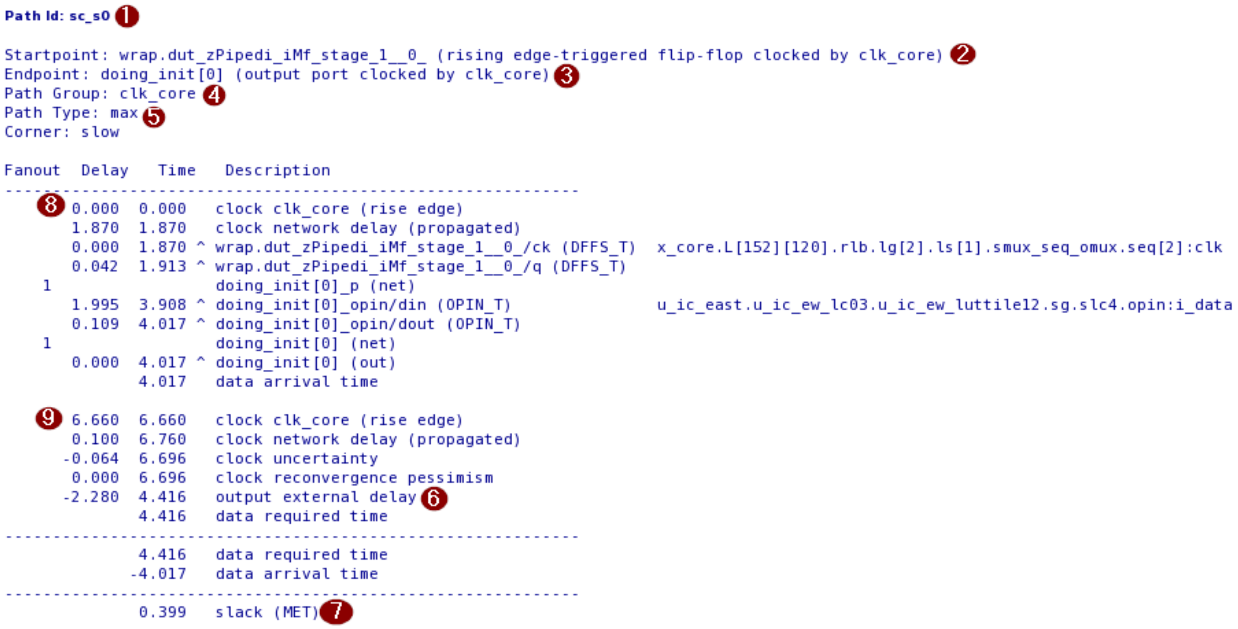
- Flop to OPIN setup timing
- Flop to OPIN hold timing

- IPIN to flop setup timing
- IPIN to flop hold timing

SDC commands are used to define the I/O timing relationships between clock and data.

Example Setup Timing Path, Flop to OPIN

In the following example, the timing path is between a flop and an OPIN in the same clock domain (clk_core).



```
Path Id: sc_s0

Startpoint: wrap.dut_zPipedi_iMf_stage_1__0_ (rising edge-triggered flip-flop clocked by clk_core)
Endpoint: doing_init[0] (output port clocked by clk_core)
Path Group: clk_core
Path Type: max
Corner: slow

Fanout  Delay   Time   Description
-------------------------------------------------------------
        0.000   0.000   clock clk_core (rise edge)
        1.870   1.870   clock network delay (propagated)
        0.000   1.870 ^ wrap.dut_zPipedi_iMf_stage_1__0_/ck (DFFS_T)  x_core.L[152][120].rlb.lg[2].ls[1].smux_seq_omux.seq[2]:clk
        0.042   1.913 ^ wrap.dut_zPipedi_iMf_stage_1__0_/q (DFFS_T)
    1                   doing_init[0]_p (net)
        1.995   3.908 ^ doing_init[0]_opin/din (OPIN_T)             u_ic_east.u_ic_ew_lc03.u_ic_ew_luttile12.sg.slc4.opin:i_data
        0.109   4.017 ^ doing_init[0]_opin/dout (OPIN_T)
    1                   doing_init[0] (net)
        0.000   4.017 ^ doing_init[0] (out)
                4.017   data arrival time

        6.660   6.660   clock clk_core (rise edge)
        0.100   6.760   clock network delay (propagated)
       -0.064   6.696   clock uncertainty
        0.000   6.696   clock reconvergence pessimism
       -2.280   4.416   output external delay
                4.416   data required time
-------------------------------------------------------------
                4.416   data required time
               -4.017   data arrival time
-------------------------------------------------------------
                0.399   slack (MET)
```

Figure 8: ***Flop to OPIN Setup Timing Path***

Below are the annotations for output setup path info (the figure above):

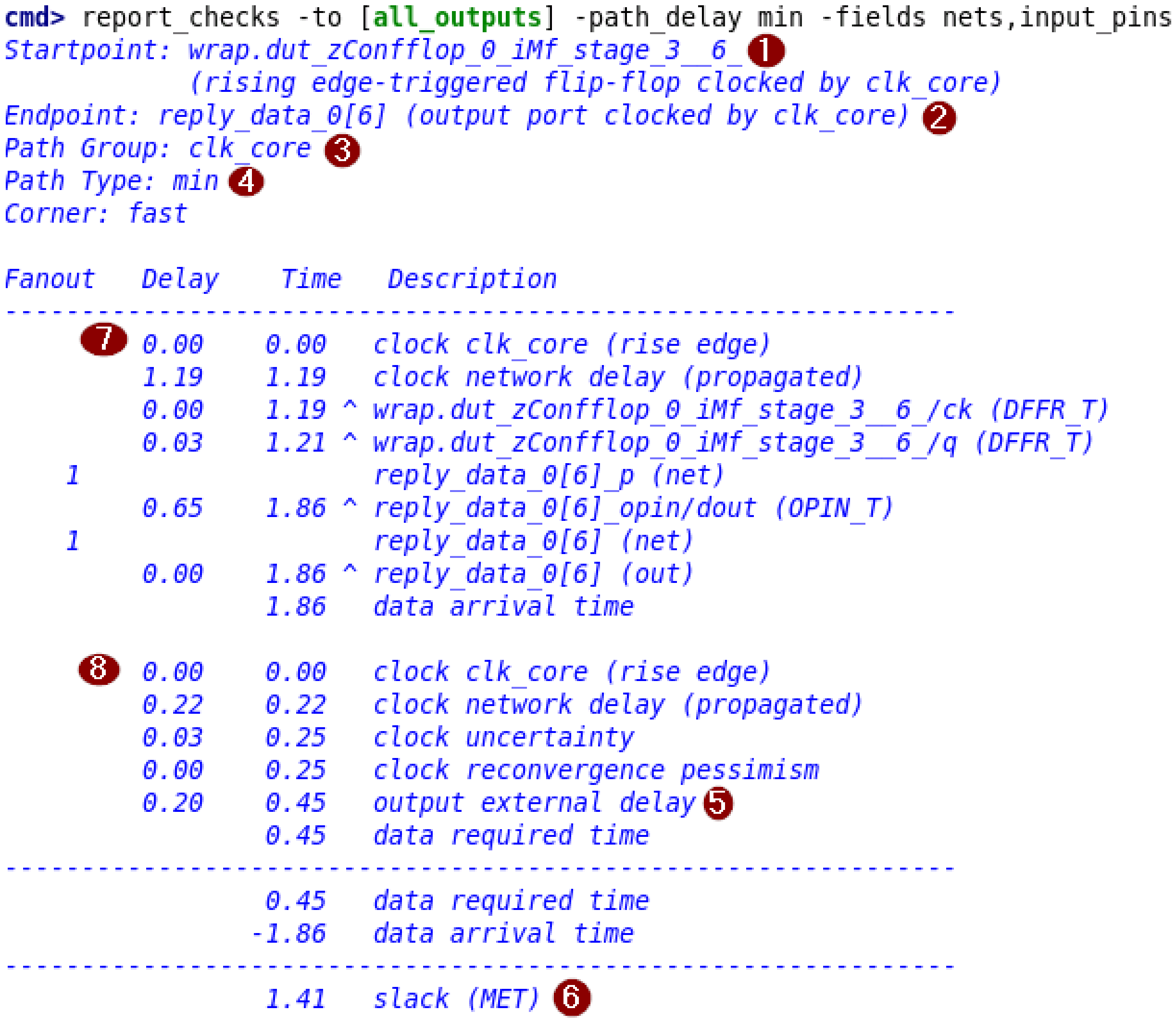
1. Path ID. sc_s0 indicates it is a setup path in slow corner.
2. Source flop.
3. output port.
4. Path group clk_core identified by capture clock.
5. Path Type max indicates setup.
6. Output external delay constraint from SDC, time data is required to arrive at OPIN before the clock. The timing constraint for this value comes from a `set_output_delay -clock -max` SDC statement, specific to this data pin. In this case the value is 2.28, which is specific to `clk_core`.
7. Slack (data required time less data arrival time. A positive slack implies MET and a negative slack implies VIOLATED.

8. Clock arrival time at the top-level pin
    a. The number displayed in the "Delay" column for the "Clock arrival time at the top level pin" comes directly from the SDC. It is equal to the applicable clock edge of the `create_clock` definition for the respective clock. The rise value is used in the above example, as the clock edge is listed as "**^**" or rise edge. By default, if these values are not explicitly set in the SDC syntax, the value here is set to time zero (0.000).
    b. There are many ways to define a clock with the `create_clock` syntax. By default, the first asserted edge is a rising edge at time zero (0.000). This setting can be modified by using the `create_clock -waveform` option.
9. Clock cycle time (second edge). By default, the value for the clock's second edge comes from the `create_clock -period` value. With the first edge rising at time 0.000, and the second edge rising at the value of the period specified. Typically data paths have one cycle to get from the source register to the capture register. This time can vary if both edges of the clock are used, creating timing paths where there are different edges between the source clock edge and the capture clock edge. If the source clock edge is rising and the capture clock edge is falling, this value would either be 50% of the `create_clock -period` or the second value of the `create_clock -waveform` values.

Example Hold Timing Path, Flop to OPIN

The following example shows a hold timing path between a flop and an OPIN in the same clock domain (`clk_core`). The timing path labels each of the lines in a typical flop to OPIN, same clock, hold timing path. The SDC on the OPIN (`set_output_delay`) models a register at the output.

> **note**
>
> The following report is generated using interactive ACE STA command `report_checks` as it does not appear in the 10 worst negative-slack (WNS) paths reported under path group `clk_core` for hold. Interactive ACE is invoked using the command `prepare_sta.` This report differs from the others in that it does not contain placement information.



```
cmd> report_checks -to [all_outputs] -path_delay min -fields nets,input_pins
Startpoint: wrap.dut_zConfflop_0_iMf_stage_3__6_
            (rising edge-triggered flip-flop clocked by clk_core)
Endpoint: reply_data_0[6] (output port clocked by clk_core)
Path Group: clk_core
Path Type: min
Corner: fast

Fanout   Delay    Time   Description
--------------------------------------------------------------
         0.00    0.00   clock clk_core (rise edge)
         1.19    1.19   clock network delay (propagated)
         0.00    1.19 ^ wrap.dut_zConfflop_0_iMf_stage_3__6_/ck (DFFR_T)
         0.03    1.21 ^ wrap.dut_zConfflop_0_iMf_stage_3__6_/q (DFFR_T)
    1                   reply_data_0[6]_p (net)
         0.65    1.86 ^ reply_data_0[6]_opin/dout (OPIN_T)
    1                   reply_data_0[6] (net)
         0.00    1.86 ^ reply_data_0[6] (out)
                 1.86   data arrival time

         0.00    0.00   clock clk_core (rise edge)
         0.22    0.22   clock network delay (propagated)
         0.03    0.25   clock uncertainty
         0.00    0.25   clock reconvergence pessimism
         0.20    0.45   output external delay
                 0.45   data required time
--------------------------------------------------------------
                 0.45   data required time
                -1.86   data arrival time
--------------------------------------------------------------
                 1.41   slack (MET)
```

Figure 9: ***Flop to OPIN Hold Timing Path***

Below are the annotations for output hold path info (the figure above):

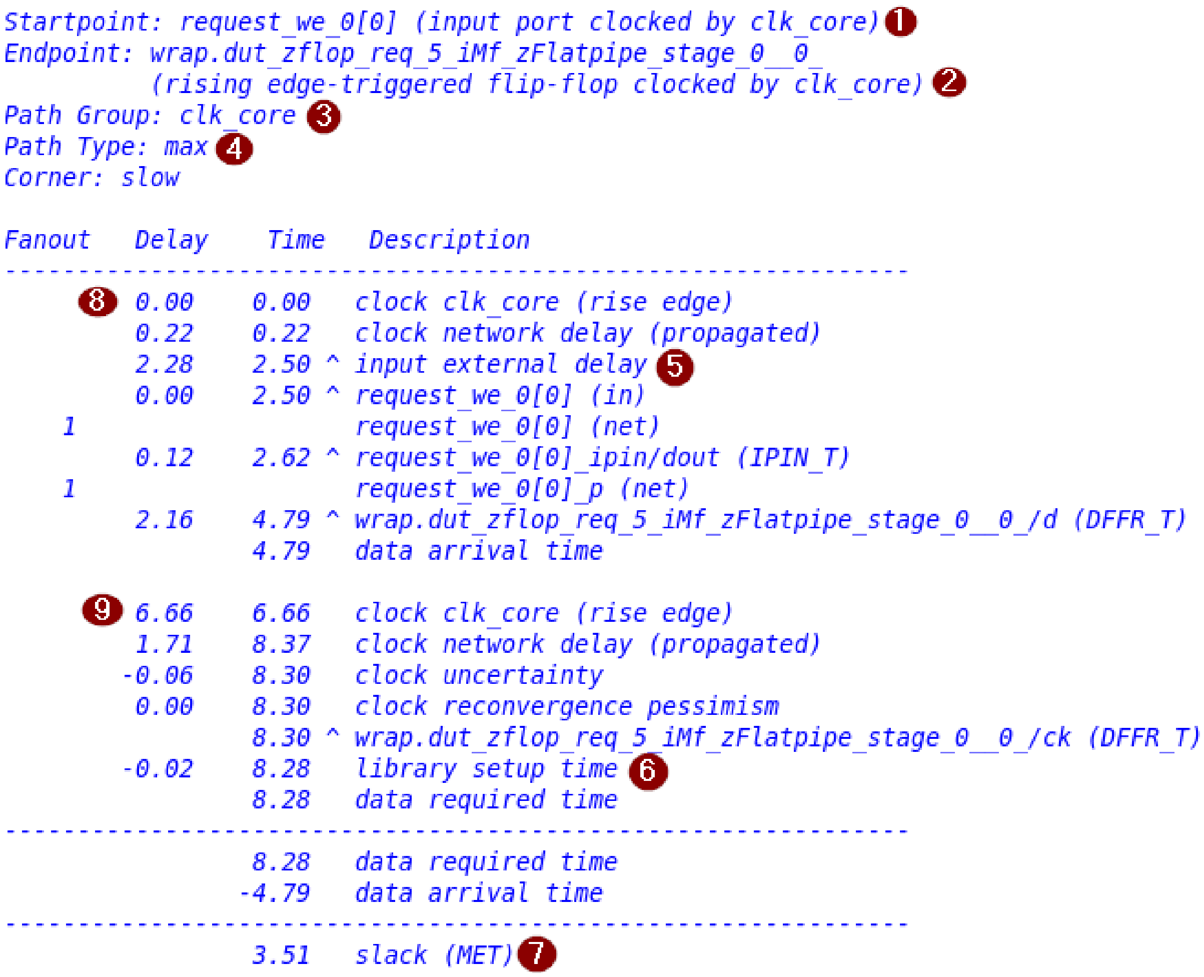
1. Source flop.
2. Output port.
3. Path group clk_core identified by capture clock.
4. Path Type min indicates hold.
5. Time data has to hold at the OPIN after clock. The timing constraint comes from a `set_output_delay -clock -min` SDC statement specific to this data pin. The value here is 0.2.
6. Slack (data arrival time less data required time). A positive slack implies MET and a negative slack implies VIOLATED.

7. Clock arrival time at the top-level pin
    a. The number displayed in the "Delay" column for the "Clock arrival time at the top level pin" comes directly from the SDC. It is equal to the applicable clock edge of the `create_clock` definition for the respective clock. The rise value is used in the above example, as the clock edge is listed as "**^**" or rise edge. By default, if these values are not explicitly set in the SDC syntax, the value here is set to time zero (0.000).
    b. There are many ways to define a clock with the `create_clock` syntax. By default, the first asserted edge is a rising edge at time zero (0.000). This setting can be modified by using the `create_clock -waveform` option.
    c. Estimated clock network delay propagated beneath clock arrival time is an estimate of the clock latency to the clock pin of the source flop. The value of this estimate changes at different stages of the flow, and is set after routing.
8. This edge is same as clock arrival edge for hold check.

Example Setup Timing Path, IPIN to Flop

The following example illustrates a timing path between an IPIN and a flop in the same clock domain (clk_core). The timing path labels each of the lines in a typical IPIN to flop one cycle, same clock, setup timing path, and indicates which of the timing values come from the SDC. The delays from the IPIN are modeled as a register using the `set_input_delay` constraints.

> **note**
>
> The following report is generated using interactive ACE STA command `report_checks`, so it is slightly different then the other reports.



```
Startpoint: request_we_0[0] (input port clocked by clk_core)❶
Endpoint: wrap.dut_zflop_req_5_iMf_zFlatpipe_stage_0__0_
          (rising edge-triggered flip-flop clocked by clk_core)❷
Path Group: clk_core ❸
Path Type: max ❹
Corner: slow

Fanout   Delay    Time   Description
--------------------------------------------------------------
      ❽  0.00    0.00   clock clk_core (rise edge)
         0.22    0.22   clock network delay (propagated)
         2.28    2.50 ^ input external delay ❺
         0.00    2.50 ^ request_we_0[0] (in)
    1                   request_we_0[0] (net)
         0.12    2.62 ^ request_we_0[0]_ipin/dout (IPIN_T)
    1                   request_we_0[0]_p (net)
         2.16    4.79 ^ wrap.dut_zflop_req_5_iMf_zFlatpipe_stage_0__0_/d (DFFR_T)
                 4.79   data arrival time

      ❾  6.66    6.66   clock clk_core (rise edge)
         1.71    8.37   clock network delay (propagated)
        -0.06    8.30   clock uncertainty
         0.00    8.30   clock reconvergence pessimism
                 8.30 ^ wrap.dut_zflop_req_5_iMf_zFlatpipe_stage_0__0_/ck (DFFR_T)
        -0.02    8.28   library setup time ❻
                 8.28   data required time
--------------------------------------------------------------
                 8.28   data required time
                -4.79   data arrival time
--------------------------------------------------------------
                 3.51   slack (MET)❼
```

Figure 10: ***IPIN to Flop Setup Timing Path***

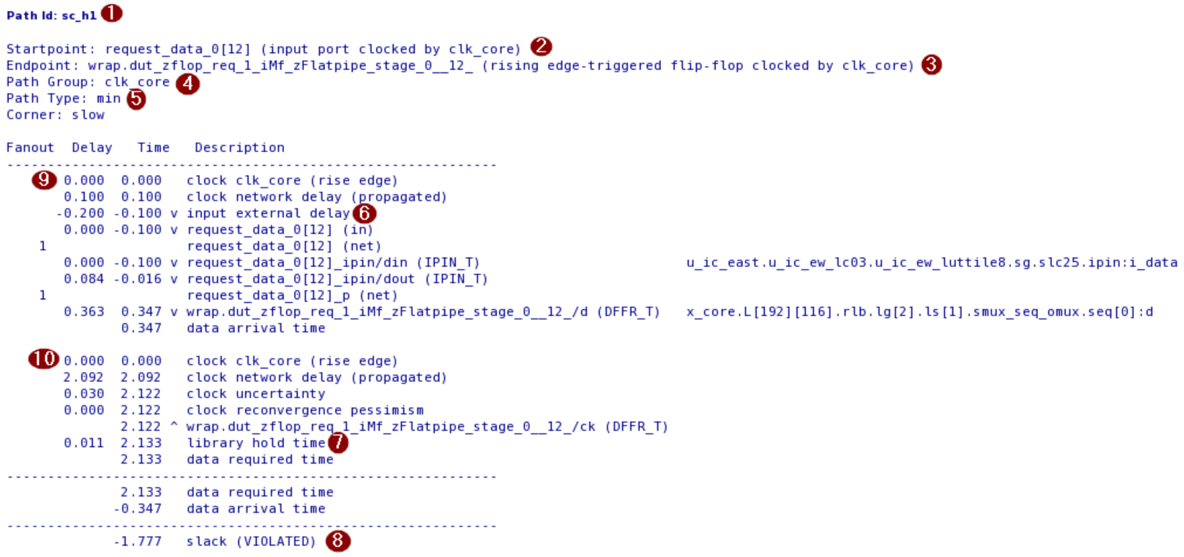
Below are the annotations for input setup path info (the figure above):

1. Source input port.
2. Capture flop.
3. Path group clk_core identified by capture clock.
4. Path Type max indicates setup.
5. The command `set_input_delay` is a SDC that models a flop that drives data to input ports.
6. Setup time, minimum time before clock edge that the data needs to be stable at the input.
7. Slack (data required time less data arrival time. A positive slack implies MET and a negative slack implies VIOLATED.

8. Clock arrival time at the top-level pin
    a. The number displayed in the "Delay" column for the "Clock arrival time at the top level pin" comes directly from the SDC. It is equal to the applicable clock edge of the `create_clock` definition for the respective clock. The rise value is used in the above example, as the clock edge is listed as "`^`" or rise edge. By default, if these values are not explicitly set in the SDC syntax, the value here is set to time zero (0.000).
    b. There are many ways to define a clock with the `create_clock` syntax. By default, the first asserted edge is a rising edge at time zero (0.000). This setting can be modified by using the `create_clock -waveform` option.
9. Clock cycle time (second edge). By default, the value for the clock's second edge comes from the `create_clock -period` value. With the first edge rising at time 0.000, and the second edge rising at the value of the period specified. Typically data paths have one cycle to get from the source register to the capture register. This time can vary if both edges of the clock are used, creating timing paths where there are different edges between the source clock edge and the capture clock edge. If the source clock edge is rising and the capture clock edge is falling, this value would either be 50% of the `create_clock -period` or the second value of the `create_clock -waveform` values.

Example Hold Timing Path, IPIN to Flop

The following timing path labels each of the lines in a typical IPIN to flop zero cycle, same clock, hold timing path, and indicates which of the timing values come from the SDC input constraints.



```
Path Id: sc_h1 ❶

Startpoint: request_data_0[12] (input port clocked by clk_core) ❷
Endpoint: wrap.dut_zflop_req_1_iMf_zFlatpipe_stage_0__12_ (rising edge-triggered flip-flop clocked by clk_core) ❸
Path Group: clk_core ❹
Path Type: min ❺
Corner: slow

Fanout  Delay   Time   Description
------------------------------------------------------------
    ❾  0.000  0.000   clock clk_core (rise edge)
       0.100  0.100   clock network delay (propagated)
      -0.200 -0.100 v input external delay ❻
       0.000 -0.100 v request_data_0[12] (in)
    1                 request_data_0[12] (net)
       0.000 -0.100 v request_data_0[12]_ipin/din (IPIN_T)                      u_ic_east.u_ic_ew_lc03.u_ic_ew_luttile8.sg.slc25.ipin:i_data
       0.084 -0.016 v request_data_0[12]_ipin/dout (IPIN_T)
    1                 request_data_0[12]_p (net)
       0.363  0.347 v wrap.dut_zflop_req_1_iMf_zFlatpipe_stage_0__12_/d (DFFR_T)   x_core.L[192][116].rlb.lg[2].ls[1].smux_seq_omux.seq[0]:d
              0.347   data arrival time

    ❿  0.000  0.000   clock clk_core (rise edge)
       2.092  2.092   clock network delay (propagated)
       0.030  2.122   clock uncertainty
       0.000  2.122   clock reconvergence pessimism
              2.122 ^ wrap.dut_zflop_req_1_iMf_zFlatpipe_stage_0__12_/ck (DFFR_T)
       0.011  2.133   library hold time ❼
              2.133   data required time
------------------------------------------------------------
              2.133   data required time
             -0.347   data arrival time
------------------------------------------------------------
             -1.777   slack (VIOLATED) ❽
```

Figure 11: ***IPIN to Flop Hold Timing Path***

Below are the annotations for input hold path info (the figure above):

1. Path ID. sc_h0 indicates it is a hold path in slow corner.
2. Source Input.
3. Capture flop.
4. Path group clk_core identified by capture clock.
5. Path Type min indicates hold.

6. External data delay. This delay value is defined in the `set_input_delay` constaint. In this case the `-min` option is used to indicate that this delay should be used when analyzing "Path Type: min" timing paths.
7. Hold time, minimum time after the clock edge that the data will need to propagate to output.
8. Slack (data arrival time less data required time). A positive slack implies MET and a negative slack implies VIOLATED.
9. Clock arrival time at the top-level pin:
    a. The number displayed in the "Delay" column for the "Clock arrival time at the top level pin" comes directly from the SDC. It is equal to the applicable clock edge of the `create_clock` definition for the respective clock. The rise value is used in the above example, as the clock edge is listed as "**^**" or rise edge. By default, if these values are not explicitly set in the SDC syntax, the value here is set to time zero (0.000).
    b. There are many ways to define a clock with the `create_clock` syntax. By default, the first asserted edge is a rising edge at time zero (0.000). This setting can be modified by using the `create_clock -waveform` option.
    c. Estimated clock network delay propagated beneath clock arrival time is an estimate of the clock latency to the clock pin of the source flop. The value of this estimate changes at different stages of the flow, and is set after routing.
10. This edge is same as clock arrival edge for hold check.

References and Support

- Refer to the*ACE User Guide* (UG070) for detailed usage information.
- Vsit Achronix Support Portal for access to our knowledge base.
- Email questions to support@achronix.com.

Revision History

Version	Date	Description
1.0	29 Jan 2021	• Initial Achronix release.

Achronix®

Data Acceleration

Achronix Semiconductor Corporation

2903 Bunker Hill Lane
Santa Clara, CA 95054
USA

Website: www.achronix.com
E-mail : info@achronix.com

Notice of Disclaimer

The information given in this document is believed to be accurate and reliable. However, Achronix Semiconductor Corporation does not give any representations or warranties as to the completeness or accuracy of such information and shall have no liability for the use of the information contained herein. Achronix Semiconductor Corporation reserves the right to make changes to this document and the information contained herein at any time and without notice. All Achronix trademarks, registered trademarks, disclaimers and patents are listed at http://www.achronix.com/legal.