cadence®

Unconstrained Paths – Debugging RAK

Rapid Adoption Kit (RAK)

Product Version: Tempus 21.11
October 2021

Contents

Introduction

In the following lab, you will learn ways to debug various problem scenarios related to **Unconstrained timing paths** by using Cadence Tempus Timing Signoff Analysis Solution. Problem statements shared in this lab are captured based on commonly reported issues. The lab comprises of 14 problem statements covering various issues related to unconstrained timing paths.

Purpose

The purpose of this document is to help you understand various techniques related to debugging unconstrained timing paths reported in Tempus. This document talks about some of the common scenarios resulting in an unconstrained timing path and some of the common techniques/commands to debug these unconstrained paths.

Lab Instructions

The lab database, scripts, and references can be found in the 'Attachments' sections below. The TAR ball (unconstrained_timing_paths_debugging_testcase.tar.gz) contains the testcase and a README file.

1. Untar the tar ball and go to the following directory:
   cd unconstrained_timing_paths_debugging_testcase/cmmmc/work
2. Invoke Tempus with any latest version.

   Note: Testcase in this RAK is tested with the Tempus 21.11-s131_1 release.

3. Run the following command to load the design:

```
source run.tcl
  OR
tempus -init run.tcl
```

After the testcase is loaded, go through the README file or the following "problem statements" in detail and try to debug or answer the mentioned problems.

You will find solutions to all problems mentioned in the later part of this document. It is recommended that you go through these problems and try to find out answers before referring to the Solutions section.

Problem Statements

There are 14 timing paths shared below and none of the paths have reported constrained. These paths are either reported as unconstrained path or not reported as unconstrained paths in some cases.

You are suggested to run all paths mentioned below and try to debug the reason for the path not being reported as a constrained path.

Path 1

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[1]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D
```

No constrained timing paths found with the given description.

Path 2

```
> report_timing -from SPI_INST/bit_cnt_reg[0]/Q -to
SPI_INST/spi_sr_reg[4]/D
```

No constrained timing paths found with the given description.

Path 3

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[9]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/D
```

No constrained timing paths found with the given description.

Path 4

```
> report_timing -from scan_se -to RESULTS_CONV_INST/go_reg/SE
```

No constrained timing paths found with the given description.

Path 5

```
 > report_timing -from TDSP_CORE_INST/EXECUTE_INST/arp_reg/Q -to
RESULTS_CONV_INST/r852_reg[8]/D
```

No constrained timing paths found with the given description.

Path 6

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q -
to TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D
```

No constrained timing paths found with the given description.

Path 7

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[8]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[26]/D
```

No constrained timing paths found with the given description.

Path 8

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[0]/Q -to
ARB_INST/ECO1reg_present_state_reg[2]/D
```

No constrained timing paths found with the given description.

Path 9

```
> report_timing -from
TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/Q -to
ARB_INST/present_state_reg[1]/D
```

No constrained timing paths found with the given description.

Path 10

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[12]/Q -
through_fall DATA_SAMPLE_MUX_INST/FE_OFC17_n_23/A -to
RAM_256x16_TEST_INST/RAM_256x16_INST/D[0]
```

No constrained timing paths found with the given description.

Path 11

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/D
```

No constrained timing paths found with the given description.

Path 12

```
> report_timing -from
TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[0]/Q -to
TDSP_CORE_INST/PROG_BUS_MACH_INST/data_out_reg[6]/D
```

No constrained timing paths found with the given description.

Path 13

```
> report_timing -from port_pad_data_in[15]
```

No constrained timing paths found with the given description.

Path 14

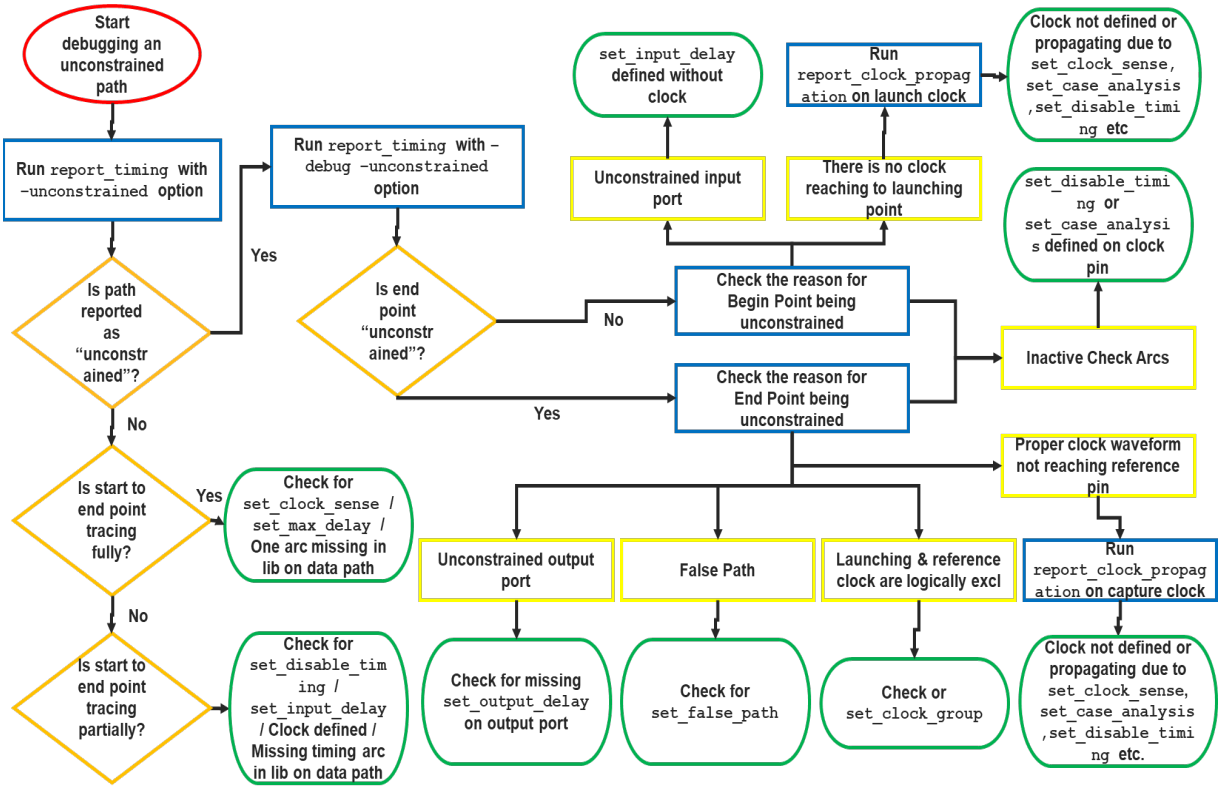
```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[14]/Q -
to TDSP_CORE_INST/EXECUTE_INST/skip_one_reg/D
```

No constrained timing paths found with the given description.

Debugging Flowchart

A path cannot be reported as constrained due to missing or incorrect constraints, path exceptions, timing-library-related issues, and so on.

The following flowchart guides you to debug reasons for most of these paths:



Solutions

In this section, the debugging approach of all paths that were shared earlier in the Problem Statements section are explained with details.

Debug for Path 1

- Run `report_timing` with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[1]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D -unconstrained -debug unconstrained

 Path Unconstrained Reason:
-------------------------------
All the capturing arcs of the end point 'TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D'
are unable to constrain the path
end point : 'TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D'
signal transition : 'rise'
launching clock : 'm_clk'
      ------------------------------------------------------------------------------
       RefPin   ArcType   WhenCond   RefClock               Reason
      ------------------------------------------------------------------------
       CK ^     HOLD      -          -          incompatible slack computation mode

       CK ^     SETUP     -          -          inactive check arc: User Disable

      ------------------------------------------------------------------------------
```

The tool gives the reason as `inactive check arc: User Disable`.

- Run the `report_inactive_arcs` command on this instance to identify the reason for the inactive arc.

```
> report_inactive_arcs TDSP_CORE_INST/EXECUTE_INST/p_reg[31] -view func_slow_RCMAX
   ------------------------------------------------------------------------------
    From                          To                              ArcType
Sense                       Reason                       View
   ------------------------------------------------------------------------------
   TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/CK
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D   hold_rising                 non_unate
User Disable               func_slow_RCMAX
   TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/CK
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D   setup_rising                non_unate
User Disable               func_slow_RCMAX
   ------------------------------------------------------------------------------
------------------------------------------------------------
```

You can notice from the report that the `CK->D` arc of the instance `TDSP_CORE_INST/EXECUTE_INST/p_reg[31]` is disabled.

- Alternatively, you can run the `report_analysis_coverage` command to confirm if any check arc is inactive.

```
> report_analysis_coverage TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D -verbose
untested
 ----------------------------------------------------------------------------------
                                                    TIMING CHECK COVERAGE DETAILS
    ------------------------------------------------------------------------------
---
     Pin                            Reference Pin                Check Type
Slack        Reason
    ------------------------------------------------------------------------------
---
    TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/CK ^   Setup                        UNTESTED
disable
    TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/CK ^   Hold                         UNTESTED
disable
 ----------------------------------------------------------------------------------
```

You can notice that the `CK->D` check arc is disabled.

- Run the `reset_disable_timing` command to remove the arc disabled by the `set_disable_timing` command and check if the path is reported as a constrained path now.

```
> reset_disable_timing -from CK -to D
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]

> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[1]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D
Path 1: VIOLATED Setup Check with Pin TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/CK
Endpoint:   TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D (v) checked with  leading edge
of 'm_clk'
Beginpoint: TDSP_CORE_INST/DECODE_INST/ir_reg[1]/Q  (v) triggered by  leading edge
of 'm_clk'
Path Groups: {m_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          5.744
- Setup                         1.298
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 8.045
- Arrival Time                 34.239
= Slack Time                  -26.193
```

This path is now reported as constrained.

Debug for Path 2

- Run `report_timing` with the "`-debug unconstrained`" option.

```
> report_timing -from SPI_INST/bit_cnt_reg[0]/Q -to SPI_INST/spi_sr_reg[4]/D -
unconstrained -debug unconstrained

Path Unconstrained Reason:
-------------------------------
```

```
All the capturing arcs of the end point 'SPI_INST/spi_sr_reg[4]/D' are unable to
constrain the path
end point : 'SPI_INST/spi_sr_reg[4]/D'
signal transition : 'rise'
launching clock : 'm_spi_clk'
      ------------------------------------------------------------------------
       RefPin   ArcType   WhenCond   RefClock              Reason
      ------------------------------------------------------------------------
       CK ^     HOLD      -          -          incompatible slack computation mode

       CK ^     SETUP     -          -          proper clock waveform not reaching
reference pin

      ------------------------------------------------------------------------
```

The tool gives the reason as `proper clock waveform not reaching reference pin`.

- Run the `report_clock_propagation` command to identify the reason for the clock not reaching the clock pin.

```
> report_clock_propagation -to SPI_INST/spi_sr_reg[4]/CK -clock m_spi_clk -view
func_slow_RCMAX
Clock: m_spi_clk
Point: SPI_INST/spi_sr_reg[4]/CK
View:  func_slow_RCMAX
Reason 1: The clock 'm_spi_clk' is stopped due to set_clock_sense constraint on pin
'm_spi_clk__L2_I5/A'
```

You can notice from the report that `set_clock_sense` is applied on a pin in clock path results topping clock propagation to the reference pin.

- Run `reset_clock_sense` to remove this `set_clock_sense` to check if the path is reported as constrained or not.

```
> reset_clock_sense m_spi_clk__L2_I5/A

> report_timing -from SPI_INST/bit_cnt_reg[0]/Q -to SPI_INST/spi_sr_reg[4]/D
Path 1: VIOLATED Setup Check with Pin SPI_INST/spi_sr_reg[4]/CK
Endpoint:   SPI_INST/spi_sr_reg[4]/D  (v) checked with  leading edge of 'm_spi_clk'
Beginpoint: SPI_INST/bit_cnt_reg[0]/Q (^) triggered by  leading edge of 'm_spi_clk'
Path Groups: {m_spi_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          0.781
- Setup                         1.307
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 3.074
- Arrival Time                  6.123
= Slack Time                   -3.049
```

This path is now reported as constrained.

Debug for Path 3

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[9]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/D -unconstrained -debug unconstrained
Path Unconstrained Reason:
------------------------------
All the capturing arcs of the end point 'TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/D'
are unable to constrain the path
end point : 'TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/D'
signal transition : 'rise'
launching clock : 'm_clk'
      ------------------------------------------------------------------------
       RefPin   ArcType   WhenCond   RefClock              Reason
      ------------------------------------------------------------------------
       CK ^     HOLD      -          -          incompatible slack computation
mode

       CK ^     SETUP     -          -          inactive check arc:
TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK = 1

      ------------------------------------------------------------------------
```

The tool gives reason as `inactive check arc: TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK = 1`.

- Run the `report_inactive_arcs` command on this instance to identify the reason for the inactive arc.

```
> report_inactive_arcs TDSP_CORE_INST/EXECUTE_INST/p_reg[13] -view func_slow_RCMAX

------------------------------------------------------------------------------
From                          To                           ArcType
   Sense                       Reason                     View
------------------------------------------------------------------------------
TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK   TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/D
  setup_rising                 non_unate                  constant
disabled          func_slow_RCMAX
```

You can notice from the report that the `CK->D` arc of the `TDSP_CORE_INST/EXECUTE_INST/p_reg[13]` instance is constant-disabled.

- Run the `report_case_analysis` command with the "`-propagated`" option to find the source of the constant propagating to this pin '`TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK`':

```
> report_case_analysis TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK -verbose -
propagated -view func_slow_RCMAX
```

```
Pin TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK 1 view func_slow_RCMAX is caused by
set_case_analysis 0 on pin m_clk__L19_I30/A
      ------------------------------------------------------------------------
       Pin name                                              Constant   View
Name
                                                             value

      ------------------------------------------------------------------------
      m_clk__L19_I30/A                                      0          func_slow_RCM
AX
      m_clk__L19_I30/ZN                                     1          func_slow_RCM
AX
      TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK              1          func_slow_RCM
AX
      ------------------------------------------------------------------------
```

You can notice form the report that the case analysis value of '0' is applied on the 'm_clk__L19_I30/A' pin.

- Run the `reset_case_analysis` command to reset `set_case_analysis` to check if the path is reported as a constrained path now.

```
> reset_case_analysis m_clk__L19_I30/A

> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[9]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/D
Path 1: VIOLATED Setup Check with Pin TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/CK
Endpoint:   TDSP_CORE_INST/EXECUTE_INST/p_reg[13]/D (^) checked with  leading edge
of 'm_clk'
Beginpoint: TDSP_CORE_INST/DECODE_INST/ir_reg[9]/Q  (v) triggered by  leading edge
of 'm_clk'
Path Groups: {m_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          5.668
- Setup                         0.582
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 8.686
- Arrival Time                 33.430
= Slack Time                  -24.744
```

This path is now reported as a constrained path.

Debug for Path 4

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from scan_se -to RESULTS_CONV_INST/go_reg/SE -unconstrained -debug
unconstrained
Path Unconstrained Reason:
------------------------------
All the capturing arcs of the end point 'RESULTS_CONV_INST/go_reg/SE' are unable to
constrain the path
end point : 'RESULTS_CONV_INST/go_reg/SE'
signal transition : 'rise'
launching clock : 'm_dsram_clk'
```

```
      ------------------------------------------------------------------------
       RefPin   ArcType   WhenCond    RefClock              Reason
      ------------------------------------------------------------------------
       CK ^     HOLD      -           -           incompatible slack computation
mode

       CK ^     SETUP     -           m_rcc_clk   false path
      ------------------------------------------------------------------------
```

The tool gives the reason as `false path`.

- Run the `report_timing` command with the `-path_exceptions` option to get details of the false path applied on this path.

```
> report_timing -from scan_se -to RESULTS_CONV_INST/go_reg/SE -unconstrained -debug
unconstrained -path_exceptions all
     Applied exceptions:
   ------------------------------------------------------------------------------
              From                       Through                  Late
View Name
   ------------------------------------------------------------------------------
   scan_se                      FE_OFC413_scan_se/Z          false             scan
_slow_RCMAX
   ------------------------------------------------------------------------------
```

You can notice from the report that the false path is applied from the `scan_se` port and through the `FE_OFC413_scan_se/Z` pin.

- Run the `reset_path_exception` command to reset this false path constraint and check if the path is reported as a constrained path now.

```
> reset_path_exception -type false_path -from scan_se -through
FE_OFC413_scan_se/Z
> report_timing -from scan_se -to RESULTS_CONV_INST/go_reg/SE
Path 1: MET Setup Check with Pin RESULTS_CONV_INST/go_reg/CK
Endpoint:   RESULTS_CONV_INST/go_reg/SE (^) checked with trailing edge of
'm_rcc_clk'
Beginpoint: scan_se                     (v) triggered by  leading edge of
'm_dsram_clk'
Path Groups: {m_rcc_clk}
Analysis View: scan_slow_RCMAX
Other End Arrival Time         21.513
- Setup                         1.479
+ Phase Shift                   0.000
+ Cycle Adjustment             40.000
- Uncertainty                   0.400
= Required Time                59.634
- Arrival Time                 18.803
= Slack Time                   40.831
```

You can notice that the path is reported as a constrained path now.

Debug for Path 5

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/EXECUTE_INST/arp_reg/Q -to
RESULTS_CONV_INST/r852_reg[8]/D -unconstrained -debug unconstrained
Path Unconstrained Reason:
------------------------------
All the capturing arcs of the end point 'RESULTS_CONV_INST/r852_reg[8]/D' are
unable to constrain the path
end point : 'RESULTS_CONV_INST/r852_reg[8]/D'
signal transition : 'rise'
launching clock : 'm_clk'
      ------------------------------------------------------------------------
       RefPin   ArcType   WhenCond    RefClock              Reason
      ------------------------------------------------------------------------
       CK ^     HOLD      -          -           incompatible slack computation
mode

       CK ^     SETUP     -          m_rcc_clk   launching and reference clocks are
logically exclusive

      ------------------------------------------------------------------------
```

The tool gives the reason as `launching and reference clocks are logically exclusive`, which indicates that there is a `set_clock_group` applied between these two clocks.

- Run `report_timing` with the `-path_exception` option to crosscheck if this is reported as the false path.

```
> report_timing -from TDSP_CORE_INST/EXECUTE_INST/arp_reg/Q -to
RESULTS_CONV_INST/r852_reg[8]/D -unconstrained -debug unconstrained -
path_exceptions all
      ------------------------------------------------------------------------
     Applied exceptions:
    ------------------------------------------------------------------------
              From                          To                  Late
View Name
    ------------------------------------------------------------------------
    "m_clk"                     "m_rcc_clk"                 false            scan
_slow_RCMAX
    ------------------------------------------------------------------------
```

You can notice from the report that the false path is set between two clocks, `m_clk` and `m_rcc_clk`.

- Run `report_clocks` with the `-group` option to check for the existing clock groups.

```
> report_clocks -groups {m_clk m_rcc_clk} -view scan_slow_RCMAX
------------------------------------------------------------------
```

```
                        Clock Groups Descriptions
---------------------------------------------------------------------
Total logically exclusive groups: 1
Clock Group : clk_grp
View : scan_slow_RCMAX
 -group { m_clk }
 -group { m_rcc_clk }

Total asynchronous groups: 0

Total physically exclusive groups: 0
```

- Check SDC for `set_clock_group` applied between these two clocks.

```
> set_clock_groups -name log_grp -logically_exclusive  -group  [get_clocks {m_clk}]
-group  [get_clocks {m_rcc_clk}]
```

- Run `reset_clock_group` on this clock group to check if a constrained path is reported now.

```
> reset_clock_groups -logically_exclusive -name log_grp

> report_timing -from TDSP_CORE_INST/EXECUTE_INST/arp_reg/Q -to
RESULTS_CONV_INST/r852_reg[8]/D
Path 1: VIOLATED Setup Check with Pin RESULTS_CONV_INST/r852_reg[8]/CK
Endpoint:   RESULTS_CONV_INST/r852_reg[8]/D       (^) checked with trailing edge of
'm_rcc_clk'
Beginpoint: TDSP_CORE_INST/EXECUTE_INST/arp_reg/Q (^) triggered by  leading edge of
'm_clk'
Path Groups: {m_rcc_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          3.464
- Setup                         0.754
+ Phase Shift                   0.000
- Uncertainty                   0.400
= Required Time                 2.310
- Arrival Time                 19.181
= Slack Time                  -16.871
```

You can notice that the path is reported as a constrained path now.

Debug for Path 6

- Run `report_timing` with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q -to
TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D -unconstrained -debug unconstrained
############################################################
No unconstrained timing paths with given description found.
Paths may be constrained or  may not exist.
```

The tool does not report this path either as unconstrained or constrained.

- Run `report_fanin` command on the end point to find out if paths between these two points are fully traceable or not.

Here is the output of the `report_fanin` command on the end point:

```
> report_fanin -to TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D -view func_slow_RCMAX
############################################################
Fanin Network for View: func_slow_RCMAX
Fanin Network for Pin: TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D
       ------------------------------------------------------------------------
        Source Pin              Sink Pin               Cell Type         Timing
Sense      Arc        Case

Disabled   Analysis
       ------------------------------------------------------------------------
       TDSP_CORE_INST/EXECUTE_INST/n1020D25736/ZN
TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D   (net)             -                 -
-
       ------------------------------------------------------------------------
```

You will find that this does not trace to the '`TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q`' begin point.

- Check the pin between the begin and end points where tracing of the path is getting stopped.

You can use the following sample script, which traces all fanouts from the begin point and find out the pin that has only one fanout and the arc from that point is either disabled or missing in the library.

```
> foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q] {
set pin [get_pins $fanout]
if {[sizeof_collection [all_fanout -from [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
echo " Pin [get_object_name $pin] has one fanout and its arc is either disabled or
missing in library"
 }
 }
```

The output is as follows:

The `TDSP_CORE_INST/EXECUTE_INST/n1020D25736/A` pin has one fanout and its arc is either disabled or missing in the library.

Similarly, use the `all_fanin` command from the end point to find the instance that has one fanin and the arc to that point is either disabled or missing in the library.

```
> foreach_in_collection fanin [all_fanin -to
TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D] {
set pin [get_pins $fanin]
```

```
if {[sizeof_collection [all_fanin -to [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
echo "Pin [get_object_name $pin]  has one fanin and its arc is either disabled or
missing in library"
  }
  }
```

The `TDSP_CORE_INST/EXECUTE_INST/n1020D25736/ZN` pin has one fanin and its arc is either disabled or missing in the library.

You can notice from the output of both scripts that path tracing is stopped at the input and output pins of the same instance. It seems the arc is either disabled or missing for this instance / cell.

- Run the `report_inactive_arc` command to the instance reported to confirm if the arc ending to this pin is disabled.

```
> report_inactive_arcs TDSP_CORE_INST/EXECUTE_INST/n1020D25736 -view
func_slow_RCMAX
   ------------------------------------------------------------------------------
    From                        To                          ArcType
Sense                      Reason                      View
   ------------------------------------------------------------------------------
---
   TDSP_CORE_INST/EXECUTE_INST/n1020D25736/A
TDSP_CORE_INST/EXECUTE_INST/n1020D25736/ZN   combinational
negative_unate             User Disable              func_slow_RCMAX
   ------------------------------------------------------------------------------
```

You can notice from the report that the `A -> ZN` arc of the `TDSP_CORE_INST/EXECUTE_INST/n1020D25736` instance is disabled.

- Run the `reset_disable_timing` command to reset the applied `set_disable_timing` on this arc and check if this path is reported as a constrained path now.

```
> reset_disable_timing -from A -to ZN TDSP_CORE_INST/EXECUTE_INST/n1020D25736
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q -to
TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D
Start delay calculation (fullDC) (16 T). (MEM=6554.84)
End delay calculation. (MEM=7011.56 CPU=0:00:13.4 REAL=0:00:02.0)
End delay calculation (fullDC). (MEM=7011.56 CPU=0:00:14.8 REAL=0:00:02.0)
Path 1: VIOLATED Setup Check with Pin TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/CK
Endpoint:   TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/D (^) checked with  leading edge
of 'm_clk'
Beginpoint: TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q  (v) triggered by  leading edge
of 'm_clk'
Path Groups: {m_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          5.328
- Setup                         0.549
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 8.379
```

```
- Arrival Time                  33.796
= Slack Time                   -25.417
```

This concludes that this path was unconstrained due to `set_disable_timing` applied on the data path.

Note: If a path is not getting reported either as constrained or unconstrained, sometimes, it may become difficult to figure out what instance or pin in the path results in the path not reported either as constrained or unconstrained. If you have the knowledge of the expected timing path, you may also use the `-point_to_point` option of the `report_timing` command to figure out such a pin or instance.

For example, you can do `report_timing -point_to_point` from the start point to any intermediate point in the path and check if you can get the path reported. If it is getting reported, the pin or instance causing the problem might be falling beyond this point and if it is not getting reported, the pin or instance causing the problem might be falling between these two points. In that case, you may further pick any intermediate point between these two points and repeat the same exercise. So, this will be a hit-and- trial method to find out the intermediate pin or instance causing the problem.

In the above example, if you pick any intermediate point (for example, `TDSP_CORE_INST/EXECUTE_INST/n1020D25736/ZN`) and run `report_timing -point_to_point` from the start point, you will notice that it does not report any path.

It means that the pin or instance causing the problem might be falling between these two points.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q -to
TDSP_CORE_INST/EXECUTE_INST/n1020D25736/ZN -point_to_point
############################################################
No unconstrained timing paths with given description found.
Paths may be constrained or  may not exist.
```

Further, if you pick any other intermediate point (for example, `TDSP_CORE_INST/EXECUTE_INST/n1020D25736/A`) and run `report_timing -point_to_point`, you can notice that this reports the path.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q -to
TDSP_CORE_INST/EXECUTE_INST/n1020D25736/A -point_to_point
############################################################
Path 1:Endpoint:   TDSP_CORE_INST/EXECUTE_INST/n1020D25736/A (v) (unconstrained
output)
Beginpoint: TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q   (^) triggered by  leading
edge of 'm_clk'
Arrival Time                   27.769
Analysis View: scan_slow_RCMAX
```

```
      ------------------------------------------------------------------------
      Instance                                                        Arc
Cell       Delay  Arrival

Time
      ------------------------------------------------------------------------
      TDSP_CORE_INST/DECODE_INST/ir_reg[11]                           Q ^
-          -      0.000
      TDSP_CORE_INST/DECODE_INST/FE_OFC192_ir_11_                     A ^ -> Z v
BUF_X16    0.460  0.460
```

It means that the problem is between `DSP_CORE_INST/EXECUTE_INST/n1020D25736/A` and `TDSP_CORE_INST/EXECUTE_INST/n1020D25736/Z` points.

Debug for Path 7

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[8]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[26]/D -unconstrained -debug unconstrained
No unconstrained timing paths with given description found.
Paths may be constrained or  may not exist.
```

The tool does not report this path either as unconstrained or constrained.

- Run `report_fanin` command on the end point to find out if paths between these two points are fully traceable or not.

Here is the output of the `report_fanin` command on the end point:

```
> report_fanin -to TDSP_CORE_INST/EXECUTE_INST/p_reg[26]/D -view func_slow_RCMAX
############################################################
Fanin Network for View: func_slow_RCMAX
Fanin Network for Pin: TDSP_CORE_INST/EXECUTE_INST/p_reg[26]/D
       ------------------------------------------------------------------------
        Source Pin             Sink Pin               Cell Type         Timing
Sense      Arc        Case

Disabled   Analysis
       ------------------------------------------------------------------------
       TDSP_CORE_INST/EXECUTE_INST/n0361D/ZN
TDSP_CORE_INST/EXECUTE_INST/p_reg[26]/D   (net)             -                -
-
       ------------------------------------------------------------------------
.
.
.

       ------------------------------------------------------------------------
       TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/FE_OFC88_n_339/Z
TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0510A/A2   (net)             -
-          -
```

```
       TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/Fn0049D/ZN
TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0510A/A1   (net)             -
-          -
       TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0973A/ZN
TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0966A/A2   (net)             -
-          -
       TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0720A14177/ZN
TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0966A/A1   (net)             -
-          -
       TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0493A/ZN
TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0916A14084/A2   (net)
-                  -          -
       TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0966A/ZN
TDSP_CORE_INST/MPY_32_INST/M16X16_INST/mul_8_14/p0916A14084/A1   (net)
-                  -          -
       ------------------------------------------------------------------------
```

You will find that it does not trace to the 'TDSP_CORE_INST/DECODE_INST/ir_reg[8]/Q' begin point.

- Check the pin between the begin and end points where tracing of the path is getting stopped.

You can use the following sample script, which traces all fanouts from the begin point and find out the pin that has only one fanout and the arc from that point is either disabled or missing in the library.

```
> foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/DECODE_INST/ir_reg[8]/Q] {
set pin [get_pins $fanout]
if {[sizeof_collection [all_fanout -from [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
echo " Pin [get_object_name $pin] has one fanout and its arc is either disabled or
missing in library"
 }
 }
```

The output of is as follows:

The TDSP_CORE_INST/FE_OFC253_port_address_0_/A pin has one fanout and its arc is either disabled or missing in the library.

Similarly, use the all_fanin command from the end point to find the instance that has one fanin and the arc to that point is either disabled or missing in the library.

```
 > foreach_in_collection fanin [all_fanin -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[26]/D] {
set pin [get_pins $fanin]
if {[sizeof_collection [all_fanin -to [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
echo "Pin [get_object_name $pin] has one fanin and its arc is either disabled or
missing in library"
  }
```

```
  }
```

The `TDSP_CORE_INST/FE_OFC253_port_address_0_/Z` pin has one fanin and its arc is either disabled or missing in the library.

You can notice from the output of both scripts that path tracing is stopped at the input and output pins of the same instance. It seems the arc is disabled for this instance or missing in the library.

- Run `report_inactive_arc` to the instance reported to confirm whether the arc is disabled or not.

```
> report_inactive_arcs TDSP_CORE_INST/FE_OFC253_port_address_0_
   ------------------------------------------------------------------------------
    From                          To                          ArcType
Sense                      Reason                    View
   ------------------------------------------------------------------------------
```

This report is blank, indicating that there is no timing arc disabled to this pin.

- Check if there is any arc missing for this instance.

```
> sizeof_collection [get_arcs -of_objects [get_cells
TDSP_CORE_INST/FE_OFC253_port_address_0_]]
0
```

You can notice that it does not show any arc.

Alternatively, you can run the `report_cell_instance_timing` command and you will find that there is no arc present for this cell.

```
>  report_cell_instance_timing TDSP_CORE_INST/FE_OFC253_port_address_0_
      ---------------------------------------------------
                              Instance TDSP_CORE_INST/FE_OFC253_port_address_0_ of
                              BUF_X64
      ---------------------------------------------------
            Arc            Slew
      ---------------------------------------------------
        From    To      In      Out     Load   Delay   Phase
      ---------------------------------------------------
```

- Check the library for the above instance (the `BUF_X64` cell) using you will find that there is no timing arc existing for this cell.

```
> report_instance_library -instance  TDSP_CORE_INST/FE_OFC253_port_address_0_ -view
func_slow_RCMAX
Instance              : TDSP_CORE_INST/FE_OFC253_port_address_0_
Cell                  :
BUF_X64
```

```
Analysis View          :
func_slow_RCMAX
Power Domain           : Default
Library/Libset         : FreePDK45_lib_v1.0/libset_slow
Library File           : ./libs/liberty/FreePDK45_lib_v1.0_worst.lib
Binding Status         : All PG Pins Voltage Match
Op Cond                : P->1.000000, V->0.800000, T->25.000000 (TypTyp_0.8V_25C)
Cell Type              : gateCell
```

Debug for Path 8

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[0]/Q -to
ARB_INST/ECO1reg_present_state_reg[2]/D -unconstrained -debug unconstrained
############################################################
No unconstrained timing paths with given description found.
Paths may be constrained or  may not exist.
```

The tool does not report this path either as unconstrained or constrained.

- Run `report_fanin` command on the end point to find out if paths between these two points are fully traceable or not.

Here is the output of the `report_fanin` command on the end point:

```
> report_fanin -to ARB_INST/ECO1reg_present_state_reg[2]/D -view func_slow_RCMAX
############################################################
Fanin Network for View: func_slow_RCMAX
Fanin Network for Pin: ARB_INST/ECO1reg_present_state_reg[2]/D
       ------------------------------------------------------------------------------
---------------------------
        Source Pin             Sink Pin               Cell Type         Timing
Sense      Arc        Case

Disabled   Analysis
       ------------------------------------------------------------------------------
---------------------------
       ARB_INST/ECO1inst_8/ZN ARB_INST/ECO1reg_present_state_reg[2]/D   (net)
-                -          -
       ------------------------------------------------------------------------------
---------------------------.
.
.
      ------------------------------------------------------------------------------
-------------------------
       ARB_INST/present_state_reg[0]/CK   ARB_INST/present_state_reg[0]/Q   DFFS_X1
non_unate         -          -
       ARB_INST/present_state_reg[1]/CK   ARB_INST/present_state_reg[1]/Q
SDFFR_X2          non_unate         -          -
       TDSP_CORE_INST/DATA_BUS_MACH_INST/as_reg/CK
TDSP_CORE_INST/DATA_BUS_MACH_INST/as_reg/Q   SDFFR_X2          non_unate         -
-
```

```
       TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/CK
TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/Q   SDFFR_X2          non_unate
-          -
       TDSP_DS_CS_INST/p6962A/A   TDSP_DS_CS_INST/p6962A/ZN   INV_X2
negative_unate    -          -
       ------------------------------------------------------------------------------
------------------------
```

You can notice that it does not trace to the '`TDSP_CORE_INST/DECODE_INST/ir_reg[0]/Q`' begin point.

- Check the pin between the begin and end points where tracing of the path is getting stopped.

You can use the following sample script, which traces all fanouts from the begin point and find out the pin that has only one fanout and the arc from that point is either disabled or missing in the library.

```
foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/DECODE_INST/ir_reg[0]/Q] {
set pin [get_pins $fanout]
if {[sizeof_collection [all_fanout -from [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
echo " Pin [get_object_name $pin] has one fanout and its arc is either disabled or
missing in library "
 }
 }
```

This script does not give any output.

Similarly, you can run `all_fanin` from the end point to find the instance that has one fanin and its arc to that point is either disabled or missing in the library.

```
foreach_in_collection fanin [all_fanin -to ARB_INST/ECO1reg_present_state_reg[2]/D]
{
set pin [get_pins $fanin]
if {[sizeof_collection [all_fanin -to [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
echo " Pin [get_object_name $pin] has one fanin and its arc is either disabled or
missing in library "
  }
  }
```

This script also does not give any output.

This means that the begin-to-end point path is not traceable due to any arc disabled or the arc missing in the library.

This may happen if there is a `set_input_delay` or `create_clock` defined on the data path.

Checking `is_clock_used_as_data` and `is_clock_used_as_clock` properties will give you an indication of any such constraint on the data path.

- You can check if both `is_clock_used_as_data` and `is_clock_used_as_clock` are reported as "`false`". You can use the following sample script to find any such pin on this path:

```
set path1 [report_timing -to ARB_INST/ECO1reg_present_state_reg[2]/D  -nworst 100 -
collection -view func_slow_RCMAX]
set lp [get_property $path1 launching_point]

foreach_in_collection g $lp {
if {[get_property $g is_clock_used_as_data]=="false" && [get_property $g
is_clock_used_as_clock]=="false"} {
echo " Pin [get_object_name $g] has property is_clock_used_as_data &
is_clock_used_as_clock as false "
}
}
```

The output is as follows:

```
Pin TDSP_DS_CS_INST/p6962A/A has property is_clock_used_as_data &
is_clock_used_as_clock as false
```

- You may run the `report_timing` command from the `TDSP_DS_CS_INST/p6962A/A` pin to the `ARB_INST/ECO1reg_present_state_reg[2]/D` end point and this path is reported as a valid constraint path. This means that `TDSP_DS_CS_INST/p6962A/A` is also constrained as a valid start point.

```
> report_timing -from TDSP_DS_CS_INST/p6962A/A -to
ARB_INST/ECO1reg_present_state_reg[2]/D
       ############################################################
       Path 1: MET Setup Check with Pin ARB_INST/ECO1reg_present_state_reg[2]/CK
       Endpoint:   ARB_INST/ECO1reg_present_state_reg[2]/D (^) checked with
       leading edge of 'm_clk'
       Beginpoint: TDSP_DS_CS_INST/p6962A/A               (^) triggered by
       leading edge of 'm_clk'
       Path Groups: {m_clk}
       Analysis View: func_slow_RCMAX
       Other End Arrival Time          5.710
       - Setup                         1.289
       + Phase Shift                   4.000
       - Uncertainty                   0.400
       = Required Time                 8.021
       - Arrival Time                  5.007
       = Slack Time                    3.015
            Clock Rise Edge                       0.000
            + Input Delay                         0.400
            = Beginpoint Arrival Time             0.400
```

You can notice that the input delay is applied on this path.

You can also check `set_input_delay` specified on this pin in SDC. You can notice the following statement in the SDC:

```
> set_input_delay 0.4 TDSP_DS_CS_INST/p6962A/A -clock m_clk
```

- Run the `reset_input_delay` command to reset the applied `set_input_delay` on this pin and check if this path is reported as a constrained path now.

```
> reset_input_delay TDSP_DS_CS_INST/p6962A/A -clock m_clk

> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[0]/Q -to
ARB_INST/ECO1reg_present_state_reg[2]/D
Path 1: VIOLATED Setup Check with Pin ARB_INST/ECO1reg_present_state_reg[2]/CK
Endpoint:   ARB_INST/ECO1reg_present_state_reg[2]/D (^) checked with  leading edge
of 'm_clk'
Beginpoint: TDSP_CORE_INST/DECODE_INST/ir_reg[0]/Q  (^) triggered by  leading edge
of 'm_clk'
Path Groups: {m_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          5.710
- Setup                         1.289
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 8.021
- Arrival Time                 19.404
= Slack Time                  -11.383
```

This concludes that this path was unconstrained due to the `set_input_delay` constraint applied on the data pin.

Debug for Path 9

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/Q -to
ARB_INST/present_state_reg[1]/D -unconstrained -debug unconstrained
#############################################################
No unconstrained timing paths with given description found.
Paths may be constrained or  may not exist.?
```

The tool does not report this path either as unconstrained or constrained.

- Run `report_fanin` command on the end point to find out if paths between these two points are fully traceable or not.

Here is the output of the `report_fanin` command on the end point:

```
> report_fanin -to ARB_INST/present_state_reg[1]/D -view func_slow_RCMAX
#############################################################
Fanin Network for View: func_slow_RCMAX
```

```
Fanin Network for Pin: ARB_INST/present_state_reg[1]/D
       ------------------------------------------------------------------------------
--------------------------
        Source Pin             Sink Pin               Cell Type         Timing
Sense      Arc        Case

Disabled   Analysis
       ------------------------------------------------------------------------------
--------------------------
       ARB_INST/FE_OFCC305_n_22/Z   ARB_INST/present_state_reg[1]/D   (net)
-                -          -
       ------------------------------------------------------------------------------
--------------------------
       ARB_INST/FE_OFCC305_n_22/A   ARB_INST/FE_OFCC305_n_22/Z   BUF_X4
positive_unate    -          -
       ------------------------------------------------------------------------------
--------------------------
       ARB_INST/ECO1inst_4/ZN ARB_INST/FE_OFCC305_n_22/A   (net)             -
-          -
       ------------------------------------------------------------------------------
--------------------------
       ARB_INST/ECO1inst_4/A  ARB_INST/ECO1inst_4/ZN INV_X2
negative_unate    -          -
       ------------------------------------------------------------------------------
--------------------------
```

You can notice that it does not trace to the '`TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/Q`' begin point.

- Check the pin between the begin and end points where tracing of the path is getting stopped.

You can use the following sample script, which traces all fanouts from the begin point and finds out the instance that has only one fanout and the arc from that point is either disabled or missing in the library.

```
foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/Q] {
set pin [get_pins $fanout]
if {[sizeof_collection [all_fanout -from [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
echo " Pin [get_object_name $pin] has one fanout and its arc is either disabled or
missing in library "
 }
 }
```

This script does not give any output.

Similarly, you can run `all_fanin` from the end point to find the instance that has one fanin and its arc to that point is either disabled or missing in the library.

```
foreach_in_collection fanin [all_fanin -to ARB_INST/present_state_reg[1]/D] {
set pin [get_pins $fanin]
if {[sizeof_collection [all_fanin -to [get_object_name $pin]]] == "1" &&
[get_property [get_cells -of_objects $pin] is_disable_timing] == "true"} {
```

```
echo " Pin [get_object_name $pin] has one fanin and its arc is either disabled or
missing in library "
  }
  }
```

This script also does not give any output.

This concludes that the begin-to-end point is not traceable due to any arc disabled or the arc missing in the library.

This may happen if there is a `set_input_delay` or `create_clock` defined on the data path.

Checking `is_clock_used_as_data` and `is_clock_used_as_clock` properties will give you an indication of any such constraint on the data path.

- You can check if both `is_clock_used_as_data` and `is_clock_used_as_clock` are reported as "`false`". You can use the following sample script to find any such pin on this path:

```
set path1 [report_timing -to ARB_INST/present_state_reg[1]/D -nworst 100 -
collection -view func_slow_RCMAX]
set lp [get_property $path1 launching_point]

foreach_in_collection g $lp {
if {[get_property $g is_clock_used_as_data]=="false" && [get_property $g
is_clock_used_as_clock]=="false"} {
echo " Pin [get_object_name $g] has property is_clock_used_as_data &
is_clock_used_as_clock as false "
}
}
```

This script does not give any output.

- You can check if `is_clock_used_as_data` is reported as "`true`" and `is_clock_used_as_clock` is reported as "`false`". You can use the following sample script to find any such pin on this path:

```
set q [report_timing -to ARB_INST/present_state_reg[1]/D  -nworst 1000 -collection
-view func_slow_RCMAX]
set lp [get_property $q launching_point]

foreach_in_collection g $lp {
if {[get_property $g is_clock_used_as_data]=="true" && [get_property $g
is_clock_used_as_clock]=="false"} {
 echo "Pin [get_object_name $g] has property is_clock_used_as_data as true and
is_clock_used_as_clock as false"
}
}
```

The output is as follows:

```
Pin ARB_INST/ECO1inst_4/A has property is_clock_used_as_data as true and
is_clock_used_as_clock as false
```

- You may run the `report_timing` command from the `ARB_INST/ECO1inst_4/A` to the `ARB_INST/present_state_reg[1]/D` end point and this path is reported as a valid constraint path. This means that `ARB_INST/ECO1inst_4/A` is also constrained as a valid start point.

```
> report_timing -from ARB_INST/ECO1inst_4/A -to ARB_INST/present_state_reg[1]/D
#############################################################
Path 1: MET Setup Check with Pin ARB_INST/present_state_reg[1]/CK
Endpoint:   ARB_INST/present_state_reg[1]/D (^) checked with  leading edge of
'm_clk'
Beginpoint: ARB_INST/ECO1inst_4/A           (v) triggered by trailing edge of
'vclkm'
Path Groups: {m_clk}
Analysis View: func_fast_RCMIN
Other End Arrival Time          4.326
- Setup                         1.033
+ Phase Shift                   6.000
- Uncertainty                   0.400
= Required Time                 8.893
- Arrival Time                  5.952
= Slack Time                    2.941
```

You can further check SDC and find that the `vlkm` clock is defined at the `ARB_INST/ECO1inst_4/A` pin.

```
> create_clock -name vclkm -period 8 [get_pins ARB_INST/ECO1inst_4/A]
```

- Run the `reset_clock` command to reset the applied clock and check if this path is reported as a constrained path now.

```
> reset_clock vclkm
> report_timing -from TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/Q -to
ARB_INST/present_state_reg[1]/D
Path 1: VIOLATED Setup Check with Pin ARB_INST/present_state_reg[1]/CK
Endpoint:   ARB_INST/present_state_reg[1]/D                     (v) checked with
leading edge of 'm_clk'
Beginpoint: TDSP_CORE_INST/DATA_BUS_MACH_INST/bus_request_reg/Q (^) triggered by
leading edge of 'm_clk'
Path Groups: {m_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          5.817
- Setup                         1.085
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 8.331
- Arrival Time                 12.534
= Slack Time                   -4.202
```

This concluded that this path was unconstrained due to the clock definition applied on the data pin.

Debug for Path 10

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[12]/Q -through_fall
DATA_SAMPLE_MUX_INST/FE_OFC17_n_23/A -to RAM_256x16_TEST_INST/RAM_256x16_INST/D[0]
-unconstrained -debug unconstrained
############################################################
No unconstrained timing paths with given description found.
Paths may be constrained or  may not exist.
```

The tool does not report this path either as unconstrained or constrained.

Run the `report_fanin` command on the end point to find out if paths between these two points are fully traceable or not.

Here is the output of the `report_fanin` command on the end point:

```
> report_fanin -to RAM_256x16_TEST_INST/RAM_256x16_INST/D[0] -view func_slow_RCMAX
############################################################
Fanin Network for View: func_slow_RCMAX
Fanin Network for Pin: RAM_256x16_TEST_INST/RAM_256x16_INST/D[0]
       ------------------------------------------------------------------------------
--------------------------
        Source Pin             Sink Pin               Cell Type         Timing
Sense      Arc        Case

Disabled   Analysis
       ------------------------------------------------------------------------------
--------------------------
       FE_OFC16_ds_datain_0_/Z   RAM_256x16_TEST_INST/RAM_256x16_INST/D[0]   (net)
-                -          -
       ------------------------------------------------------------------------------
--------------------------.
.
.

       ------------------------------------------------------------------------------
--------------------------
       TDSP_CORE_INST/DECODE_INST/ir_reg[12]/CK
TDSP_CORE_INST/DECODE_INST/ir_reg[12]/Q  SDFFS_X2          non_unate         -
-
       TDSP_CORE_INST/DECODE_INST/ir_reg[13]/CK
TDSP_CORE_INST/DECODE_INST/ir_reg[13]/Q   SDFFS_X2          non_unate         -
-
       TDSP_CORE_INST/DECODE_INST/ir_reg[11]/CK
TDSP_CORE_INST/DECODE_INST/ir_reg[11]/Q   SDFFS_X2          non_unate         -
-
       TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9364A/B2
TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9364A/ZN   OAI21_X1          negative_unate
-          -
       TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9364A/B1
TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9364A/ZN   OAI21_X1          negative_unate
-          -
```

```
       TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9364A/A
TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9364A/ZN   OAI21_X1          negative_unate
-          -
       TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9583A/A2
TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9583A/ZN   NAND2_X1          negative_unate
-          -
       TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9583A/A1
TDSP_CORE_INST/TDSP_CORE_GLUE_INST/p9583A/ZN   NAND2_X1          negative_unate
-          -
       ------------------------------------------------------------------------
--------------------------
```

You can notice that it does trace to the 'TDSP_CORE_INST/DECODE_INST/ir_reg[12]/Q' begin point.

- Check if there are instances between the begin point and end point, which are missing either the rise or fall timing arc in the library.
  Such instances can be found using the logic that has the property of either `delay_max_rise/delay_min_rise` as `NA` or `delay_max_fall/delay_min_fall` as `NA`.

Following sample script uses the same logic to find out the instance having either rise or fall arc missing:

```
foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/DECODE_INST/ir_reg[12]/Q -only_cells ] {
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_max_rise] ==
"NA"} {
echo "Max Rise arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_min_rise] ==
"NA"} {
echo "Min rise arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_max_fall] ==
"NA"} {
echo "Max fall arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_min_fall] ==
"NA"} {
echo "Min fall arc is missing in [get_object_name $fanout] "
}
}
```

The output is as follows:

```
Max Rise arc is missing in TDSP_CORE_INST/DECODE_INST/FE_OFC191_ir_12_
Min rise arc is missing in TDSP_CORE_INST/DECODE_INST/FE_OFC191_ir_12_
Max Rise arc is missing in RESULTS_CONV_INST/FE_OFCC469_n_2012
Min rise arc is missing in RESULTS_CONV_INST/FE_OFCC469_n_2012
Max Rise arc is missing in FE_OFC16_ds_datain_0_
Min rise arc is missing in FE_OFC16_ds_datain_0_
```

You can notice that all above instances have the rise arc missing. If you check the library for above instances (`BUFX_16`), you will find that there is no rise arc existing for this cell.

```
> report_instance_library -instance TDSP_CORE_INST/DECODE_INST/FE_OFC191_ir_12_ -
view func_slow_RCMAX
Instance              : TDSP_CORE_INST/DECODE_INST/FE_OFC191_ir_12_
Cell                  : BUFX_16
Analysis View         : func_slow_RCMAX
Power Domain          : Default
Library/Libset        : FreePDK45_lib_v1.0/libset_slow
Library File          : ./libs/liberty/FreePDK45_lib_v1.0_worst.lib
Binding Status        : All PG Pins Voltage Match
Op Cond               : P->1.000000, V->0.800000, T->25.000000 (TypTyp_0.8V_25C)
Cell Type             : gateCell
```

Debug for Path 11

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/D -unconstrained -debug unconstrained
############################################################
No unconstrained timing paths with given description found.
Paths may be constrained or  may not exist.
```

The tool does not report this path either as unconstrained or constrained.

- Run `report_fanin` command on the end point to find out if paths between these two points are fully traceable or not.

Here is the output of the `report_fanin` command on the end point.

```
> report_fanin -to TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/D -view func_slow_RCMAX
############################################################
Fanin Network for View: func_slow_RCMAX
Fanin Network for Pin: TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/D
       ------------------------------------------------------------------------------
--------------------------
        Source Pin             Sink Pin               Cell Type         Timing
Sense      Arc        Case

Disabled   Analysis
       ------------------------------------------------------------------------------
--------------------------
       TDSP_CORE_INST/EXECUTE_INST/n0439D/ZN
TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/D   (net)             -                -
-
.
.
.
       TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/CK
TDSP_CORE_INST/EXECUTE_INST/acc_reg[1]/Q   SDFF_X2           non_unate         -
-
```

```
       TDSP_CORE_INST/DECODE_INST/ir_reg[2]/CK
TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q   SDFFR_X1          non_unate         -
-
       TDSP_CORE_INST/EXECUTE_INST/p_reg[2]/CK
TDSP_CORE_INST/EXECUTE_INST/p_reg[2]/Q   SDFF_X2           non_unate         -
-
       ------------------------------------------------------------------------
-------------------------
```

You can notice that it does trace to the 'TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q' begin point.

- Check if there are instances between the begin point and end point, which are missing either the rise or fall timing arc in the library.

Such instances can be found using the logic that has the property of either delay_max_rise/delay_min_rise as NA or delay_max_fall/delay_min_fall as NA.

Following sample script uses the same logic to find out the instance having either the rise or fall arc missing:

```
foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q  -only_cells ] {
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_max_rise] ==
"NA"} {
echo "Max Rise arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_min_rise] ==
"NA"} {
echo "Min rise arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_max_fall] ==
"NA"} {
echo "Max fall arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_min_fall] ==
"NA"} {
echo "Min fall arc is missing in [get_object_name $fanout] "
}
}
```

This script does not give any output.

This concludes that there is no instance between the begin and end points, which has either the rise or fall arc missing.

- Check if there is any clock sense applied between begin and end points. To find such pins, you can use the logic that no clock and data phase will be reaching that pin and the tool will show the arrival_window property as NA.

```
> foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q] {
    set pin [get_pins $fanout]
    if {[get_property $pin arrival_window] == "NA"} {
    echo " Pin with arrival window as NA is [get_object_name $pin] "
    return
    }
    }
```

The output is as follows:

The pin with the arrival window as NA is TDSP_CORE_INST/DECODE_INST/FE_OFC203_ir_2_/A.

- Run the `report_cell_instance_timing` command to check clock and data phases on this pin. You will notice that no phase is reaching this pin.

```
> report_cell_instance_timing TDSP_CORE_INST/DECODE_INST/FE_OFC203_ir_2_
############################################################
      ------------------------------------------------------------------------------
                                          Instance
TDSP_CORE_INST/DECODE_INST/FE_OFC203_ir_2_ of BUF_X16
      ------------------------------------------------------------------------------
        Pin     Dir    Propagated   Arrival   Required   Slack   Phase
                          Slew
      ------------------------------------------------------------------------------
           A      IN                                                (func_slow_RCMAX)
```

This confirms that no clock and data phase is reaching this pin. You can check SDC on the above pin for the `set_clock_sense` constraint.

```
> set_clock_sense  -stop { all }  [get_pins {
TDSP_CORE_INST/DECODE_INST/FE_OFC203_ir_2_/A }]
```

- Run the `reset_clock_sense` command to reset the applied clock sense and check if this path is reported as a constrained path now.

```
> reset_clock_sense TDSP_CORE_INST/DECODE_INST/FE_OFC203_ir_2_/A
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q -to
TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/D
Path 1: VIOLATED Setup Check with Pin TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/CK
Endpoint:   TDSP_CORE_INST/EXECUTE_INST/p_reg[27]/D (v) checked with  leading edge
of 'm_clk'
Beginpoint: TDSP_CORE_INST/DECODE_INST/ir_reg[2]/Q  (^) triggered by  leading edge
of 'm_clk'
Path Groups: {m_clk}
Analysis View: func_slow_RCMAX
Other End Arrival Time          5.745
- Setup                         1.195
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 8.150
- Arrival Time                 33.489
= Slack Time                  -25.339
```

This concluded that this path was unconstrained due to the clock sense applied on the data pin.

Debug for Path 12

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[0]/Q -to
TDSP_CORE_INST/PROG_BUS_MACH_INST/data_out_reg[6]/D
############################################################
No constrained timing paths with given description found.
Paths may be unconstrained (try '-unconstrained' option) or  may not exist.
```

The tool does not report this path either as unconstrained or constrained.

- Run `report_fanin` command on the end point to find out if paths between these two points are fully traceable or not.

Here is the output of the `report_fanin` command on the end point:

```
> report_fanin -to TDSP_CORE_INST/PROG_BUS_MACH_INST/data_out_reg[6]/D -view
func_slow_RCMAX
############################################################
Fanin Network for View: func_slow_RCMAX
Fanin Network for Pin: TDSP_CORE_INST/PROG_BUS_MACH_INST/data_out_reg[6]/D
       ------------------------------------------------------------------------------
--------------------------
        Source Pin             Sink Pin               Cell Type         Timing
Sense      Arc        Case

Disabled   Analysis
       ------------------------------------------------------------------------------
--------------------------
       TDSP_CORE_INST/PROG_BUS_MACH_INST/p11493A344/Z
TDSP_CORE_INST/PROG_BUS_MACH_INST/data_out_reg[6]/D   (net)             -
-          -
       ------------------------------------------------------------------------------
--------------------------.
.
       ------------------------------------------------------------------------------
--------------------------
       TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[2]/CK
TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[2]/Q   SDFFR_X2
non_unate         -          -
       TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[0]/CK
TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[0]/Q   SDFFR_X2
non_unate         -          -
       ------------------------------------------------------------------------------
--------------------------
```

You can notice that it does trace to the '`TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[0]/Q`' begin point.

- Check if there are instances between the begin point and end point, which are missing either the rise or fall timing arc in the library.

Such instances can be found using the logic that has the property of either `delay_max_rise/delay_min_rise` as `NA` or `delay_max_fall/delay_min_fall` as `NA`:

Following sample script uses the same logic to find out the instance having either the rise or fall arc missing:

```
foreach_in_collection fanout [all_fanout -from
TDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[0]/Q  -only_cells ] {
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_max_rise] ==
"NA"} {
echo "Max Rise arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_min_rise] ==
"NA"} {
echo "Min rise arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_max_fall] ==
"NA"} {
echo "Max fall arc is missing in [get_object_name $fanout] "
}
if {[get_property [get_arcs -of_objects [get_cells $fanout]] delay_min_fall] ==
"NA"} {
echo "Min fall arc is missing in [get_object_name $fanout] "
}
}
```

This script does not give any output.

This concludes that there is no instance between the begin and end points, which has missing either the rise or fall arc.

- Check if there is any clock sense applied between the begin and end points. To find such pins, you can use the logic that no clock and data phase will be reaching the pin and the tool will show the `arrival_window` property as `NA`.

```
> foreach_in_collection fanout [all_fanout -
fromTDSP_CORE_INST/PROG_BUS_MACH_INST/present_state_reg[0]/Q   ] {
    set pin [get_pins $fanout]
    if {[get_property $pin arrival_window] == "NA"} {
    echo " Pin with arrival window as NA is [get_object_name $pin] "
    return
    }
    }
```

This script does not give any output. This confirms that there is no clock sense applied on the data path.

- Check if `set_max_delay` or `set_min_delay` is applied between the begin and end points. If there is any such constraint, it will break the path.

Run `all_fanin` to the end point.

```
 > all_fanin -to TDSP_CORE_INST/PROG_BUS_MACH_INST/data_out_reg[6]/D -
startpoints_only
            TDSP_CORE_INST/PROG_BUS_MACH_INST/FE_OFCC315_FE_OFN9_n_21/A
ROM_512x16_0_INST/CLK TDSP_CORE_INST/PROG_BUS_MACH_INST/data_out_reg[6]/CK
```

Run the following command on the intermediate pin to check the `delay_max_rise/fall` and `delay_min_rise/fall` properties. If this returns any value, it means `set_max_delay` or `set_min_delay` is applied.

```
> get_property [get_pins
TDSP_CORE_INST/PROG_BUS_MACH_INST/FE_OFCC315_FE_OFN9_n_21/A] delay_max_rise
0.800
> get_property [get_pins
TDSP_CORE_INST/PROG_BUS_MACH_INST/FE_OFCC315_FE_OFN9_n_21/A] delay_max_fall
0.800

> get_property [get_pins
TDSP_CORE_INST/PROG_BUS_MACH_INST/FE_OFCC315_FE_OFN9_n_21/A] delay_min_rise
NA
> get_property [get_pins
TDSP_CORE_INST/PROG_BUS_MACH_INST/FE_OFCC315_FE_OFN9_n_21/A] delay_min_fall
NA
```

Since it returns the `delay_max_rise/fall` value, it means `set_max_delay` is applied. Run `write_sdc` or check in SDC to confirm if `set_max_delay` is applied on that pin. You will find the following constraint in SDC:

```
> set_max_delay 0.8  -from [get_pins
{TDSP_CORE_INST/PROG_BUS_MACH_INST/p11493A352/A}]  -to [get_pins
{TDSP_CORE_INST/PROG_BUS_MACH_INST/FE_OFCC315_FE_OFN9_n_21/A}]
```

Note: If `all_fanin -to <pinName> -startpoints_only` traces fully, that is, until the start point, check the `delay_max_rise/fall` and `delay_min_rise/fall` properties on the end point. If it returns any value, it means that `set_max_delay or set_min_delay` is applied on the end point.

Debug for Path 13

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
 > report_timing -from port_pad_data_in[15] -unconstrained -debug unconstrained

Path Unconstrained Reason:
------------------------------
The state of global 'timing_apply_default_primary_input_assertion' is false
The launching point 'port_pad_data_in[15]' is an unconstrained input port
```

The tool gives the reason as `unconstrained input port`.

- This message comes if input ports are not constrained to a clock.
- Check the SDC constraint file if the input port is constraint using the `set_input_delay` command with the `-clock` option.

If you apply `set_input_delay` with the `-clock` option on this port, a constraint timing path will be reported.

```
> set_input_delay 0.2 port_pad_data_in[15] -clock ck2
> report_timing -from port_pad_data_in[15]
Path 1: MET Setup Check with Pin
TDSP_CORE_INST/PORT_BUS_MACH_INST/data_out_reg[15]/CK
Endpoint:   TDSP_CORE_INST/PORT_BUS_MACH_INST/data_out_reg[15]/D (v) checked with
leading edge of 'm_clk'
Beginpoint: port_pad_data_in[15]                                (v) triggered by
leading edge of 'ck2'
Path Groups: {m_clk}
Analysis View: scan_fast_RCMIN
Other End Arrival Time          4.321
- Setup                         1.129
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 6.792
- Arrival Time                  0.751
= Slack Time                    6.041
```

Debug for Path 14

- Run the `report_timing` command with the "`-debug unconstrained`" option.

```
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[14]/Q -to
TDSP_CORE_INST/EXECUTE_INST/skip_one_reg/D -unconstrained -debug unconstrained

Path Unconstrained Reason:
------------------------------
There is no clock reaching to the launching point
TDSP_CORE_INST/DECODE_INST/ir_reg[14]/CK
```

The tool gives the reason as `There is no clock reaching to the launching point`.

- Run the `report_clock_propagation` command to identify the reason the clock is not reaching the clock pin.

```
> report_clock_propagation -to TDSP_CORE_INST/DECODE_INST/ir_reg[14]/CK -clock
m_clk -view scan_slow_RCMAX

Clock: m_clk
Point: TDSP_CORE_INST/DECODE_INST/ir_reg[14]/CK
View:  scan_slow_RCMAX
Reason 1: The clock 'm_clk' propagation stops due to the inactive timing arc:
```

```
 (m_clk__L18_I14/A->m_clk__L18_I14/ZN)
```

You can notice from this report that the `A->ZN` arc is inactive for the `m_clk__L18_I14` instance.

- Run the `report_inactive_arc` command on this instance to identify the reason for the inactive arc.

```
>   report_inactive_arcs m_clk__L18_I14 -view scan_slow_RCMAX
   ----------------------------------------------------------------------------
    From                          To                            ArcType
Sense                       Reason                      View
   ----------------------------------------------------------------------------
   m_clk__L18_I14 /A             m_clk__L18_I14 /ZN             combinational
negative_unate              User Disable                scan_slow_RCMAX
   ----------------------------------------------------------------------------
```

You can notice from the report that the arc between `A->Z` of the `m_clk_L19_I6` instance is disabled.

- Run `reset_disable_timing` to reset disabled timing arcs and check if this path is reported as a constrained path now.

```
> reset_disable_timing -from A -to ZN m_clk__L18_I14
> report_timing -from TDSP_CORE_INST/DECODE_INST/ir_reg[14]/Q -to
TDSP_CORE_INST/EXECUTE_INST/skip_one_reg/D

Path 1: VIOLATED Setup Check with Pin TDSP_CORE_INST/EXECUTE_INST/skip_one_reg/CK
Endpoint:   TDSP_CORE_INST/EXECUTE_INST/skip_one_reg/D (^) checked with  leading
edge of 'm_clk'
Beginpoint: TDSP_CORE_INST/DECODE_INST/ir_reg[14]/Q    (^) triggered by  leading
edge of 'm_clk'
Path Groups: {m_clk}
Analysis View: scan_slow_RCMAX
Other End Arrival Time          5.813
- Setup                         1.198
+ Phase Shift                   4.000
- Uncertainty                   0.400
= Required Time                 8.215
- Arrival Time                 13.366
= Slack Time                   -5.150
```

This concluded that this path was unconstrained due to `set_disable_timing`.

Support

Cadence Learning and Support Portal provides access to support resources, including an extensive knowledge base, access to software updates for Cadence products, and the ability to interact with Cadence Customer Support. Visit https://support.cadence.com.

Feedback

Email comments, questions, and suggestions to content_feedback@cadence.com.