

Vivado Design Suite Tutorial

Design Analysis and Closure Techniques

UG938 (v2019.1) August 12, 2019

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
08/12/2019 Version 2019.1	
General Updates	Validated for 2019.1.

Table of Contents

Revision History	2
Chapter 1: Tutorial Overview	5
Introduction.....	5
Tutorial Description.....	5
Software Requirements.....	5
Locating Tutorial Design Files.....	5
Chapter 2: Lab 1: Setting Waivers with the Vivado IDE	6
Introduction.....	6
Step 1: Starting the Vivado IDE.....	6
Step 2: Generating the CDC Report.....	7
Step 3: Waiving a Single CDC Violation.....	8
Step 4: Generating a Report for Waived Violations.....	12
Step 5: Generating a Text Report with Details for Waived Violations.....	13
Step 6: Waiving Multiple CDC Violations.....	14
Step 7: Exporting Waivers.....	17
Step 8: Using the create_waiver Command.....	18
Step 9: Waiving Multiple CDC Violations.....	19
Step 10: Waiving Multiple DRC Violations.....	21
Step 11: Generating a Summary Report for Waived Violations.....	26
Step 12: Using Waiver Commands.....	29
Summary.....	30
Chapter 3: Lab 2: Using Report QoR Suggestions	31
Introduction.....	31
Step 1: Understanding the Design.....	31
Step 2: Running Report QoR Suggestions.....	34
Step 3: Understanding the Report.....	35
Step 4: Run with Suggestions.....	41
Step 5: Design Closure.....	44
Summary.....	45

Appendix A: Additional Resources and Legal Notices..... 46

Please Read: Important Legal Notices..... 46

Tutorial Overview

Introduction

This tutorial uses the Vivado® design rules checker (`report_drc`), clock domain crossing checker (`report_cdc`), and quality of results enhancer (`report_qor_suggestions`) to analyze example designs for issues, and shows you how to take corrective actions.

Tutorial Description

Lab 1 walks you through creating waivers for CDC, methodology, and DRC violations.

Lab 2 is a guide to using the `report_qor_suggestions` (RQS) command.

Note: The designs used in this tutorial are intended to exhibit issues for demonstration purposes, and should not be used as a reference for designs outside this tutorial.

Software Requirements

This tutorial requires that the 2019.1 Vivado® Design Suite software release or later is installed.

For a complete list of system and software requirements, see the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* ([UG973](#)).

Locating Tutorial Design Files

1. Download the [reference design files](#) from the Xilinx® website.
2. Extract the ZIP file contents into any write-accessible location.

This tutorial refers to the location of the extracted ZIP file contents as `<Extract_Dir>`.

Lab 1: Setting Waivers with the Vivado IDE

Introduction

In the Vivado[®] Design Suite, you can use the waiver mechanism to waive clock domain crossing (CDC), design rule check (DRC), or methodology check violations. After a violation is waived, it is no longer reported by the `report_cdc`, `report_drc`, or `report_methodology` commands. Waived checks are also filtered out from the mandatory DRCs run at the start of the implementation commands, such as `opt_design`, `place_design`, and `route_design`. For more information, see this [link](#) in the *Vivado Design Suite User Guide: Design Analysis and Closure Techniques (UG906)*.



IMPORTANT! *The content of the waiver is built with the objects that exist when the waiver is created. However, if an instance referenced inside a waiver is replicated by Vivado[®], the replicated instance is automatically added to the waiver and saved in subsequent checkpoints and XDC.*

This lab shows how to set waivers with the Vivado integrated design environment (IDE) using both menu commands and the Tcl Console. The lab focuses on CDC waivers, but the methods for waiving DRC and methodology violations are similar.

Step 1: Starting the Vivado IDE

This lab uses a Vivado design checkpoint (`.dcp` file), which is a snapshot of a design. When you launch the Vivado IDE using a design checkpoint, a subset of the Vivado IDE functionality is available.



TIP: *To launch the Vivado Tcl Shell on Windows, select **Start** → **All Programs** → **Xilinx Design Tools** → **Vivado <version>** → **Vivado <version> Tcl Shell**.*

1. From the command line or the Vivado Tcl Shell, change to the directory where the lab materials are stored:

```
cd <Extract_Dir>/src/lab1
```

- To start the Vivado IDE with the design checkpoint loaded, enter the following:

```
vivado my_ip_example_design_placed.dcp
```

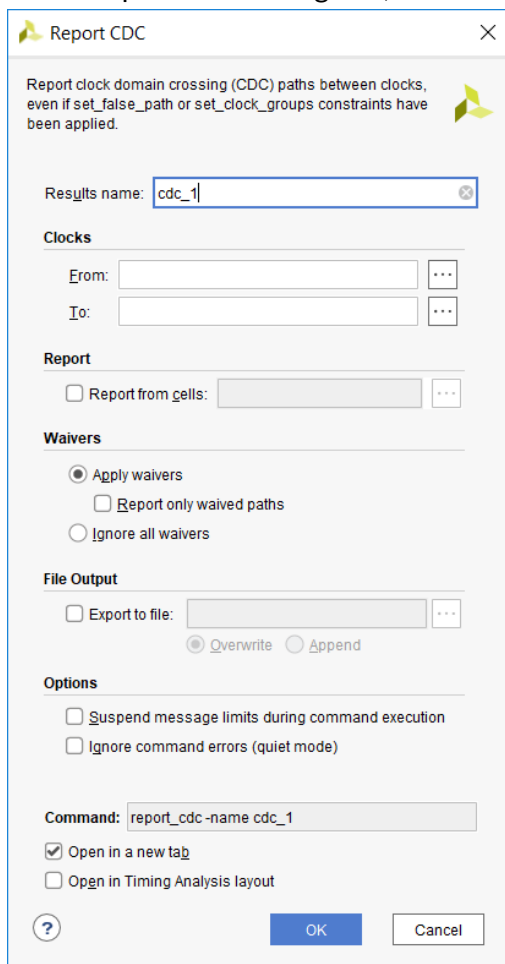


TIP: You can disregard the critical warnings about the unbounded GT locations.

Step 2: Generating the CDC Report

In this step, you generate the CDC report to view the associated CDC violations.

- Select **Reports** → **Timing** → **Report CDC**.
- In the Report CDC dialog box, leave the default settings as-is, and click **OK**.



The screenshot shows the 'Report CDC' dialog box with the following settings:

- Results name:** cdc_1
- Clocks:** From: [empty], To: [empty]
- Report:** Report from cells: [empty]
- Waivers:** Apply waivers (selected), Report only waived paths, Ignore all waivers
- File Output:** Export to file: [empty], Overwrite (selected), Append
- Options:** Suspend message limits during command execution, Ignore command errors (quiet mode)
- Command:** report_cdc -name cdc_1
- Open in a new tab:** checked
- Open in Timing Analysis layout:** unchecked

The Summary (by clock pair) section of the CDC Report appears as follows.

Severity	Source Clock	Destination Clock	CDC Type	Exceptions	Endpoints	Safe	Unsafe	Unknown	No ASYNC_REG
Critical	my_ip_glblck	my_ip_axi_aclk	No Common Primary Clock	False Path	2	1	1	0	0
Critical	my_ip_axi_aclk	my_ip_drpclk	No Common Primary Clock	False Path	2	0	2	0	0
Critical	my_ip_axi_aclk	my_ip_glblck	No Common Primary Clock	False Path	942	12	351	579	185
Info	my_ip_drpclk	my_ip_axi_aclk	No Common Primary Clock	False Path	1	1	0	0	0
Info	input port clock	my_ip_drpclk	No Common Primary Clock	False Path	2	2	0	0	0
Info	my_ip_glblck	my_ip_drpclk	No Common Primary Clock	False Path	6	6	0	0	0
Info	my_ip_drpclk	my_ip_glblck	No Common Primary Clock	False Path	2	2	0	0	0

The Summary (by CDC type) section appears as follows.

Severity	ID	Count	Description
Critical	CDC-1	536	1-bit unknown CDC circuitry
Critical	CDC-4	4	Multi-bit unknown CDC circuitry
Critical	CDC-10	187	Combinational logic detected before a synchronizer
Critical	CDC-11	2	Fan-out from launch flop to destination clock
Critical	CDC-13	170	1-bit CDC path on a non-FD primitive
Critical	CDC-14	5	Multi-bit CDC path on a non-FD primitive
Warning	CDC-15	10	Clock enable controlled CDC structure detected
Info	CDC-3	9	1-bit synchronized with ASYNC_REG property
Info	CDC-9	5	Asynchronous reset synchronized with ASYNC_REG property

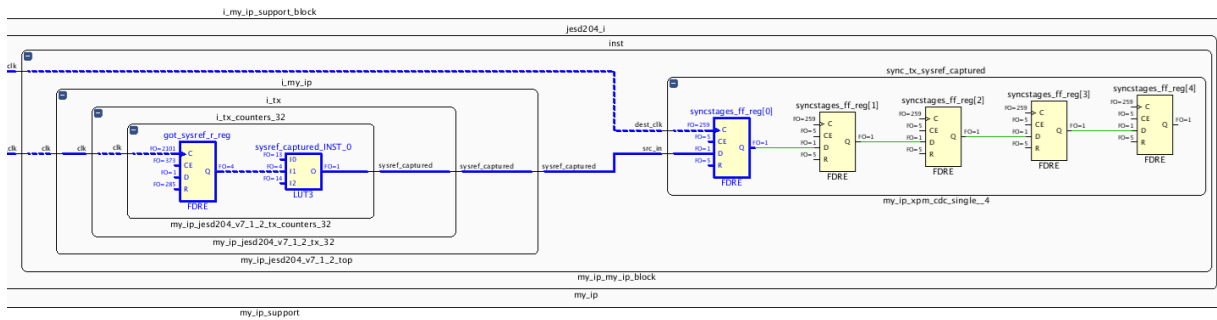
Step 3: Waiving a Single CDC Violation

The my_ip_glblck to my_ip_axi_aclk clock pair includes one Critical CDC-10 violation due to combinational logic on the CDC path. This step covers how to waive the CDC-10 violation.

Severity	ID	Description	Depth	Exception	Source (From)	Destination (To)	Category
Critical	CDC-10	Combinational logic detected before a synchronizer	5	False Path	I_my_ip_supp...ysref_r_reg/C	I_my_ip_supp...s_ff_reg[0]/D	Unsafe
Info	CDC-3	1-bit synchronized with ASYNC_REG property	5	False Path	I_my_ip_supp...ot_sync_reg/C	I_my_ip_supp...s_ff_reg[0]/D	Safe

- To view a schematic of the violation, select the CDC-10 row in the CDC Report, and click the Schematic toolbar button

Note: Alternatively, you can press F4 to generate the schematic. However, using the toolbar button provides a more detailed schematic that includes all the levels of the downstream synchronizer.



- To waive the violation, select the **CDC-10** row in the CDC Report, right-click, and select **Create Waiver**.
- In the Create Waiver dialog box, enter a description, and click **OK**.

Create Waiver
✕

Create waiver for 1 cdc path

User:

Description:

Tags:

Tcl Command Preview

```
create_waiver -type CDC -id CDC-10 -from [get_pins i_my_ip_support_block/jesd204_i/inst/i_my_ip/i_tx/i_tx_counters
```

?
OK
Cancel



IMPORTANT! A waiver tracks the date the waiver was added, the user that added the waiver, and a description of why the violation was waived. The date is automatically added by the system. The Tags field is an optional description or list of keywords that can be used for documentation purposes.

- After the waiver is created, check the CDC Report.

To indicate that a waiver was created, the CDC-10 row is gray and disabled.

Note: Rows are only disabled in the Report CDC result window from which the waivers were created.

Severity	ID	Description	Depth	Exception	Source (From)	Destination (To)	Category
Critical warning	CDC-10	Combinational logic detected before a synchronizer	5	False Path	I_my_ip_support_sysref_r_reg/C	I_my_ip_support_es_f_reg0/D	Unsafe
Info	CDC-3	1-bit synchronized with ASYNC_REG property	5	False Path	I_my_ip_support_ot_sync_reg/C	I_my_ip_support_es_f_reg0/D	Safe

- To see the impact of the CDC-10 waiver, select **Reports** → **Timing** → **Report CDC** to rerun Report CDC.

Note: When a waiver is created or deleted, you must rerun Report CDC, Report DRC, or Report Methodology to see the updated results.

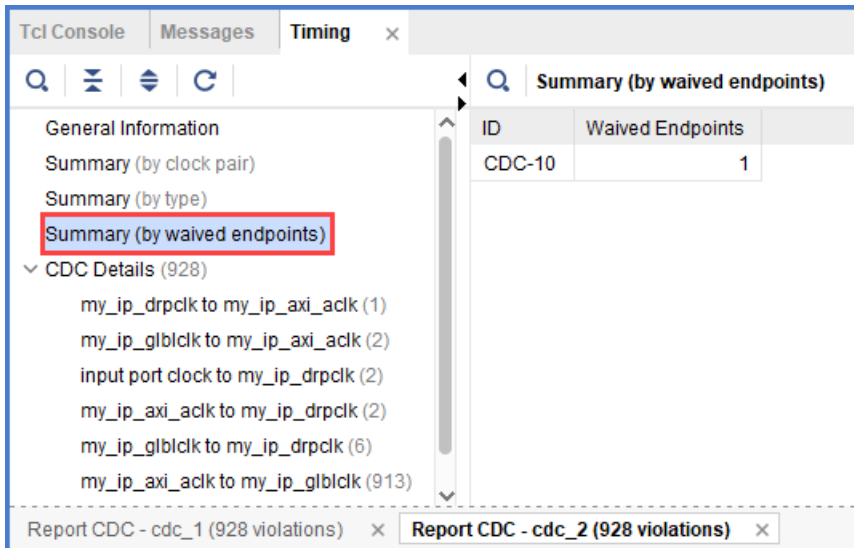
- See the CDC Report to view the updated information.

The differences from the previous Summary by clock pair and Summary by type sections are highlighted in red in the following figures.

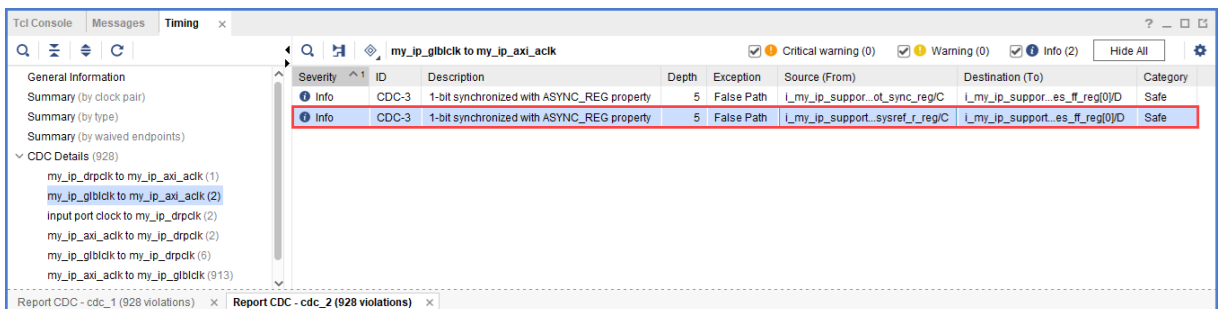
Severity	Source Clock	Destination Clock	CDC Type	Exceptions	Endpoints	Safe	Unsafe	Unknown	No ASYNC_REG
Critical	my_ip_axi_ack	my_ip_drpclk	No Common Primary Clock	False Path	2	0	2	0	0
Critical	my_ip_axi_ack	my_ip_glbclk	No Common Primary Clock	False Path	942	12	351	579	185
Info	my_ip_glbclk	my_ip_axi_ack	No Common Primary Clock	False Path	1	1	0	0	0
Info	my_ip_glbclk	my_ip_axi_ack	No Common Primary Clock	False Path	2	2	0	0	0
Info	input port clock	my_ip_drpclk	No Common Primary Clock	False Path	2	2	0	0	0
Info	my_ip_glbclk	my_ip_drpclk	No Common Primary Clock	False Path	6	6	0	0	0
Info	my_ip_drpclk	my_ip_glbclk	No Common Primary Clock	False Path	2	2	0	0	0

Severity	ID	Count	Description
Critical	CDC-1	536	1-bit unknown CDC circuitry
Critical	CDC-4	4	Multi-bit unknown CDC circuitry
Critical	CDC-10	186	Combinational logic detected before a synchronizer
Critical	CDC-11	2	Fan-out from launch flop to destination clock
Critical	CDC-13	170	1-bit CDC path on a non-FD primitive
Critical	CDC-14	5	Multi-bit CDC path on a non-FD primitive
Warning	CDC-15	10	Clock enable controlled CDC structure detected
Info	CDC-3	10	1-bit synchronized with ASYNC_REG property
Info	CDC-9	5	Asynchronous reset synchronized with ASYNC_REG property

You can also view a summary with the list of waived endpoints.

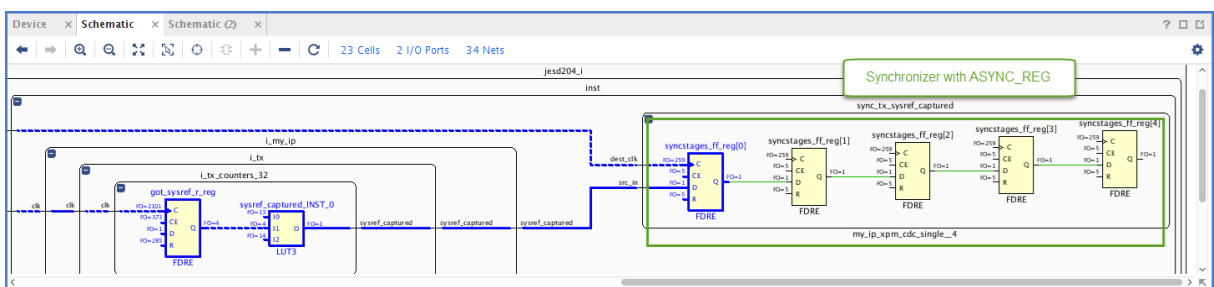


The detailed section for the `my_ip_glblclk` to `my_ip_axi_aclk` CDC shows that the Critical CDC-10 was replaced with an Info CDC-3.



7. Select the new CDC-3 row, and click the Schematic toolbar button . Double click on the Q pin of the output register to expand the schematic to match what is shown in the following figure.

The CDC path includes a 5-level synchronizer on the output of the selected destination register. This is the reason the CDC-10 was replaced with CDC-3 for this topology, as shown in the following figure.





IMPORTANT! In this lab, Report CDC only reports a single violation per endpoint and per clock pair. When multiple violations apply to a specific endpoint, only the violation with the highest precedence is reported. Because CDC-10 has a higher precedence than CDC-3, only CDC-10 is reported when both CDC-10 and CDC-3 apply to the same endpoint. For more information on CDC rules precedence, see this [link](#) in the Vivado Design Suite User Guide: Design Analysis and Closure Techniques (UG906).



TIP: To report all of the CDC violations for each endpoint regardless of the precedence rules, use the command line option `-all_checks_per_endpoint`.

Step 4: Generating a Report for Waived Violations

You can generate a report for the CDC, DRC, or methodology check violations that were waived. This step shows how to generate a report for waived CDC violations using the Tcl Console as well as the Vivado® IDE menu commands.

Generating a Text Report for Waived Violations

1. In the Tcl Console, enter:

```
report_cdc -waived
```

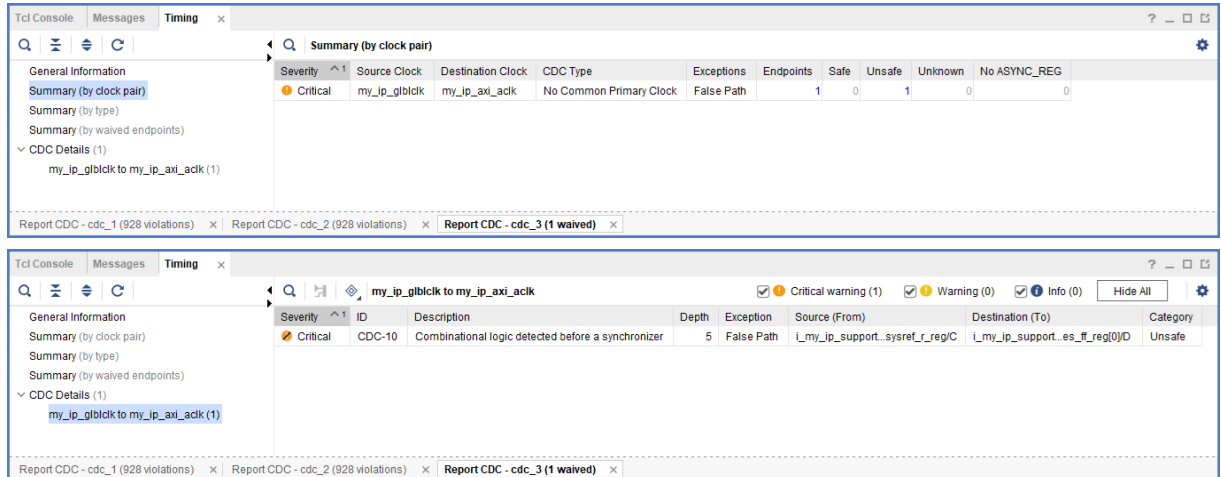
2. In the CDC report, verify that a single CDC-10 violation is listed, because only one waiver was created.

CDC Report						
ID	Severity	Count	Description	Depth	Level	Destination (To)
CDC-10	Critical	1	Combinational logic detected before a synchronizer	5	Fail	_reg/C 1_my_ip_support_block/jesd204_i/inst/sync_tx_sysref_captured/syncstages_ff_reg[01]/D
Waived Endpoints						
ID	Waived Endpoints					
CDC-10	1					
Source Clock: my_ip_glblclk						
Destination Clock: my_ip_axi_aclk						
CDC Type: No Common Primary Clock						
Row	ID	Severity	Description	Depth	Level	Destination (To)
1	CDC-10	Critical	Combinational logic detected before a synchronizer	5	Fail	_reg/C 1_my_ip_support_block/jesd204_i/inst/sync_tx_sysref_captured/syncstages_ff_reg[01]/D

Generating a Vivado IDE CDC Report for Waived Violations

1. Select **Reports** → **Timing** → **Report CDC**.
2. In the Report CDC dialog box, enable Report only waived paths, and click **OK**.
3. In the CDC Report, check the Summary (by clock pair) and CDC Details to verify that a single CDC-10 violation is listed.

Note: The icon next to the violation shows that the violation was waived 🚫.



Step 5: Generating a Text Report with Details for Waived Violations

In this step, you generate text reports with additional details, including a list of all of the rules and all of the violations regardless of the waivers.

Generating a List of Rules with Waived Violations

1. In the Tcl Console, enter:

```
report_cdc -details -show_waiver
```

2. Verify that the `my_ip_glbclk to my_ip_axi_aclk` CDC-10 violation is waived and the two CDC-3 violations are not waived.

Note: In the text report, all of the rules are reported, whether they were waived or not. The Waived column indicates the status of the rule.

ID	Severity	Count	Description	Depth	Exception	Source (From)	Destination (To)	Waived
CDC-1	Critical	536	1-bit unknown CDC circuitry					
CDC-3	Info	10	1-bit synchronized with ASYNC_REG property					
CDC-4	Critical	4	Multi-bit unknown CDC circuitry					
CDC-9	Info	5	Asynchronous reset synchronized with ASYNC_REG property					
CDC-10	Critical	186	Combinational logic detected before a synchronizer					
CDC-11	Critical	2	Fan-out from launch flop to destination clock					
CDC-13	Critical	170	1-bit CDC path on a non-FD primitive					
CDC-14	Critical	5	Multi-bit CDC path on a non-FD primitive					
CDC-15	Warning	10	Clock enable controlled CDC structure detected					
ID Waived								
CDC-10		1						Y
Source Clock: my_ip_glbclk Destination Clock: my_ip_axi_aclk CDC Type: No Common Primary Clock								
Row	ID	Severity	Description	Depth	Exception	Source (From)	Destination (To)	Waived
1	CDC-3	Info	1-bit synchronized with ASYNC_REG property	5	False Path	I_my_ip_support_b...	s_32/got_sync_reg/C	N
2	CDC-3	Info	1-bit synchronized with ASYNC_REG property	5	False Path	I_my_ip_support_b...	s_32/got_sysref_r_reg/C	N
3	CDC-10	Critical	Combinational logic detected before a synchronizer	5	False Path	I_my_ip_support_b...	s_32/got_sysref_r_reg/C	Y

Generating a List of All Violations Regardless of the Waivers

1. In the Tcl Console, enter:

```
report_cdc -no-waiver
```

2. In the text report, verify that the table matches the original report from Report CDC before the CDC-10 waiver was created.

CDC Report										
Severity	Source Clock	Destination Clock	CDC Type	Exceptions	Endpoints	Safe	Unsafe	Unknown	No ASYNC_REG	
Critical	my_ip_glbclk	my_ip_axi_aclk	No Common Primary Clock	False Path	2	1	1	0	0	0
Critical	my_ip_axi_aclk	my_ip_drpclk	No Common Primary Clock	False Path	2	0	2	0	0	0
Critical	my_ip_axi_aclk	my_ip_glbclk	No Common Primary Clock	False Path	942	12	351	579	185	0
Info	my_ip_drpclk	my_ip_axi_aclk	No Common Primary Clock	False Path	1	1	0	0	0	0
Info	input port clock	my_ip_drpclk	No Common Primary Clock	False Path	2	2	0	0	0	0
Info	my_ip_glbclk	my_ip_drpclk	No Common Primary Clock	False Path	6	6	0	0	0	0
Info	my_ip_drpclk	my_ip_glbclk	No Common Primary Clock	False Path	2	2	0	0	0	0

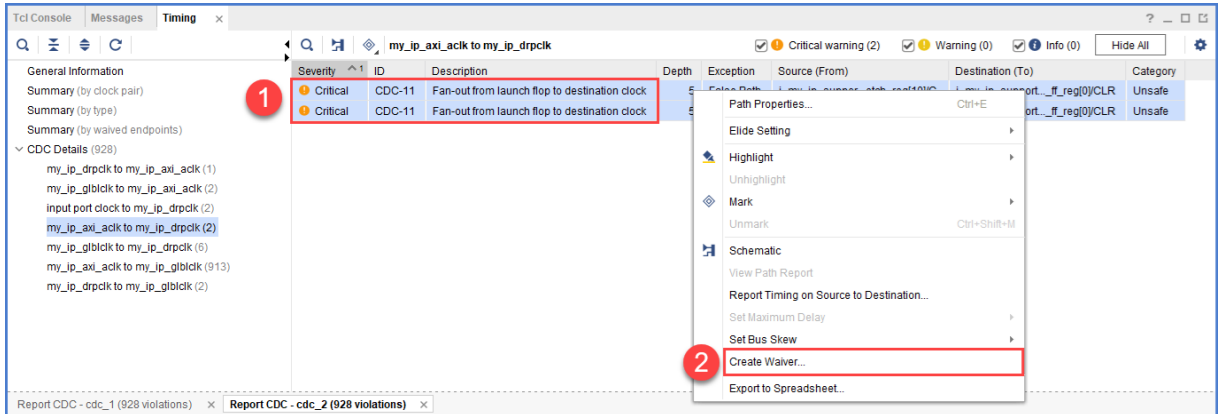


TIP: You can also generate a list of all violations regardless of the waivers from the Vivado IDE. Select **Reports** → **Timing** → **Report CDC**. In the Report CDC dialog box, enable **Ignore all waivers**, and click **OK**.

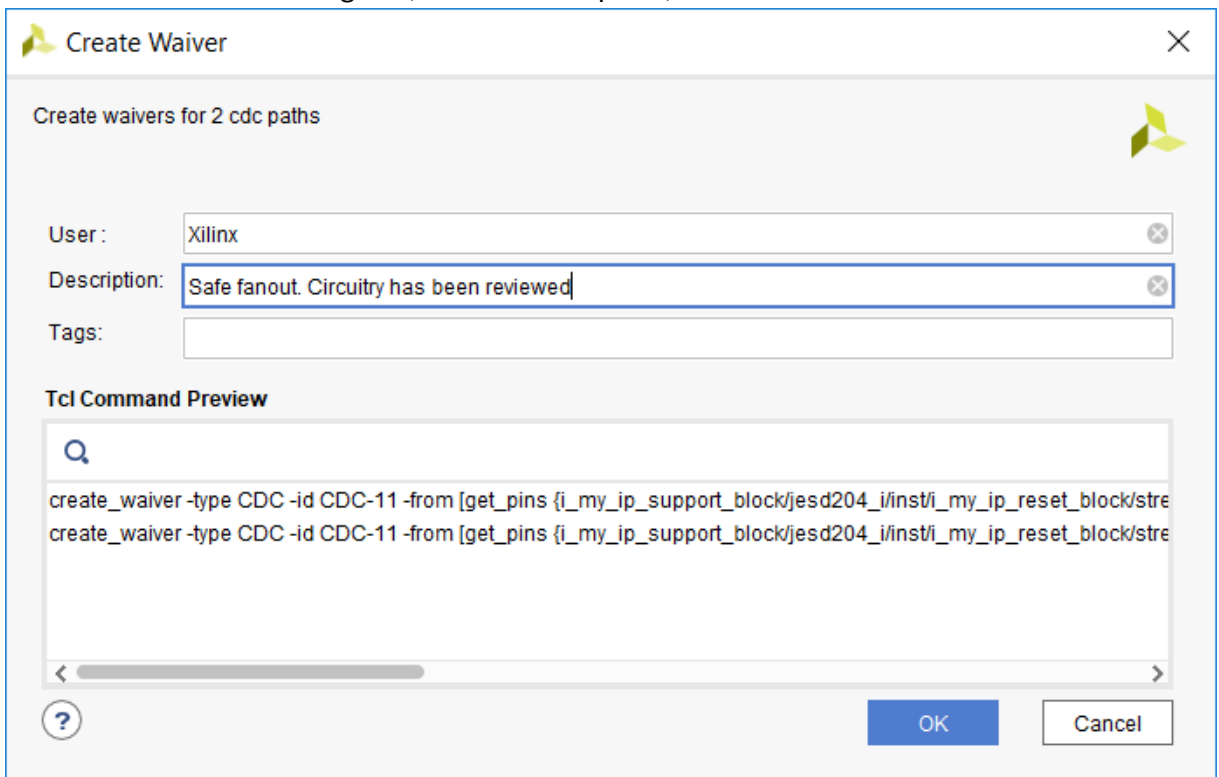
Step 6: Waiving Multiple CDC Violations

The my_ip_axi_aclk to my_ip_drpclk CDC includes two Critical CDC-11 violations. This step covers how to waive both CDC-11 violations simultaneously.

1. To waive the violations, select the **CDC-11** rows in the CDC Report, right-click, and select **Create Waiver**.



2. In the Create Waiver dialog box, enter a description, and click **OK**.



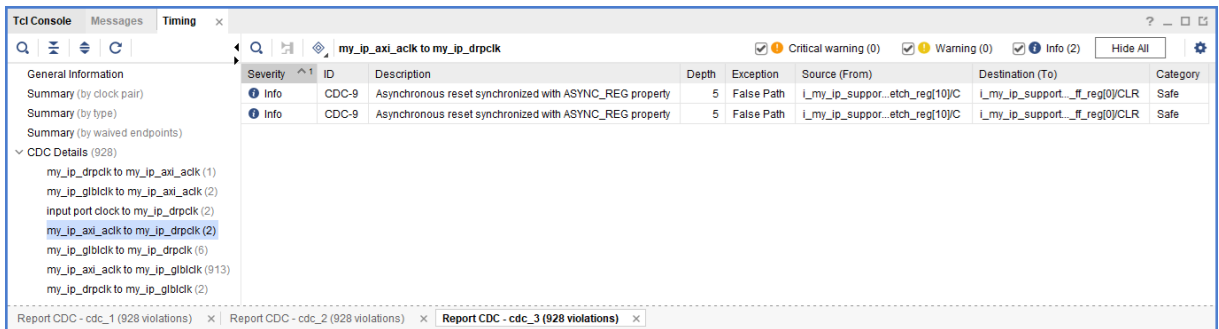
In the Timing Report, the two selected rows are disabled when the waivers are created.


Note: One waiver is created for each selected row. In this example, two waivers are created.

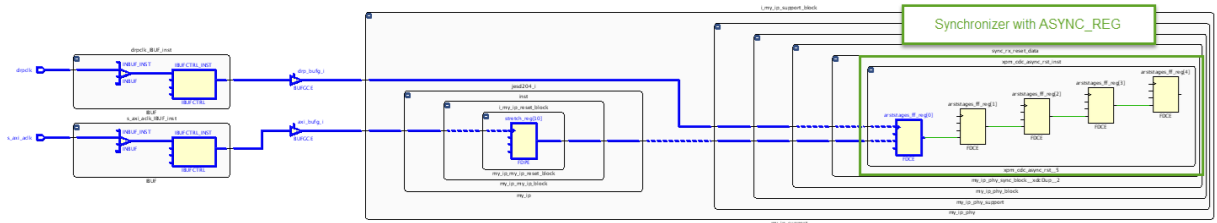
Severity	ID	Description	Depth	Exception	Source (From)	Destination (To)	Category
Critical	CDC-11	Fan-out from launch flop to destination clock	5	False Path	i_my_ip_supp...tch_reg[10]/C	i_my_ip_supp...ff_reg[0]/CLR	Unsafe
Critical	CDC-11	Fan-out from launch flop to destination clock	5	False Path	i_my_ip_supp...tch_reg[10]/C	i_my_ip_supp...ff_reg[0]/CLR	Unsafe

3. Select **Reports** → **Timing** → **Report CDC** to rerun Report CDC. In the Report CDC dialog box, make sure that Report only waived paths is unchecked, and click **OK**.
4. In the CDC Report, look at the my_ip_axi_aclk to my_ip_drpcclk CDC.

The two Critical CDC-11 violations were replaced with two Info CDC-9 violations. Based on the CDC precedence rules, waiving CDC-11 unmarks CDC-9 for this circuit.

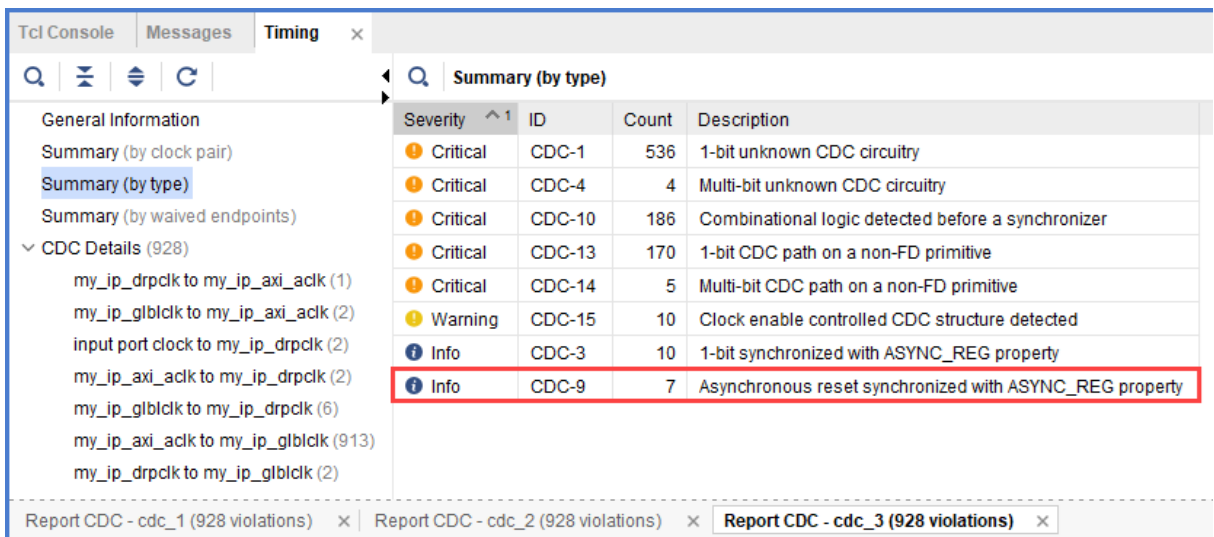


- To view a schematic of the violation, select the CDC-9 row in the CDC Report, and click the Schematic toolbar button .
- Verify that there is a 5-level synchronizer on the destination clock domain.



- Compare the new Summary (by type) information with the information from the previous CDC Report.

In the updated CDC Report, the two CDC-11 violations are no longer listed. Instead, there are two new CDC-9 violations.



- Look at the Summary (by waived endpoints) information.

In the updated CDC Report, there are three waived endpoints. This number is different from the number of waived violations (2), because CDC-11 is a multi-bit violation.

The screenshot shows the 'Timing' window in the Vivado IDE. On the left, there is a navigation pane with options: 'General Information', 'Summary (by clock pair)', 'Summary (by type)', 'Summary (by waived endpoints)' (which is highlighted), and 'CDC Details (928)'. On the right, the 'Summary (by waived endpoints)' table is displayed:

ID	Waived Endpoints
CDC-10	1
CDC-11	2

9. Generate different text reports and compare the results with previous reports.

For example, you can run the following Tcl commands:

```
report_cdc -details
report_cdc -details -waived
report_cdc -details -show_waiver
report_cdc -details -no_waiver
```

The following report was generated using the `report_cdc -details -waived` Tcl command and shows that three violations were waived.

The screenshot shows the output of the `report_cdc -details -waived` command. It displays a table of waived violations:

ID	Waived
CDC-10	1
CDC-11	2

Below this, the report details the source and destination clocks and provides a list of waived violations with their descriptions, depths, and exceptions.

Step 7: Exporting Waivers

In this step, you export waivers with the `write_waivers` Tcl command.

Note: The XDC output file can be imported using the `read_xdc` or `source` Tcl commands.

1. To export the CDC waivers, enter: `write_waivers -type cdc waivers.xdc`.



TIP: Alternatively, because there are no DRC or methodology waivers, you can enter:

`write_waivers waivers.xdc` or `write_xdc -type waiver waivers.xdc`.

2. Open the `waivers.xdc` file to view the three waivers.

Note: The following example is reformatted to better show the different command line options.

```
create_waiver -type CDC -id {CDC-10} -user "Xilinx" \
  -desc "This is a safe CDC per review with the team" \
  -from [get_pins i_my_ip_support_block/jesd204_i/inst/i_my_ip/i_tx/
i_tx_counters_32/got_sysref_r_reg/C] \
  -to [get_pins {i_my_ip_support_block/jesd204_i/inst/
sync_tx_sysref_captured/syncstages_ff_reg[0]/D}] \
  -timestamp "<timestamp>" ;#1

create_waiver -type CDC -id {CDC-11} -user "Xilinx" \
  -desc "Safe fanout. Circuitry has been released" \
  -from [get_pins {i_my_ip_support_block/jesd204_i/inst/
i_my_ip_reset_block/stretch_reg[10]/C}] \
  -to [get_pins {i_my_ip_support_block/i_jesd204_phy/inst/
jesd204_phy_block_i/sync_rx_reset_data/xpm_cdc_async_rst_inst/
arststages_ff_reg[0]/CLR}] \
  -timestamp "<timestamp>" ;#1

create_waiver -type CDC -id {CDC-11} -user "Xilinx" \
  -desc "Safe fanout. Circuitry has been released" \
  -from [get_pins {i_my_ip_support_block/jesd204_i/inst/
i_my_ip_reset_block/stretch_reg[10]/C}] \
  -to [get_pins {i_my_ip_support_block/i_jesd204_phy/inst/
jesd204_phy_block_i/sync_tx_reset_data/xpm_cdc_async_rst_inst/
arststages_ff_reg[0]/CLR}] \
  -timestamp "<timestamp>" ;#2
```

Step 8: Using the create_waiver Command

Waivers added from the Report CDC dialog box are created using the `create_waiver` command. You can view these commands as follows.

Note: You can use the `create_waiver` command line command for CDC, DRC, and methodology waivers. The options differ slightly depending on whether you are creating a CDC, DRC, or methodology waiver. For more information, including information on the different options, see the [create_waiver](#) command in the *Vivado Design Suite Tcl Command Reference Guide* (UG835).

1. Open the Vivado® journal file (`vivado.jou`) to see the three distinct `create_waiver` commands issued by the Vivado IDE.
2. Scroll through the history of the Tcl Console to see the same three `create_waiver` commands.



TIP: The `-from` and `-to` options are used to specify the startpoints and endpoints. When a waiver is set from the Report CDC dialog box, both `-from` and `-to` are specified to match the exact violation. However, you can specify a CDC waiver using only the `-from` option or only the `-to` option, but more paths might be waived than expected.

Step 9: Waiving Multiple CDC Violations

In this step, you waive multiple CDC violations simultaneously.

1. In the CDC Report, view the `my_ip_axi_aclk` to `my_ip_glblclk` CDC under CDC Details.

This crossing has five CDC-14 violations, which are multi-bit violations. The five CDC-14 violations all start from the same two register clock pins:

```
i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[2:1]/C
```



TIP: You can sort the table by the column ID to more easily see the five CDC-14 violations.

Severity	ID	Description	Depth	Exception	Source (From)	Destination (To)	Category
Warning	CDC-15	Clock enable control. C structure detected	0	False Path	i_my_ip_support..odes_reg[1]C	i_my_ip_support..ced_nls_r_regD	Safe
Critical	CDC-14	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support..des_reg[2:1]C	i_my_ip_support..NST/TXDATA[2:1]	Unknown
Critical	CDC-14	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support..des_reg[2:1]C	i_my_ip_support..NST/TXDATA[2:1]	Unknown
Critical	CDC-14	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support..des_reg[2:1]C	i_my_ip_support..NST/TXDATA[2:1]	Unknown
Critical	CDC-14	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support..des_reg[2:1]C	i_my_ip_support..NST/TXDATA[2:1]	Unknown
Critical	CDC-13	1-bit CDC path on a non-FD primitive	0	False Path	L_my_ip_support..odes_reg[2]C	L_my_ip_support..NST/TXCTRL[2]0	Unsafe
Critical	CDC-13	1-bit CDC path on a non-FD primitive	0	False Path	L_my_ip_support..odes_reg[2]C	L_my_ip_support..NST/TXCTRL[2]1	Unsafe
Critical	CDC-13	1-bit CDC path on a non-FD primitive	0	False Path	L_my_ip_support..odes_reg[2]C	L_my_ip_support..NST/TXCTRL[2]3	Unsafe
Critical	CDC-13	1-bit CDC path on a non-FD primitive	0	False Path	L_my_ip_support..odes_reg[1]C	L_my_ip_support..NST/TXDATA[0]	Unsafe

2. Because `i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[*]/C` matches five pins and you only need to target two of those five pins, construct the list of startpoints as follows:

```
set startpoints [list \
  [get_pins i_my_ip_support_block/jesd204_i/inst/
  tx_cfg_test_modes_reg[1]/C] \
  [get_pins i_my_ip_support_block/jesd204_i/inst/
  tx_cfg_test_modes_reg[2]/C] \
]
```

3. To waive the five CDC-14 violations, use the `create_waiver` Tcl command with the `-from` option:

```
create_waiver -type {CDC} -id {CDC-14} -user {Xilinx} -desc {No more CDC
14!} from $startpoints
```

4. From the Vivado IDE, select **Reports** → **Timing** → **Report CDC** to rerun Report CDC.
5. In the CDC Report, verify that the CDC-14 violations are no longer reported in the Summary section.

Severity	ID	Count	Description
Critical	CDC-1	536	1-bit unknown CDC circuitry
Critical	CDC-4	4	Multi-bit unknown CDC circuitry
Critical	CDC-10	186	Combinational logic detected before a synchronizer
Critical	CDC-13	170	1-bit CDC path on a non-FD primitive
Warning	CDC-15	10	Clock enable controlled CDC structure detected
Info	CDC-3	10	1-bit synchronized with ASYNC_REG property
Info	CDC-9	7	Asynchronous reset synchronized with ASYNC_REG property

6. To report only the waived violations, enter:

```
report_cdc -details -waived
```

The following figure shows the waived CDC violations in two different tables. The first table shows the 5 CDC-14 violations waived as multi-bit violations. The second table shows the 10 single-bit violations, calculated by multiplying the 5 multi-bit violations by 2 bits per multi-bit violation.

ID	Severity	Count	Description
CDC-10	Critical	1	Combinatorial logic detected before a synchronizer
CDC-11	Critical	2	Fan-out from launch flop to destination clock
CDC-14	Critical	5	Multi-bit CDC path on a non-FD primitive

ID	Waived
CDC-10	1
CDC-11	2
CDC-14	10

Row	ID	Severity	Description	Depth	Exception	Source (From)
1	CDC-10	Critical	Combinatorial logic detected before a synchronizer	5	False Path	i_my_ip_support_block/jesd204_i/inst/i_my_ip/tx/tx_counters_32/got

Row	ID	Severity	Description	Depth	Exception	Source (From)
1	CDC-11	Critical	Fan-out from launch flop to destination clock	5	False Path	i_my_ip_support_block/jesd204_i/inst/i_my_ip_reset_block/stretch_reg[10]/C
2	CDC-11	Critical	Fan-out from launch flop to destination clock	5	False Path	i_my_ip_support_block/jesd204_i/inst/i_my_ip_reset_block/stretch_reg[10]/C

Row	ID	Severity	Description	Depth	Exception	Source (From)	Destination
1	CDC-14	Critical	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[2:1]/C	i_my_ip_suppr
2	CDC-14	Critical	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[2:1]/C	i_my_ip_suppr
3	CDC-14	Critical	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[2:1]/C	i_my_ip_suppr
4	CDC-14	Critical	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[2:1]/C	i_my_ip_suppr
5	CDC-14	Critical	Multi-bit CDC path on a non-FD primitive	0	False Path	i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[2:1]/C	i_my_ip_suppr

7. To export all the waivers inside a script and verify that a total of four waivers were added, enter:

```
write_waivers -type cdc waivers.xdc -force
```

Note: Because the `waivers.xdc` file already exists, the `-force` option must be specified to override the file.



TIP: Alternatively, because there are no DRC or methodology waivers, you can enter:

```
write_waivers waivers.xdc -force or write_xdc -type waiver waivers.xdc
-force.
```

The list of waivers inside `waivers.xdc` appears as follows.

```
#
# WRITE CDC Waivers
# cmd: write_waivers -type cdc -file waivers.xdc -force
current_instance -quiet
create_waiver -type CDC -id {CDC-10} -user "Xilinx" -desc "This is a safe CDC per review with the team" -from [get_pins {i_my_ip_support_block/jesd204_i/inst/i_my_ip/i_tx/i_tx_counters_32/got_
create_waiver -type CDC -id {CDC-11} -user "Xilinx" -desc "Safe Fanout, Circuitry has been released" -from [get_pins {i_my_ip_support_block/jesd204_i/inst/i_my_ip_reset_block/stretch_reg[10]/C
create_waiver -type CDC -id {CDC-11} -user "Xilinx" -desc "Safe Fanout, Circuitry has been released" -from [get_pins {i_my_ip_support_block/jesd204_i/inst/i_my_ip_reset_block/stretch_reg[10]/C
create_waiver -type CDC -id {CDC-14} -user "Xilinx" -desc "No more CDC-14!" -from [list [get_pins {i_my_ip_support_block/jesd204_i/inst/tx_cfg_test_modes_reg[1]/C}] [get_pins {i_my_ip_support_
#
current_instance -quiet
```

8. To import the `waivers.xdc` file, enter:

```
read_xdc waivers.xdc
```

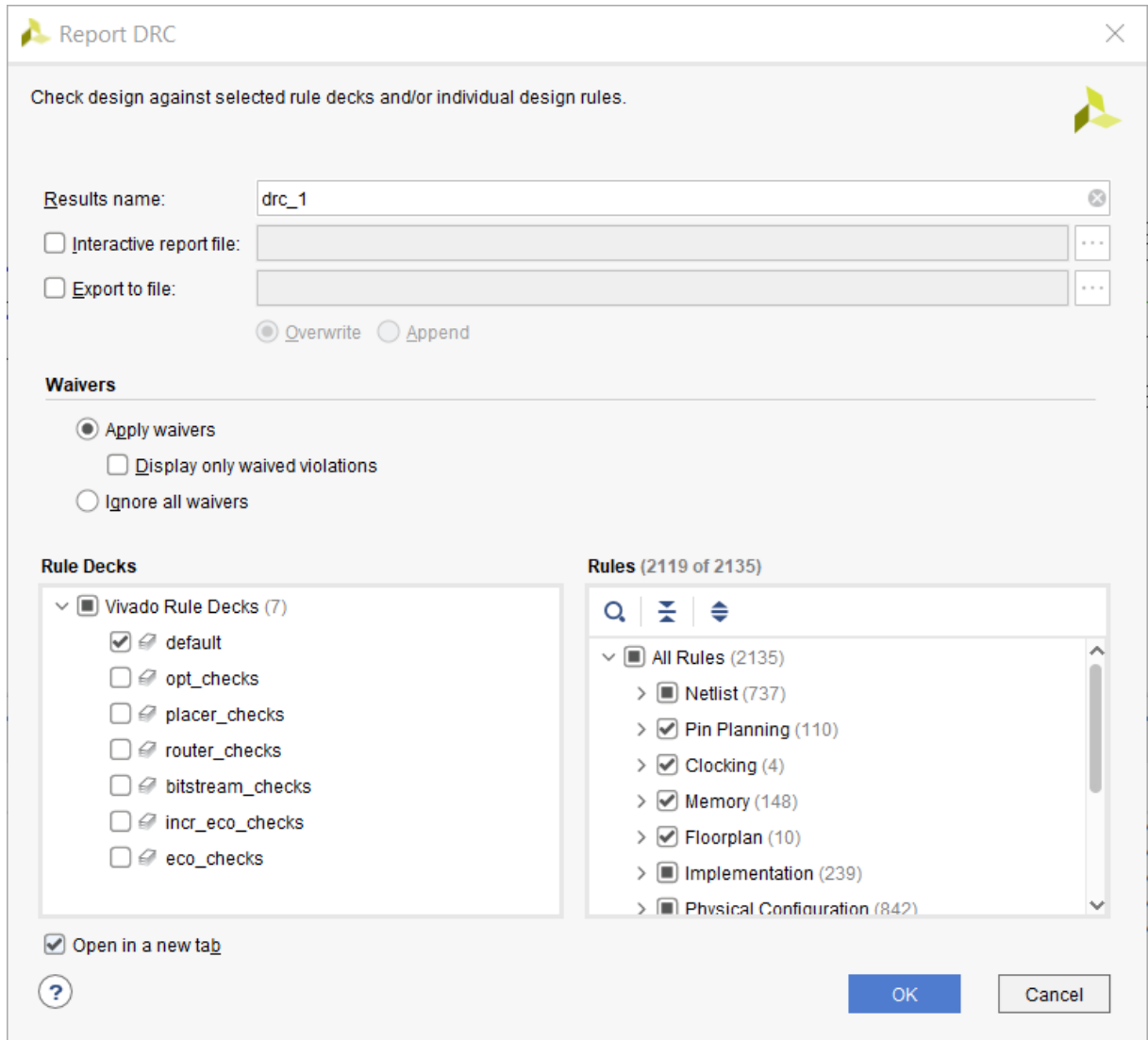
The following warnings show that duplicate waivers were not added to the existing waivers. Only waivers that are exact duplicates of existing waivers are rejected.

```
WARNING: [Vivado_Tcl 4-935] Waiver ID 'CDC-10' is a duplicate and will
not be added again.
WARNING: [Vivado_Tcl 4-935] Waiver ID 'CDC-11' is a duplicate and will
not be added again.
WARNING: [Vivado_Tcl 4-935] Waiver ID 'CDC-11' is a duplicate and will
not be added again.
WARNING: [Vivado_Tcl 4-935] Waiver ID 'CDC-14' is a duplicate and will
not be added again.
```

Step 10: Waiving Multiple DRC Violations

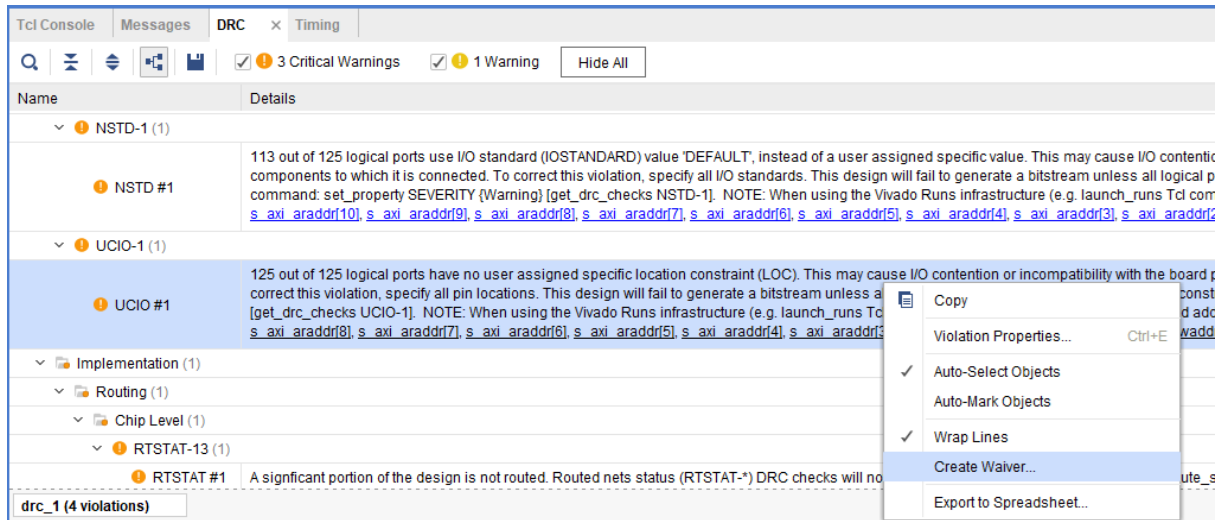
In this step, you waive multiple DRC violations simultaneously.

1. Select **Reports** → **Report DRC**.
2. In the Report DRC dialog box, leave all settings at their default, and click **OK**.

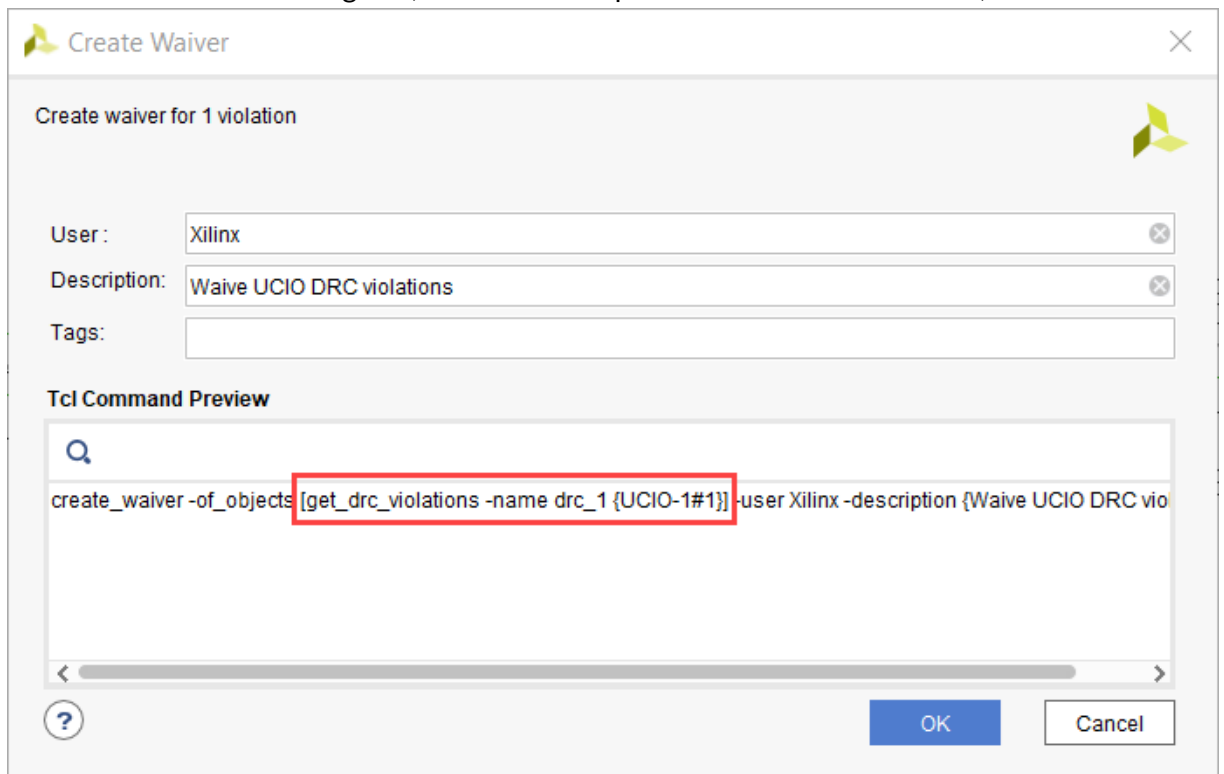


- In the DRC Report, right-click **UCIO#1**, and select **Create Waiver** to create a waiver for the UCIO-1 violations.

Note: The UCIO#1 violation combines 125 individual violations into a single violation. Similarly, the NSTD#1 violation covers 113 ports.



4. In the Create Waiver dialog box, look at the output in Tcl Command Preview, and click **OK**.



5. To generate the `drc_waivers.xdc` file and verify that the waiver is waiving all 125 objects, enter:

```
write_waivers -type DRC drc_waivers.xdc
```

6. In the XDC file, look at the expanded port list, and notice that some of the strings from the violations message were converted to wildcards (*).

Strings are automatically converted to wildcards for UCIO-1, NSTD-1, TIMING-15, and TIMING-16 type violations. For UCIO-1, the numbers of objects in the violations are replaced with wildcards, because the numbers of elements are not meaningful.

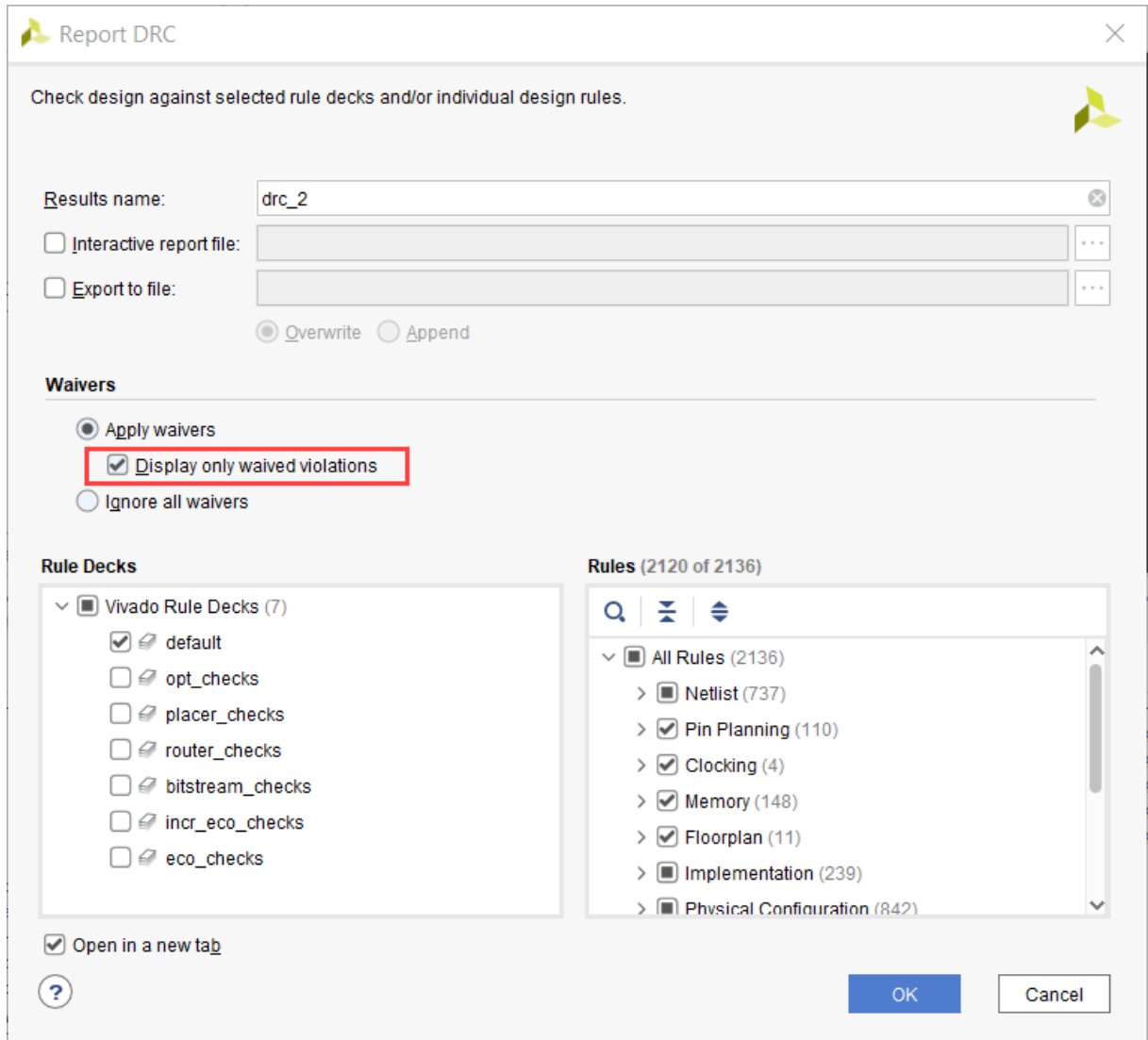
```
#
# WRITE DRC WAIVERS
# cmd: write_waivers -type DRC drc_waivers.xdc
current_instance -quiet
create_waiver -type DRC -id {UCIO-1} -user "Xilinx" -desc "Waive UCIO DRC violations" -objects [get_ports { refclk0p gblclkp refclk0n tx_start_of_frame[3] tx_start_of_multiframe[3] gblclkn
tx_reset drpclk tx_start_of_multiframe[2] txp[3] tx_start_of_multiframe[0] tx_start_of_frame[2] tx_start_of_frame[1] s_axi_rready tx_sync tx_sysref tx_aresetn s_axi_rdata[1] s_axi_rvalid
s_axi_rresp[0] s_axi_rresp[1] s_axi_rdata[0] s_axi_rdata[2] s_axi_rdata[3] s_axi_rdata[4] s_axi_rdata[9] s_axi_rdata[5] s_axi_rdata[10] s_axi_rdata[6] s_axi_rdata[7] s_axi_rdata[8]
s_axi_rdata[15] s_axi_rdata[11] s_axi_rdata[12] s_axi_rdata[13] s_axi_rdata[16] s_axi_rdata[17] s_axi_rdata[18] s_axi_rdata[14] s_axi_rdata[21] s_axi_rdata[22] s_axi_rdata[27]
s_axi_rdata[23] s_axi_rdata[19] s_axi_rdata[25] s_axi_rdata[26] s_axi_rready s_axi_rdata[28] s_axi_rdata[24] s_axi_rdata[30] s_axi_rdata[31] s_axi_araddr[12] s_axi_arvalid s_axi_rdata[29]
s_axi_araddr[7] s_axi_araddr[3] s_axi_araddr[4] s_axi_araddr[5] s_axi_araddr[6] s_axi_bvalid s_axi_araddr[8] s_axi_araddr[10] s_axi_bready s_axi_wstrb[0] s_axi_wstrb[1] s_axi_araddr[8]
s_axi_bresp[1] s_axi_wready s_axi_wvalid s_axi_wdata[0] s_axi_bresp[0] s_axi_wstrb[2] s_axi_araddr[11] s_axi_wdata[4] s_axi_wdata[1] s_axi_wstrb[3] s_axi_wdata[2] s_axi_wdata[3]
s_axi_wdata[9] s_axi_wdata[5] s_axi_wdata[6] s_axi_wdata[7] s_axi_wdata[8] s_axi_wdata[14] s_axi_wdata[10] s_axi_wdata[11] s_axi_wdata[12] s_axi_wdata[13] s_axi_wdata[19] s_axi_wdata[15]
s_axi_wdata[16] s_axi_wdata[17] s_axi_wdata[18] s_axi_wdata[24] s_axi_wdata[20] s_axi_wdata[28] s_axi_wdata[26] s_axi_wdata[27] s_axi_wdata[26] s_axi_wdata[26] s_axi_wdata[26] s_axi_wdata[26]
tx_start_of_multiframe[1] txp[1] tx_start_of_frame[0] txp[2] txn[1] txn[3] txn[2] s_axi_awaddr[11] txn[0] txp[0] s_axi_bready s_axi_bresp[0] s_axi_bresp[1] s_axi_bresp[2] s_axi_bresp[3]
s_axi_awaddr[9] s_axi_awaddr[2] txn[4] s_axi_awaddr[5] s_axi_awaddr[6] s_axi_awaddr[3] s_axi_awaddr[4] ] -strings { "*" } -strings { "*" } -timestamp "Wed Mar 14 22:57:14 GMT 2018" ;#1
```

7. To delete the DRC waiver and rewrite the waiver using wildcards to target a subset of the ports objects, enter:

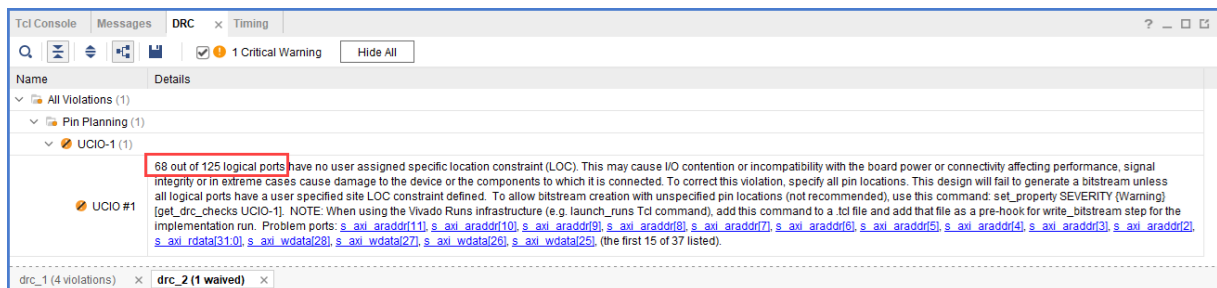
```
delete_waivers [get_waivers -type drc]
create_waiver -type DRC -id {UCIO-1} -user "Xilinx" -desc "Waive
selected UCIO violations" -objects [get_ports { s_axi_rdata[*]
s_axi_wdata[*] s_axi_araddr[*] } ] -strings { "*" } -strings { "*" }
```

Note: This command only covers a subset of the original 125 objects.

8. Select **Reports** → **Report DRC** to rerun Report DRC.
9. In the Report DRC dialog box, select **Display only waived violations** to report only waived violations, and click **OK**.



In the DRC Report, verify that only 68 violations are waived out of 125.



IMPORTANT! You cannot waive *READONLY* or *NODISABLE* violations. For example, if you enter:

```
create_waiver -type DRC -id RTSTAT-1 -description "Waive RTSTAT-1"
```

The Vivado tools issue the following error:

```
ERROR: [Vivado_Tcl 4-934] Waiver ID 'RTSTAT-1' is READONLY or NODISABLE
and cannot be waived.
These Factory designations specify that a check is required and may not be
overridden by user action.
```

Step 11: Generating a Summary Report for Waived Violations

This step covers how to use the `report_waivers` Tcl command to generate a summary report for CDC, DRC, and methodology waivers.



IMPORTANT! Before running the `report_waivers` command, you must rerun Report CDC, Report DRC, or Report Methodology to ensure that added or removed waivers are included in the statistics reported by `report_waivers`.

1. To rerun Report CDC, enter:

```
report_cdc
```

2. To rerun Report DRC, enter:

```
report_drc
```

Note: You do not need to rerun Report Methodology, because no methodology waivers were set.

3. To create a summary report, enter:

```
report_waivers
```

By default, `report_waivers` reports only waived violations. The following figure shows the UCIO-1, CDC-10, CDC-11, and CDC-14 rules, which have defined waivers.

```

-----
Table Of Contents
1. REPORT SUMMARY
2. REPORT DETAILS (DRC)
3. REPORT DETAILS (METHODOLOGY: no waivers)
4. REPORT DETAILS (CDC)
-----

1. REPORT SUMMARY
-----
Waiver Type  Total Vios  Remaining Vios  Waived Vios  Used Waivers  Set Waivers
-----
DRC          239        171            68           1             1
METHODOLOGY  0          0              0            0             0
CDC          957        944            13           4             4
Note: This report is based on the most recent report_drc/report_methodology/report_cdc runs.
-----

2. REPORT DETAILS (DRC)
-----
Rule  Severity  Description  Total Vios  Remaining Vios  Waived Vios  Used Waivers  Set Waivers
-----
UCIO-1*  Critical Warning  Unconstrained Logical Port  125  57  68  1  1
-----

4. REPORT DETAILS (CDC)
-----
Rule  Severity  Description  Total Vios  Remaining Vios  Waived Vios  Used Waivers  Set Waivers
-----
CDC-10  Critical  Combinational logic detected before a synchronizer  187  186  1  1  1
CDC-11  Critical  Fan-out from launch flop to destination clock  2  0  2  2  2
CDC-14  Critical  Multi-bit CDC path on a non-FD primitive  10  0  10  1  1
Note: Any 'Rule' which is flagged by '*' is an aggregating message and its counts are based on the number of objects represented,
rather than the number of messages.
    
```

Note the number of waived objects and total violations:

- The aggregating DRCs are reported as 1 violation per object inside the violation. Because there are 113 objects in NSTD-1, 125 objects in UCIO-1 plus 1 in RTDAT-13, a total number of 239 DRC violations are reported in the Summary table.
 - The Report Summary table reports all of the violations.
 - The Report Details tables only report the check IDs that have one or more waivers.
4. To generate detailed tables with all of the rules, including rules with no waivers, enter:

```
report_waivers -show_msgs_with_no_waivers
```

The following figure shows the report with all DRC and CDC rules reported in the Report Details.

Table Of Contents

- REPORT SUMMARY
- REPORT DETAILS (DRC)
- REPORT DETAILS (METHODOLOGY: no waivers)
- REPORT DETAILS (CDC)

1. REPORT SUMMARY

Waiver Type	Total Vios	Remaining Vios	Maived Vios	Used Waivers	Set Waivers
DRC	239	171	68	1	1
METHODOLOGY	0	0	0	0	0
CDC	957	944	13	4	4

Note: This report is based on the most recent report_drc/report_methodology/report_cdc runs.

2. REPORT DETAILS (DRC)

Rule	Severity	Description	Total Vios	Remaining Vios	Maived Vios	Used Waivers	Set Waivers
UCIO-1*	Critical Warning	Unconstrained Logical Port	125	57	68	1	1
NSTD-1*	Critical Warning	Unspecified I/O Standard	113	113	0	0	0
RTSTAT-13	Critical Warning	Insufficient Routing	1	1	0	0	0

4. REPORT DETAILS (CDC)

Rule	Severity	Description	Total Vios	Remaining Vios	Maived Vios	Used Waivers	Set Waivers
CDC-10	Critical	Combinational logic detected before a synchronizer	187	186	1	1	1
CDC-11	Critical	Fan-out from launch flop to destination clock	2	0	2	2	2
CDC-14	Critical	Multi-bit CDC path on a non-FD primitive	10	0	10	1	1
CDC-1	Critical	1-bit unknown CDC circuitry	536	536	0	0	0
CDC-3	Info	1-bit synchronized with ASYNC_REG property	9	9	0	0	0
CDC-4	Critical	Multi-bit unknown CDC circuitry	28	28	0	0	0
CDC-9	Info	Asynchronous reset synchronized with ASYNC_REG property	5	5	0	0	0
CDC-13	Critical	1-bit CDC path on a non-FD primitive	170	170	0	0	0
CDC-15	Warning	Clock enable controlled CDC structure detected	10	10	0	0	0

Note: Any 'Rule' which is flagged by '*' is an aggregating message and its counts are based on the number of objects represented, rather than the number of messages.

5. To run Report Methodology, enter:

```
report_methodology
```

6. To generate detailed tables with all of the rules, including rules with no waivers, enter:

```
report_waivers -show_msgs_with_no_waivers
```

The exact statistics are reported, as shown in the following figure.

Note: This figure does not include the Report Details (CDC) section.

1. REPORT SUMMARY

Waiver Type	Total Vios	Remaining Vios	Maived Vios	Used Waivers	Set Waivers
DRC	239	171	68	1	1
METHODOLOGY	125	125	0	0	0
CDC	957	944	13	4	4

Note: This report is based on the most recent report_drc/report_methodology/report_cdc runs.

2. REPORT DETAILS (DRC)

Rule	Severity	Description	Total Vios	Remaining Vios	Maived Vios	Used Waivers	Set Waivers
UCIO-1*	Critical Warning	Unconstrained Logical Port	125	57	68	1	1
NSTD-1*	Critical Warning	Unspecified I/O Standard	113	113	0	0	0
RTSTAT-13	Critical Warning	Insufficient Routing	1	1	0	0	0

3. REPORT DETAILS (METHODOLOGY)

Rule	Severity	Description	Total Vios	Remaining Vios	Maived Vios	Used Waivers	Set Waivers
LUTAR-1	Warning	LUT drives async reset alert	8	8	0	0	0
TIMING-9	Warning	Unknown CDC Logic	1	1	0	0	0
TIMING-10	Warning	Missing property on synchronizer	1	1	0	0	0
TIMING-18*	Warning	Missing input or output delay	115	115	0	0	0

Step 12: Using Waiver Commands

In this step, you run additional commands related to the waivers.

1. To return a collection of CDC waiver objects, enter:

```
get_waivers -type cdc
```

The following CDC waivers are returned:

```
CDC-10#1 CDC-11#1 CDC-11#2 CDC-14#1
```

2. To filter the list of waivers to only return CDC-14 waivers, enter:

```
get_waivers -filter {ID == CDC-14}
CDC-14#1
```

3. To report all of the properties on a CDC waiver object, enter:

```
report_property [lindex [get_waivers -type cdc] end]
```

The following properties are returned:

Property	Type	Read-only	Value
CLASS	string	true	cdc_waiver
DESCRIPTION	string	false	No more CDC-14!
ID	string	true	CDC-14
INDEX	string	true	1
NAME	string	true	CDC-14#1
OBJECT_COUNTS	string	true	pins:2
SCOPE	string	true	
TAGS	string	false	
TIME	string	true	<timestamp>
TYPE	string	true	CDC
USED_CNT	string	true	10
USER	string	true	Xilinx

Note: You cannot retrieve the design objects attached to a waiver object.

4. To delete all of the previously created CDC-14 waivers, enter:

```
delete_waivers [get_waivers -filter {ID == CDC-14}]
```

Note: After a waiver object is deleted, the waiver no longer applies and the violations that it waived are reported again.

5. To delete all of the remaining CDC waivers, enter:

```
delete_waivers [get_waivers -type cdc]
```

Summary

In this lab, you accomplished the following:

- Waived CDC and DRC violations
- Generated reports for waived violations
- Exported waivers
- Used waiver commands

Lab 2: Using Report QoR Suggestions

Introduction



IMPORTANT! Windows users must install the patch located at: www.xilinx.com/support/answers/72633.html in order to run this lab.

The `report_qor_suggestions` (RQS) command enables the Vivado® Design Suite tools to analyze a design and provide automated solutions for enhancing QoR. The command can be run on an open design after synthesis or after any stage in the implementation flow. RQS evaluates the design in five key areas and suggests fixes or improvements in these areas. The five areas are utilization, clocking, constraints, congestion, and timing. Recommendations from RQS can take the following forms:

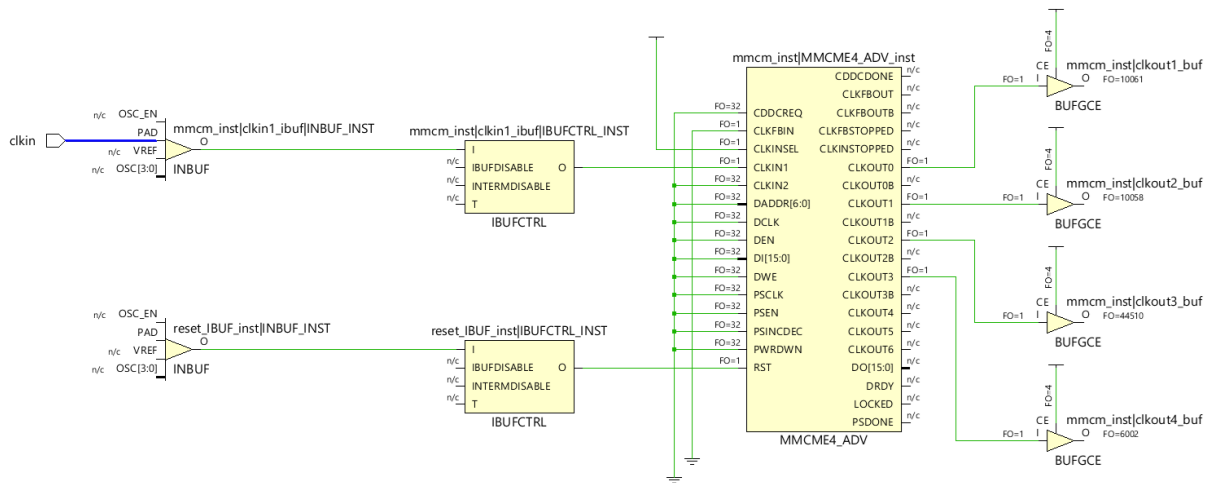
- RQS objects. These can add:
 - Switches to a given command
 - Properties to a given design object
- Text recommendations that require intervention by the user.

This tutorial will cover how to work with the RQS objects contained within RQS files in a project based environment. Non project flow steps are also covered but not explicitly run.

Step 1: Understanding the Design

This lab uses a pre-built design to demonstrate some of the features of RQS. Suggestions are triggered by the design of the RTL and the placement of blocks using floorplanning. The pre-built design contains the following modules:

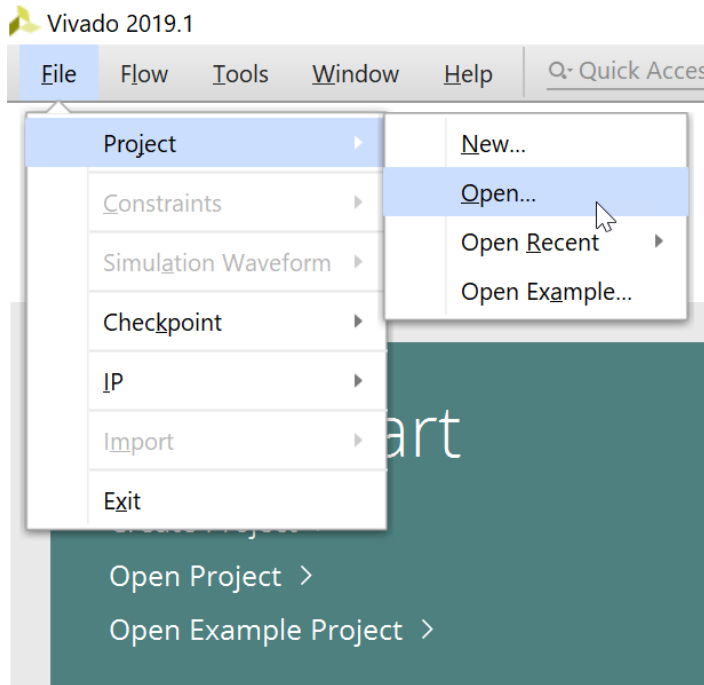
- **Clocking Module:** The main clocking circuit for the design resides in `clocking_module.vhd`. For simplicity, RST is tied to GND and LOCKED is left disconnected. The structure of this block is shown in the following figure.



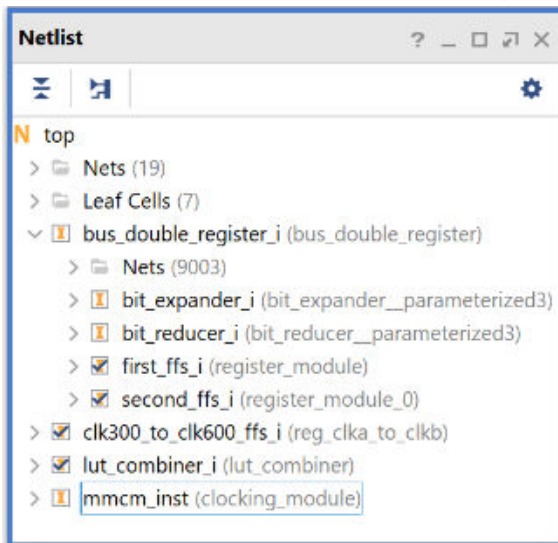
- Reg CLKA to CLKB Module:** This module contains a synchronous CDC for a large bus. It registers input data using CLKA and then passes it to a register on the CLKB domain to be passed to the output. Registering large buses on different related clock domains can impact hold slack (WHS/THS) and setup slack (WNS/TNS).
- Bus Double Register Module:** This module instantiates two sub-level modules that are connected by a wide data bus. The modules can be placed apart from each other using floorplanning. This has the effect of adding routing to the area between the two modules. The desired outcome is to create enough routing congestion to trigger suggestions.
- LUT Combiner Module:** This module registers the input data three times. A circuit then takes the XOR of these bits and registers the output. Because there are three common inputs, it will create LUT combining, which can cause congestion.
- Bit Expander and Bit Reducer Modules:** These modules enable the expansion and contraction of internal data widths so that the design does not run out of I/Os. The modules take an arbitrary data width and expand or contract it to or from a desired size. The expansion and contraction logic creates many logic levels and should be untimed. When untimed, they are ignored by `report_qor_suggestions`.

The following steps cover opening the project and examining the placement of the floorplanned modules:

- In the Vivado® Design Suite, go to **File** → **Project** → **Open** and select the project located in `<extract_Dir>/lab2/1_InitialRun`.

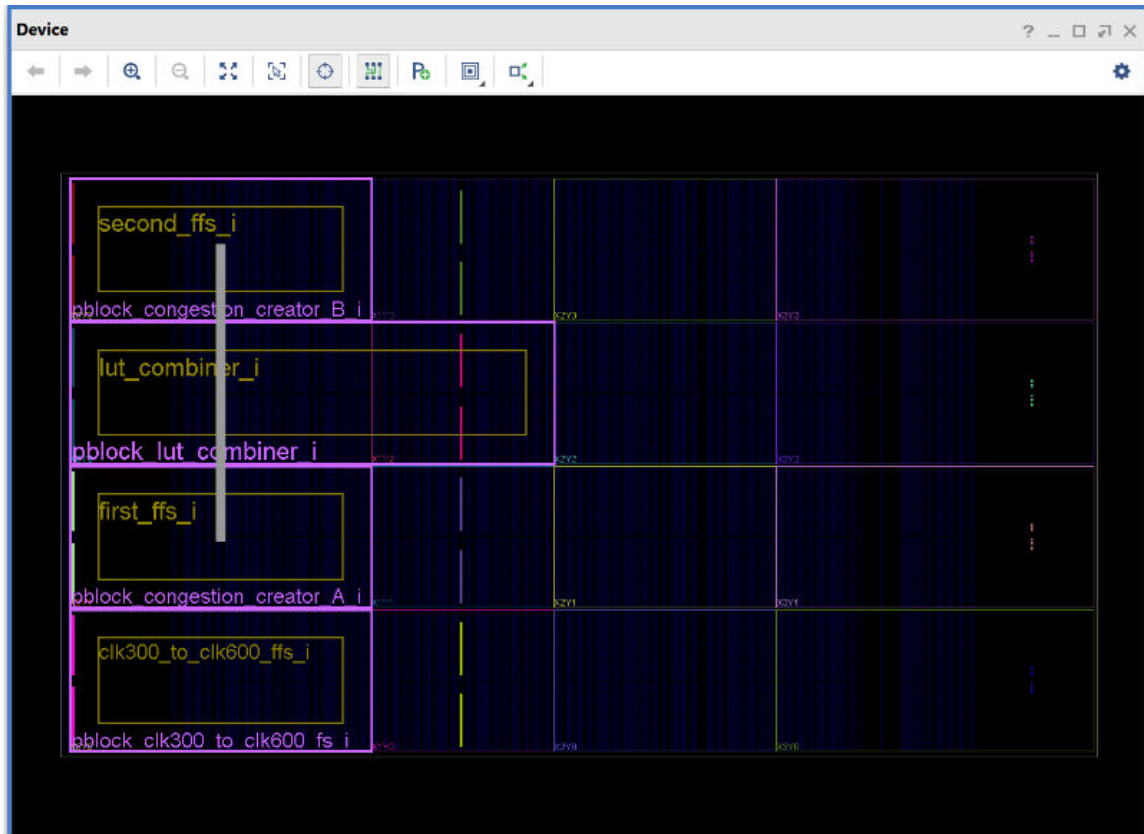


2. In the Flow Navigator, click **Open Synthesized Design**.
3. In the Netlist view, look at the hierarchy.



All modules at the top level are independent of each other. There are no timing paths between them. There should also be no timing paths between these modules and the I/Os.

4. In Device view, look at the four pblocks. These have been added to help trigger suggestions on the design without requiring a highly utilized design.



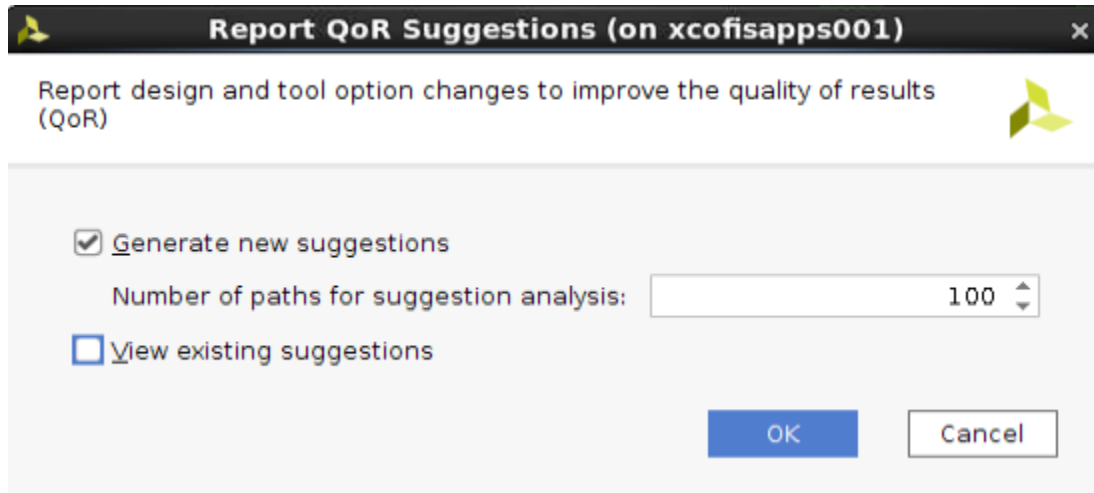
TIP: When a block or blocks are selected, you can investigate the design further by pressing F4 to open the schematic tools.

As can be seen in the previous figure, the LUT-combiner module is placed in between the Bus Double Register sub-modules. This placement forces nets across the LUT-combined module to create routing congestion; the effect mimics a real-world design, but is rigid. RQS is tuned to real-world designs. The placement of the Reg CLKA to CLKB module is contained in the `clk300_to_clk600_ffs_i` pblock.

Step 2: Running Report QoR Suggestions

This step covers running the `report_qor_suggestions` command to generate a report. The command can be run on an open design at any stage of the implementation flow after synthesis. In project mode, this is typically after synthesis or implementation. In non-project mode, this can be after `synth_design`, `link_design`, `opt_design`, `place_design`, `phys_opt_design`, or `route_design`.

1. In the Vivado® IDE, from the pull down menus, click **Reports** → **Report QoR Suggestions...** to bring up the dialog box shown in the following figure:



The equivalent Tcl command is:

```
report_qor_suggestions -quiet -name qor_suggestions_1
```

The command will:

- Examine the design and generate new suggestions
- Generate a report on the suggestions

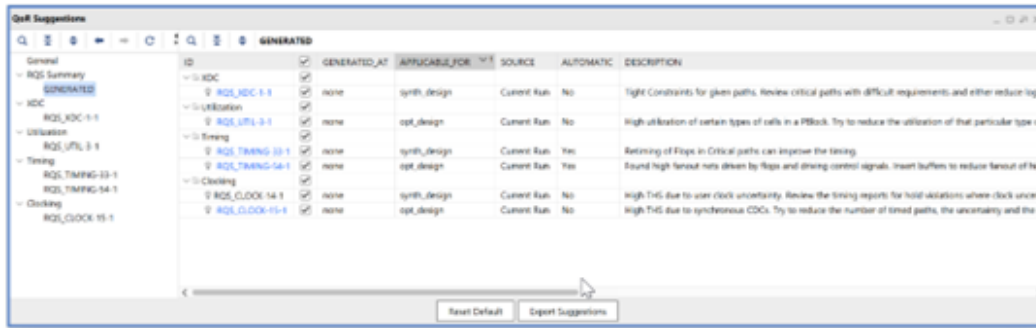
The report opens automatically in the integrated design environment (IDE). Due to the interactive nature of the report, only one instance of the report can be open at any time.

Note: By default, the RQS command reports on the 100 worst failing paths per clock group. You can change the number of paths that RQS uses for the analysis of timing-critical paths by modifying the `-max_paths` switch. Increasing this number generates more suggestions, but on paths that are reducing in criticality. This may help in the later stages of design closure once all the key items are resolved.

Step 3: Understanding the Report

This step explains the different sections of the generated QoR Suggestions report. On the left of the report window, you can navigate to the different sections of the report; on the right, more information is provided.

1. In the generated report, under RQS Summary, select **GENERATED**. This brings up the report section shown in the following figure:



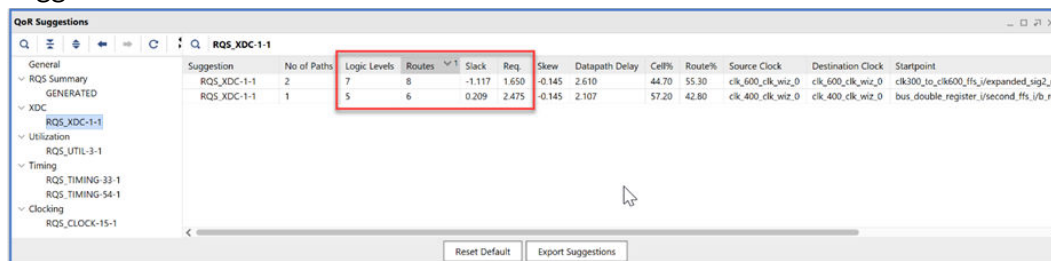
The GENERATED section provides a list of all the suggestions that have just been generated at this stage of the current run. Each suggestion has a description that details the reason for the suggestion. Additionally, for each suggestion the following information is provided:

Table 1: RQS Summary Column Description

Item	Description	Comment
GENERATED_AT	This shows what stage of the design the suggestion was generated at. Typical values <code>place_design</code> or <code>route_design</code>	As we progress through the design stages, the decisions that the tool makes are based on the information available at the time. Additionally, information accuracy increases after placement and again after routing. Some early suggestions may not be required once you have run through the flow. Some early suggestions may be required to solve issues later in the flow.
APPLICABLE_FOR	This shows what stage must be rerun in order for the suggestion to take effect.	Most suggestions are executed at either <code>synth_design</code> or <code>opt_design</code>
SOURCE	Details where the suggestion was generated	<code>current_run</code> or if from a previous run, a file name that contained the suggestion.
AUTOMATIC	Details whether the suggestion is executed automatically or the user must manually intervene	Automatic suggestions will either recommend a switch to the tool or a property to be added to a cell or net

Under the other sections of the report there are details about the individual suggestions that have been generated.

- Click on the `RQS_XDC_1_1` hyperlink. This will take you to the details section for this suggestion:




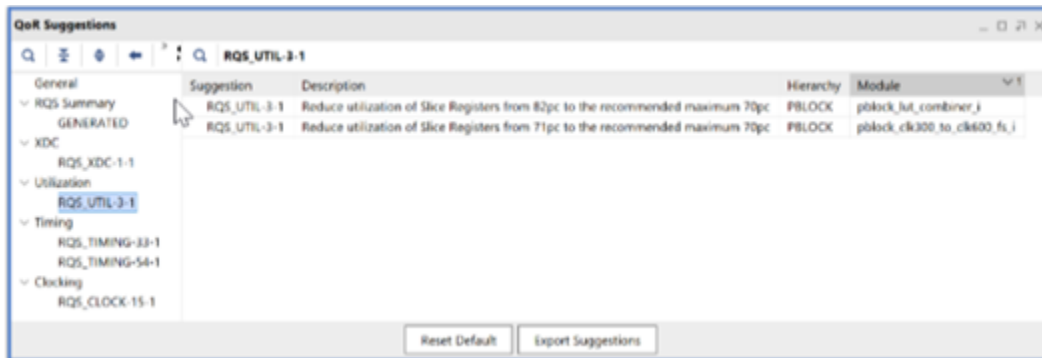
The suggestion description says that the timing constraint is too tight for the given path(s). The XDC section contains two detailed paths for the this suggestion. Some suggestions can target more than one object in a design.

The first path has a large negative slack which would stand out in a timing report. Timing paths use net delays that are optimal, this gives the tools the correct order to place and route them. Close analysis shows this is a 600 MHz path with seven logic levels. This is a path that will need to be fixed. It is not possible to fix every path automatically. For this tutorial, assume that a false path constraint has been missed. Enter the following in the Tcl console to add this:

```
set_false_path -to [get_cells clk300_to_clk600_ffs_i/bit_reducer_i/
tmp_reg]
```

The second path does not fail timing but is still flagged. This check also looks at paths by giving approximate net and LUT budgets. This is done because sometimes the optimal route is not always available and the tools can overestimate timing slack. This second check is less critical but it is still worth examining early in the design cycle.

3. Click on the back arrow button  to go back to the GENERATED view.
4. In the GENERATED view, click on **RQS_UTIL-3-1**. These suggestions examine utilization of different primitive types within a pblock.



RQS examines the utilization of the full design, pblocks and SLRs. In addition, it also looks at control sets and compares design numbers against thresholds from a characterized model that may move thresholds depending on the design.

For this case, there is high register usage in both pblocks. RQS provides a general text recommendation to reduce the utilization of this primitive type. Sometimes it will provide automatic suggestions too. It is also worth noting that `opt_design` has not been run yet. This could further reduce utilization but this is not certain until you have run the command. RQS models do not change between `synth_design` and `opt_design`. Another way to resolve this is to increase the size of the pblocks.

5. Click **Window** → **Physical Constraints**. In the Physical Constraints window, select the `clk300_to_clk600_ffs_i` pblock and view the Statistics tab in the Pblock Properties window. This will display the utilization of the pblock. You can see that the CLB utilization is 71.85%.

Site Type	Available	Used	% Util
CLB LUTs	13920	2675	19.22
LUT as Logic	13920	2675	19.22
LUT as Memory	5760	0	0.00
CLB Registers	27840	20004	71.85
Register as Flip Flop	27840	20004	71.85

- To resize the pblock, add the following command to the Tcl console:

```
resize_pblock pblock_clk300_to_clk600_fs_i \
-add {SLICE_X0Y0:SLICE_X34Y59} -remove
{CLOCKREGION_X0Y0:CLOCKREGION_X0Y0} \
-locs keep_all
```

The updated pblock utilization is now 59.54%. This is lower than the threshold. Leave the other pblock as-is. For this particular case, the constraints on pblocks are being used to generate routing congestion. In typical designs, pblock utilization at 82% would cause a failure.

- Click on the back arrow button to go back to the GENERATED view.
- Examine the TIMING-54-1 suggestion. The suggestion is recommending to insert a BUFG to lower general routing usage. It is an automatic suggestion. It has APPLICABLE_FOR == opt_design so there is no need to restart the run as the current stage is after synth_design.

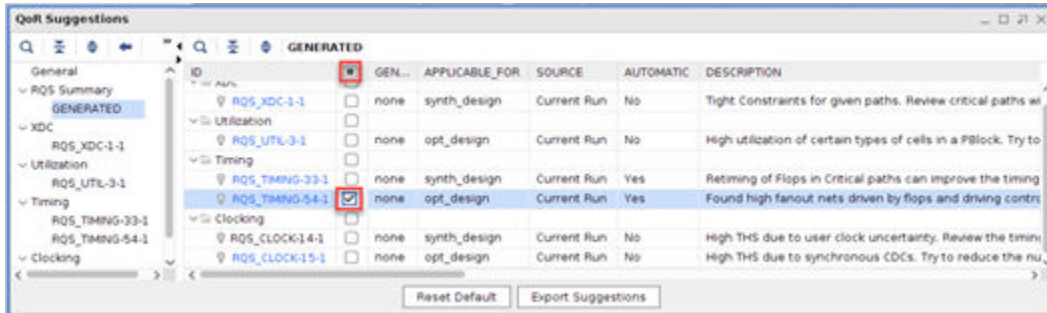
General	GEN...	APPLICABLE_FOR	SOURCE	AUTOMATIC	DESCRIPTION
RQS_SUMMARY					
GENERATED					
Utilization					
RQS_UTIL-3-1		opt_design	Current Run	No	High utilization of certain types of cells in a PBlock. Try to
Timing					
RQS_TIMING-33-1		synth_design	Current Run	Yes	Retiming of Flops in Critical paths can improve the timing
RQS_TIMING-54-1		opt_design	Current Run	Yes	Found high fanout nets driven by flops and driving contr
Clocking					
RQS_CLOCK-14-1		synth_design	Current Run	No	High THS due to user clock uncertainty. Review the timin
RQS_CLOCK-15-1		opt_design	Current Run	No	High THS due to synchronous CDCs. Try to reduce the nu

Clicking through to the details sections for this suggestion shows that the signal has a fanout of 10000 as shown in the following figure:

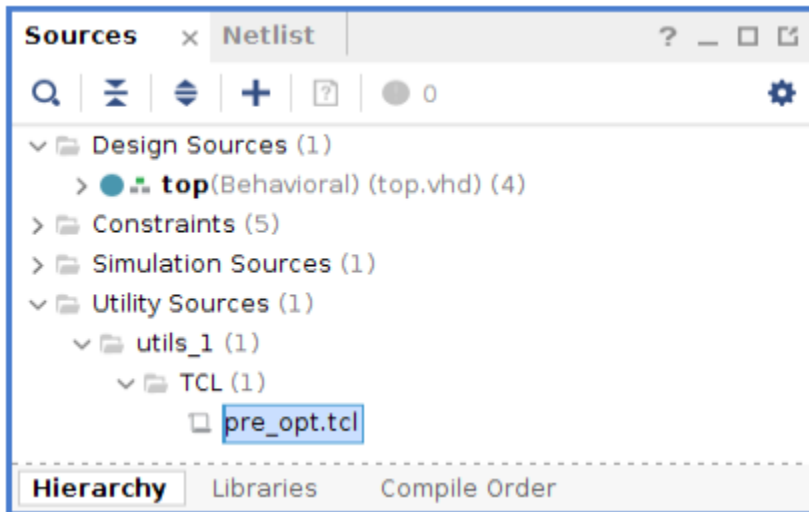
Suggestion	No of Paths	Logic Levels	Routes	Slack	Req.	Skew	Fanout	Datapath Delay	Cells	Route%	Source
RQS_TIMING-54-1	1	0	1	2.499	3.300	-0.145	10000	0.521	25.70	74.30	clk_30

- Click on the back arrow button to go back to the GENERATED view.

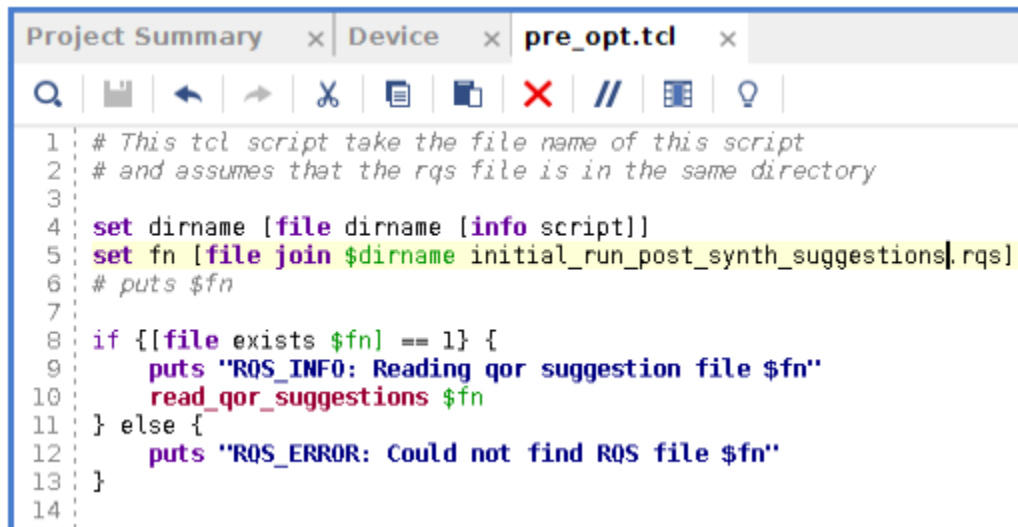
10. Select the automated suggestions to be exported to the run. First deselect the checkbox in the title heading of the column to deselect all the entries in the column, then select the checkbox associated with RQS_TIMING-54-1. The selections are shown in the following figure:



11. Click the **Export Suggestions** button and select the location as `<extract_Dir>/lab2/initial_run_post_synth_suggestions.rqs`
12. In order for the suggestions to be added to the run, the `read_qor_suggestions` command must be executed. In project mode, this must be done using a TCL hook, before the `opt_design` stage. The `pre_opt.tcl` file has been created and added to the project for you to do this. In the sources window, click **Utility Sources** → **utils_1** → **TCL** to open the `pre_opt.tcl` file.



The file contains the following contents:



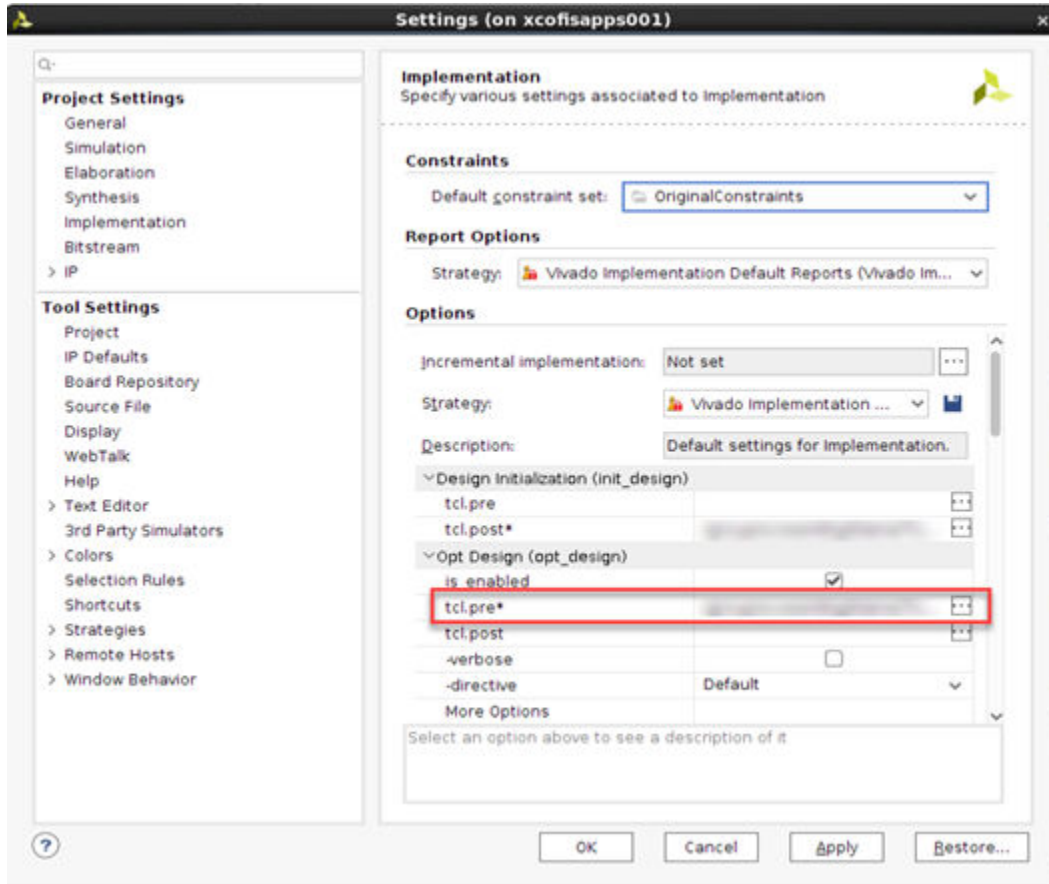
```

1  # This tcl script take the file name of this script
2  # and assumes that the rqs file is in the same directory
3
4  set dirname [file dirname [info script]]
5  set fn [file join $dirname initial_run_post_synth_suggestions.rqs]
6  # puts $fn
7
8  if {[file exists $fn] == 1} {
9      puts "RQS_INFO: Reading qor suggestion file $fn"
10     read_qor_suggestions $fn
11 } else {
12     puts "RQS_ERROR: Could not find RQS file $fn"
13 }
14
    
```

For this TCL file to work correctly the following must be true:

- a. The TCL file must be specified as a TCL hook at the pre-opt_design stage for the implementation run
- b. The RQS file must reside in the same directory as this TCL script
- c. When specifying this as an implementation run tcl hook, the file should not be copied into the project as the RQS file will not be automatically copied and it will break the required relationship mentioned in b

Next ensure this file is setup in implementation runs option. Under the Flow Navigator Select **Settings** → **Implementation** → **opt_design**.



Note: The pblock resizing and false path constraints are added inside the `post_link.tcl` script. This should normally get added to the user constraints, but to prevent the run from being out of date these have been added via a TCL script.

Step 4: Run with Suggestions

You will now examine what happens when a suggestion is applied and how it is reported. Then you will add further suggestions to the RQS file.

1. In the Flow Navigator, click **Open Implemented Design**.
2. While the run is opening, take the opportunity to examine the log file for the implementation run. In the Reports tab, select `impl_1_route_implementation_log_0` to open up the implementation log file.

Locate the following in the log file (including `<extract_dir>` explicitly and not replacing the string):

```
source <extract_dir>/lab2/pre_opt.tcl
```

Following the sourcing of the TCL script there are two messages immediately after:

```
RQS_INFO: Reading qor suggestion file <extract_dir>/lab2/
initial_run_post_synth_suggestions.rqs
INFO: [Vivado_Tcl 4-1103] Successfully read QoR suggestions file:
<extract_dir>/initial_run_post_synth_suggestions.rqs
```

The RQS_INFO message comes from the TCL file. It is worth adding some messages in the TCL file for debugging filepath issues due to the complexities of runs being launched in directories that are not where the project sources reside. This is not required when running in a non-project flow.

The INFO: [Vivado_Tcl 4-1103] message is generated by `read_qor_suggestions` to show that the file has been read.

Because the RQS file is binary, it cannot be read in a text editor. Therefore, after reading the RQS file, we have added the following command to the `pre_opt.tcl` script to show what suggestions have been read:

```
report_qor_suggestions -of_objects [get_qor_suggestions]
```

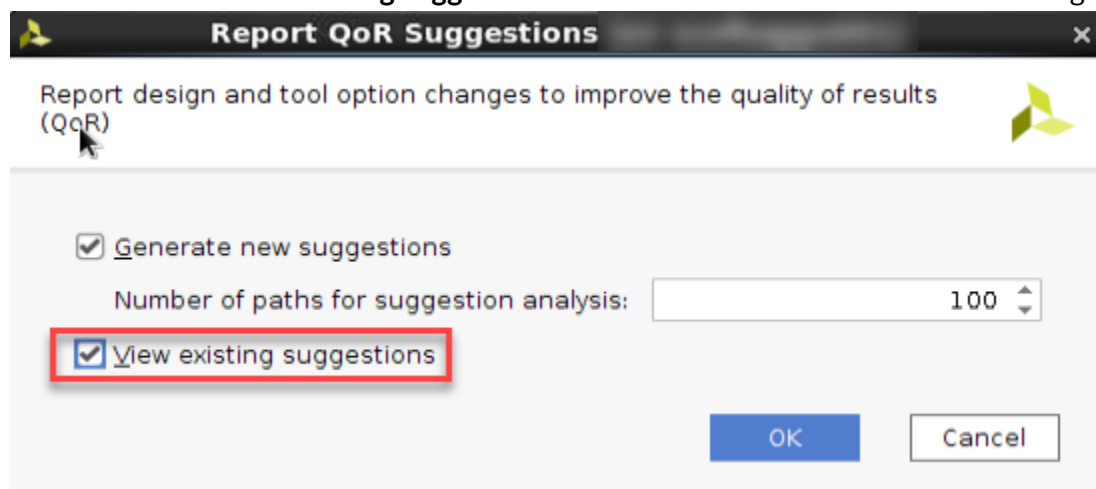
The command will report all the suggestions that are in memory. As you are running this before any calls to `report_qor_suggestions` that generate new suggestions (RQS generates new suggestions when called without `-of_objects` switch), it will only show suggestions from the file.

Finally, search for the following

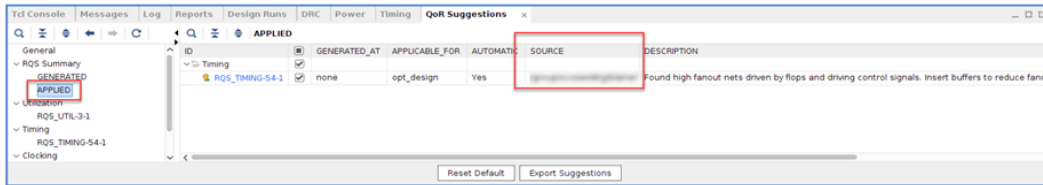
```
Phase 4 BUFG optimization
INFO: [Opt 31-318] Inserted a BUFGCE for CLOCK_BUFFER_TYPE property on
net: clk300_to_clk600_ffs_i/a_r
Phase 4 BUFG optimization | Checksum: 115417bbf
```

It is at this point where the suggestion is executed and the BUFGCE is inserted into the netlist.

3. With the Implemented Design now open, click **Reports** → **Report QoR Suggestions....** This time ensure the **View existing suggestions** box is checked as shown in the following figure:

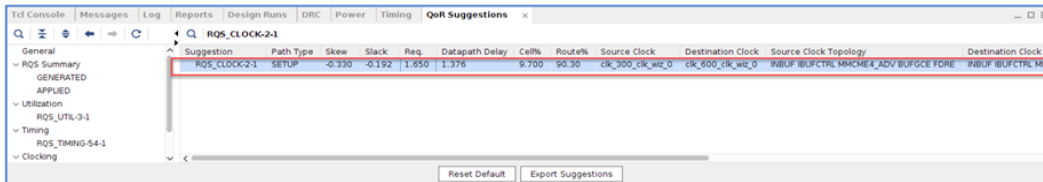


4. In the report, under RQS Summary select **APPLIED**. This shows all the AUTOMATIC suggestions that have been applied. User suggestions are not analyzed. Here you can see that RQS_TIMING-54-1 is applied:



Also note that the source of this suggestion is not the current_run.

5. Click **GENERATED**. There are now new suggestions for you to examine. The reasons for this are:
 - a. As you progress through the implementation stages, early estimates become hardened numbers and accuracy increases. RQS will not only use the updated numbers but different thresholds as confidence increases that action is or is not required.
 - b. Some suggestions are not available in the early design stages. For example, congestion suggestions become available only when there is placement information.
6. Examine the other suggestions. Note the suggestion RQS_CLOCK-2-1. This is a manual suggestion that recommends changing the clock implementation to use BUFDC_DIVs instead of BUFDCs. Details of this suggestion can also be found in the *UltraFast Design Methodology Guide for the Vivado Design Suite (UG949) - Synchronous CDC*.
7. Click on the **RQS_CLOCK-2-1** hyperlink to go to the details section for this suggestion.



8. Next double click on the path to open up the timing report. In the timing report click on the **Clock Uncertainty** hyperlink. You can see that there is a Phase Error (PE) element that is shown. Phase Error gets added when the source and destination clocks are from different MMCM output pins. The suggestion requests you to use the same MMCM output pin, connect them to BUFDC_DIVs with different divisors. This modification requires a manual RTL modification.

$\frac{\sqrt{((TS)^2 + (DJ)^2)}}{2} + PE$	
Total System jitter (TSJ)	0.071ns
Discrete jitter (DJ)	0.092ns
Phase Error (PE)	0.120ns

9. The RQS file can be modified to have both EXISTING suggestions and newly GENERATED suggestions. This time using TCL, export the following suggestions:

- RQS_TIMING-54-1 - This is the previous suggestion we exported under the APPLIED section
- RQS_CLOCK-1-1 - This is a new suggestion to apply CLOCK_DELAY_GROUP

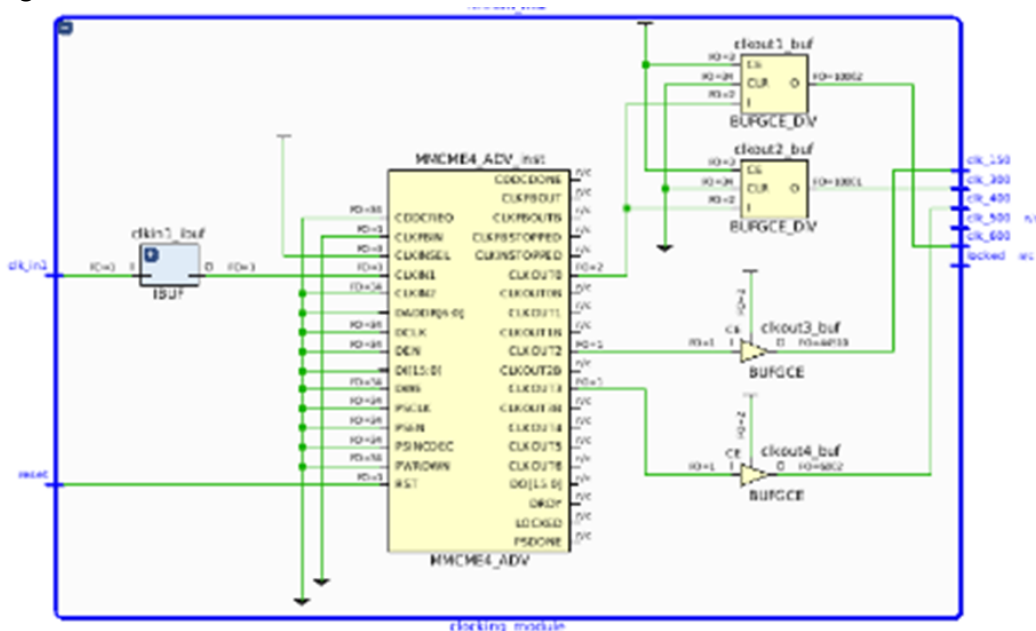
The TCL syntax for this is:

```
write_qor_suggestions -of_objects [list RQS_TIMING-54-1 RQS_CLOCK-1-1] \
-force <extract_dir>/lab2/initial_run_post_route_suggestions.rqs
```

10. Select **File** → **Close Project** to close the project.

Step 5: Design Closure

1. In the Vivado® Design Suite, go to **File** → **Project** → **Open** and select the project located in <extract_dir>/lab2/2_synth_fixes.
2. A completed implementation run exists already for you. In the Flow Navigator, select **Open Implemented Design**. Whilst the design is opening, use the time to examine the updated `clocking_module.vhd` and the implementation log file.
3. When the design is open, in the Netlist window, select `mmcm_inst` and press **F4**. Examine the clocking change that has been implemented. The updated clocking is shown in the following figure:



4. Select **Timing** tab → **Inter-clock paths** → `clk_300 to clk600_clk_wiz` → select first path. Next examine the clock uncertainty. You can see that the phase error is now reduced to 0.000.

Clock Uncertainty Equation	
$((TSJ)^2 + DJ^2)^{1/2} / 2 + PE$	
Total System Jitter (TSJ)	0.071ns
Discrete jitter (DJ)	0.081ns
Phase Error (PE)	0.000ns

- In the timing report, select **Design Timing Summary** and confirm that the run has closed timing:

Design Timing Summary					
Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	0.004 ns	Worst Hold Slack (WHS):	0.010 ns	Worst Pulse Width Slack (WPWS):	0.151 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	77512	Total Number of Endpoints:	77512	Total Number of Endpoints:	70583

All user specified timing constraints are met.

- You can also run `report_qor_suggestions` on the latest revision of this design and see further RQS suggestions. On a timing closed design, suggestions are limited as there are no timing failures but congestion suggestions are still available to improve the design
- Close the design.

Summary

In this lab, you used RQS to conduct a complex analysis of a demonstration design. You firstly examined the reports that showed RQS provided recommendations to solve implementation problems, then generated an RQS file and added it to a project implementation run. The Vivado® implementation tools executed these suggestions automatically for you. You subsequently performed further analysis and generated more suggestions, ultimately achieving design closure.

Additional Resources and Legal Notices

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby **DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE**; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at <https://www.xilinx.com/legal.htm#tos>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at <https://www.xilinx.com/legal.htm#tos>.

AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING

OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

Copyright

© Copyright 2019 Xilinx, Inc. Xilinx, the Xilinx logo, Alveo, Artix, Kintex, Spartan, Versal, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.