AN 922: Using the ECO Compilation Flow

Online Version

Send Feedback

ID: 683873

Version: 2022.03.31

Contents

1. AN 922: Using the ECO Compilation Flow.. 3
1.1. Step 1: Open the Design Example Project... 4
1.2. Step 2: Run the ECO Flow..5
1.3. Step 3: Implement Targeted ECOs.. 6
1.3.1. Modify the Lutmask..6
1.3.2. Change Routing Connections... 8
1.3.3. Tie Off Input Port to GND..9
1.3.4. Modify Slew Rate, Current Strength, and Delay Chain................................. 10
1.3.5. Place A Node in a New Location.. 11
1.3.6. Create a Wire LUT Atom...14
1.4. ECO Tcl Script Example..16
1.5. ECO Command Limitations.. 16
1.6. Document Revision History for AN 922: Using the ECO Compilation Flow.................... 18

1. AN 922: Using the ECO Compilation Flow

In a typical FPGA project development cycle, the specification of the programmable logic portion of the design can change during the design process. The Intel® Quartus® Prime software supports these last-minute, targeted *engineering change orders* (ECOs), even after full compilation is complete. This application note demonstrates implementation of ECO's with an example design.

ECOs typically occur during the design verification stage. For example, during verification you may determine that the design requires a small change, such as a netlist connection change, correcting a LUT logic error, or placing a node in a new location. Implementing an ECO change, rather than changing RTL and fully recompiling the design, requires significantly less time, and changes only the affected logic.

You specify the ECO commands in a Tcl script using the `::quartus::eco` package.

Table 1. ECO Command Quick Reference

<table>
<tr><th>ECO Change</th><th>ECO Commands</th></tr>
<tr><td>Route</td><td>make_connection -from <src> -to <dst> -port <port>
remove_connection -from <src> -to <dst> -port <port></td></tr>
<tr><td>Tie-Off</td><td>make_connection –tieoff <VCC/GND> -to <node> -port <port></td></tr>
<tr><td>Lutmask</td><td>modify_lutmask -to <node> [-eqn <lut equation>] [-mask 0x00]</td></tr>
<tr><td>Slew Rate</td><td>modify_io_slew_rate <value> -to <pin_name></td></tr>
<tr><td>Current Strength</td><td>modify_io_current_strength <value> -to <pin_name></td></tr>
<tr><td>Delay Chains</td><td>modify_io_delay_chain <value> -type <io_type> -to <pin_name></td></tr>
<tr><td>Update MIF</td><td>update_mif_files</td></tr>
<tr><td>IOPLL Ref Clock (Intel Stratix® 10 devices only)</td><td>adjust_pll_refclk -to <pll name> -refclk <freq></td></tr>
<tr><td>Create New Node</td><td>create_new_node -type <LUT|FF> -name <name></td></tr>
<tr><td>Remove Node</td><td>remove_node -name <name></td></tr>
<tr><td>Place Node</td><td>place_node -name <name> [-location <location>]</td></tr>
<tr><td>Unplace Node</td><td>unplace_node -name <name></td></tr>
<tr><td>Create Wirelut</td><td>create_wirelut -from <src> -to <dst> -port <port> [-location <location>]</td></tr>
</table>

The activities of this application note are divided into the following sections:

- Step 1: Open the Design Example Project on page 4
- Step 2: Run the ECO Flow on page 5
- Step 3: Implement Targeted ECOs on page 6

Note: The Intel Quartus Prime Pro Edition software supports ECOs only for Intel Stratix 10 and Intel Agilex™ devices.

1.1. Step 1: Open the Design Example Project

This application note includes a design example and Tcl script files that demonstrate use of various ECO commands. You can download and restore the design example to follow along with the application note steps in the Intel Quartus Prime Pro Edition software.

1. Save the design example project archive file to hard drive:

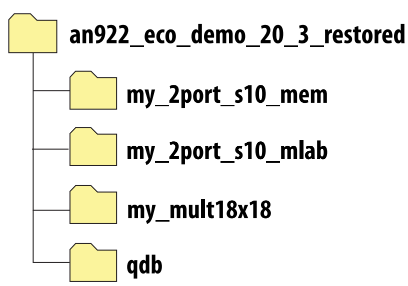
   an922_eco_demo_20_3.qar

2. In the Intel Quartus Prime Pro Edition software version 20.3, click **Project ➤ Restore Archived Project**, and then specify the `an922_eco_demo_20_3.qar` design example archive for the **Archive name**.

   This design example is verified with Intel Quartus Prime Pro Edition software version 20.3, but can be adapted for later versions with slightly different results.

3. For **Destination folder**, specify a new directory to contain the restored design example project files. The project directory also contains the `eco_demo_1.tcl` through `eco_demo_6.tcl` Tcl scripts for this design example.

Figure 1. Design Example Directory Structure



4. The **eco_demo** project revision opens in the Intel Quartus Prime Pro Edition software.

Related Information

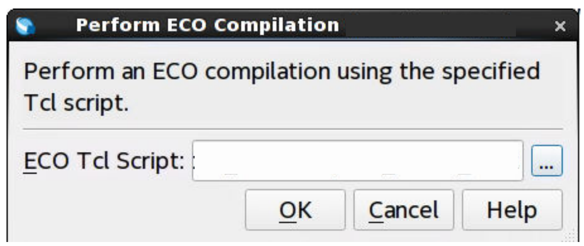
Using the ECO Compilation Flow, Intel Quartus Prime Pro Edition User Guide: Design Optimization

1.2. Step 2: Run the ECO Flow

The following steps describe how to setup, run, and view the results for any ECO command:

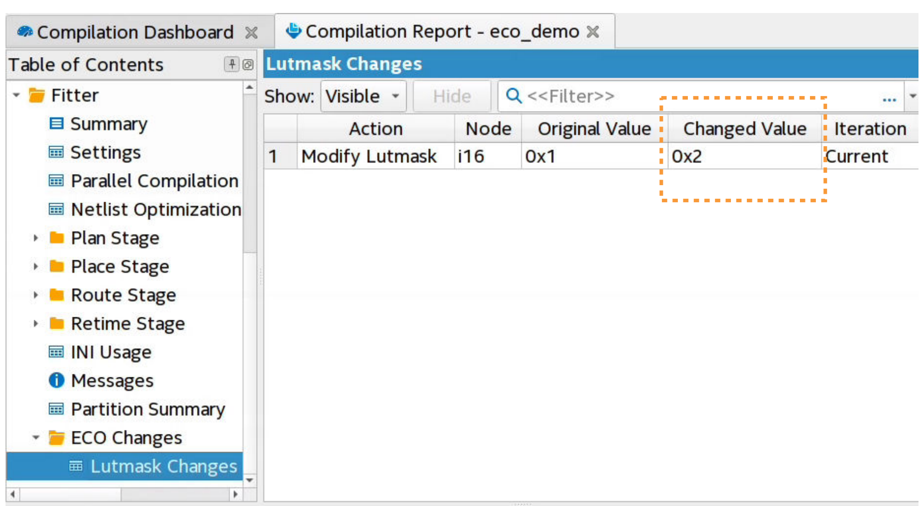
1. Determine if ECO commands support a change you want to make in a compiled design, by reviewing the "ECO Command Quick Reference" table and ECO Command Limitations on page 16.
2. Create a Tcl script that calls the ECO command, as ECO Tcl Script Example on page 16 shows. The design example includes six `eco_demo_<n>.tcl` files for use with this example.
3. To run a full compilation, click **Processing ➤ Start Compilation**. ECO commands allow you make targeted changes even after a full compilation is complete. Close the Timing Analyzer that opens when full compilation is complete.
4. To run ECO commands, click **Processing ➤ Start ➤ Perform ECO Compilation**.

Figure 2. Perform ECO Compilation



5. Specify a Tcl Script to implement one or more ECOs, as Step 3: Implement Targeted ECOs on page 6 describes.
6. View the ECO results in post-fit analysis tools, such as the Compilation Report, Timing Analyzer, Netlist Viewer, or Chip Planner. To view ECO changes in the Fitter report, click **Processing ➤ Compilation Report ➤ Fitter ➤ ECO Changes**.

Figure 3. Example of ECO Changes Report



1.3. Step 3: Implement Targeted ECOs

Run the provided Tcl scripts to implement the following ECOs:

Table 2. ECO Commands and Tcl Scripts

ECO Modification	ECO Command | Script
Modify the Lutmask on page 6	modify_lutmask | eco_demo_1.tcl
Change Routing Connections on page 8	make_connection | eco_demo_2.tcl
Tie Off Input Port to GND on page 9	modify_io_slew_rate | eco_demo_3.tcl
Modify Slew Rate, Current Strength, and Delay Chain on page 10	modify_io_delay_chain | eco_demo_4.tcl
Place A Node in a New Location on page 11	place_node | eco_demo_5.tcl
Create a Wire LUT Atom on page 14	create_wirelut | eco_demo_6.tcl

As an alternative to the GUI methods, you can use the following commands to run the ECO Tcl scripts:

```
$ quartus_fit -s
load_package eco
project_open <project_name>
eco_load_design
...
eco_commit_design
project_close
```

1.3.1. Modify the Lutmask

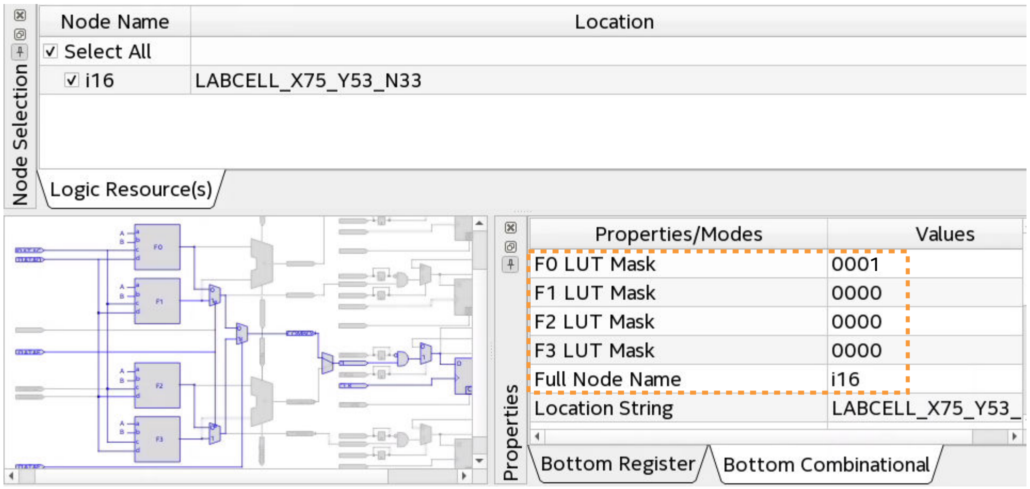
You can specify the `modify_lutmask` command to modify the lutmask to invert a pin in your design.

Run the `eco_demo_1.tcl` Tcl script to modify node `i16` with a lutmask hexadecimal value of 2. `eco_demo_1.tcl` makes changes regarding this RTL in `eco_demo.v`:

```
lab_and6 <= (inputa_6_reg[2] && inputa_6_reg[1] && inputa_6_reg[0] && \
     inputb_6_reg[2] && inputb_6_reg[1] && inputb_6_reg[0]);
```

1. To locate node `i16`, click **View ➤ Node Finder**, type `i16` in the **Named** field, and then click **Search**. `i16` appears in the **Nodes Found** results.
2. In the Node Finder, right-click the `i16` in **Nodes Found**, and then click **Locate Node ➤ Locate in Resource Property Viewer**. The `i16` node highlights in the Resource Property Viewer.

Figure 4. i16 in Resource Property Viewer Before Lutmask ECO

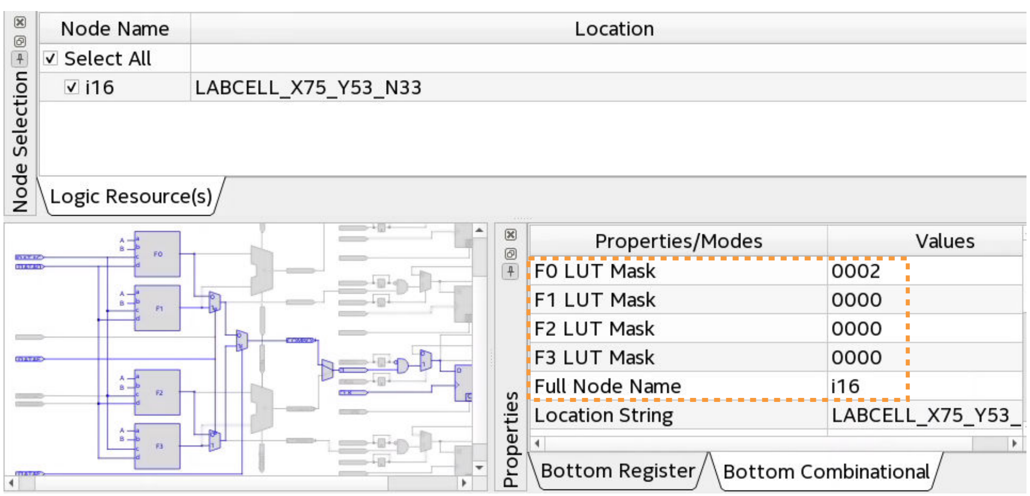


3. View the current LUT mask values in the **Bottom Combinational** tab, and then close Resource Property Viewer.
4. Click **Processing ➤ Start ➤ Perform ECO Compilation**.
5. For **ECO Tcl Script**, select `eco_demo_1.tcl` in the project directory, click **Open**, and then click **OK**. `eco_demo_1.tcl` contains the following `modify_lutmask` ECO commands. You can either modify the mask bits directly (line 2), or modify the equation (line 3).

```
###locate i16 from node finder
modify_lutmask -to i16 -mask 0x0000000000000002
#modify_lutmask -to i16 -eqn { !a & !b & c & !d & !e & !f }
```

6. Repeat steps 1 through 2 to view the change in Resource Property Viewer.

Figure 5. i16 in Resource Property Editor After Lutmask ECO Complete



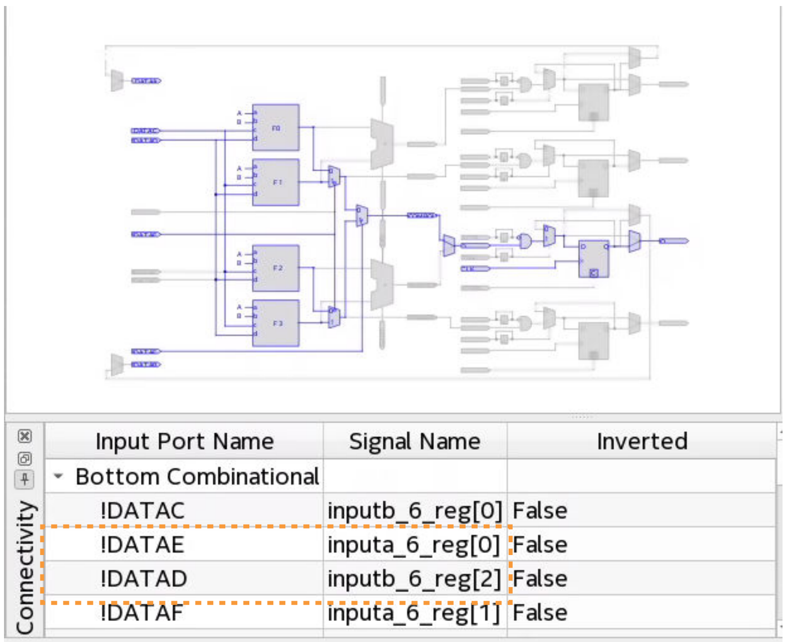
1.3.2. Change Routing Connections

You can specify the `remove_connection` and `make_connection` commands to modify the routing of the compiled design.

Follow these steps to modify the routing of node `i22` by running the `eco_demo_2.tcl` Tcl script:

1. In the **Node Finder**, find and right-click node `i22`, and then click **Locate Node ➤ Locate in Resource Property Viewer**.

Figure 6. i22 in Resource Property Viewer Before ECO



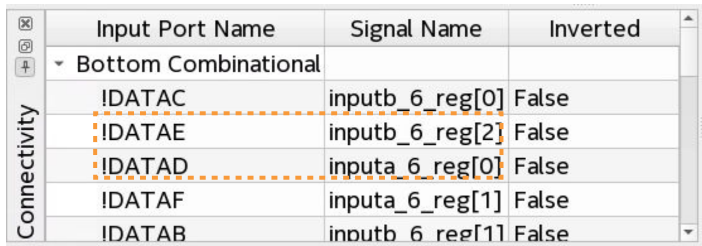
2. Click **Processing ➤ Start ➤ Perform ECO Compilation**. Specify and run the `eco_demo_2.tcl` file. `eco_demo_2.tcl` contains the following `remove_connection` and `make_connection` ECO commands:

```
remove_connection -from inputa_6_reg[0] -to i22 -port DATAE
remove_connection -from inputb_6_reg[2] -to i22 -port DATAD

make_connection -from inputb_6_reg[2] -to i22 -port DATAE
make_connection -from inputa_6_reg[0] -to i22 -port DATAD
```

3. When ECO compilation is complete, repeat step 1 to view the change in Resource Property Viewer.

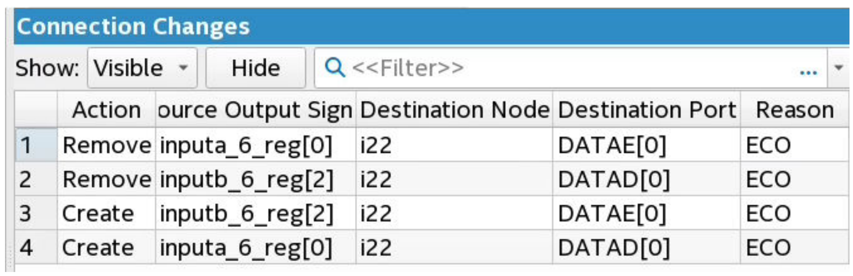
Figure 7. i22 in Resource Property Viewer After ECO Complete



Input Port Name	Signal Name	Inverted
Bottom Combinational		
!DATAC	inputb_6_reg[0]	False
!DATAE	inputb_6_reg[2]	False
!DATAD	inputa_6_reg[0]	False
!DATAF	inputa_6_reg[1]	False
!DATAB	inputb_6_reg[1]	False

4. In the Fitter section of the Compilation Report, view the Connection Changes report under the ECO Changes folder.

Figure 8. Connection Changes Report



Connection Changes

Show: Visible Hide <<Filter>>

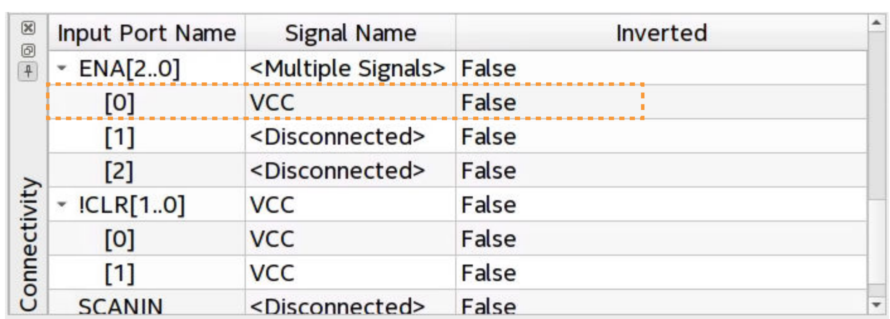
	Action	ource Output Sign	Destination Node	Destination Port	Reason
1	Remove	inputa_6_reg[0]	i22	DATAE[0]	ECO
2	Remove	inputb_6_reg[2]	i22	DATAD[0]	ECO
3	Create	inputb_6_reg[2]	i22	DATAE[0]	ECO
4	Create	inputa_6_reg[0]	i22	DATAD[0]	ECO

1.3.3. Tie Off Input Port to GND

To tie off the DSP `ENA[0]` port to `GND`, follow these steps:

1. In the **Node Finder**, find and right-click node `my_dsp_inst0`, and then click **Locate Node ➤ Locate in Resource Property Viewer**.
2. Click the **DSP Elements** tab. In the **Connectivity** pane, locate **ENA[2..0]** in the **Input Port Name** column. `ENA[0]` is set to VCC.

Figure 9. `my_dsp_inst0` Before ECO Change



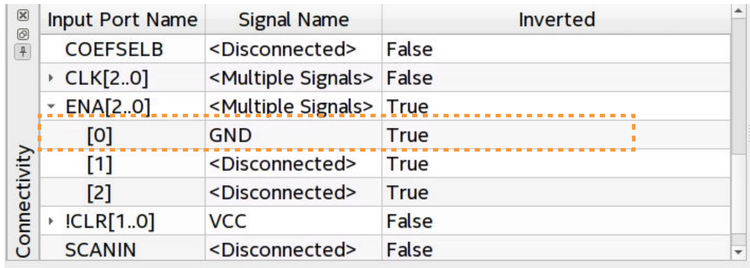
Input Port Name	Signal Name	Inverted
ENA[2..0]	<Multiple Signals>	False
[0]	VCC	False
[1]	<Disconnected>	False
[2]	<Disconnected>	False
!CLR[1..0]	VCC	False
[0]	VCC	False
[1]	VCC	False
SCANIN	<Disconnected>	False

3. Click **Processing ➤ Start ➤ Perform ECO Compilation**. Specify and run `eco_demo_3.tcl`. `eco_demo_3.tcl` contains the following `make_connection` ECO command:

```
make_connection -tieoff GND -to {my_dsp_inst0|mult_inst|lpm_mult_0|\
   lpm_mult_component|auto_generated|mult_0~mac} -port {ENA[0]}
```

4. Repeat steps 1 and 2 to view the ECO change in Resource Property Viewer.

Figure 10. `my_dsp_inst0` **After ECO Change**



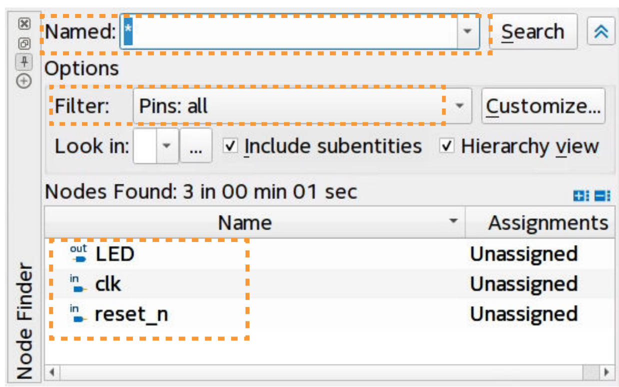
Input Port Name	Signal Name	Inverted
COEFSELB	<Disconnected>	False
CLK[2..0]	<Multiple Signals>	False
ENA[2..0]	<Multiple Signals>	True
[0]	GND	True
[1]	<Disconnected>	True
[2]	<Disconnected>	True
!CLR[1..0]	VCC	False
SCANIN	<Disconnected>	False

1.3.4. Modify Slew Rate, Current Strength, and Delay Chain

You can use ECO commands to modify I/O slew rate, current strength, and delay chains. The following steps describe making all three modifications with the provided `eco_demo_4.tcl` script.

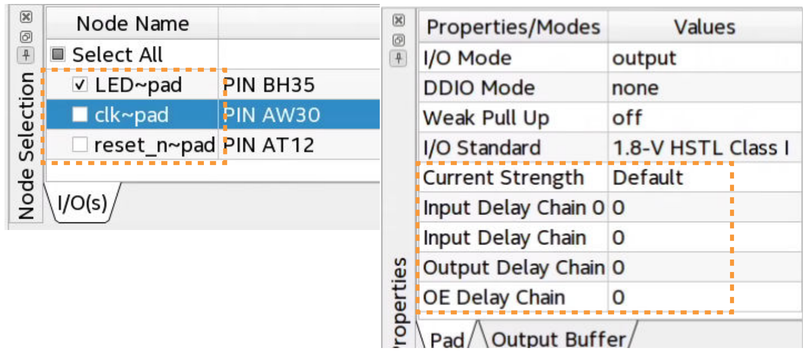
1. In the **Node Finder**, find nodes `clk`, the output pin LED, and the output pin `reset_n`.

Figure 11. **Node Finder Options**



2. Right-click the multi-selected nodes, and then click **Locate Node ➤ Locate in Resource Property Viewer**. Note the existing values for current strength, slew rate, and delay chains. Turn on or off display of properties in the **Node Selection** pane.

Figure 12. Resource Property Editor Before ECO Changes

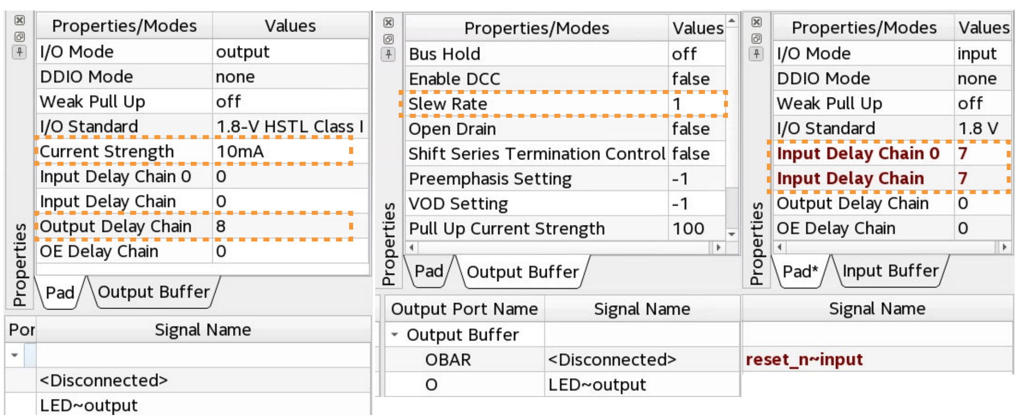


3. Click **Processing ➤ Start ➤ Perform ECO Compilation**. Specify and run `eco_demo_4.tcl`. `eco_demo_4.tcl` contains the following ECO commands:

```
modify_io_slew_rate 1 -to LED
modify_io_current_strength 10mA -to LED
modify_io_delay_chain 7 -type input -to reset_n
modify_io_delay_chain 8 -type output -to LED
```

4. Repeat steps 1 through 2 to view the changes in the Resource Property Viewer. The **Pad** tab displays the **Current Strength** and **Output Delay Chain** for LED.

Figure 13. Change to Delay Chains, Current Strength, and Slew Rate



1.3.5. Place A Node in a New Location

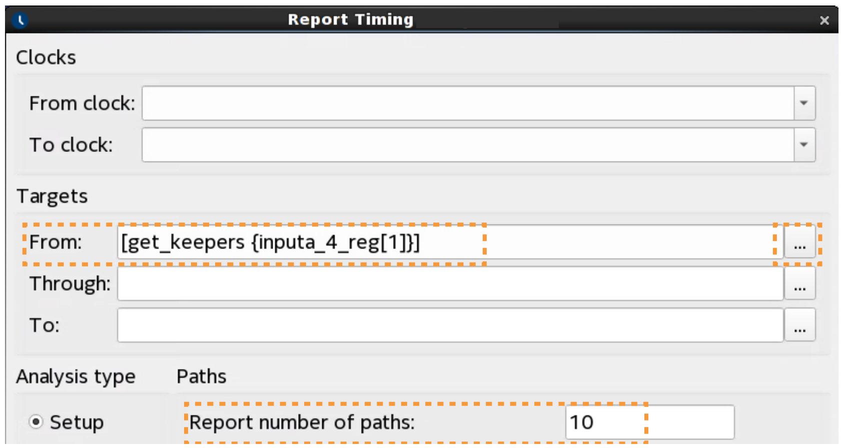
You can specify the LAB location for a node by using the `place_node` command to place an existing node in a new location. In this example, the target coordinates for nodes `i11` and `i8` are (24, 63).

Note: ECO's pertain to the physical node. Therefore the physical names in this example, such as "i8", can change from software release to release.

Follow these steps to place `i11` and `i8` in a new LAB location by running the `eco_demo_5.tcl` Tcl script.

1. Click **Tools ➤ Timing Analyzer**.
2. To observe the location of the LAB in the Timing Analyzer, click **Reports ➤ Custom Reports ➤ Report Timing**.
3. Under **Targets**, click the **...** button to search for `inputa_4_reg[1]` with the **Collection** of **get_keepers**.

Figure 14. Report Timing for inputa_4_reg[1]



4. For the **Setup** option, specify **10** for **Report number of paths**, and then click **OK**.
5. View the Report Timing report in the Timing Analyzer.

Figure 15. Node i8 in Path inputa_4_reg[1] to lab_and4 at Location (75,53)

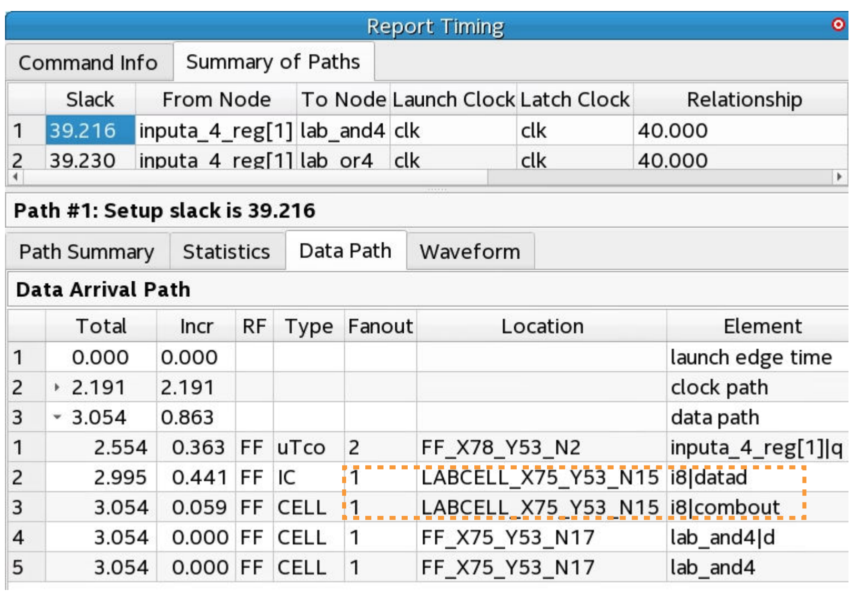
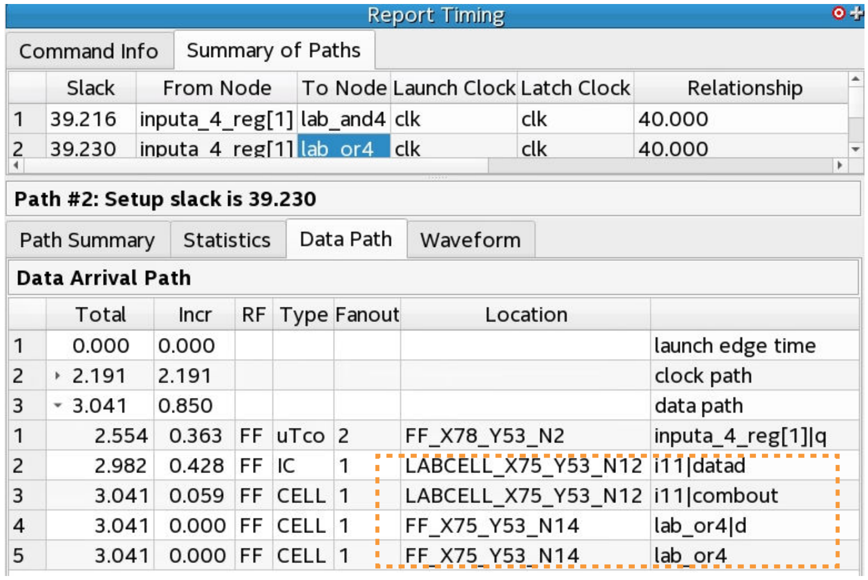


Figure 16. Node i11 in Path inputa_4_reg[1] to lab_or4 at Location (75,53)



Report Timing

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship
1	39.216	inputa_4_reg[1]	lab_and4	clk	clk	40.000
2	39.230	inputa_4_reg[1]	lab_or4	clk	clk	40.000

Path #2: Setup slack is 39.230

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

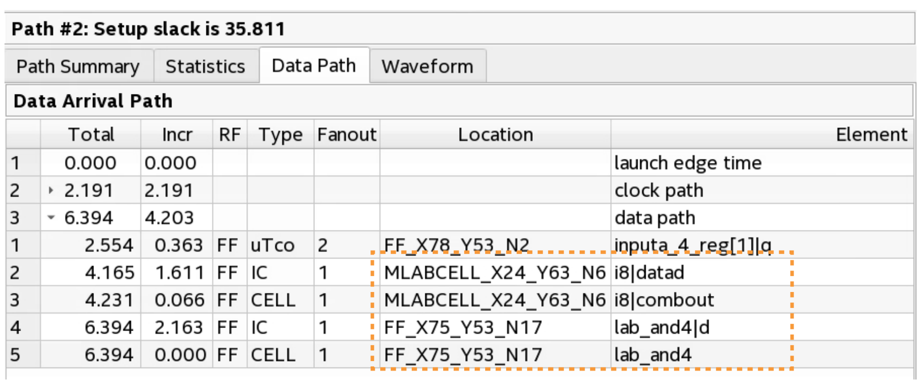
	Total	Incr	RF	Type	Fanout	Location	
1	0.000	0.000					launch edge time
2	2.191	2.191					clock path
3	3.041	0.850					data path
1	2.554	0.363	FF	uTco	2	FF_X78_Y53_N2	inputa_4_reg[1]|q
2	2.982	0.428	FF	IC	1	LABCELL_X75_Y53_N12	i11|datad
3	3.041	0.059	FF	CELL	1	LABCELL_X75_Y53_N12	i11|combout
4	3.041	0.000	FF	CELL	1	FF_X75_Y53_N14	lab_or4|d
5	3.041	0.000	FF	CELL	1	FF_X75_Y53_N14	lab_or4

6. Click **Processing ➤ Start ➤ Perform ECO Compilation**, and select the `eco_demo_5.tcl` script. `eco_demo_5.tcl` contains the following `place_node` ECO commands:

```
place_node -name i11 -location "X24 Y63"
place_node -name i8 -location "X24 Y63 X24 Y63"
```

7. Repeat step 1 through 5 to observe the new LAB location in the Timing Analyzer.

Figure 17. New Location of i8 at (24,63) in Timing Analyzer



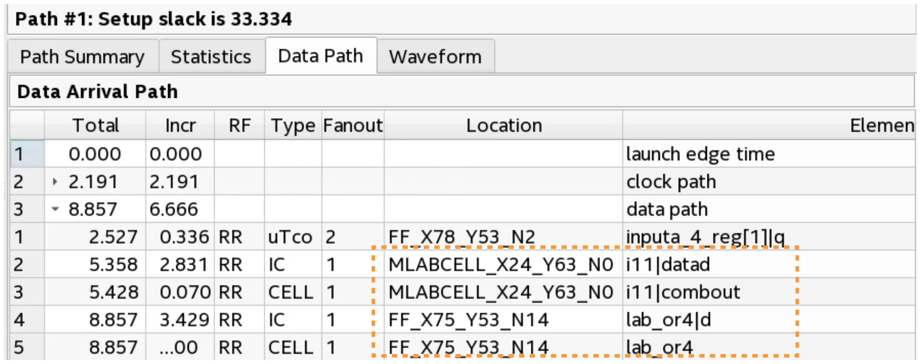
Path #2: Setup slack is 35.811

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	2.191	2.191					clock path
3	6.394	4.203					data path
1	2.554	0.363	FF	uTco	2	FF_X78_Y53_N2	inputa_4_reg[1]|q
2	4.165	1.611	FF	IC	1	MLABCELL_X24_Y63_N6	i8|datad
3	4.231	0.066	FF	CELL	1	MLABCELL_X24_Y63_N6	i8|combout
4	6.394	2.163	FF	IC	1	FF_X75_Y53_N17	lab_and4|d
5	6.394	0.000	FF	CELL	1	FF_X75_Y53_N17	lab_and4

Figure 18. New Location of i11 at (24,63) in Timing Analyzer


Path #1: Setup slack is 33.334

Path Summary | Statistics | Data Path | Waveform

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Elemen
1	0.000	0.000					launch edge time
2	2.191	2.191					clock path
3	8.857	6.666					data path
1	2.527	0.336	RR	uTco	2	FF_X78_Y53_N2	inputa_4_reg[1]|q
2	5.358	2.831	RR	IC	1	MLABCELL_X24_Y63_N0	i11|datad
3	5.428	0.070	RR	CELL	1	MLABCELL_X24_Y63_N0	i11|combout
4	8.857	3.429	RR	IC	1	FF_X75_Y53_N14	lab_or4|d
5	8.857	...00	RR	CELL	1	FF_X75_Y53_N14	lab_or4

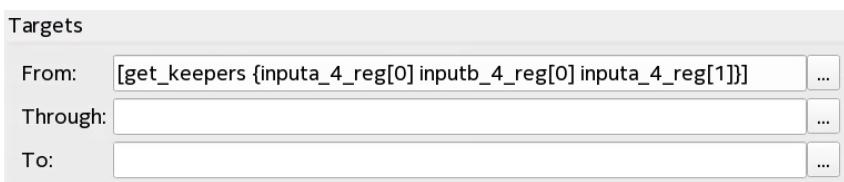
1.3.6. Create a Wire LUT Atom

You can specify the `create_wirelut` command to create a multiple of wire LUT atoms.

Follow these steps to create 3 wire LUTs from `inputa_4_reg[0]` by running the `eco_demo_6.tcl` Tcl script.

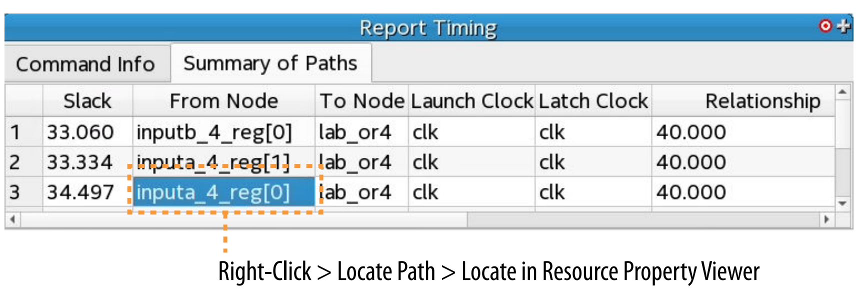
1. In the Timing Analyzer, click **Reports ➤ Custom Reports ➤ Report Timing**.
2. Under **Targets**, specify `inputa_4_reg[0]`, `inputb_4_reg[0]`, and `inputa_4_reg[1]` for **From**.

Figure 19. Report Timing Targets



3. In the Timing Analyzer, right-click `inputa_4_reg[0]`, and then click **Locate Path > Locate in Resource Property Viewer**.

Figure 20. Locate Path in Resource Property Viewer


Report Timing

Command Info | Summary of Paths

	Slack	From Node	To Node	Launch Clock	Latch Clock	Relationship
1	33.060	inputb_4_reg[0]	lab_or4	clk	clk	40.000
2	33.334	inputa_4_reg[1]	lab_or4	clk	clk	40.000
3	34.497	inputa_4_reg[0]	lab_or4	clk	clk	40.000

4. In the Node Selection pane, select node `i11` under **Arrival Data**. You can view the node `i11` as the `COMBOUT` in the **Top Combinational** list. The `inputa_4_reg[0]` node connects with the DATAE port.

Figure 21. Nodes Found in Resource Property Viewer Before ECO

Input Port Name	Signal Name	Inverted
!DATAC	inputb_4_reg[1]	False
!DATAD	inputa_4_reg[1]	False
!DATAE	inputa_4_reg[0]	False
!DATAF	<Disconnected>	False
!DATAB	<Disconnected>	False
!DATAA	inputb_4_reg[0]	False

Output Port Name	
Top Combinational	
SUMOUT	<Disconnected>
COUT	<Disconnected>
COMBOUT	i11
Bottom Combinational	
COMBOUT	<Disconnected>

5. To add a wire LUT from the targeted node, click **Processing ➤ Start ➤ Perform ECO Compilation**. Specify and run `eco_demo_6.tcl`. `eco_demo_6.tcl` contains the following `create_wirelut` ECO command:

```
create_wirelut -name my_delay_wirelut_1 -from inputa_4_reg[0] \
     -to i11 -port DATAA -location "X23 Y61 X23 Y61"
create_wirelut -name my_delay_wirelut_2 -from inputb_4_reg[0] -to i11 \
   -port DATAD -location "X23 Y61 X23 Y61"
create_wirelut -name my_delay_wirelut_3 -from inputa_4_reg[1] -to i11 \
     -port DATAC -location "X23 Y61 X23 Y61"
```

6. Repeat steps 1 through 4 to view the change in Resource Property Viewer.

Figure 22. New Wire LUT Atom After ECO Command

Input Port Name	Signal Name	Inverted
Top Combinational		
!DATAC	my_delay_wirelut_3	False
!DATAD	my_delay_wirelut_2	False
!DATAE	inputa_4_reg[0]	False
!DATAF	<Disconnected>	False
!DATAB	<Disconnected>	False
!DATAA	my_delay_wirelut_1	False

1.3.6.1. Running Incremental Flow with Signal Probe

You can route the internal signal to pins and remove the reserve pins to probe the signal with Signal Probe.

You can add assignments to route the internal signal to pins; for example:

```
set_global_assignment -name CREATE_SIGNALPROBE_PIN test_pin1
set_instance_assignment -name CONNECT_SIGNALPROBE_PIN test_pin1 -to addressr_reg
```

Recompile the design with the command: `quartus_sh –flow recompile` *`<design>`*

You can check the connection in the compilation report, or in the Fitter report, *`<design_name>`*`.fit.rpt`.

Figure 23. Connections to Signal Probe Pins in Compilation Report

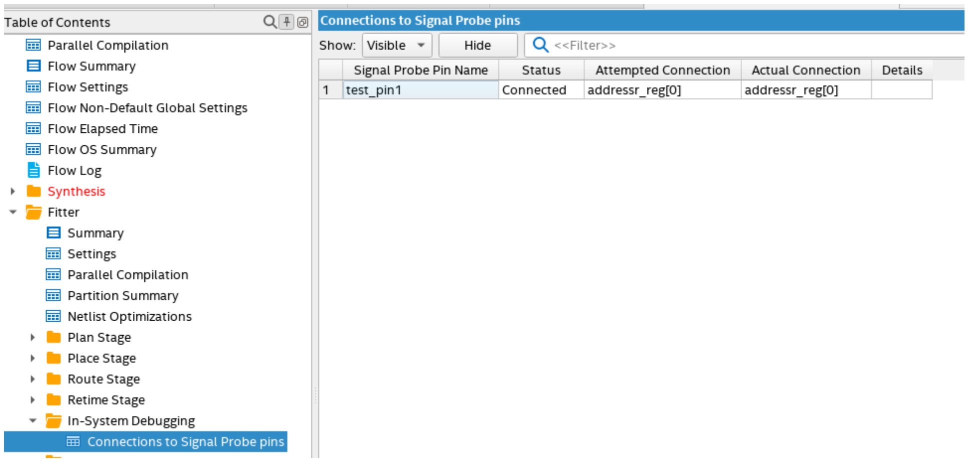


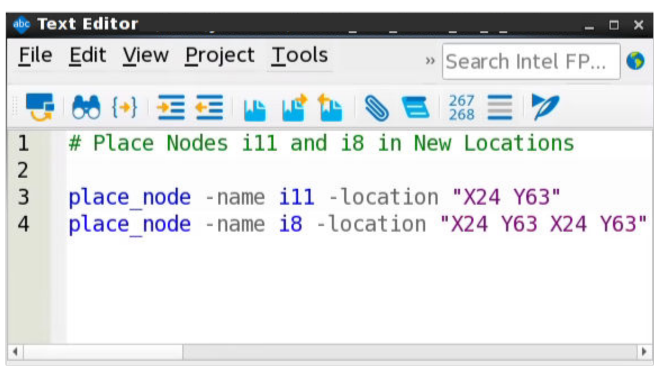
Figure 24. Connections to Signal Probe Pins in Fitter Report

```
+----------------------------------------------------------------------------------------+
; Connections to Signal Probe pins                                                       ;
+------------------------+-----------+----------------------+--------------------+---------+
; Signal Probe Pin Name ; Status    ; Attempted Connection ; Actual Connection ; Details ;
+------------------------+-----------+----------------------+--------------------+---------+
; test_pin1              ; Connected ; addressr_reg[0]      ; addressr_reg[0]    ;         ;
+------------------------+-----------+----------------------+--------------------+---------+
```

1.4. ECO Tcl Script Example

The following shows an example ECO Tcl script that places existing nodes in new locations:

Figure 25. ECO Tcl Script Example



1.5. ECO Command Limitations

The ECO commands have the following limitations due to connection dependencies within Intel FPGA devices.

- You cannot use ECO commands to modify dedicated connections.
- You cannot modify dedicated connections within a single ALM. This limitation applies to direct connections between LUT and flip-flop nodes.
- You can connect from or to a Hyper-Register. However, you cannot remove connections from or to a Hyper-Register because removing a connection from a Hyper-Register would leave the routing dangling. As an alternative, you can use `make_connection` to change a Hyper-Register connection immediately, without removing the previous connection first.
- Use of the `place_node` command with `location` arguments does not overwrite Partial Reconfiguration region constraints.
- If a LAB already has the maximum number of legal connections where a node is placed, the `place_node` or `make_connection` commands can fail, preventing the connection to the first placed node that cannot be legalized. You can then either move the original node to a different location, or move other nodes from the LAB to free up routing resources.
- The Fitter may fail to apply some I/O related ECO modifications, such as `modify_io_slew_rate`, `modify_io_current_strength`, and `modify_io_delay_chain`, if called using a command-line Tcl script or in interactive context. That is, any case that calls the `eco_load_design` command directly. To ensure all I/O modifications are applied successfully, use the standard ECO Tcl script approach this document describes.

The recommended order for creating and placing new LUTs or new flipflops is:

1. Create the node by using the `create_new_node` command.
2. Make connections to and from the node by using the `make_connection` command.
3. Update the lutmask by using the `modify_lutmask` command.
4. Place the node by using the `place_node` command.

This flow ensures that analysis includes all routing requirements when determining a legal placement for the new node. For example:

Create a new LUT in an exact location

```
set lut_name new_lut
create_new_node –name $lut_name –type lut
make_connection –from input1 –to $lut_name –port DATAA
make_connection –from input2 –to $lut_name –port DATAB
make_connection –from $lut_name –to output_dest –port DATAD
modify_lutmask –to $lut_name –eqn {A&B}
place_node –name $lut_name –location “X80 Y80 X85 Y95”
```

Create a new Flipflop in an exact location

```
set ff_name new_ff
create_new_node –name $ff_name –type ff
make_connection –from input1 –to $ff_name –port DATAA
make_connection –from input2 –to $ff_name –port DATAB
make_connection –from $ff_name –to output_dest –port DATAD
modify_lutmask –to $ff_name –eqn {A&B}
place_node –name $ff_name –location “X80 Y80 X85 Y95”
```

Note: To minimize issues with name matching caused by escaped characters, it can be useful to surround entity names with {} characters, instead of "". This technique is particularly useful if entity names contain backslashes or any other special characters.

1.6. Document Revision History for AN 922: Using the ECO Compilation Flow

Document Version	Changes
2022.03.31	• Added *Running Incremental Flow with Signal Probe* topic. • Added *Create a new FlipFlop in an exact location* code sample to the *ECO Command Limitations* topic.
2020.09.28	• Initial release.