The Easy Way to Achieve ECO Success

Functional ECO Methodology Guide

Achieving the Smallest ECO Patch through RTL-to-RTL Comparison

EasylogicECO White Paper
WP-ECO-01

Engineering Change Order (ECO) is an incremental design method used to revise existing ASIC designs. Once a new RTL code changes the ASIC function, an ECO task modifies a portion of the original netlist to create a revised netlist that aligns with the new ASIC function. This process is known as Functional ECO.

Functional change requests have become increasingly common in recent years due to the rising complexity of ASIC design, shorter project cycles, and more importantly, evolving market demands. When an RTL change is necessary, re-spinning the entire design always causes project delays. A successful ECO task offers a shortcut for incorporating the required changes swiftly. In the late design stage, a successful ECO can significantly reduce the project delays from months to just a few days, while upholding same quality results and the integrity of design flow.

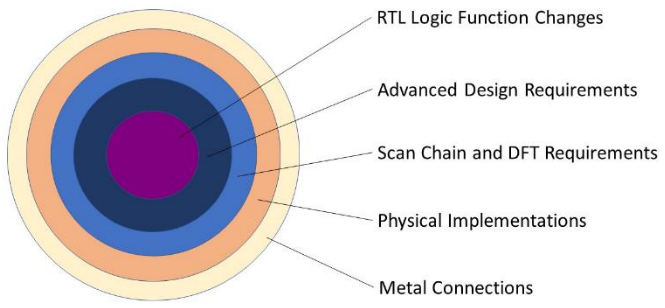
However, implementing a functional ECO is a complex process. This article discusses the challenges related to identifying the exact functionality changes and proposes an efficient design methodology that leads to creating the most precise ECO patch.

Introduction

A functional ECO task begins with an RTL code change. However, the resulting design change may extend beyond what is apparent in the downstream implementations. The functional ECO process encounters two primary challenges:

1. Assessing the extent of required netlist-level design changes and implementing all necessary fixes.
2. Fixing the circuit at various stages of the design flow.

Challenge 1 involves assessing the impact of an ECO on the overall design. For instance, consider a scenario where a straightforward RTL change is made to the interface portion of an IP, and the IP is instantiated multiple times in the design. In such a case, the functional ECO task can potentially become a disruptive experience.



RTL Logic Function Changes
Advanced Design Requirements
Scan Chain and DFT Requirements
Physical Implementations
Metal Connections

Figure 1: Types of data that might require correction in a functional ECO task

For challenge 2, the required revisions in a functional ECO depend on the stage of the design project. Designers might need to modify not only the design logic but also various downstream design data. Figure 1 illustrates the types of data that might require correction.

This white paper will focus on the first challenge: making the most precise changes.

Issues With the Traditional Methodology: Using G2G Comparison To Identify Functional Discrepancies

One of the most critical issues associated with the common functional ECO design methodology is that many designers initiate their functional ECO tasks with a Gate-to-Gate (G2G) comparison between the original design and the revised design (Figure 2). In this article, we will investigate the issues associated with this approach.

Functional comparison based on two gate-level netlists, where Synthesized Netlist is generated from the revised RTL, is commonly perceived as an easier approach to human interpretation and seems more logical, as the original netlist serves as the reference. However, G2G-based ECO method often brings about three significant problems.

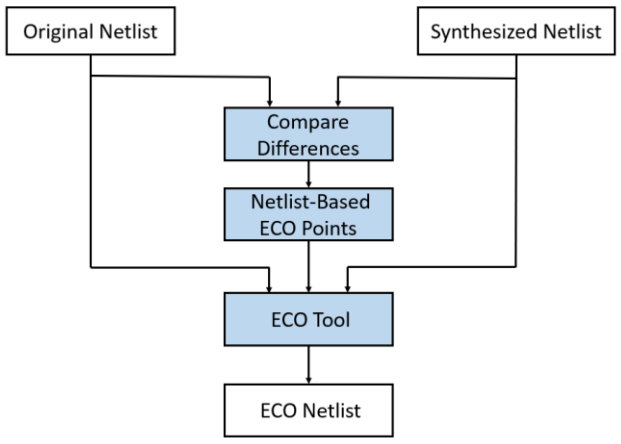


Figure 2: G2G netlist-based methodology

(1) This approach often creates false ECO points

The Synthesized Netlist may exhibit unexpected changes because of logic optimization during the RTL synthesis process. In some cases, signals present in the RTL may no longer exist in the netlist. Let's examine a simple example to illustrate this.

```
// Original RTL

module adder (
        input [3:0] operand_1,
        input [3:0] operand_2,
        input [3:0] operand_3,
        output [4:0] result);
wire [4:0] n1;

assign n1 = operand_1 + operand_2;
assign result = n1 + operand_3;
endmodule
```

```
// Revised RTL

module adder (
        input [3:0] operand_1,
        input [3:0] operand_2,
        input [3:0] operand_3,
        output [4:0] result);
wire [4:0] n1;

assign n1 = operand_1 + (operand_1 + operand_2) ^ 4'b0011;
assign result = n1 + operand_3;
endmodule
```

A required ECO change on "*assign n1*" statement in the Original RTL presents a challenge in locating the *n1* signal in the gate-level netlist. This is because the synthesis tool breaks down the signal to reduce the area. The absence of the signal can be attributed to the following reasons:

1. The wire *n1* in RTL is internal, which means that the synthesis tool has the flexibility to utilize it in any way if the output functions are correctly composed. Depending on the optimization goal, the synthesis

tool may choose to keep the signal *n1* or decompose it into minterms for constructing the output functions.

2. Regardless of whether the signal *n1* in RTL is retained, its name may be changed in the synthesized netlist. This makes it challenging for the designer to trace the corresponding signal in the netlist.

The above-mentioned ECO task involves modifying the function of *output: result* based on the new specification of *n1* in RTL. Consequently, the designer may need to invest significant effort in tracing the signal paths that drive all bits of *output:result* to create the smallest feasible patch.

(2) Large patch size

When either the Original Netlist or the Synthesized Netlist, or both, has optimized out some RTL signals from the netlist, the comparison can become a serious problem. The G2G equivalence check is likely to create false discrepancies, resulting in false ECO points for the functional ECO task. As a result of the ECO operation, an unnecessarily large patch circuit is generated, often failing to meet timing constraints.

(3) Long ECO turnaround time

Tool execution times include 2 major portions:

1. Logical equivalence check (LEC) between the original netlist and synthesized netlist to identify ECO points.
2. ECO tool execution to create the patch and prepare the incremental constraints for the subsequent tool in the design flow.

The ECO turnaround time is the sum of the above-mentioned portions. Now, let's break it down.

1. Turnaround time of LEC runs

   G2G equivalence check requires a significant amount of execution time. Additionally, preparing constraints and parameters to assist in LEC, as well as user diagnosis afterwards to eliminate false non-equivalences, are necessary. Iterations must continue until the designer no longer detects any false errors. Only at this stage can the designer assume that all non-equivalent results in LEC are related to ECO.

   However, if any non-equivalent results exist that are not directly related to functional ECO, they will be treated as required changes, resulting in unnecessary patches.

   Furthermore, in a G2G comparison, the equivalence check might result in unexpected Abort states, as the synthesized netlist can be significantly different from the original netlist in the datapath due to the revised function.

2. ECO tool execution time

   The execution time for ECO can range from a few hours to a few days for a sub-system in the ASIC. Typically, the tool execution time increases exponentially as the size or complexity of the ECO design increases. This is because the tool faces difficulties in balancing functionality, timing calculations, and resource limitations. As a result, the tool often returns a failure if the design fails to meet the timing or resource requirements.

   It is understandable that users feel deep frustration when the result comes back empty.

Benefits An Effective Functional ECO Solution Should Offer

Based on previous discussions, a functional ECO methodology should address these issues effectively:

1. False ECO points generated by G2G comparison.
2. Uncontrollable long turnaround time.

In other words, a functional ECO methodology should offer the following benefits (Figure 4):

1. Result accuracy:
   Pinpointing required functional changes precisely and developing the patch without introducing unnecessary circuitry due to erroneous ECO points.

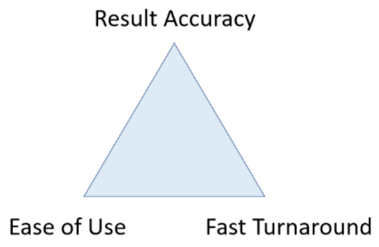


Figure 4: Required deliverables of a functional ECO methodology

2. Ease of use:
   Minimizing required user knowledge of the design, reducing manual efforts in implementing solutions, and facilitating intuitive human interactions during the ECO process.

3. Fast turnaround:
   Ensuring the shortest ECO turnaround time from the initial RTL design change throughout the entire logical and physical implementation process.

Thankfully, functional ECO methodology has improved remarkably over the last decade, driven by the rapidly evolving electronics market and growing demand for ECO solutions. Furthermore, the availability of specialized logic optimization technology tailored for ECO requirements has further fueled the progress in this field.

As a result, ECO designers benefit from this emerging technology, resulting in significant increases in both ECO turnaround time and success rate.

Solution: An RTL-Based Functional ECO Methodology

Advanced functional ECO technology, based solely on RTL-to-RTL comparison (Figure 5), offers the following five key benefits to help save engineering resources, reduce project time, and minimize required user involvement:

1. Higher accuracy for ECO points
2. Increased design freedom
3. Smaller size of the ECO patch
4. Preservation of the golden RTL design
5. Reduced total ECO turnaround time.

Now, let’s explore the details of each benefit.

1. Higher Accuracy in Identifying ECO Goals

   R2R comparison provides a more intuitive functionality comparison and an efficient approach in identifying ECO goals. It helps

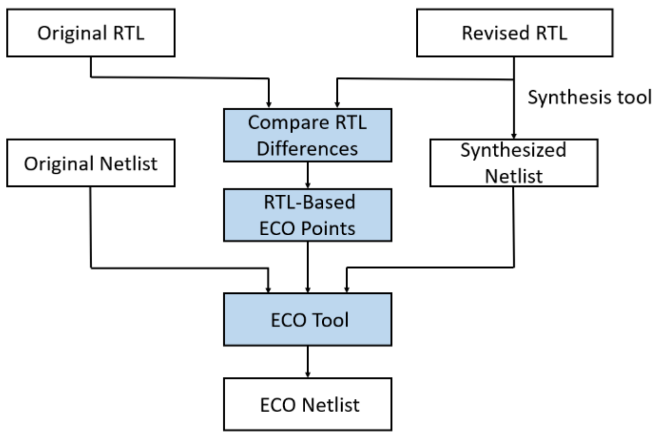


Figure 5: RTL-based functional ECO methodology

prevent the functional recognition errors that commonly occur with G2G and the false non-equivalent reports that may arise due to differences in synthesis strategies employed during netlist preparation.

2. Higher Degree of Design Freedom

   Logic optimization technology for implementing functional ECO solutions has undergone significant advancements. The technology has evolved to the point where it can precisely determine how an RTL signal is represented in the gate-level netlist. These advancements include the introduction of advanced algorithms, automated flows, and improved debugging capabilities.

   In the Verilog design example discussed earlier, designers can freely modify the function of *n1* at the RTL level without worrying about how it is optimized in the gate-level netlist.

3. Smaller ECO Patch Size

   The analysis based on RTL functional differences provides an accurate guideline for the ECO strategy. When combined with a comprehensive understanding of RTL mapping and optimization operations that transform RTL code into a gate netlist, the guideline becomes crucial for identifying ECO points downstream.

   Subsequently, the strategy is applied to optimize the patch logic, resulting in a reduced patch size.

4. Preserving the Golden RTL Design

   Maintaining the integrity of the ASIC design flow, especially the golden RTL code, is equally important in ECO design. Since the RTL-level code serves as the centerpiece of the entire digital design, many designers prefer to stay at the RTL level and may hesitate to venture into the unfamiliar territory of gate-level circuits. R2R comparison plays a key role in preserving the design flow.

5. Reduced Tool Execution Time and ECO Turnaround Time

   R2R comparison delivers a fast runtime while ensuring scalability. It is evident that the execution time of gate-level netlist comparison is significantly longer than that of RTL, especially when dealing with large design sizes. In equivalence checks, the execution times of R2R are typically 10 times faster or more compared to G2G, although this may vary depending on the specific design's characteristics. In general, R2R equivalence checks complete within a few hours, even for complex modules, while G2G equivalence checks for the same module might take several days.

   The outcome of R2R comparison, which provides accurate and concise ECO points, further accelerates ECO tool execution. Due to the smaller resulting patch, the optimization time is reduced.

Conclusion

In summary, leveraging functional ECO is a strategic imperative for ASIC companies to enhance product competitiveness and drive higher profitability. By implementing RTL-based functional ECO flow, EasylogicECO enables ASIC designers to concentrate on the functional change itself, without the need to worry about netlist-level details. This approach not only enhances design productivity but also results in a more robust and competitive ASIC product.

About the author

Kager Tsai
Vice President of Technical Support, Easy-Logic Technology

Since joining Easy-Logic Technology in 2021, Kager has been instrumental in leading his team to offer top-notch technical support to users globally. With 18 years of experience in the CAD industry and has worked for companies such as SiS, MStar, MTK, and Cadence, Kager is proficient in the ASIC front-end tool flow, especially in formal verification and ECO-related applications. Kager also participated in the development of many advanced ECO tool features.