Easylogic *The Easy Way to ECO Success*

RTL-Based Full-Module Functional ECO Methodology

EasylogicECO White Paper
WP-RTL-01

Engineering Change Order (ECO) is an incremental design method that revises an existing ASIC design. Once a new RTL code changes the ASIC function, an ECO task will modify a minimum portion of the netlist including instances and connections, keeping most of the netlist intact, to match the new ASIC function. This process is called Functional ECO.

Functional change requests during any ASIC design stage have become much more frequent in recent years as the complexity of ASIC design keeps increasing, project cycles keep shortening, and more importantly, market demands keep evolving. Once a spec change is inevitable, re-spinning the design always causes project delay. A successful ECO task provides a shortcut for a design to quickly adopt the required changes. In a late design stage, ECO success can drastically reduce the project delay from a few months to only a few days, upholding same quality results and the integrity of design flow.

Despite the immense value ECO brings to the ASIC project, implementing a functional ECO is always a challenging task. This article discusses the issues associated with the current functional ECO solutions and suggests a new design methodology.

Introduction

Functional ECO can bring tremendous business value to the chip success. However, it is a notoriously sophisticated process itself. While designers are responsible for addressing all aspects of the ECO change, the complex nature of such changes sometimes exceed their ability to manage independently as the required change might traverse beyond what the RTL code change looks like. The complexity of functional ECO tasks scares away many project leaders; they jump to conclusion quickly that a design re-spin is necessary when a spec revision is requested.

From a technically perspective, a functional ECO process faces two major challenges –

1. Fixing the circuit across various stages of the design flow, and
2. Assessing the scope of design changes required for the existing circuit and implementing all necessary fixes

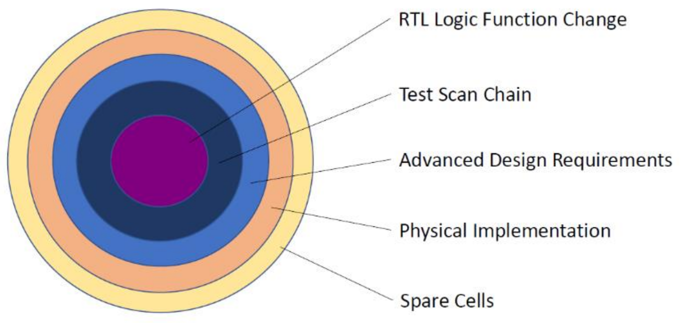

Figure 1: Ripple effect of the ECO-related design changes

A functional ECO task originates from an RTL code change. Depending on the stage design at which the ECO is requested, designers might need to address not only the design logic, but also the related implementations of scan chains, specific design requirements like low-power cells, P&R data with standard cells, and masks using spare cells if the design is already fabricated or close to completion. Figure 1 shows the types of design data that need to be fixed.

Another challenge the designers face is the extent to which an ECO change affects the design. A simple example is that an RTL modification is made to an IP that is instantiated multiple times in the chip, and the functional change occurs within the IP's interface portion. Now, this functional ECO task has become an unsettling experience.

This white paper will specifically focus on the second part, which entails assessing the extent of design change's impact and fixing them all.

Issues of the Common Functional ECO Approach in the Industry

Most designers follow these guidelines when they work on functional ECO tasks:

1. Gate-to-Gate (G2G) comparison to identify functional differences
2. Partitioning the design down to sub-module level for ECO operation
3. Enduring the unbearable long tool execution time and slow task turnaround

As stated above, the ECO process itself is a complex task. However, the root cause of ECO problems lies in the common design methodology used in the current ASIC design industry, which has not kept up with modern design practices. Outlined below, we will look at the issues associated with each of the guidelines.

(1) Issues with using G2G comparison to identify functional differences

Functional comparison based on 2 gate-level netlists (Figure 2), where Synthesized Netlist is generated from the revised RTL, is perceived as an easier approach to human eyes, and to be more logical, as the original netlist is the reference netlist. However, G2G-based ECO method is often accompanied by two major problems.

1. Signal tracing issues in the original netlist

This approach often creates false ECO points. The Synthesized Netlist may exhibit unexpected changes due to logic optimization during the synthesis process. Sometimes, the signal in RTL simply does not exist in the netlist anymore. Let's look at a simple example:

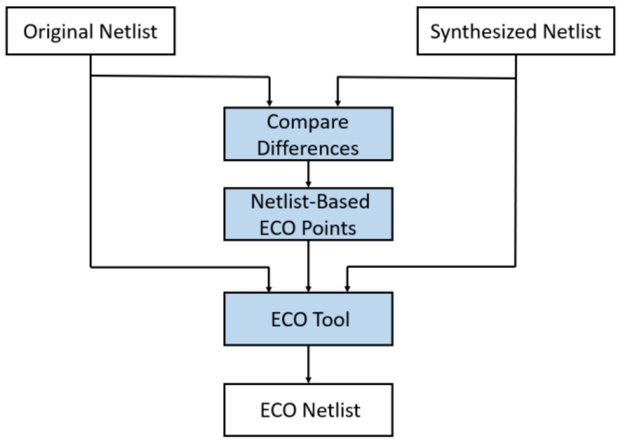

Figure 2: G2G netlist-based methodology

```
// 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, locating n1 signal in the gate-level netlist becomes challenging as the synthesis tool breaks down the signal to make the area smaller. The missing signal can be attributed to the following reasons:

1. The wire n1 in RTL is internal. That means it is up to the synthesis tool to "use" it in any way as long as the output functions are correctly composed. The synthesis tool may keep the signal n1 or decompose it into minterms for constructing the output functions, depending on the optimization goal.

2. No matter whether the signal n1 in RTL is kept, its name may be changed in the synthesized netlist. It would be hard for the engineer to trace the corresponding signal in the netlist either way.

The above-mentioned ECO task is to modify the function of the output: result based on the new specification of n1 in RTL. As a result, the designer may need to spend much effort to trace the signal paths driving all bits of the output:result to create the smallest feasible patch.

2. Large patch size

When either the Original netlist, or the Synthesized netlist, or both, have optimized out some RTL signals from the netlist, it can become a serious problem. The G2G equivalence check will likely create false discrepancies, thus false ECO points. The result of the ECO operation is an unnecessarily large patch circuit, which often fails to meet timing constraints.

(2) Issues with ECO using small sub-modules

To understand the terms used in the discussion, Figure 3 shows a full ASIC design and its building blocks. The top level of the design hierarchy is A (whole chip). The whole chip is divided into logical modules B – H, with B, C, D, E serving as system modules of A, and F, G, and H as sub-modules.

Module-based design approach enables the designer to develop the chip in a hierarchical fashion, where each module can be designed and tested independently. Due to considerations of efficiency and result quality, the size of system module is usually below 5 million logic units.

Partitioned blocks within a system module are typically referred to as either “sub-modules" or "blocks". Sub-modules are individual functional units that are divided within a system module to perform a specific function.

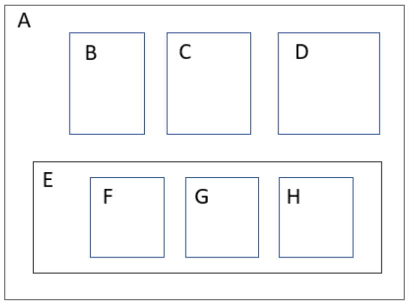

Figure 3: Example of ASIC design hierarchy

Most designers prefer to use a divide-and-conquer methodology when dealing with functional ECO tasks due to the high risks associated with G2G comparison, including potential false errors and long execution time. Designers perform the ECO task within each sub-module, which is sized based on the acceptable execution time for G2G comparison and the ECO tool time according to user’s preference.

Except the ECO cases that involve only simple yet localized changes, partitioned sub-modules undoubtedly created a huge challenge in terms of total ECO turnaround time, human involvement, and quality of result. The reasons are:

1. Extraction efforts for obtaining boundary constraints

The example below (Figure 4) demonstrates the challenges faced during the ECO task.

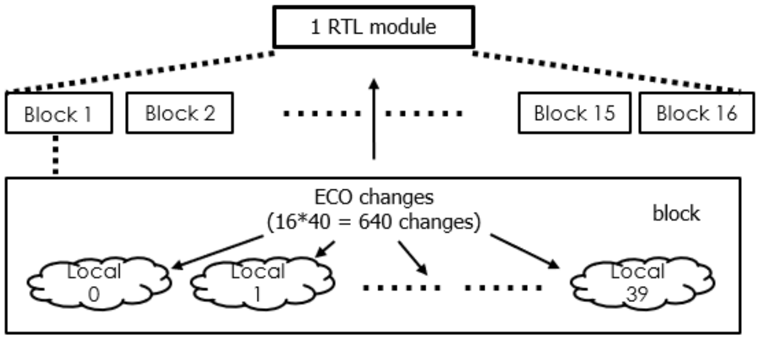

Figure 4: Example of the scope of ECO impact

An IP instantiated 40 times in a block, and the block is replicated 16 times in the module. One line of RTL code change in the IP needs to be carefully examined 640 times during the ECO task to make sure the design stays intact.

The challenge here is on the development of boundary constraints for each block involved in the ECO. Only with the correct boundary constraints for each block can we ensure that functional logic, DFT, and other design requirements, are not mishandled, leading to unnecessary modifications or even incorrect ECO results. Boundary constraints are developed from boundary conditions of the block, which include block interconnections (pin constraints), clock buffer tree, disable DFT, etc. Designers often make mistakes when manually extracting boundary constraints for each individual block.

2. Heavy human involvements

In addition to extracting boundary conditions for functional logic, designers often find themselves duplicating similar efforts across multiple design steps, such as formal verification, DFT, and others.

Table 1 below summarizes the Pros and Cons of performing ECO operations at sub-module level.

Sub-Module ECO Methodology	Analysis
Quality of result	Good if associated changes are within a sub-module
Execution time of ECO tool	Acceptable
Synthesis run time of revised RTL	Acceptable
User's design knowledge required	▪ Number of design blocks involved in the ECO ▪ Tracking design impact across block boundaries ▪ Boundary conditions of each block to identify
Developing block constraints	Tedious and time consuming, repeating the process for each sub-module with varying constraint conditions.
Formal verification	Tedious and time consuming, repeating the process for each sub-module with varying constraint conditions.

Table 1: Analysis of the ECO design work based on sub-modules

(3) Enduring the unbearable long tool execution times and total ECO turnaround time

On top of designer's manual work, the lengthy execution times of tools also contribute significantly to the total ECO turnaround time. As previously mentioned, tool execution times play a crucial role in the designer's decision on how small each sub-module should be. However, as each tool execution time decreases due to using smaller design blocks, it results in an increased amount of designer's manual efforts (Figure 5).

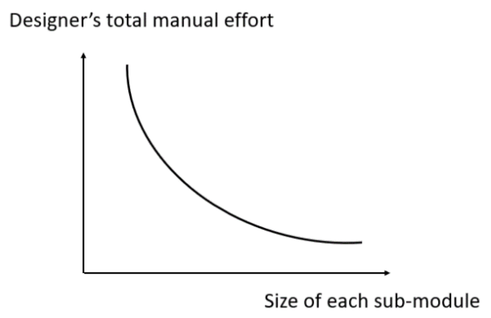

Figure 5: Trade-off between size of sub-module and designer's effort

Tool execution times include 3 major portions:

1. Synthesis of the revised RTL to generate the synthesized netlist,
2. Logical equivalence check (LEC) between original netlist and synthesized netlist to identify ECO points,
3. ECO tool to create the patch, as well as preparing the incremental instructions for the subsequent tool in the design flow.

Total ECO turnaround time is the sum of the above 3 portions, once the constraints for each block are successfully developed, multiplies the number of blocks that need ECO changes if not processed in parallel. Let's break it down.

1. Turnaround time of synthesis and LEC runs

Both RTL synthesis and G2G equivalence check collectively require a substantial amount of execution time. In addition, preparing constraints and parameters to assist LEC, and the user diagnosis afterwards to eliminate false non-equivalences, are required. Iterations must continue until the designer sees no more false errors. At this stage, the designer then assumes that all the non-equivalent results in LEC are ECO related.

However, if any non-equivalent results not directly related to the functional ECO exist, it will be treated as required changes and create unnecessary patches.

2. ECO tool execution time

The ECO execution time ranges from a few hours to a few days for a sub-module. Normally the tool execution time increases exponentially when the size, or the complexity, of ECO design increases as the tool struggles to balance among result functionality, timing calculations and resource limitations. Often, the result is returned as a failure, as it doesn't meet the timing or resource requirements. The deep frustration users feel when the result comes back empty is easily understandable.

Analysis

Drawing from the aforementioned discussions, a contemporary Functional ECO methodology should effectively address these issues:

1. False ECO points generated by G2G comparison,
2. Dividing full module into small blocks for ECO,
3. Uncontrollable long turnaround time.

In other words, a Functional ECO methodology should offer the following deliverables (Figure 6):

1. Result accuracy:

Pinpointing required functional changes precisely, and developing the patch without adding unnecessary circuit due to erroneous ECO points

2. Ease of use:
   Minimizing required designer knowledge across the whole design, manual efforts in solution implementation, and human interactions in the ECO process

3. Fast turnaround:
   Delivering the shortest ECO turnaround time starting from RTL design change through the entire logical and physical implementation process

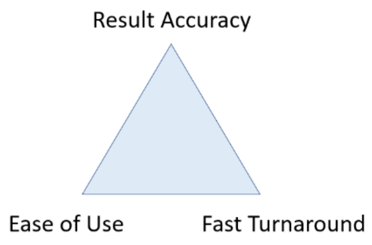

Figure 6: Required solution deliverables

Thankfully, functional ECO methodology has made remarkable improvements over the last decade, driven by the rapid-moving electronics market, increased demand for ECO solutions, and the proactive response from EDA vendors. Additionally, the availability of specialized logic optimization technology tailored for ECO requirements has further fueled the progress in this area.

Notably, logic optimization technology for implementing functional ECO solutions has undergone significant advancements, including the introduction of advanced algorithms, automated flows, and improved debugging capabilities. As a result, ECO designers benefit from the emerging technology, and ECO success rate has increased considerably.

Solution: Insights into modern ECO technology at the forefront of the industry

Advanced functional ECO technology offers the following three key features to help save engineering resources, reduce project time, and minimize required user involvements:

1. Functional comparison solely based on RTL,
2. Module-based ECO operation without requiring sub-module partitions,
3. Short tool execution time and reduced total ECO turnaround.

These are the details of each feature, let’s get a closer look.

(1) Why is RTL-based functional ECO methodology a better solution?

RTL-level ECO design methodology (Figure 5), namely RTL-to-RTL (R2R) comparison, enables pinpointing functional changes with much greater precision than what G2G methodology can offer. RTL-level ECO flow offers these strong benefits:

1. Higher accuracy for ECO Points

   R2R comparison can avoid functional recognition errors that easily occur with G2G, and the false non-equivalent reports that may arise due to differences in synthesis strategies employed during netlist preparation.

   Furthermore, in a G2G comparison, the equivalence check might result in unexpected Abort states as the

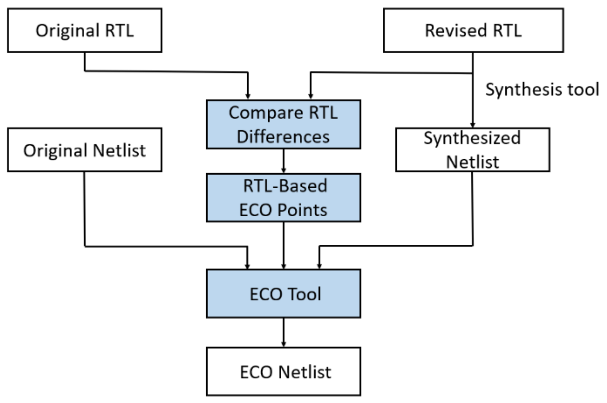

Figure 5: RTL-based ECO methodology

synthesized netlist might be significantly different from the original netlist in datapath due to the revised function. R2R comparison can be a more efficient approach in identifying ECO points, while G2G comparison often results in wasted time and effort.

2. Higher degree of design freedom

Modern ECO technology has evolved to being able to tell exactly how an RTL signal is represented in the gate-level netlist. On the Verilog design example discussed earlier, designers can freely change the function of n1 in RTL without worrying about how it is optimized in the gate-level netlist.

3. Smaller patch size

The analysis based on RTL functional differences provides an accurate action guideline for the ECO strategy. A comprehensive grasp of RTL mapping and optimization operations that transform RTL code into a gate netlist is crucial for identifying ECO points downstream, and subsequently in optimizing the patch logic, which results in a smaller patch size.

4. Maintaining the golden RTL design

Keeping the integrity of the ASIC design flow, namely the golden RTL code, is equally important to the ECO design. As RTL-level code is the center stage of the entire digital design, many designers prefer to remain at the RTL-level and may not venture into the unfamiliar territory of gate-level circuits. R2R comparison helps maintain the design flow.

(2) Benefits of module-based ECO operation

As tracking signal connections between sub-modules and extracting boundary conditions for each block are time-consuming and error prone to humans, a Module-Based ECO flow, without partitioning into sub-modules, eliminates the need for designers to spend excessive time and effort, while significantly improving the result accuracy of the results.

The development of specialized algorithms specifically tailored for functional ECO applications has significantly reduced the tool runtime, even for large designs. Revisiting the previous example shown in Figure 4, an automatic ECO solution can seamlessly perform ECO tasks on the entire module by successfully achieving these two critical success factors:

1. Tracking the extent of impact across 640 design blocks to identify the required changes, and
2. Extracting conditions for all required changes using an innovative logic synthesis and optimization algorithm.

The extracted conditions are converted into automatic ECO constraints, which are then utilized by the ECO execution to create an optimized patch for the entire module.

The same module-based operation applies to post-ECO verification. Performing equivalence check between the revised RTL and the ECO netlist on the entire module is recommended.

In conjunction with RTL-based functional comparison, Module-Based ECO flow presents an ideal approach to greatly reduce the challenges that designers face.

(3) Minimizing tool execution times and ECO turnaround time

The turnaround time for ECO tasks is of paramount importance. Different from a normal ASIC design process, functional ECO tasks are typically design-related assignments that come with predefined conditions and limited resources. As these tasks are subject to significant time pressure, but also have an unpredictable success rate, the operation time needed to complete an ECO process is strategically crucial in effectively performing these tasks.

After the design project transitions into the layout phase, the resources for functional ECO become limited, and the physical task becomes significantly more challenging compared to logical phase. The most challenging scenario is post-mask ECO, where success depends on the available spare resources around the physical vicinity of the ECO point in the layout. Furthermore, the netlist might have been optimized, or flattened, during the P&R process and becomes unrecognizable, adding another layer of difficulty to the ECO task. During this stage, the back-end layout plays a crucial role in the outcome of the ECO process, and designer's making minimal changes in the RTL code does not automatically guarantee a higher likelihood of success.

Table 2 below presents a compelling illustration of the comparison of various code changes in an ECO task.

The designer conducted meticulous experiments on four versions of RTL code change for an ECO task that yield an identical functional outcome. The finding revealed that Version 4 delivered the optimal ECO result, followed by Version 2, while version 2 and version 3 failed to meet the timing requirements.

It's important to highlight that the designer was unable to accurately predict the ECO outcome solely based on the RTL code change, underscoring the need for thorough evaluation to determine the most effective solution.

Revised RTL	Design Size (gates)	Patch Size (gates)	Recycled gates	Gate count change
Version 1	3815	280	98	+182
Version 2	3815	41	10	+31
Version 3	3815	678	199	+479
Version 4	3815	29	4	+25

Table 2: Experimenting with multiple RTL code changes for the best ECO outcome

In this scenario, the runtime of the tool becomes a critical determinant of the ECO success. The ability to quickly access whether the ECO is feasible, or nearing success, is essential. If an RTL code change doesn't yield the desired outcome, alternative approaches for RTL modification may need to be explored for the ECO task. Attaining an extremely short operation time is imperative to effectively accomplish this objective.

To deliver the shortest possible total ECO turnaround time, a few techniques can be deployed:

1. R2R for functional comparison

   Almost all ECO flows require re-synthesis to create a new netlist from the revised RTL. Since a comparison either R2R (revised RTL vs original RTL) or G2G (Synthesized netlist vs original netlist) must be performed to identify ECO points, it is obvious that the execution time of gate-level netlist comparison is much longer than that of RTL.

   When design sizes are large, the execution times of R2R are typically 10 times or more faster compared to G2G in equivalence checks, although the execution time depends on the specific design's characteristics, such as the number of registers and gates, the complexity of the logic, and the job environment setup, such as the tool deployed, and the constraints applied. In general, R2R equivalence checks typically complete within a few hours, even for complex modules, while G2G equivalence checks for the same module might take several days.

   R2R comparison delivers a manageable and fast run time, it also ensures scalability. The outcome of R2R comparison, namely accurate and concise ECO points, further speeds up ECO tool execution.

2. Trade-off between patch size and timing accuracy

The ECO algorithm should prioritize patch size over timing accuracy, striking a trade-off between those two factors. The approach brings significant values to designers:

1. Accessing the acceptability of the RTL change in meeting implementation requirements by utilizing a significantly reduced amount of time compared to full-blown timing-based optimization,
2. Allowing testing multiple variations of RTL change for the required function,
3. Minimal changes in the subsequent P&R steps.

There are 3 major techniques to help achieve the goal:

1. Optimization priority
   Reducing the number of gates in the patch significantly enhances the probability of achieving successful timing closure, particularly in an ECO process pertaining to layout. Meanwhile, prioritizing gate count optimization over timing accuracy can considerably reduce tool run time.

2. Timing estimation
   A well-designed timing estimation algorithm that guides the trade-off process is a key component for an efficient ECO process. The estimation creates a target range of logic levels suitable for the patch, the ECO optimization step then takes the estimation results as constraints.

3. Physical aware
   The ECO mapping and optimization algorithm carefully selects from the available types of available gates in the vicinity of the physical ECO point, creating the shortest logic path by balancing the trade-off between the levels of gate logic, resource functionality, and physical locations.

3. Generating the necessary instructions for the subsequent tools to operate effectively with

   As ECO process encompasses the entire ASIC design flow, from RTL code to GDS tape-out, it involves a variety of critical design tools and requires information exchanges with other tools (Figure 6).

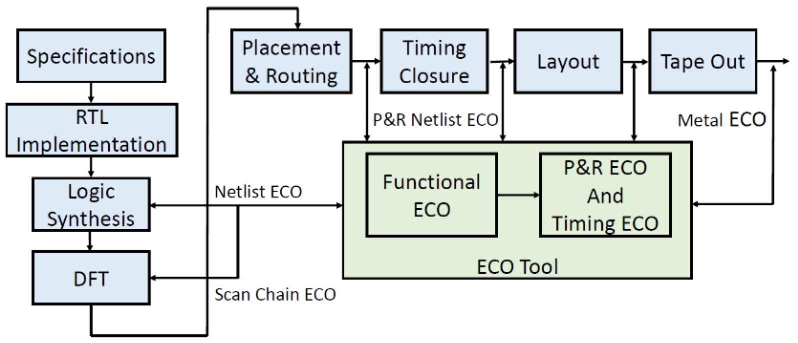

Figure 6: Required ECO data integration in an ECO flow

As most ASIC companies deploy a multi-vendor design flow to take full advantage of the strength of each individual design tool, closed-environment ECO solutions may not always be the preferred solution compared to standalone functional ECO tool. A standalone ECO tool has the following advantages:

1. Plug-and-play to preserve the user's investment of the current ASIC design flow,
2. Information exchanges through various standard formats reduce user’s efforts in overcoming the design flow integration challenges.

The ECO tool should always generate the action references for the subsequent design steps, such as ECO TCLs for formal verification and P&R tools, enabling users to easily convert the references to the constraints of downstream tools.

A tightly integrated automatic ECO design flow that seamlessly adapts to a mixed-vendor tool environment is crucial for achieving efficient ECO tasks and minimizing the total turnaround time.

Conclusion

After careful analysis of the problems at hand, functional ECO is a major challenge for designers who are using a traditional methodology. However, our study has shown that the emerging ECO technology has the right approach to overcome those challenges and bring about significant benefits for accurate ECO results and short turnaround time.

I believe that by implementing RTL-based full-module functional ECO flow, ASIC designers will be able to focus on the function change itself, which will not only boost their design productivity but also lead to a more vigilant and more competitive ASIC product. Furthermore, my analysis indicates that RTL-based full-module functional ECO methodology is a more robust and more cost-effective option, and it can be easily integrated into existing systems.

In summary, leveraging functional ECO is a strategic imperative for ASIC companies to enhance product competitiveness and drive higher profitability for the organization. Commercial ECO tools, such as EasylogicECO provided by Easy-Logic Technology, which utilize innovative ECO methodologies, are already accessible in the market. I highly advocate for ASIC designers to seriously consider implementing this new ECO design flow to optimize their design process.

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.