2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION JAPAN

Quality Driven Analysis of Clock Tree Network using "Accelerated Clock Reference Model Generator"

Tejas Dipakkumar Dalal, Samsung Semiconductor India Research, Bengaluru, India (tejas.dalal@samsung.com)

Giridhar S, Samsung Semiconductor India Research, Bengaluru, India (giridhar.s@samsung.com)

Jeevan Nataraju, Samsung Semiconductor India Research, Bengaluru, India (j.nataraju@samsung.com)

Garima Srivastava, Samsung Semiconductor India Research, Bengaluru, India (s.garima@samsung.com)

***Abstract*—Clock tree network (CTN) is one of the complex design structures in SOC that controls clock supply for all the synchronized logic on the chip. It constitutes numerous clock component instances to employ Dynamic Voltage Frequency Scaling (DVFS) and Clock Gating (CG) schemes, effectively contributing to overall performance and power optimization [1]. Ensuring verification of such a network with legacy methodology is demanding, inconsistent and mostly restricted. This paper presents a holistic verification strategy for CTN with clock reference model (CRM) that encompasses all the clock components of the network under verification. The CRM generation is automated using "Accelerated clock reference model generator" (ACRMG). It is an in-house developed tool, modelled with a custom algorithm to trace clock tree path for each IP in SOC and generate the CRM. It is a single click solution that makes it adaptable for frequent design changes during the chip development cycle. The proposed solution is validated with mobile and automotive SOCs having multi-die/chiplet-based architecture. With ACRMG, the overall TB environment setup time is reduced from 2 days to less than 1 day resulting in faster bug resolution.**

Keywords—System On Chip(SOC); Clock Tree Network (CTN); Clock Reference Model (CRM); Accelerated Clock Reference Model Generator (ACRMG);

I. INTRODUCTION

The world is advancing towards rapid growth and development. With technology fueling a large part of this global movement, the demand for System On Chip (SOC) driving these numerous technologies has also increased. This swift progression has a strong impact on SOC development cycle, with time to market being very crucial to meet this ever-increasing demand. The SOC development has also witnessed rapid advancement in design to meet the Power-Performance-Area (PPA) requirements. This has resulted in increase of design complexity demanding innovative solutions and optimized approach to expedite the verification process, while ensuring better quality and reliability of the product.

The Clock Tree Network CTN in an SOC essentially spans across the chip as it drives all the synchronized logic of the digital design. Its scale, constituting components, along with its diverse functional features, collectively classify it as one of the complex and critical designs in any SOC. Application of conventional verification strategies to CTN is restricted due to its complexity. Further, it is also demanding as it would be time consuming, and tiresome to consider frequent design change for PPA optimizations, and is largely prone to human errors. Achieving signoff metrics on the other hand, is a key criterion to ascertain the design quality of the CTN. As any bug in clock tree network can propagate to multiple subsystems/IPs, or affect data/control path operation, it is critical to qualify all the associated clock components to gain confidence. Though assertion based functional coverage act as a key contributor in accomplishing the signoff metrics, these comprehensive coverage reports are mostly available towards the end of verification cycle with a central regression. Any coverage hole detected at this stage can largely hinder the overall chip development cycle, and additionally would require extended effort for analysis and timely closure of the same. Automation has helped in achieving verification of such complex designs effortlessly. The proposed solution emphasizes the significance of automation in verification of SOC Clock Tree Network (CTN).

The paper presents CTN design overview, verification complexity and operational flow of the proposed solution. It discusses the limitations of conventional methodology, and enhancements employed to overcome the same. The results quantify the process improvement achieved relative to legacy flow.

II. ARCHITECTURE

Clock Tree Network (CTN) is primarily comprised of clock source, router (multiplexer, divider), Q-Channel Manager (QCH _MNGR) [2] and clock gate components to supply clock to all the IPs in SOC. The clock source and routers together contribute to Dynamic Voltage Frequency Scaling (DVFS). The QCH_MNGR and the clock gate components contribute to hardware (HW) and software (SW) controlled power optimization respectively. An example of CTN is shown in Figure 1. Here, the clock components within a subsystem (SS) driving all its IPs are collectively referred to as Clock Control Unit (CCU). Such CCUs for each SS, together constitute SOC CTN.

The same design can be interpreted as a graph with nodes and arcs representing clock components and their interconnections, respectively as shown in Figure 2. A premium class mobile SOC roughly holds more than 3000 clock component units [1]. Hence, at the scale of SOC, this graph is a complex network comprising multiple thousands of such nodes. This topology is subjected to frequent change for PPA and timing optimizations during the development cycle.

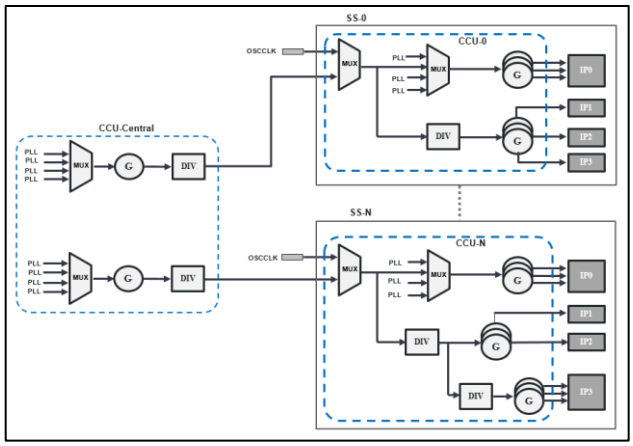

Figure 1. Example Clock Tree Network

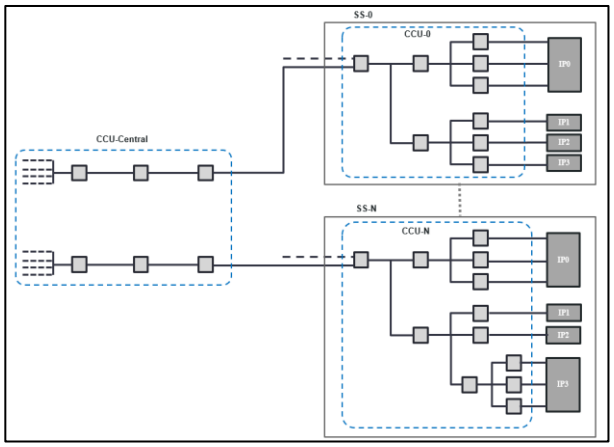

Figure 2. Topological representation of Clock Tree Network

III. METHODOLOGY

A. *Limitations with conventional flow*

The conventional verification procedure for CTN is to develop verification environment analyzing the specification manually with limited automation. The verification environment briefly includes assertions and clock frequency checkers. This flow was determined to be an inconsistent approach restricting thorough CTN verification and largely hindering verification cycle due to following limitations that compromises verification quality.

- Multiple steps to sparsely generate CTN TB files, further stitched manually to enable verification. This process typically consumed 6hrs for every RTL release.
- Need for extended manual effort, consuming roughly 2 to 3 days for CTN TB environment setup with sanity runs.
- Bug resolution turnaround time (TAT) was typically 2 RTL release cycles due to delayed CTN TB development as indicated in Figure 18.
- Assertion checker enablement at later stage of verification cycle due to complex network and extended manual effort for binding with RTL.

B. *ACRMG Flow*

Accelerated Clock Reference Model Generator (ACRMG) is a tool developed in-house to automate CRM generation by analyzing every branch, node and associated feature of CTN. A high-level execution flow is illustrated in Figure 3.

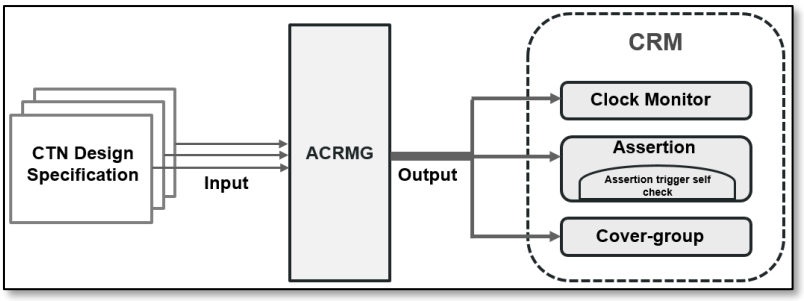

Figure 3. Accelerated Clock Reference Model Generator execution flow

The CTN design specification given as an input to ACRMG, encapsulates the network component details, connectivity, supported features and configurations. The tool parses this specification and processes the data with custom algorithm by analyzing each path from clock source to destination IP in CTN. The analyzed data including component characteristics, inter-component relation and all the associated feature information, is then translated to binds, assertions and cover-groups mapping every component to an associated TB feature checker.

C. CRM Components

CRM generated by ACRMG encapsulates the following TB components as indicated in Figure 3, to ensure rigorous check and verification completeness of the CTN:

- Clock Monitor: It monitors the target IP frequency for all supported DVFS corners and clock gating.
- Assertion checkers with assertion trigger self-check: The assertions ensure functional integrity of each CTN component, and the assertion trigger self-check is a custom logic integrated within assertion property to monitor if the same is covered during simulation. This helps in faster assertion coverage closure.
- Cover-groups: The cover-groups ensure all associated features of the CTN are covered.

The generated checkers and cover-groups together ensure verification quality. The custom algorithm accelerates this overall flow to complete analysis and generation instantaneously narrowing down the delay between RTL release and CRM setup.

D. ACRMG Structural Implementation

ACRMG structural implementation is shown in Figure 4. The overall flow is categorized into two steps:

a) STEP_1: Data processing and refinement

The CTN design specification given as input to ACRMG is a collection of different clock related specification files (in different format). These specifications are parsed and processed by the tool in to extract and consolidate:

- Clock component characteristics
- Inter component relation
- Associated feature information (like sign-off frequency, IP clock list, etc.)

b) STEP_2: CRM generation with processed data

The intermediate refined data is stored as a processed data package that is read to fetch relevant information to generate the CRM components. The tool is enabled with user input to control different phases of the flow such as data processing, intermediate data selection and the CRM generation.

As indicated in Figure 17, the total time in parsing and generating CRM for a mid-range mobile SOC is less than 4 minutes. For any incremental specification update, associated CRM TB files are re-generated accordingly.

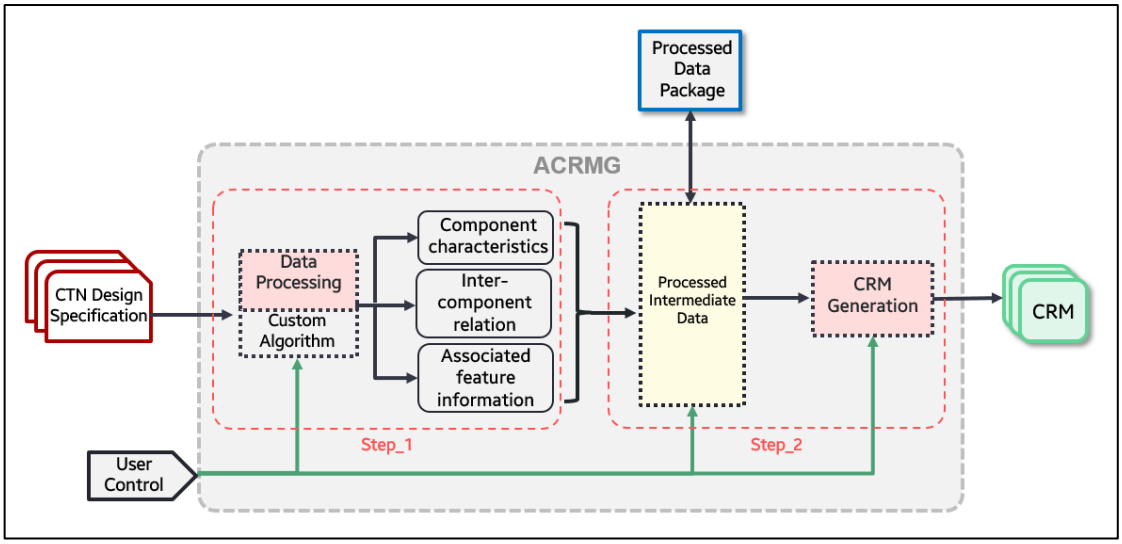

Figure 4. ACRMG Structural Implementation

E. *Assertion Trigger Self Check*

Assertion trigger self-check logic is integrated with every assertion instance during generation. As shown in Figure 5, a counter initialized to zero is included with every assertion instance. The assertion checks are enabled based on the feature under check during simulation. In run phase, the counter is incremented every time its associated assertion evaluates to true. At the end of run phase, the test is converged to fail if the counter value is zero, indicating that the intended assertion did not evaluate to true even once in the simulation. This avoids vacuous pass scenarios, which converge as an assertion coverage hole if not detected. This scheme ensures that all intended features are validated for a given scenario.

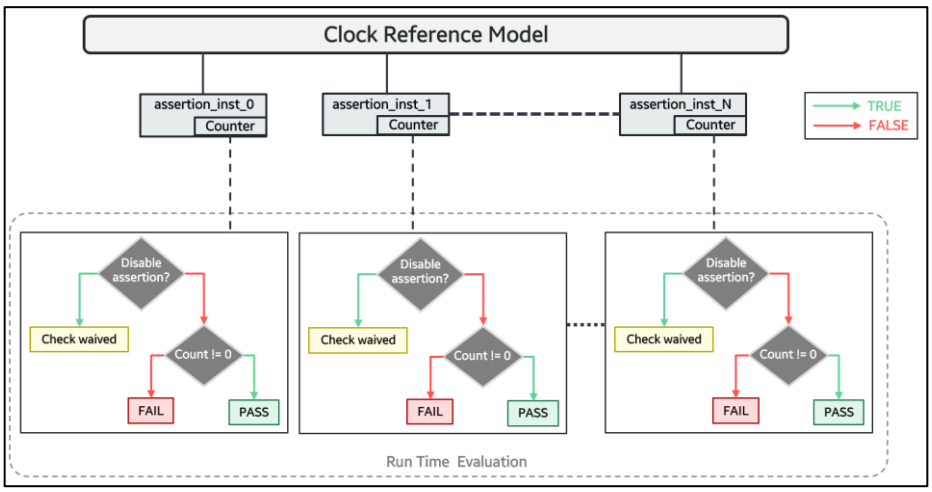

Figure 5. Assertion trigger self-check

F. *Property waiver auto translation*

As indicated in Figure 5, certain assertion checks can be disabled to waive the check. These are exceptional properties identified and waived based on their applicability, post assertion trigger self-check failure analysis. The waived properties are captured in a file in user readable format. As shown in Figure 6, this waiver file is auto translated to coverage refine format that is directly fed as an input to the coverage tool to exclude these pre-reviewed waivers. This prevents the need for multiple reviews of assertion coverage hole if any as the same would be reviewed and waived as part of assertion trigger self-check failure analysis. Figure 7 indicates the assertion coverage data achieved with first set of regression without any review for a mid-range mobile SOC. With the generated refine file from this flow, the assertion coverage is expected to be 100%.

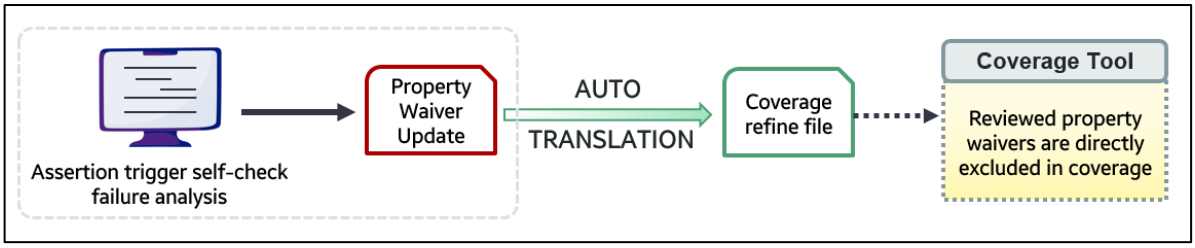

Figure 6. Property Waiver Auto Translation

IV. RESULTS

The holistic approach effectively contributed towards improving verification quality by catching functional bugs and network connectivity issues early in the verification cycle. The following sections briefly discuss the overall process improvement achieved with the ACRMG flow.

A. Comprehensive Coverage

As shown in Figure 7 and Figure 8, nearly 22K+ assertions, 10K+ cover-groups and 2K+ clock monitors are generated for a mid-range mobile SOC CTN using ACRMG. These numbers proportionally increase to 42K+ assertions, 20K+ cover-groups and 4.5K+ clock monitors for a large-size automotive SOC as shown in Figure 9 and Figure 10. The assertions and cover-groups indicate the design complexity with the number of clock component instances employed in CTN. The clock monitor instances generated correspond to the target IP clocks being monitored.

Both the indicated coverage reports are generated from first set of regression result without any review, with nearly 100 uncovered functional bins in mobile SOC, and nearly 480 uncovered functional bins in automotive SOC that are feasible to review.

A.1 Mid-Range Mobile SOC

Overall	94.94%	20042 / 24104 (83.15%)
Code	n/a	0 / 0 (n/a)
FSM	n/a	0 / 0 (n/a)
Functional	94.94%	20042 / 24104 (83.15%)
Assertion	94.94%	20042 / 24104 (83.15%)
CoverGroup	n/a	0 / 0 (n/a)
FaultNode	n/a	0 / 0 (n/a)

Figure 7. Assertion coverage of mobile SOC

Overall	99.03%	10517 / 10617 (99.06%)
Code	n/a	0 / 0 (n/a)
FSM	n/a	0 / 0 (n/a)
Functional	99.03%	10517 / 10617 (99.06%)
Assertion	n/a	0 / 0 (n/a)
CoverGroup	99.03%	10517 / 10617 (99.06%)
FaultNode	n/a	0 / 0 (n/a)

Figure 8. Functional coverage of mobile SOC

A.2 Large-Size Automotive SOC

Overall	96.46%	42276 / 49657 (85.14%)
Code	n/a	0 / 0 (n/a)
FSM	n/a	0 / 0 (n/a)
Functional	96.46%	42276 / 49657 (85.14%)
Assertion	96.46%	42276 / 49657 (85.14%)
CoverGroup	n/a	0 / 0 (n/a)
FaultNode	n/a	0 / 0 (n/a)

Figure 9. Assertion coverage of Automotive SOC

Overall	98.59%	20859 / 21329 (97.8%)
Code	n/a	0 / 0 (n/a)
FSM	n/a	0 / 0 (n/a)
Functional	98.59%	20859 / 21329 (97.8%)
Assertion	n/a	0 / 0 (n/a)
CoverGroup	98.59%	20859 / 21329 (97.8%)
FaultNode	n/a	0 / 0 (n/a)

Figure 10. Functional coverage of Automotive SOC

B. *Process Improvement with Assertion Trigger Self-Check*

Figure 11 illustrates the assertion coverage achieved without assertion trigger self-checks for a mid-range mobile SOC, with nearly 1/4th of the assertions uncovered. The coverage data is retrieved with first set of regression without any review. In order to review the uncovered bins, the assertion trigger self-check logic was integrated to the TB and all the scenarios were rerun with a second regression. The logic was able to identify and report all the vacuous pass scenarios converging a set of test cases to fail which were previously converged to pass. Post failure analysis, certain test sequences were updated to target and cover a set of uncovered properties and the rest were identified to be not applicable due to exceptional cases. The assertion coverage achieved with second regression post failure analysis was 94.47% as shown in Figure 12. The relative coverage improvement of nearly 22% indicates the process improvement achieved with the assertion trigger self-checks. It ensures that all the associated assertions are covered during simulation, without allowing the same to converge as a coverage hole towards the end of verification cycle.

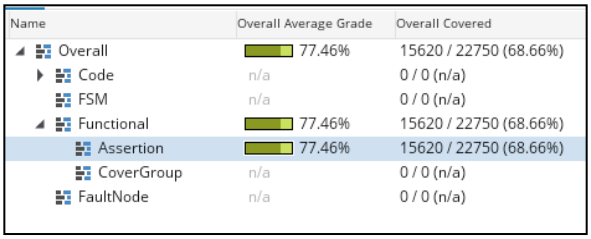

Figure 11. Assertion coverage of mobile SOC (without assertion trigger self-check)

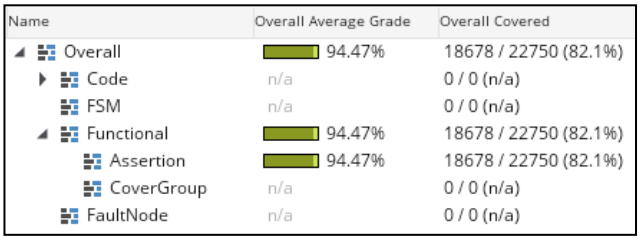

Figure 12. Assertion coverage of mobile SOC (with assertion trigger self-check)

C. *Coverage closure*

The uncovered assertion properties shown in Figure 7 and Figure 9 are pre-reviewed waivers in simulation that are auto-translated to coverage waivers by ACRMG. The small number of uncovered functional bins as shown in Figure 8 and Figure 10 are feasible to review. Both, the assertion and functional coverage converge to 100% post review as indicated in Figure 13 and Figure 14 for Mid-Range Mobile SOC, and in Figure 15 and Figure 16 for Large-Size Automotive SOC.

C.1 Mid-Range Mobile SOC

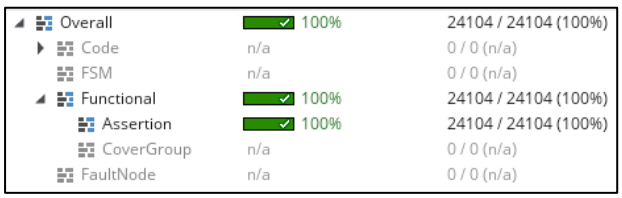

Figure 13. Assertion coverage of mobile SOC

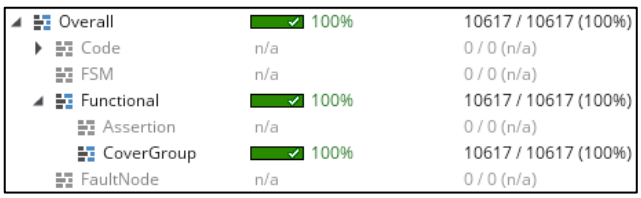

Figure 14. Functional coverage of mobile SOC

C.2 Large-size Automotive SOC

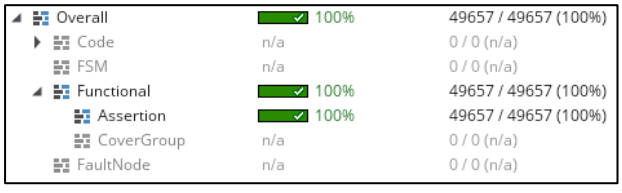

Figure 15. Assertion coverage of Automotive SOC

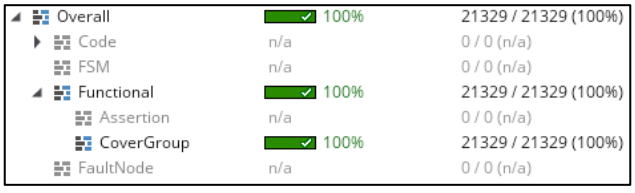

Figure 16. Functional coverage of Automotive SOC

D. Accelerated Generation

The overall CTN TB generation time with conventional flow was nearly 6 hours with extended manual effort to integrate sparsely generated TB components. With ACRMG, the analysis and the CRM generation time was less than 4 minutes for a mid-range mobile SOC CTN as shown in Figure 17, demonstrating the efficiency of the tool and the custom approach. This extensively eased TB generation for every RTL release cycle and the quick generation time enabled DV to reproduce the CRM for any design change instantaneously.

```
Time taken =  0:03:48.563217
```

Figure 17. Clock Reference Model generation time (hh:mm:ss)

E. Reduced Turn Around Time (TAT) for bug resolution

As indicated in Figure 18, general DV execution flow involves an RTL release followed with verification release a day after for DV to execute, with typical RTL release cycle of one week. Post verification release, CRM generation and CTN TB environment setup is done for DV to enable execution. With conventional flow, which typically takes 2 to 3 days, bug detection and reporting gets delayed. This in turn results in missing the bug fix window for every RTL release cycle. Consequently, the reported bugs are fixed in alternate releases delaying the verification timeline. This issue was overcome with ACRMG to limit the CTN environment setup time to less than one day with CRM generation, reducing the bug resolution TAT to 1 RTL release cycle where the bugs detected are fixed in the immediate next RTL release.

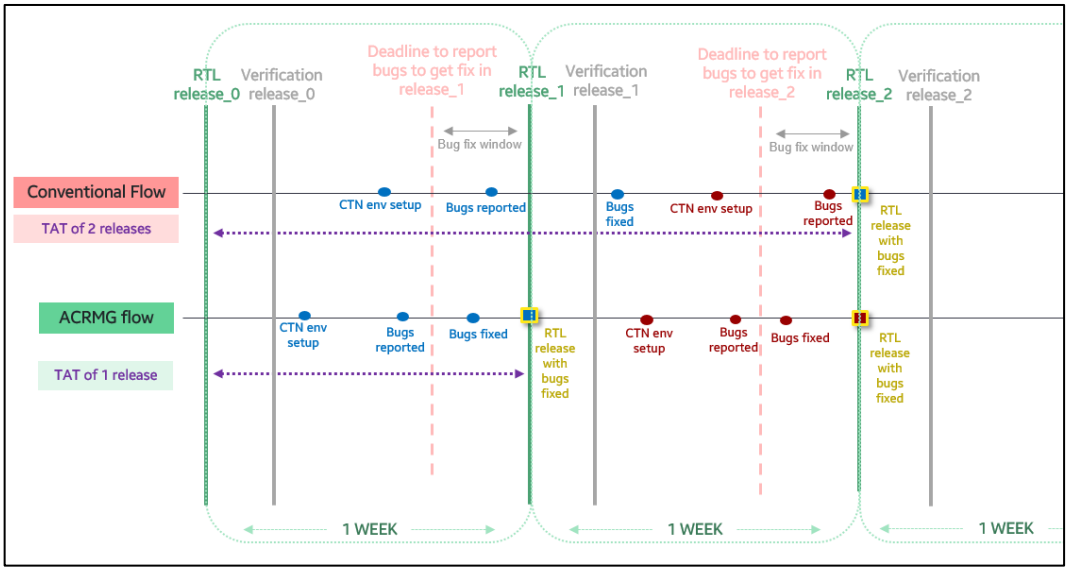

Figure 18. Bug resolution turnaround time

F. Architectural Bug Findings

ACRMG integration and flow helped in updating CTN design architecture by finding critical bugs as listed below:

a) Connectivity mismatch affecting thermal management was detected as part of assertion trigger self-check failure analysis.
b) CTN component dependency misses were identified which affected clock wake-up latency on critical data paths.
c) The CTN component interconnectivity issues were identified early in verification cycle as component specific assertion checks were part of CRM from the start.

V. CONCLUSION

With increasing design complexity and demand for SOCs to drive diverse applications like smartphones, automotive, etc., it is very critical to cut-down the chip development cycle without compromising on the performance and quality. The proposed ACRMG solution adapts to this swift progression that demands innovative techniques and methodologies. It demonstrates the importance of automation in verification of SOC CTN, and the limitations of the traditional approaches.

The proposed approach accelerates the CTN design specification analysis and the CRM generation flow. The CRM is structured with robust TB components (clock monitor, assertion modules and functional cover-groups), integrated with enhancements such as assertion trigger self-check and auto waiver translation. This ensures qualitative analysis of the design with a comprehensive verification strategy.

ACRMG was integrated and validated for mid-range mobile SOCs and large size automotive SOCs. The results presented indicate substantial process improvement easing CTN verification with CRM development and assertion coverage, early bug detection and reduced TAT of one release label for bug resolution. This approach could be applied to any CTN topology as long as it can be viewed as a graph to utilize the automation benefits of ACRMG.

REFERENCES

[1] J. -G. Lee, Y. Choi, H. Jeon, J. -J. Lee and D. Shin, "Fully Automated Hardware-Driven Clock-Gating Architecture With Complete Clock Coverage for 4 nm Exynos Mobile SOC," in IEEE Journal of Solid-State Circuits, vol. 58, no. 1, pp. 90-101, Jan. 2023

[2] AMBA Low Power Interface Specification, ARM Standard IHI 0068C (ID091216)