An Automated Method for Adding Resiliency to Mission-Critical SoC Designs

Author

Mary Ann White
Product Management Director

Abstract

Adding safety measures to system-on-chip (SoC) designs in the form of radiation-hardened elements or redundancy is essential in making mission-critical applications in the Aerospace and Defense (A&D), cloud, automotive, robotics, medical, and Internet-of-Things (IoT) industries more resilient against random hardware failures that occur. Designing for reliable and resilient functionality does impact semiconductor development where these safety measures have generally been inserted manually by SoC designers. Manual approaches can often lead to errors that cannot be accounted for. Synopsys has created a fully automated implementation flow to insert various types of safety mechanisms, which can result in more reliable and resilient mission-critical SoC designs.

This paper discusses the process of implementing the safety mechanisms/measures (SM) in the design to make them more resilient and analyze their effectiveness from design inception to the final product.

Introduction

With SoC designs, it is important not only to show that design functionality is correct, but also that the design works properly in the context of the system and is safe over its targeted operational lifetime. Many different types of faults or failures can be encountered in the various stages of the semiconductor life cycle, whether they be systematic or random types that can show up in either manufacturing or during the in-field operation stage where piece parts have been deployed in production use in various mission-critical applications.

Systematic failures are permanent types of faults that can be found and corrected during the IC development process. For example, design for manufacturing (DFM) tools such as physical verification can easily identify signal shorts or opens that may have inadvertently made it into the design. Functional simulation with a robust test bench might identify safety-related issues with the design. In both cases, the design issues can be fixed, even before the design has made it to silicon, making the SoC more reliable.

In addition, it is essential to ensure that electronic design automation (EDA) tools used to design the IC will not introduce safety issues that can cause systematic failures. A robust tool qualification and assessment process should be followed to ensure that the tools are "safe" to deploy for mission-critical semiconductor design.

But what about SoC designs out in the field that might fail due to unexpected events which can cause temporary malfunctions? Semiconductor devices must be resilient to these types of failures by either recovering from or mitigating them.

Reliability vs. Resiliency

There is a distinct difference between the concepts and objectives of reliability and resiliency, especially for semiconductor devices. Reliability refers to the ability of the SoC to operate properly (e.g., no failures or silicon device degradation) over a long period of time. In contrast, resiliency refers to the ability of the semiconductor device to “recover” from unexpected failures, such as single event upsets (SEU) caused by solar flares. In both cases, any malfunction of the SoC design can have serious consequences, especially in mission-critical designs.

Electronic IC designs should be reliable for a certain operational lifetime. For example, automotive devices should be reliable for a minimum of 10 years in order to be ISO 26262 compliant. Aging models and silicon failure rates can be considered during the development phase to account for down-the-road aging effects. In addition, the design methodology can deploy the use of DFM, test, and simulation EDA tools for reliability by improving the manufacturing process, and through the performance of rigorous testing and validation.

However, random hardware failures from SEUs, like alpha particles and radiation from solar flares, must be accounted for during the design process to ensure the devices are resilient. This is where the notion of designing in safety measures or safety mechanisms is essential for mission-critical designs to make them more resilient by minimizing the impact and offering protection from unexpected SEUs.

SMs are inserted as “backup” systems using redundant components, enabling rapid and automatic recovery to ensure that critical operations are protected. These SMs allows the in-field SoC design to recover from any unexpected SEUs and remain fully operational.

SMs, generally in the form of redundancy or radiation-hardened elements, added to SoC designs can monitor and detect the occurrence of random faults and help the system reach a safe state if a fault manifests. Additionally, these SMs can often be used to mitigate attacks that cause targeted faults.

Redundant Systems for SoC Designs

Random hardware faults represent uncontrollable events like SEUs, which can cause glitches or loss of state. They can’t be outright eliminated, so their risks must be mitigated through redundancy and other SMs.

There are several types of redundancy that can be inserted into the design. In general, dual modular redundant (DMR) systems will perform error detection where recovery back to a safe state or operation will need to be designed in. However, a triple modular redundant (TMR) system, will perform error correction so that devices will continue to operate in the event of an SEU failure.

Typical DMR and TMR systems implemented in semiconductor designs can be applied to different components such as flip flops, cores, and finite state machines (via parity encoding). For mission-critical designs, the use of radiation-hardened dual interlocked storage cell (DICE) flops is prevalently in use. DICE flops contain redundancy at the transistor- rather than the cell-level.

Design Implementation Challenges

Implementation of redundant-type SMs for semiconductor device resiliency is not new; however, insertion of them has mostly been a manual process. The challenge is not only to insert the redundant elements in manually, but also they have physical placement constraints, such that they are placed a certain distance apart so that SEUs will not affect the all of the redundant elements. In addition, routing considerations must be accounted for so that cascading failures or common mode interference (CMI) will not affect reset, power, or clock network signals.

These manual methods are not very scalable in that they typically require hundreds, if not thousands, of lines of scripting that often need to be customized for each project. In addition, manual insertion of the SM redundant elements can affect tradeoffs in power, performance, and area (PPA) goals where it may be difficult to analyze the effects.

Synopsys synthesis and physical implementation tools have a fully automated approach to inserting the SMs to make mission-critical designs much more resilient. Synthesis can automatically insert the elements while the place and route (P&R) tool will take care of the physical implementation challenges such as placement distance and routing independence of signal nets.

EDA Flow for Automated SM Insertion

SMs are captured using a Safety Specification Format (SSF), informing the Synopsys EDA tools which SMs are to be inserted, analyzed, and verified in the mission-critical design. A comparison of a typical implementation flow and the automated Synopsys implementation flow that inserts safety mechanisms/measures is shown in Figure 1 below:

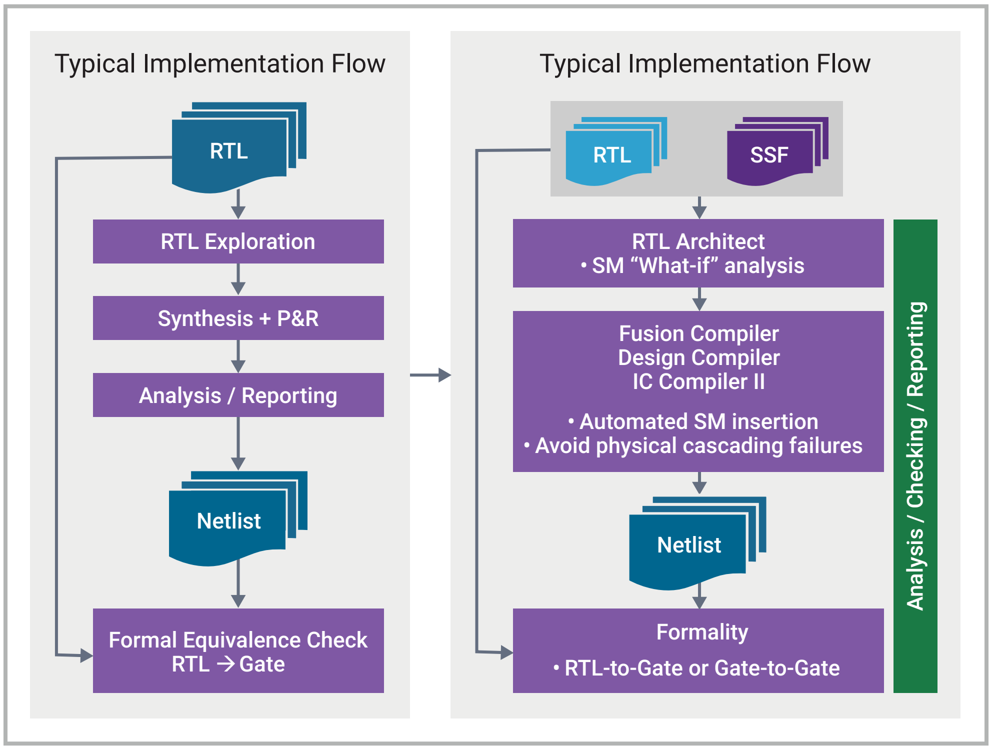


Figure 1: Safety Flow for Resilient Designs

The need for SMs can be identified, analyzed, inserted, and then verified using the automated safety flow. Examples of redundant SMs that can be automatically implemented include the insertion of dual core lock step (DCLS), and dual- or triple-modular redundant (DMR/TMR) registers. Equally important is that the physical implementation tools should be aware of and mitigate potential cascading failures such that reset, power, or clock networks should properly be isolated.

A. Automated Safety Register Redundancy Insertion

There are several ways of implementing redundancy on flip-flops, and the one chosen depends on tradeoffs of resiliency vs PPA goals.

The most resilient designs rely on TMR flops where three identical flip-flops are connected in parallel. Their output is combined into a majority voting mechanism to determine the correct output. Essentially, data is duplicated through these three flops where single-bit errors that can occur through SEUs (e.g., noise or radiation interference), are detected and corrected. In this case, it is highly unlikely that the SEU will affect all three flip-flops, so the failure is mitigated, and the design operates normally. Of the various strategies, this consumes the most silicon area.

The second alternative is using dual-mode redundancy (DMR); however, this is useful only for error detection and not correction. Other logic to determine what to do when an error occurs would need to be implemented.

Used in many mission-critical designs, critical paths can simply be hardened with fault-tolerant registers to reduce the chance of failure. In this case, a fault is neither corrected nor detected but is simply made much less likely to succumb to an SEU failure. This consumes the least silicon area of the alternatives since redundancy is built in at the transistor level; but statistically, it offers lower resiliency against an SEU vs. TMR flops, especially for advanced smaller geometry processes.

Another newer approach is to apply parity error protection schemes to groups of registers. This would offer a smaller area vs DMR and TMRs; however, the tradeoff could be operation at a slower frequency. Synopsys synthesis tools, such as Design Compiler® or Fusion Compiler™, can automatically insert any of these register-based SMs noted in Figure 2.

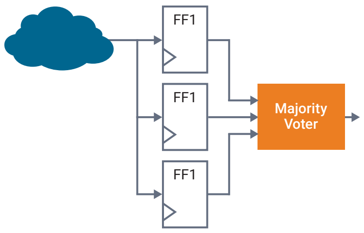
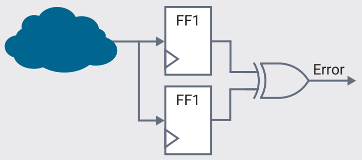
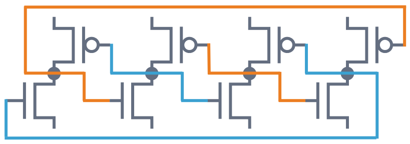
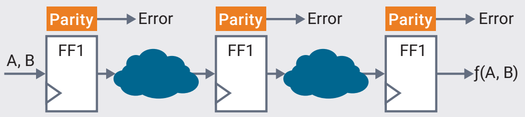
Safety Register Strategy	Description	Design Example
Triple Modular Redundancy (TMR)	• Three registers sample the input state • Majority voting logic • Output is self-corrected	
Dual Modular Redundancy (DMR)	• Two registers sample the input state • Error detection only	
Fault Tolerant (FT)	• Dual Interlocked Storage Cell (DICE) • Resist SEU event (rad-hard) • Reduced area • Available in special libraries	
Error Protection Scheme (Parity)	• Safety applied to register groups / busses • Types: Parity (odd/even), EDC, ECC • Reduced area vs TMR / DMR	 Parity Error Parity Error Parity Error A, B FF1 FF1 FF1 ƒ(A, B)

Figure 2: Safety Register Strategies

B. Physical Implementation of TMRs

Synopsys place and route tools, such as Fusion Compiler and IC Compiler™ II, legalize the placement of the registers and ensure that routes adhere to the freedom from common mode interference requirements. It places the registers with the physical distance as defined in the SSF intent. As an extra safety measure, it can insert additional supply taps on either side of the registers to ensure further resilience against any SEU.

Figure 3 illustrates how Synopsys tools address the physical implementation requirement for TMRs. It shows how physical distance requirements will be met such that an SEU will not affect all three of the registers at the same time, insertion of supply taps on either side of the registers as an additional SM, and how safety register clocks/resets are split with independent physical routes and buffering to avoid cascading failures on the nets.

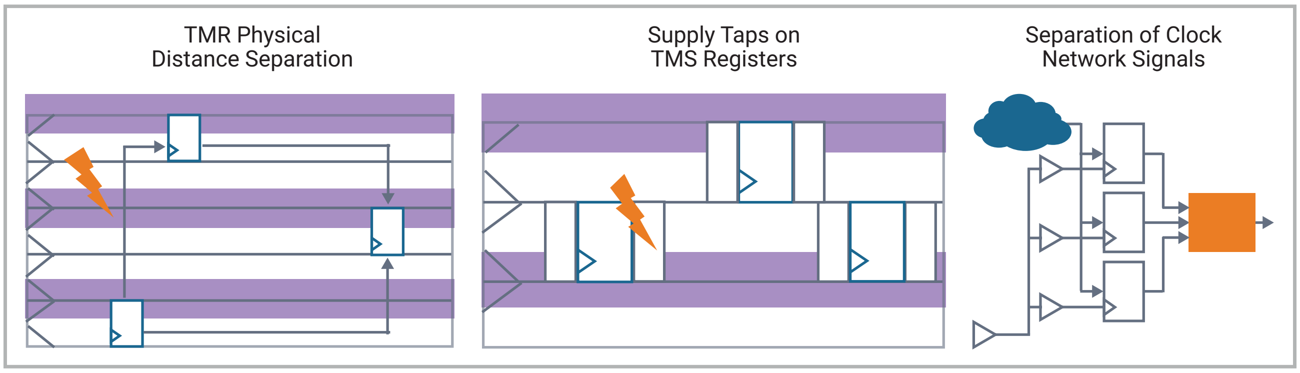

TMR Physical Distance Separation
Supply Taps on TMS Registers
Separation of Clock Network Signals

Figure 3: Physical Implementation Requirements of TMR Registers

Design Compiler, Fusion Compiler, and IC Compiler II all provide robust error checking to ensure that the SMs have been implemented properly along with visual aids such as coloring a TMR group and its voting logic the same color as shown in Figure 4.

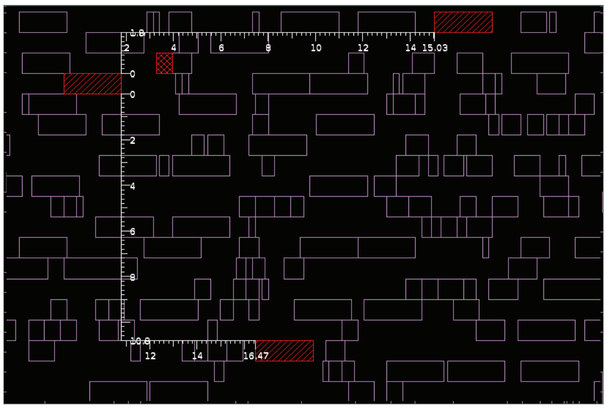


Figure 4: Physical Layout of TMR w/Voter

C. Redundant Core Design Implementation

Dual Core Lock Step (DCLS) is another approach to designing redundancy in SoCs. DCLS uses two identical cores that run in lockstep, essentially executing the same instruction sets at the same time. The output of the two cores is compared to ensure that they produce identical results. Any mismatch, such as what can happen with an SEU, gets detected as an error.

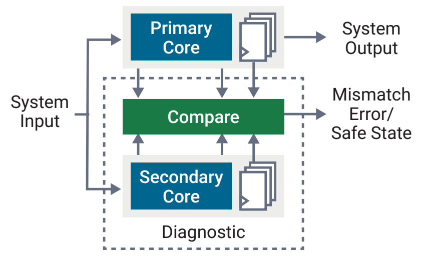
Redundant Core Strategy	Description	Design Example
Dual Core Lock Step (DCLS)	• Two cores run in lockstep (same instruction sets) • Output of cores produce identical results • Any mismatch from a fault gets detected as an error	 Primary Core System Output System Input Compare Mismatch Error/ Safe State Secondary Core Diagnostic

Figure 5: DCLS Safety Mechanism/Measure

DCLS is often used in mission-critical applications to ensure the resilience of the semiconductor design. As previously stated, dual redundancy is often used for error detection – in the case of DCLS, further logic must be designed to correct the error and get the cores back in lockstep. Like TMR registers, Triple Core Lock Step (TCLS) can be deployed for error correction in addition to error detection; however, this is generally not used since core designs tend to be area intensive.

D. Physical Implementation of DCLS

Similar to TMR registers, requirements for DCLS physical implementation include the following to avoid common mode failures (see Figure 6):

- Physical distance separation between primary and secondary cores
- Avoid shared clustering of flops in both cores during clock tree synthesis and generation of clock networks
- Physical routes from one core should not be shared with the other core as much as possible

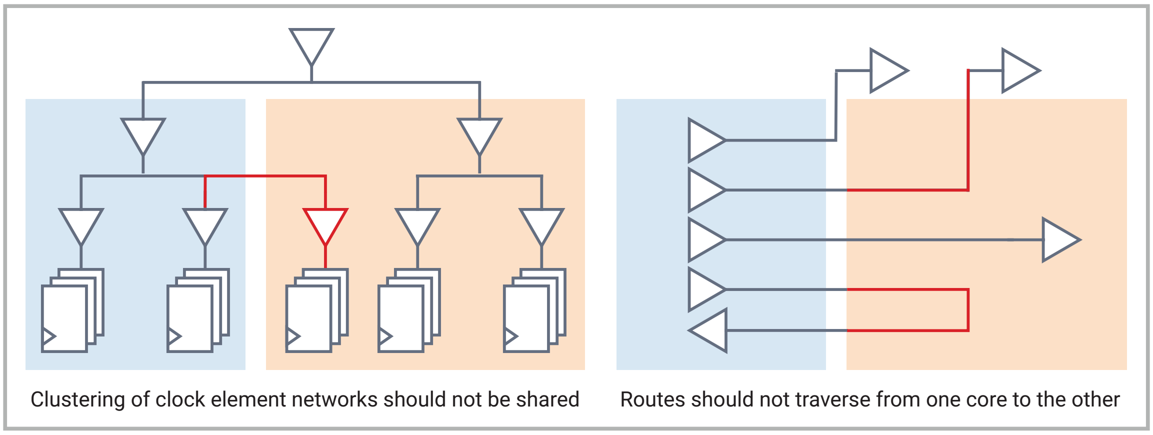

Clustering of clock element networks should not be shared
Routes should not traverse from one core to the other

Figure 6: DCLS Physical Implementation Requirements

In the example of implementing DCLS in a design, the cores should not only be physically separated, but any signal or clock routes should not traverse from one core to the other since an SEU on any shared route can cause both cores to experience a failure. Examples of a Synopsys ARC-based DCLS implementation with and without safety awareness are shown in Figure 7. Cores are not only physically separated by the distance specified in SSF, but routes will be 'localized' within each of the cores such that net traversals from one core to the other are minimized.

Placement, clock tree synthesis, optimization, routing, and reporting engines in Fusion Compiler and IC Compiler II have all been updated to include safety-awareness to meet all of the SM requirements for redundant registers and cores.

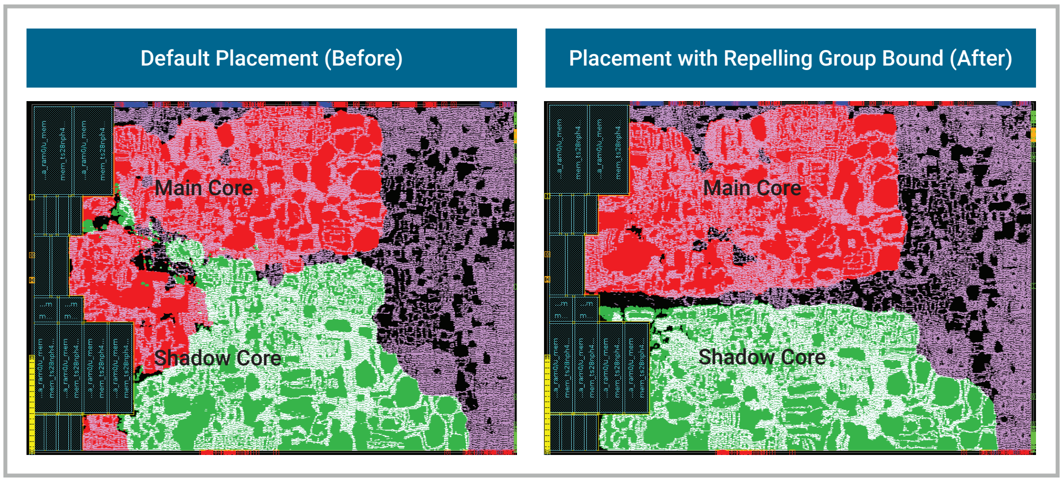

Default Placement (Before)
Placement with Repelling Group Bound (After)
Main Core
Shadow Core
Main Core
Shadow Core

Figure 7: Safety-aware Implementation of DCLS

Conclusion

In summary, reliability focuses on avoiding failure over the long-term operation of semiconductor designs, while resiliency ensures quick recovery from unexpected events and soft errors that can occur from a single event upset. Both concepts are important in mission-critical semiconductor design. They require the implementation of different safety mechanism/measure strategies and trade-offs to achieve the desired level of performance and robustness.

Previously these strategies have been inserted manually which potentially requires thousands of lines of custom scripting and can be prone to human error; but Synopsys provides a fully automated approach to inserting redundancy making mission-critical designs across various industries more resilient.

Pub: May 2023