Leveraging Statistical Random Fault (SRF) Sampling for efficient Functional Safety with Reduced efforts

Gulshan Kumar Sharma
Samsung Semiconductor India Research
Bengaluru, India
gks.92@samsung.com

Sougata Bhattacharjee
Samsung Semiconductor India Research
Bengaluru, India
sougata.b@samsung.com

Akshaya Kumar Jain
Samsung Semiconductor India Research
Bengaluru, India
akshaya.jain@samsung.com

Udaykrishna J
Siemens EDA
Bengaluru, India
udaykrishna.j@siemens.com

Gaurav Goel
Siemens EDA, India
goel.gaurav@siemens.com

Arun Gogineni
Siemens EDA
USA
arun.gogineni@siemens.com

Abstract- **Functional Verification is needed to make sure that the design is meeting its requirements as per the intended specification and design's integrity is maintained but it will never tell about the design's capability to detect and recover from random hardware failures. Fault Simulation is very important to measure the design's ability to recover from hazardous failures but with the increased complexity of the Integrated circuits due to the increased number of gates and Flip Flops in the design, the total number of fault spaces is also increasing drastically which results in high amount of time to measure the Diagnostic Coverage (DC) of the specified circuit. The motivation behind writing this paper is to present an alternative way of performing fault Simulation and to calculate the Diagnostic Coverage with reduced time and effort. The paper also illustrates on the Statistical Fault Sampling methodology which as earlier mentioned is an alternative way of doing Fault Sampling and also emphasizes on the stern differences of result when simulation is performed with Full fault space as compared to fault space decided by SRF technique.**

I. INTRODUCTION

To ensure the quality of chips used in the automotive, the industry has come up with Industrial (IEC 62380) and Automotive standards (ISO 26262). All automotive chip manufacturers/providers have to be compliant with ISO 26262 standards before being used inside the automobile. Functional Safety (FuSa) verification at either SOC design or IP level design will try to inject faults in the logic specified for the safety scenario. It will observe/analyze the effects of the faults through fault simulation or fault emulation.

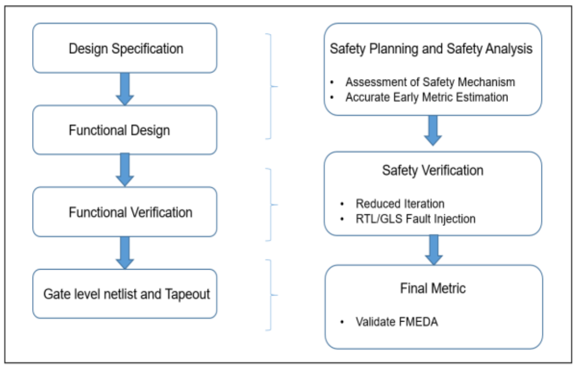

Figure 1: Traditional Flow v/s FuSa Flow

Traditional Functional safety flow starts with the availability of gate-level netlist. Analyzing the design with safety mechanisms and generating the FMEDA report [1] is one of the works of a safety engineer. If the diagnostic coverage (DC) is below the required metric, then restarting the whole lifecycle from design to verification and then safety analysis is very time-consuming. To reduce these expensive iterations, safety engineers should start working at the architecture level. Safety planning and assessment of safety mechanisms used in the design should be considered at an architectural level before the RTL is created. This safety analysis will help in predicting a near to accurate diagnostic coverage (DC) [1][2] value which can help in reaching the desired Automotive Safety Integrity Level (ASIL) of the integrated chip.

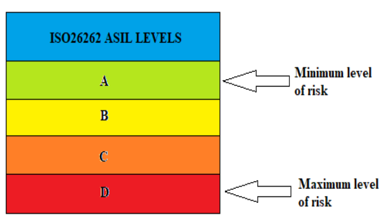

Figure 2: ASIL Level

But inspite of started working at the Architectural level, if the number of faults are more which in this case a NPU (Neural Processing Unit) Subsystem, calculating the required DC within the stipulated time is a challenging task. As Moore's law suggested, "The number of transistors will double every two years" and maybe nowadays the transistor count has increased in less than two years, the design's complexity is also increasing exponentially, and keeping in mind the needs of Functional Safety to meet the time to market of project execution, some automated methods are needed to compute the DC of the complete fault space within the stipulated time. Hence this paper introduced SRF Sampling technique [7] which will use a subset of total fault known as fault sample and sampled test vector which provides more controllability and observability to calculate the DC with much lesser time and effort as compared to full fault simulation.

II. STATISTICAL RANDOM FAULT (SRF) SAMPLING METHODOLOGY

Random hardware faults are probabilistic in nature and they can be easily estimated with statistics.
Statistical Random Fault (SRF) sampling is the technique which is used to assess the fault coverage efficiently and empirically. SRF sampling is implemented to achieve confidence in the presence of large fault population with limited resources.
SRF fault list [3][4] is a subset of full fault list which is generated for the safety block under analysis. The major advantage is in reduction of fault simulation time and Diagnostic coverage value convergence when fault space is large.

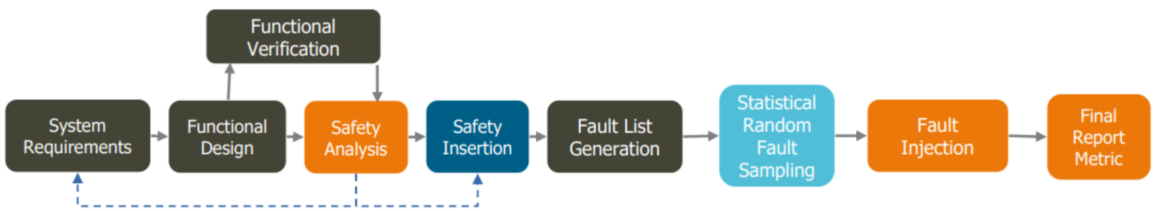

Figure1. SRF flow

Estimation of sampling size is a two-step process.
Step:1 → Get the base Sample Size using formula described in paper Statistical Fault Injection: Quantified Error and Confidence (see references)

$$n = \frac{N}{1 + e^2 \times \frac{N-1}{t^2 \times p \times (1-p)}} \qquad (1)$$

n → Sample Size required to achieve desired Error Margin and Confidence Interval
N → Total Fault Size
e → Error Margin = 1% default proposed p → probability of Error = 0.5 default which is typical
t → sensitivity analysis parameter for Confidence Interval [5]

Step:2 → use "n" calculated in previous step as input and calculate number of samples required for given Coverage (nCov) using Z-Score
• Here are the multipliers to get 4 different confidence intervals:
• 90% CI = ± [1.645 * stdev + 1/(2nCov)]
• 95% CI = ± [1.960 * stdev + 1/(2nCov)]
• 99% CI = ± [2.576 * stdev + 1/(2nCov)]
• 99.9% CI = ± [3.291 * stdev + 1/(2nCov)]

stdev → sqrt [FPC * c(1-c)/(nCov-1)]
FPC → 1 – nCov/n
c → coverage
nCov → number of samples needed for given coverage
n → number of faults in the fault universe from which the samples are taken
FPC → finite population correction (i.e. correction when sample size is a significant percentage of fault universe; the estimate will be more precise than for an infinite universe)
1/(2nCov) → this corrects for when sample size is small enough that the Gaussian distribution of estimates would be somewhat chunky instead of a smooth curve

SRF fault injection flow is similar to the traditional functional safety fault injection flow.

- It starts from safety requirements creation
- Safety analysis is done on the design (Unsafe RTL)
- Safety mechanisms are inserted into the design as per the safety requirement
- Once the safe RTL is created, then next step is creation of full fault list based on the safety block
- Multiple SRF fault list is and the sample size for the fault list is finalized based on multiple parameters
- Fault injection is done for each of the SRF fault list and the Diagnostic coverage (DC) parameter values are generated for quantitative analysis
- If the DC values are in compliance with the ASIL or safety requirements, then final FMEDA report is generated for assessment and compliance with ISO26262 automotive standard

III. Implementation Methodology

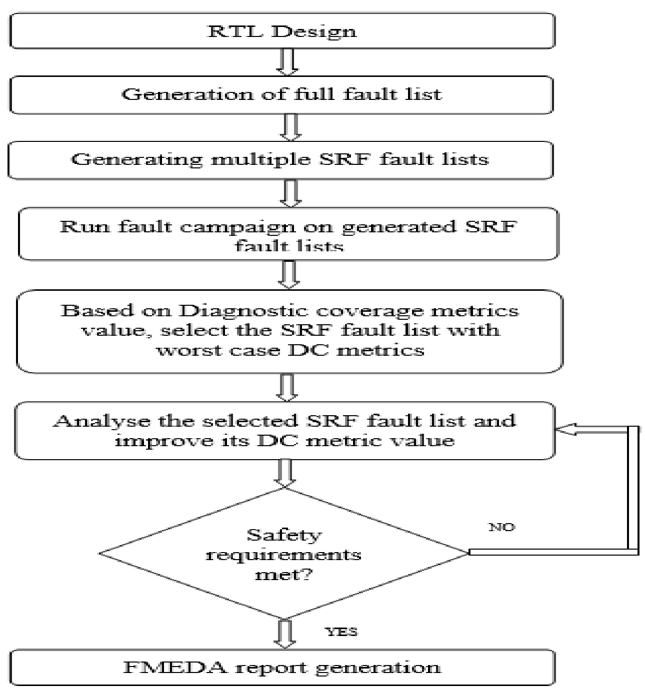

Figure 2. SRF Methodology flow

The Implemented flow is mentioned in the figure 2. The NPU RTL design with multiple sub-blocks is given as input and full fault lists of sub-blocks are generated. From the full fault list, we generated multiple SRF fault list. The SRF fault list is generated based on parameters like Confidence interval, Margin of Error, Sample size, etc. as mentioned in previous section. Once we have generated SRF fault list, then fault campaign has to be run on all the fault lists. Based on the Diagnostic coverage (DC) metric value, the SRF fault list with worst DC metric value is chosen to implement and run fault campaign on. Till the safety requirements are met, we have to analyze and improve the DC metrics. Once the requirements are met for the safety goal, the last step is the generation of FMEDA report from the fault campaign results.

IV. Results

The sub-blocks presented in results are part of NPU block. Table I contains fault campaign results on NPU sub-blocks with full fault list simulation.

Block	SRF Fault Space	Alarms Detected
TCM Core 0	5,93,532	99.58%
SHARED SRAM	3,77,67,424	Not finished
TCM Core 1	5,93,532	99.59%
BUF Mem	66,496	99.98%
LUTFIFO	2,96,104	99.96%
DMA MEM	74,896	99.98%

Table I. Full fault list fault campaign

We generated SRF fault lists with confidence interval 90% and margin of error as 1.38%-1.43% for each sub-blocks. We generated 3 SRF fault lists for each sub-block and ran fault campaign on them. The diagnostic coverage metric results for SRF fault simulation are mentioned in Table II for TCM Core 0. Based on DC metrics, we can observe that SRF2 has the least DC metric value compared to other SRF fault lists. So SRF2 is chosen to run fault campaign and for comparison with full fault list simulation results. The DC metric value is improved after adding more test bench scenarios for fault injection on Not injected nodes. Similarly, we generated SRF fault list for all the sub-blocks are ran fault simulation.

SRF methodology helps us to execute fault campaign with different seeds for a subset of faults considered for several NPU blocks. One of the block shown below is TCM Core 0 for which different detected SRF computation has been shown when executed with multiple different seeds. The worst case analysis has been considered for the final result in order to achieve the maximum coverage performed on different modules within the single block.

Block	SRF Fault space	SRF1 – Alarms detected	SRF2 – Alarms detected	SRF3 – Alarms detected
TCM Core 0	4800	74.6%	73.5%	74.78%

Table II. TCM Core 0 block SRF Fault simulation results

Table III provides results for all the blocks inside NPU with SRF fault simulation implementation. The fault campaign results clearly demonstrate that the SRF fault campaign metric is comparative with Full fault list simulation results. Another advantage is DC convergence for large fault space.

For SHARED SRAM block in Table I, we can observe that due to large fault space the convergence for DC values can become issue. This is resolved when we use SRF.

Block	SRF Fault Space	Alarms Detected
TCM Core 0	4800	99.92%
SHARED SRAM	4714	100%
TCM Core 1	4804	99.96%
BUF Mem	4708	99.98%
LUTFIFO	4510	99.97%
DMA MEM	4438	99.95%

Table III. SRF fault list fault campaign

Another advantage of using SRF fault simulation over full fault list simulation is reduction in test time or fault simulation time. As the fault space is reduced in SRF, so the fault simulation time is reduced drastically by the tool. The test time reduction achieved is more than 70% when compared to full fault list simulation. The results are tabulated in Table IV.

Block	Full Fault Space	SRF Faults	Full Fault Computation time	SRF Computation time
TCM Core 0	593532	4800	>48 hrs	~7 hrs
SHARED SRAM	1000000	4714	>72 hrs	~6 hrs
TCM Core 1	593532	4800	>48 hrs	~7 hrs
BUF Mem	66496	4708	>24 hrs	~5 hrs
LUTFIFO	296104	4510	~48 hrs	~6 hrs
DMA MEM	74896	4438	>24 hrs	~5 hrs

Table IV. SRF fault list fault campaign

V. Conclusion

In this paper, we have broadly discussed about SRF technique and it has great potential to extract the results with less time and effort in comparison with the Full Fault space simulation. The paper also discussed the necessary execution steps and techniques required to implement SRF and demonstrated results performed on the large number of blocks which justifies the results of this technique.

References

[1] ISO26262-11:2018, Clause 4.8.2, Page 56), Quantified error and confidence, 2009 Design, Automation & Test in Europe Conference & Exhibition. IEEE. April 2009, 502-506 (Reference of Page 56)

[2] Kevin Rich, Shekhar Mahatme, and Meirav Nitzen "Functional Safety Verification for ISO 26262, DVCON US 2018

[3] Michael G. Mcnamer, Subhash C. Roy and H. Troy Nagle , Fellow, IEEE, “Statistical Fault Sampling”, IEEE Transactions on Industrial Electronics, Vol. 36, NO. 2, May 1989.

[4] Agrawal, V.D., “Sampling techniques for determining fault coverage in LSI circuits” , Journal of Digital Systems , vol. 5, no. 3, pp 189 - 202 , 1981.

[5] Austemper KaleidoScope User Guide – Safety Verification

[6] Austemper SafetyScope User Guide – Safety Analysis

[7] Fault Sampling and Statistical Fault Simulation – Adam A. Ray, VLSI Testing, Springer – 2005

[8] McNamer, M. G., Roy, S. C., Nagle, H. T., “Statistical fault sampling,” IEEE Trans. Indust. Elec., vol. 36, no. 2, pp. 141-150, May 1989