2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID)

Perturbation based Workload Augmentation for Comprehensive Functional Safety Analysis

V Prasanth
Texas Instruments (India) Pvt. Ltd.
Bangalore, India
prasanth.v@ti.com

Rubin Parekhji
Texas Instruments (India) Pvt. Ltd.
Bangalore, India
parekhji@ti.com

Bharadwaj Amrutur
Indian Institute of Science
Bangalore, India
amrutur@ece.iisc.ernet.in

Abstract – **Integrated circuits are being used in many safety critical applications, wherein these circuits interact with physical systems. We are seeing a paradigm shift in the design of such systems where both digital / mixed signal circuits and physical systems are co-designed. However, the same does not hold true when it comes to safety. Safety analysis for such critical systems is still performed for the integrated circuit and physical components separately. Methods have been proposed for their co-evaluation; however, practical adoption is still limited due to analysis complexity and concerns about their comprehensiveness. Absence of a comprehensive set of workloads for safety analysis in turn impacts the dependability of results. In this paper, we examine this problem and propose a new method to incrementally augment workloads using a local perturbation technique, so as to render the augmented set to be more comprehensive. This method has been applied to several electric motor control and digital power control routines as well as a physical system for inverter operation driving an AC load using a microcontroller based system. Compared to earlier methods, the proposed method is able to identify additional critical flip-flops. For the control routines, 112 additional flip-flops (26% increase) were identified over an average of six perturbation iterations. For the inverter application, 4 additional critical flip-flops (12.5% increase) were identified over 4 perturbation iterations. These results indicate that the proposed perturbation technique is both effective and affordable.**

Keywords – ***Safety analysis, workload augmentation, fault injection.***

I. INTRODUCTION

The use of Integrated Circuit (IC) components in end applications continues to rise, particularly in critical functional safety applications like automotive, industrial and medical control. A recent study indicated that the percentage of automotive semiconductor sales is increasing at a much faster pace compared to the overall semiconductor sales [1]. Due to the complexity of the functions implemented using semiconductor devices and the complexity of the device manufacturing process, a semiconductor component can fail in multiple different ways, thus increasing the risk to the application [2].

To reduce this risk, the automotive industry has been slow in adopting the advancements in semiconductor technology. Semiconductor vendors have waited for a process technology node to mature before designing integrated circuits in that technology node into their automotive end systems. However, the emergence of new applications like autonomous and driverless vehicles is driving a change here. These new applications require upto 1000X more processing power than that required by the traditional automotive applications like power steering, emergency braking, etc [3]. The increase in processing power requirements is driving the semiconductor industry to design integrated circuits built using newer (smaller geometry) technology nodes which further accentuate the risks [4, 5].

Functional safety standards like ISO26262 [6] and IEC61508 [7] have been defined to address risks to the application due to failure of electric and electronic components, and the software / firmware executed on this hardware. The risk due to semiconductor faults is addressed by having additional mechanisms to enable timely detection of faults before they can lead to a catastrophic application failure [8]. These safety mechanisms must be optimal, i.e. incur lesser overhead on area, power, application MIPS, etc., while ensuring the required levels of safety. Fault injection [9] is a proven methodology for identifying the critical components and ascertaining the safety worthiness / robustness of the circuit. Robustness evaluation using fault injection requires the user to ascertain the impact of faults in every circuit element in each of the operating cycles of the workload. Comprehensive evaluation of safety worthiness is a computationally intensive problem and techniques have been proposed to speed up this evaluation [10, 11].

Traditional fault injection based functional safety evaluation has two major limitations. (i) Application level tolerance due to interaction with the physical system is not accounted for. (ii) Comprehensiveness of the application workloads selected for functional safety evaluation is not guaranteed [12]. The limitation in (i) causes the analysis to be pessimistic, while that in (ii) causes it to be optimistic. In this paper, we propose a new workload augmentation method based upon judicious perturbations of a given workload, to address the limitation in (ii). These perturbations are carried out using systematic analysis of a given workload to identify the input parameters and data variables, and application specific understanding to determine the impact on performance and safety due to these perturbations.

The main contributions of this paper are: (i) Different experiments are performed with alternate workloads to profile the increase in the number of critical flip-flops identified. (ii) A method is proposed for systematic perturbation of workloads whereby new workloads are generated iteratively, and are shown to be effective to detect additional critical flip-flops. (These are termed as *derivative workloads*). (iii) This method is evaluated on two examples. In one case, several closed loop routines for electric motor control and digital power conversion applications are analysed and perturbed. In the second case, a physical system for inverter operation driving an AC load using a microcontroller based system is analysed and its operation similarly perturbed. For the control routines, 112 additional flip-flops were identified (26% increase) over an average of six perturbation iterations. For the inverter application, 4 additional critical flip-flops were identified (12.5% increase) over four perturbation iterations. These results indicate that the proposed perturbation technique is both effective and affordable.

The rest of the paper is organized as follows. Section II gives a brief overview of related work in this area. Section III describes the proposed workload perturbation approach. Section IV illustrates the results obtained using the proposed approach on the set of control routines and the inverter application. Section V concludes the paper.

II. Background and Related Work

Integrated circuits used in automotive and other critical functional safety applications must go through a mandatory functional safety analysis process. This involves ascertaining the circuit behavior in presence of faults and determining whether the protection mechanisms incorporated in the chip are capable of detecting them. The set of workloads chosen for safety evaluation play an important role in ensuring analysis comprehensiveness. In order to understand this better, we have performed safety evaluation on a representative module with 25 different workloads. The results thus obtained are shown in Figure 1. The X axis denotes the workload number and Y axis denotes the number of unique flip-flops identified as critical using every new workload. It can be observed that the number of critical flip-flops identified saturates around the tenth workload and each additional workload identifies a much smaller number of critical flip-flops. We term these flip-flops as the *trailing end* of flip-flops. This paper addresses an affordable way to identify such trailing end flip-flops.

As safety analysis is becoming an integral part of IC

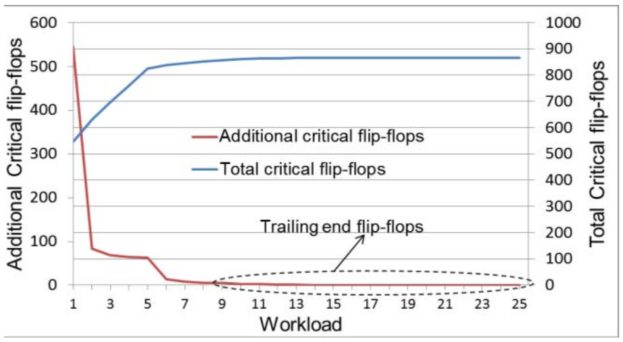

Figure 1. Number of unique critical flip-flops identified for each workload

design, addressing analysis complexity in ensuring comprehensive evaluation has become an important requirement. Several techniques have been proposed to speed up the safety analysis [10, 11] and some of the techniques have been adopted in mainstream EDA tools as well [13, 14]. Statistical and analytical methods have also been proposed in place of traditional fault injection methods to speed up functional safety analysis [15, 16]. In all these methodologies, IC safety is treated as a standalone problem independent of the application.

Real life functional safety applications are closed loop control systems with digital components interacting with physical components. The final control target for the physical system depends on various other factors which are dynamically varying. For example, cruise control application employed in automobiles has to account for variations in the road condition and slope. Given the real time nature of the application and the variability in the environment, there is an acceptable tolerance built into the application. Physical systems also have an inherent inertia, i.e. a change in the control parameter will not immediately lead to a change in the physical system operation. For example, in a motor control application a change in the actuator output takes many milliseconds to reflect as a change in the motor speed. Due to the closed loop nature of control, some computations are repeatedly performed in every loop. As a result, the effects of certain faults are over-written during the course of operation and hence can get corrected. If the circuit faults lead to a state for which the control parameters are within the acceptable tolerance threshold, the application will not be disturbed and the fault and the circuit element impacted by it (e.g. a flip-flop) are considered to be safe. Functional safety analysis without considering such effects is pessimistic, and leads to significant hardware and software overhead for making the system safe.

Analysing a physical system for application based tolerance is often infeasible due to the complexity of analysis of fault induced behaviours for such large and complex (heterogeneous) systems. In [17], the analysis pessimism problem has been addressed by incorporating the application level tolerance information during fault injection. Here, application specific tolerance information is mapped as *value tolerance* and *time tolerance*. This information is used for analysing the impact of faults at the module level. The number of critical flip-flops identified is compared for three approaches. (i) Traditional approach where fault injection is performed at the SoC level (as a collection of interacting functions). (ii) Application level (ideal) approach where fault injection is performed at the full application level and the physical system output is observed. (iii) Divide and conquer approach where the fault injection is performed at standalone functional module using *value tolerance* and *time tolerance* information derived from the application evaluation. A 4.3x reduction in the number of critical flip-flops identified is reported in (iii) as compared to that in (i). The results also indicate that some flip-flops which are uniquely detected using

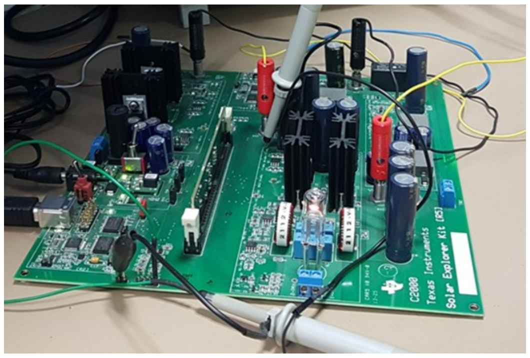
Figure 2. AC inverter application kit

the approach in (ii) escape detection using the approaches in (i) and (iii) due to limited design excitation caused by insufficient workloads. Further, some flip-flops which were identified as critical in approaches (i) and (ii) escape detection using the approach in (iii) due to a combination of insufficient workloads and tolerance considerations. Both these escape conditions indicate that the analysis based on application tolerance is optimistic, i.e. some critical flip-flops escape detection. The work in this paper addresses this issue and seeks to recover critical flip-flops which are missed in the divide and conquer (standalone functional module) approach. (These additional critical flip-flops are denoted as *trailing end* flip-flops in Figure 1).

III. Workload Augmentation Technique

Workloads play an important role in determining the comprehensiveness of the functional safety evaluation. However, there are no easy methods for upfront determination of workload comprehensiveness (or lack of it). We illustrate the impact of incomplete workloads in identifying the critical flip-flops using an inverter application. We further illustrate how workloads can be intelligently augmented (through perturbation) for better safety analysis (to identify critical flip-flops) using the same example.

A. *Inverter Application*

We used a microcontroller (MCU) based development kit [18] for studying application specific workload impact. The experimental setup consists of an AC load driven by an MCU as shown in Figure 2. The MCU has a CPU which executes the closed loop control algorithm to drive the AC load (e.g. motor) at the commanded voltage and current levels. The AC load voltage and current are measured by the MCU using the on chip Analog to Digital Converter (ADC).

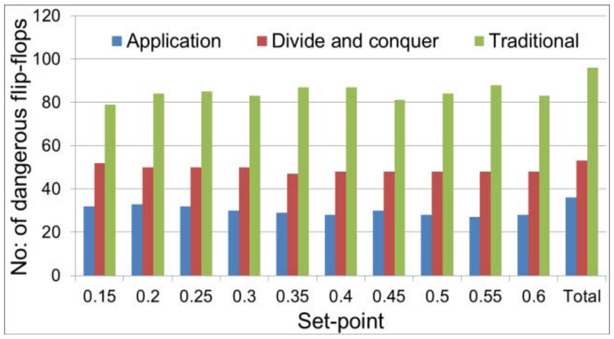

Figure 3. Critical flip-flops identified for inverter application

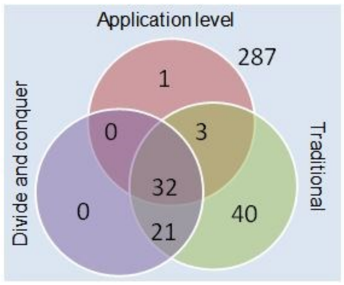

Figure 4. Critical flip-flops identified in three approaches for inverter application

The measured load current and voltage values are compared with the reference values to determine the control error. The CPU periodically executes the Proportional Integral (PI) control algorithm to minimize this error. The CPU output is used to drive the Pulse Width Modulator (PWM) which controls the power stages used to determine the voltage / current provided to the AC load. For this experiment, the CPU frequency of 60 MHz and the control loop frequency of 20 KHz are chosen. We have selected ±4% output tolerance which corresponds to the peak overshoot in current for which the control system is designed.

We have analysed the workload in the presence of faults at the different operating conditions. Each operating condition refers to a different setting where the current to the AC load is varied (referred henceforth as set-point). Fault injection is performed on 384 flip-flops (belonging to the CPU module which controls PWM) to identify the critical ones using each of the three approaches mentioned in Section II. The results thus obtained are shown Figure 3. The X axis denotes the various set-points associated with the application workload. The Y axis denotes the number of critical flip-flops identified. In order to compare the detection capability with the three approaches, we performed a detailed analysis of the flip-flops identified as critical. The results thus obtained are shown in Figure 4. We make the following observations:

(a) Application based (ideal) approach identifies the least number of critical flip-flops. On the other hand, the traditional approach identifies the highest number (indicating pessimism in the analysis).
(b) One flip-flop is uniquely identified as critical by application based approach, which is missed by the other two approaches.
(c) On the other hand, there are three flip-flops identified by application based and traditional approaches, which are missed by the divide and conquer approach.
(d) Overall, four critical flip-flops have escaped detection using the divide and conquer approach.
(e) Using the three approaches, a total of 287 flip-flops are classified as non-critical and 32 flip-flops as critical.

(f) 21 additional flip-flops are identified as critical using both the divide and conquer approach and traditional approach, while 40 flip-flops are identified as critical using only the traditional approach. (These numbers indicate the pessimism associated with these two approaches).

B. *Workload Augmentation Approach*

In order to enable the identification of critical flip-flops at the trailing end of the curve in Figure 1, a new methodology is proposed using the below three steps:

(a) The flip-flops which escape detection were *functional neighbours* of the critical flip-flops which are already identified. Functional neighbourhood consists of the set of flip-flops which implement / control the same functionality. For example, the 32 flip-flops storing the value of integration constant (*Ki*) in the Proportional Integral (PI) control function form a functional neighbourhood.

(b) A detailed analysis revealed that small perturbations in the workload are required to excite the specific conditions for controlling and observing a particular fault location. These perturbations were mapped to changes in input parameters and other embedded control parameters. The changes in values thus obtained were used to set the perturbation range. These additional workloads are called *derivative workloads*.

(c) A *derivative workload* causes better excitation in certain areas of the design, e.g. forcing the application behaviour to exceed the permissible tolerance values. This excites additional flip-flops (corresponding to the trailing end flip-flops in Figure 1).

On the other hand, an application agnostic approach to generating additional workloads with random perturbations of control / state settings may lead to the generation and / or excitation of several states and conditions which may not be functionally relevant. Such an approach is hence pessimistic. A desirable approach would be to identify the set of acceptable input conditions and internal states which influence the closed loop operation in the desired way.

An algorithm for such systematic identification and perturbation to generate derivative workloads is shown in Figure 5. The objective function for the *Workload Augmentation* algorithm is to maximize the number of critical flip-flops identified. We first identify the set of acceptable input variables and embedded control parameters. We perturb the workload within the acceptable parameter variation range and perform functional safety evaluation. If the generated workload identifies at least one additional critical flip-flop, it is accepted. The algorithm is iteratively executed to evaluate the search space. With each succeeding iteration, the number of new flip-flops identified reduces (corresponding to the *trailing end flip-flops* in Figure 1).

```
$\underline{\boldsymbol{Workload\ Augmentation(S, WL)}}$
    Set of design elements $S$
    Identify critical elements using workload $WL$,
    $S_{crit} = critical\ (S, WL)$

    for $\boldsymbol{iter} = \boldsymbol{0}$ through $\boldsymbol{max}$
        perturb workload, $WL_{iter} = perturb(WL)$
        Identify critical elements $S_{iter} = critical(S, WL_{iter})$
        If $S_{crit} \cup S_{iter} \neq S_{crit}$ , accept the workload
            $S_{crit} = S_{crit} \cup S_{iter}$
If search is not leading to new workload acceptance, exit ()

$\underline{\boldsymbol{perturb(WL)}}$
    Identify all the embedded control variables $(v)$ and input
parameters $(i)$ in Workload, $WL = f(i, v)$
        set of acceptable values of $v$, $V = acceptable\ (v)$
        set of acceptable values of $i$, $I = accptable(i)$
        $\forall v' \in V\ and\ \forall\ i' \in I, neighbor(WL) = f(i', v')$
```

Figure 5. Algorithm for workload augmentation

TABLE 1. CONTROL FUNCTIONS USED FOR EVALUATION

Function	Purpose
mppt_PnO	Perturb and observe algorithm to extract maximum power.
mppt_incc	Incremental conductance algorithm to extract maximum power.
clarke	Transformation to convert three-phase quantities into two-phase quantities.
Iclarke	Transformation to convert two-phase quadrature quantities into three-phase quantities.
park	Transformation to convert stationary reference frame to rotating reference frame.
ipark	Transformation to convert rotating reference frame to stationary reference frame.
2p2z	Two pole two zero digital control algorithm.
3p3z	Three pole three zero digital control algorithm.

IV. EXPERIMENTAL RESULTS

In order to evaluate the effectiveness of the proposed approach in identifying critical flip-flops, experiments are first conducted on a set of control functions and later repeated on the inverter application.

A. *Control Functions*

We selected a representative set of control functions as given in Table 1. This includes a mix of closed loop control applications and related critical computation functions (which are used in safety critical applications). We determined the set of acceptable input values, set of acceptable values for embedded control parameters and output tolerances. Fault injection experiments were performed to identify the set of critical flip-flops for each of the control functions, and repeated for these variations, using the *Workload_Augmentation* algorithm in Figure 5.

The results are shown in Figure 6. For initial evaluation, the maximum number of iterations was set to six, i.e. the original workload was perturbed five times. We injected faults on different memory mapped data variables. (This number varies with the function and is shown in brackets). The cumulative number of critical flip-flops identified for each iteration is shown along the Y axis. As can be seen, with successive iterations, no new trailing end flip-flops are

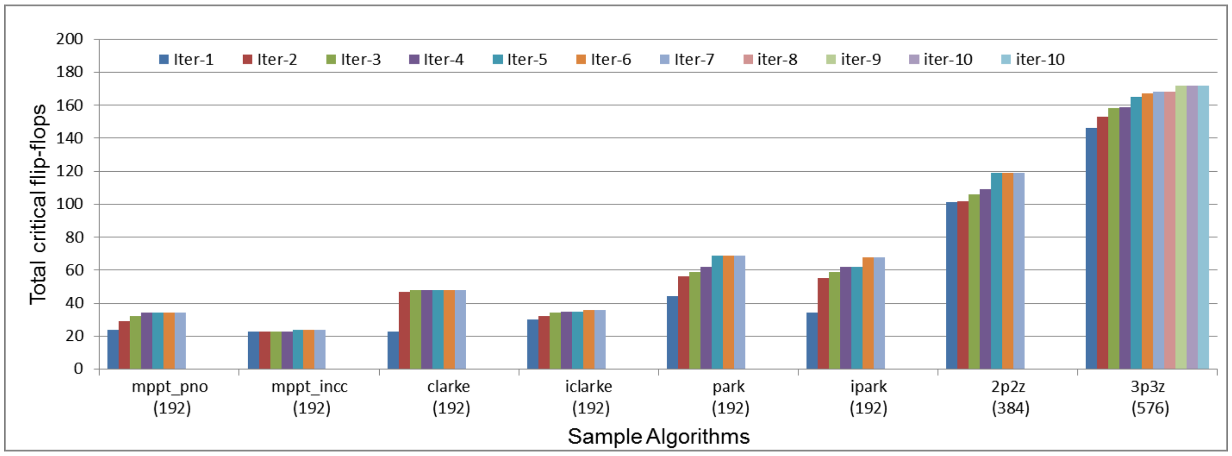

Figure 6. Variation of critical elements with workload perturbation

identified, and eventually the workload perturbation stops. (The initial iteration count was set to 6. However for *3p3z*, it was extended to 10 since in the 6th iteration also new critical flip-flops were identified).

B. *Inverter Application*

For practical workloads, exhaustive iterations using additional workloads (generated using the algorithm in Figure 5) will result in a very large number of workloads and thus increase the analysis complexity. For the inverter application, the estimated workloads are in excess of 10,000. We hence propose a more directed perturbation method, (based upon the knowledge of the application, its inputs and embedded control variables for closed loop operation), wherein variations which can result in the generation of relevant workloads specific to the application alone are considered.

While a detailed explanation of the control loop algorithm for the inverter application is beyond the scope of this paper, we can condense the analysis using the equations below.

$$\begin{aligned} y(i) &= Up + Ui \\ &= [Kp * e(i)] + [Ki * e(i) + U(i-1)] \end{aligned}$$

where $y(i), Up, Ui, e(i), Kp$ and Ki denote the PI function output, proportional path output, integral path output, error input, integration path constant and proportional path constant respectively. In order to control the output $y(i)$, the error term $e(i)$ must be suitably controlled by Kp and Ki.

For the inverter application, fault injection was performed with an initial workload to identify the critical flip-flops using each of the three approaches mentioned in Section II. The set of critical flip-flops which remain undetected using the divide and conquer approach (as explained in Section III.A) were analysed and the workload was augmented iteratively. This is indicated in Table 2, and the additional number of critical flip-flops identified is shown in Figure 7. (The dotted ellipse indicates the critical flip-flops which escaped detection for that workload). The following analysis was performed.

TABLE 2. WORKLOAD ITERATIONS FOR INVERTER

Workload	Change
WL0	Initial workload.
WL1	WL0 updated with time varying (sinusoidal) error.
WL2	WL1 updated to optimize the control loop for open loop scenario.
WL3	WL2 updated to change the integral control parameter of the closed loop.

(i) 4 flip-flops escaped detection using the initial workload *WL0* (Figure 7(a)). This is because while in the actual application, the control system error value (which is the input to the control function) in each control loop iteration shows significant variation due to the physical system behaviour, in the divide and conquer approach the same workload is executed on the module in isolation with lesser variation. The module, (in this case the CPU on which fault injection is performed and which provides controls to PWM), can be more comprehensively excited by changing the constant reference provided to the closed loop control with a continuously changing reference (e.g. a sinusoidal reference). A new workload *WL1* is thereby created. The results shown in Figure 7(b) indicate that 2 out of these 4 flip-flops are now detected.

(ii) Further analysis of the 2 unidentified critical flip-flops indicated that the control loop parameters required for open loop operation (divide and conquer approach) were different compared to those required for closed loop operation (application approach). A new workload *WL2* was generated by updating the control loop parameters to recalibrate the output value for the open loop operation. The results in Figure 7(c) indicate that 1 additional flip-flop is now detected.

(iii) The single flip-flop which escaped detection using Workload *WL2* was part of the integrator logic. In a typical PI control loop, the Kp (proportional) term impacts the loop gain and helps to reach the optimal performance point faster. The Ki (integral) term helps to remove the steady state error in the system. A new workload *WL3* was added with new value for Ki which can force the application behaviour to exceed the

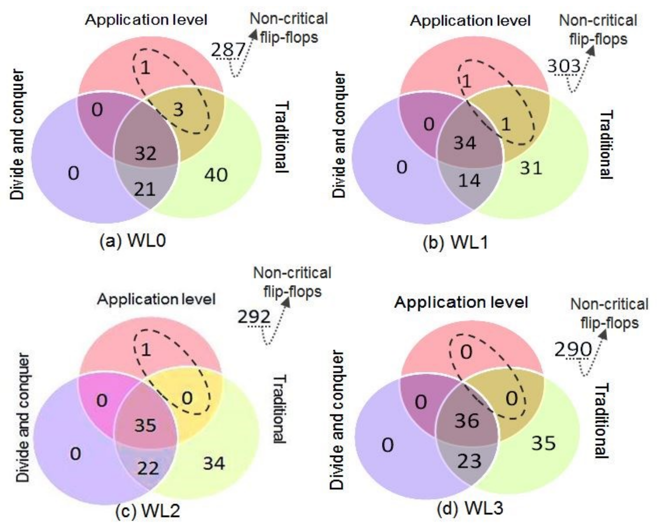

Figure 7. Critical flip-flops identified with perturbed workloads for inverter applications

permissible tolerance values. As shown in Figure 7(d), *WL3* detected all the critical flip-flops.

(iv) In this particular study, we have used the proposed approach to reduce the number of critical flip-flops which escape detection. This approach can also be utilized to perform trade-off between hardware overhead incurred and reliability gained by reducing the number of false positives, i.e. the number of non-critical flip-flops which are pessimistically marked as critical using the proposed approach. As can be seen in Figure 7, the total number of critical flip-flops identified varies across workloads. This includes flip-flops not identified as critical in the application approach as well as flip-flops so identified pessimistically. The number of flip-flops pessimistically identified using this algorithm is contained to 23 (out of 384, i.e. 5.99%), whereas with random perturbation, this number is much higher, tending to cover all the flip-flops.

The workload perturbation algorithm is carried out accordingly in Figure 5. Generic recommendations include: (a) Inputs applied and parameter changes performed in open loop analysis must be representative of the actual closed loop scenario. (ii) Flip-flops whose criticality depends on their values must be so identified and updated.

V. Conclusion

In this paper we propose a perturbation based workload augmentation technique for performing comprehensive functional safety evaluation. Experiments are performed on a set of safety critical control functions and an inverter application, and the effectiveness of the proposed method in identifying additional critical flip-flops is demonstrated. 12% to 26% additional critical flip-flops are identified. Through these experiments, we have illustrated how optimisations in safety evaluation methods using application level tolerance can be traded off with additional workloads to arrive at a more comprehensive list of critical flip-flops to meet the overall hardware overhead and reliability requirements. Together, these results indicate that the proposed perturbation technique is effective to identify additional critical flip-flops within affordable overhead of analysis complexity and pessimism.

Further investigations are planned towards reducing pessimism in the proposed approach and deriving an application agnostic approach for workload perturbation.

References

1. S. C. Stefan Burghardt and F. Weig. (2017) Mobility trends: What's ahead for automotive semiconductors. [Online]. Available: https://-www.mckinsey.com/industries/semiconductors/our-insights/mobility-trends-whats-ahead-for-automotive-semiconductors
2. R. Mariani and G. Boschi, "A systematic approach for failure modes and effects analysis of system-on-chips," in *International On-Line Testing Symposium*, 2007.
3. Technology and computing requirements for self-driving cars (2017). [Online]. Available: https://www.intel.in/content/www/in/en/-automotive/driving-safety-advanced-driver-assistance-systems-self-driving-technology-paper.html
4. R. C. Baumann, "Radiation induced soft errors in advanced semiconductor technologies," *IEEE Tran. on Device and Materials Reliability*, 2005.
5. D. Lorenz, G. Georgakos, and U. Schlichtmann, "Aging analysis of circuit timing considering NBTI and HCI," in *International On-Line Testing Symposium*, 2009.
6. *ISO 26262, International standard for functional safety of electrical and electronic systems in production automobiles*, Std., 2011.
7. *IEC 61508, International standard for functional safety of electrical/electronic/programmable electronic safety-related systems*, Std., 2010.
8. V. Prasanth, D. Foley, and S. Ravi, "Demystifying automotive safety and security for semiconductor developer," in *International Test Conference*, 2017, pp. 1–10.
9. A. Benso and P. Prinetto, *Fault injection techniques and tools for embedded systems reliability evaluation*. Springer Science & Business Media, 2003.
10. F. Kriebel, S. Rehman, D. Sun, P. V. Aceituno, M. Shafique, and J. Henkel, "Acsem: Accuracy-configurable fast soft error masking analysis in combinatorial circuits," in *Design, Automation & Test in Europe*, 2015.
11. S. Mirkhani, S. Mitra, C.-Y. Cher, and J. Abraham, "Efficient soft error vulnerability estimation of complex designs," in *Design, Automation & Test in Europe*, 2015.
12. V. Prasanth, R. Parekhji, and B. Amrutur, "Improved methods for accurate safety analysis of real-life systems," in *Asian Test Symposium*, 2015.
13. Cadence Incisive Functional Safety Simulator. [Online]. Available: https://www.cadence.com/content/dam/cadence-www/global/en_US/-documents/tools/system-design-verification/incisive-functional-safety-simulator-ds.pdf
14. Synopsys: Z01X Functional Safety Assurance. [Online]. Available: https://www.synopsys.com/verification/simulation/z01x-functional-safety.html
15. H. Cho, S. Mirkhani, C.-Y. Cher, J. A. Abraham, and S. Mitra, "Quantitative evaluation of soft error injection techniques for robust system design," in *Design Automation Conference*, 2013.
16. L. Chen, M. Ebrahimi, and M. B. Tahoori, "Cep: correlated error propagation for hierarchical soft error analysis," *Journal of Electronic Testing*, 2013.
17. V. Prasanth, R. Parekhji, and B. Amrutur, "Safety analysis for integrated circuits in the context of hybrid systems," in *International Test Conference*, 2017.
18. Texas Instruments Development Kit Application Note. [Online]. Available: http://www.ti.com/tool/TMDSSOLARPEXPKIT