Functional safety is a critical concern for all automotive products, and the most complex and least understood part of it is safety from random faults (faults due to unpredictable natural phenomena rather than design bugs). ISO 26262, “Road vehicles – Functional safety” sets out the requirements for safe designs. In this article, we present a simple, easy step-by-step methodology to comprehend and achieve functional safety from random faults based on Questa® simulation and the fault-injection accelerator from Optima.

INTRODUCTION

The computers are fleeing their cages. Until recently, people interacted with computers in a virtual world of screens and mice. That world had many security risks but relatively few safety risks, mostly electrocution or having a PC fall on your foot. But in the last few years a new wave of computers is invading the real world, and physically interacting with it. This trend is expected to explode in the near future, with self-driving cars and drones leading the rush. This raises totally new safety concerns for the teams designing the semiconductor parts used in these markets. In the good old days, a HW bug would cause a blue-screen and everyone would blame Microsoft®. Nowadays, a HW bug can trigger a criminal trial for involuntary manslaughter.

To prevent such problems, at least for the Automotive market, The International Standards Organization ISO published in 2011 the first version of ISO 26262, “Road vehicles – Functional safety”. The second revision is being completed now and should be published in about a year. While focused on road vehicles, this standard can be easily adapted to related areas that do not yet have their own safety standard, such as drones, since it is in fact an adaptation of IEC 61508, the basic standard for Functional Safety of all Electrical/ Electronic/Programmable Electronic Safety-related Systems.

This article discusses functional safety. The International Electrotechnical Commission IEC, who own the ultimate standard in this area, define safety as freedom from unacceptable risk of physical injury or of damage to the health of people, either directly, or indirectly as a result of damage to property or to the environment. Functional safety is the part of the overall safety that depends on a system or equipment operating correctly in response to its inputs. Functional safety is the detection of a potentially dangerous condition resulting in the activation of a protective or corrective device or mechanism to prevent hazardous events arising or providing mitigation to reduce the consequence of the hazardous event.

The following discussion is based on ISO 26262, and so targets people in the Automotive market. But it is general enough to be useful for anyone who worries about the functional safety of their semiconductor products.
THE TYPES OF SAFETY ISSUES

Safety issues fall into two main categories: systemic and random faults. Systemic faults are those that are repeatable, hence predictable. A more common name for them is design bugs. Random faults are unpredictable (except in the aggregate), and are due to the complex interaction between the product and its environment.

Safety from systemic faults, also known as bug prevention, detection and recovery, is a well-known discipline. Safety from random faults, on the other hand, is much less understood. This article will discuss how to achieve safety from random faults, and to do so with a reasonable cost.

Random faults fall into two further categories: permanent and transient faults. Permanent faults, such as a burn-out of a wire, are faults that remain faulty and so can be tested for. Permanent faults can occur at any location in the product, and so are modeled on all electrical nodes. Transient faults, on the other hand, disappear after a short while. Typically, transient faults are due to the effects of a cosmic radiation particle hitting the product, dispersing some electrons, and subsiding.

Transient faults can occur at any location in the product. However, the extensive use of ECC/EDC schemas for memories (see below) means that transient faults on memories can be ignored as a solved problem. Locations that are combinational logic gates, on the other hand, seldom cause harm to the product since the logic value of any gate is only relevant for a very small percentage of the time (only when that gate is in the active computation branch and only when the wave of final results goes through that gate). So as a matter of practice, transient faults are only investigated for registers.

ENSURING SAFETY FROM RANDOM FAULTS

Safety from random faults is a statistical goal. No design can ever be 100% free of random faults. Instead, a goal is set for the probability of failure. These are usually defined in terms of FIT, where 1 FIT is defined as one failure in every 10^9 hours, or once every 114,155 years. The predicted probability must be lower than the goal set for the specific product being designed.

Prevention of random faults (of both types) is an expensive endeavor. The most common and generic approach to it is with redundancy, sacrificing costs to achieve safety. Examples of redundancy include dual modular redundancy (DMR, aka lockstep) where duplicating the hardware and comparing results enables fault detection; triple modular redundancy (TMR) where having three copies enable not only detection but also correction; error detection and correction (EDC) and error-correcting code (ECC) schemas that are used for memories and busses and achieve similar goals with a smaller cost than full duplication; and more. Obviously, the cost (in Silicon area, power consumption, etc.) for these approaches can be 2-3X that of the unprotected design.

Detection of random faults is usually based on the frequent running of (SW or HW) tests with known results, and checking if the right answer is produced. This can only be applied to permanent faults, and usually does not detect all the faults. For every given design and test a number, called the test coverage, indicates the ratio of faults detected by the test out of all possible faults. If we can show that a design can fail due to a fault in one of two possible locations, and only one of these faults causes the test to give the wrong result, then that test has a coverage of 50%. So the probability of
the product to be harmed by a permanent fault can be derated by the detection coverage (assuming the tests are run frequently enough).

**Recovery** from random faults is usually applied to transient faults only, since permanent faults have an unbounded impact on the behavior of the product. On the other hand, transient faults can dissipate after some time, and the design is then said to have recovered from that fault. So the probability of the product to be harmed by a transient fault at a specific location can be derated by the probability that such a fault will dissipate harmlessly, and the total probability of transient fault harm is the sum for all locations.

This last discussion raises a new possibility for **prevention**. If the probability that a fault on a given location will dissipate harmlessly is known, it becomes possible to apply redundancy on a location-by-location basis. Specifically, since transient faults are computed for register locations only, those registers with a high probability of harm can be selectively implemented using a protective design (e.g., DICE®), using a technique known as selective hardening.

**THE NEED FOR ACCURATE DATA**
The discussion above requires two types of data:

1. Test coverage for permanent faults
2. The probability that a fault will dissipate harmlessly for transient faults

Test fault coverage is a well-known technique in manufacturing test and DFT. There it is used to determine how good manufacturing tests are in detecting faults and filtering out bad products from reaching the customer. In recent years, structural test and ATPG techniques have replaced test fault coverage techniques, but still, some functional test is still used for fault detection, and the same methodology can be applied. The typical approach is fault-testing, using gate level simulation to process fault by fault (or in small fault batches).

Fault dissipation probability is a new technique, with little support in methodology and CAD tools. Again, the usual approach is to apply simulation, in this case usually RTL simulation.

The basic flow of using simulation for both types of data collection can be seen in Figure 1.

**THE BASIC FLOW: HOW TO MAKE THE DESIGN SAFE FROM RANDOM FAULTS**
The basic design flow to protect your design begins by partitioning your design into memory

![Figure 1. Using simulation for data collection](image)
blocks and random-logic blocks. Memory blocks have a well-understood protection mechanism in ECC/EDC, so that it is just a question of selecting the appropriate approach given the specific requirements and constraints of the design.

For random-logic blocks, a key decision is whether or not to use redundancy. If the design constraints allow for the extra cost in area and power, then redundancy is very easy to implement. Just decide on the relevant level (product, unit, gate), the number of copies (2, 3, more), and the type of redundancy, and you are done.

If redundancy is not affordable, then you must consider permanent and transient faults separately. For permanent faults, the easiest way is to apply DFT / ATPG techniques to generate high-coverage tests. The downside, besides the need to pay for some extra area to account for structural test HW, is that these tests require a hard reset before and after they run. So they can be applied only in cases where the product can be taken offline, tested and restored to use every millisecond or so. In other cases, a functional test must be written and evaluated.

For transient faults, the next decision is whether full flop hardening is applicable. Full flop hardening means the implementation of all flops in a way that minimizes transient fault probability, with the usual area & power penalty. If the constraints prevent this option, then you must apply selective hardening. This overall flow can be seen in Figure 2 below.

**SELECTIVE HARDENING**

Selective hardening is the process of determining which register should be implemented using what technology. It is predicated on two assumptions:

1. That every register can be implemented in a number of ways, and that these ways differ in their susceptibility to transient faults, in their area and in their power dissipation. Examples can include a regular register, a DICE register, and a TMR register which is three parallel registers with voting.
2. That for every register, the probability that a transient fault on it will dissipate harmlessly is known with a high accuracy, given a specific SW workload.
Under these two assumptions, it is clear to see how different assignment of implementation options to the various registers will lead to different overall results of safety, area and power. Proper trade off techniques are then utilized to best match the design goals and constraints.

While the first assumption is simple, it is less clear how to meet the second assumption. First, it is important to understand why it depends on a specific SW workload. Since for almost every register in a design it is possible to write a SW workload for which no faults on that register ever dissipate, taking the worst-case approach leads to assuming all registers have 0% dissipation rate. This is an unrealistic over-design. In fact, most safety-sensitive HW has very precise SW that is expected to run on it. Thus, that SW should be used and registers that, for that SW, have a high dissipation rate should be treated accordingly.

For a given SW and a given register, then, a simulation can be made of the results of a fault happening on that register at cycle X of the SW. The results of the simulation should show whether or not that specific fault has dissipated harmlessly. This should be repeated, either for all cycles or for a large enough sample of cycles. The results of this process, presented as the percentage of faults that dissipated out of the overall faults simulated, is a good approximation of the overall probability of that register, with higher accuracy the more simulations were performed. This process should then be repeated for all registers in the design.

THE NEED FOR FAST FAULT SIMULATION
We have seen that both permanent and transient fault safety require, in certain cases, a large number of simulations. For permanent fault test coverage, a simulation of the entire test is required once per register. For transient selective hardening, a number of simulations of the reference workload is required per register. These are very high numbers.

The reference simulation in these cases would be a run of Mentor Questa® RTL or gate-level simulation. However, even with the latest speedups, the total machine-years of simulations can easily reach thousands of years, with the associated TTM impact and the engineering, computer and license costs.

THE OPTIMA-SE ULTRA-FAST SIMULATION SOLUTION
Optima Design Automation (www.optima-da.com, info@optima-da.com) is an Israeli startup that addresses the problems of ensuring safety for electronic devices. Its unique and ultra-fast technology enables fault simulation up to 100,000X faster than regular simulations, while keeping full compatibility with and integration with Questa. Thus, thousands of years can become mere weeks of computer time.
Optima-SE analyses your design, indicates hot-spots and areas of concern, and creates a unique spreadsheet of data for your selective hardening work. Its easy-to-use controls enable you to quickly and easily apply selective hardening to designs with millions of registers, seeing the resulting safety, area and power implications immediately and quickly converging on the right solution.

Contact us today for an evaluation of this unique technology on your own design, to see what this safety solution can do for you.

END NOTES


VERIFICATION ACADEMY
The Most Comprehensive Resource for Verification Training

27 Video Courses Available Covering
• SystemVerilog OOP
• Formal Verification
• Intelligent Testbench Automation
• Metrics in SoC Verification
• Verification Planning
• Introductory, Basic, and Advanced UVM
• Assertion-Based Verification
• FPGA Verification
• Testbench Acceleration
• PowerAware Verification
• Analog Mixed-Signal Verification

UVM and Coverage Online Methodology Cookbooks
Discussion Forum with more than 7850 topics
Verification Patterns Library

www.verificationacademy.com