Understanding SW Test Libraries (STL) for safety-related integrated circuits and the value of white-box SIL2(3) – ASILB(D) YOGITECH faultRobust STL

White Paper n. 001/2014

Riccardo Mariani
C.T.O. YOGITECH S.p.A.
# Table of Contents

LEGAL NOTICE .................................................................................................................. 2
Abstract ................................................................................................................................. 3
About SW Test Library (STL) ............................................................................................... 3
Requirements of IEC 61508 and ISO 26262 for STL ................................................................. 3
About black-box SW Test Libraries: the ARM Cortex-M3 example ........................................ 3
About YOGITECH faultRobust Methodology (fRMethodology) ............................................... 4
Using fRMethodology to specify, implement and verify STL .................................................. 5
Key features of YOGITECH faultRobust STL (fRSTL) .......................................................... 6
Using YOGITECH fRSTL in single, dual MCU or multi-cores ............................................... 7
Summary ............................................................................................................................... 8
Availability of YOGITECH faultRobust STL products ............................................................ 8
References ............................................................................................................................ 8
Glossary ............................................................................................................................... 8
About YOGITECH ................................................................................................................ 9
LEGAL NOTICE

Information of this product is subject to change without notice according to continuous product development and improvements.

All product details and product usage described in this document are given by YOGITECH S.p.A. in a good faith.

YOGITECH S.p.A shall not be liable for any loss or damage deriving from the incorrect use of information contained in this document, or any error or omission in such information, or any incorrect use of the product.

Trademarks mentioned in this document are the exclusive property of their respective owners.
Abstract

Self-test by software (often referred to as SW test library) is one of the methods to provide diagnostic coverage for safety-related integrated circuits. It is particularly suitable for circuits that have limited or no HW diagnostic measures. This white paper describes the requirements for STLS in relation to popular functional safety standards such as IEC 61508 and ISO 26262. Furthermore, this paper demonstrates that a real compliance with these requirements is achieved only if the STL is specified, implemented and verified by means of a white-box approach which goes into detail of the integrated circuits to be tested, and which uses fault injection at gate-level to measure the actual diagnostic coverage provided by the STL. This white paper illustrates how this is accomplished by YOGITECH faultRobust SW test libraries – based on YOGITECH’s faultRobust Methodology. It also describes the advantages of YOGITECH’s SW test libraries in terms of their very limited code size, short execution time, flexibility and reusability. Finally, this white paper shows how YOGITECH SW test libraries can be used for both single and dual microcontroller designs, or in multicores.

About SW Test Library (STL)

Self-test by software is one of the methods of providing diagnostic coverage for safety-related integrated circuits. It is referred to as an SW test library (STL), a diagnostic SW test (DST), or a core-self test (CST). In essence, an STL is an SW program which is periodically executed in the field by a processing unit, with the aim of testing that the processing unit itself, or another part of the integrated circuit, is operating as designed. STLS are suitable for circuits that have limited or no HW diagnostic measures, and may also be used to complement safety mechanisms of integrated circuits that have HW support for safety.

The following section describes the requirements of popular functional safety standards such as IEC 61508 and ISO 26262 for the specification, design and verification of those STLS.

Requirements of IEC 61508 and ISO 26262 for STL

According to IEC 61508 standard (ref. IEC 61508-7 A.3.2), the STL is realized by software functions which perform self-tests using a data pattern which tests structures such as the data registers, the address registers and the instruction decoder. According to the standard (IEC 61508-2, Table A.4), the “Maximum diagnostic coverage considered achievable” is “Medium”, i.e. 90%.

Very similar requirements exist in ISO 26262, the functional safety standard for the automotive industry. For example, according to ISO 26262-5 D.2.3.1, STLS are used to detect, as early as possible, failures in the processing unit and other sub-elements consisting of physical storage (e.g., registers) or functional units (e.g., instruction decoders) by means of software. For ISO 26262 (ref. ISO 26262-5 Annex D Table D-4), the achieved diagnostic coverage depends on the quality of the STL, and it is typically “Medium”, i.e. 90%.

Moreover, ISO 26262-5 Table D.1 clearly states that “Stuck-at gate level” needs to be covered by the STL. It further states that (ref. ISO 26262-5 D.2.3.1) determining the actual coverage of the tested gates requires extensive fault simulation.

Both functional safety standards agree that – as far as the development process of an STL is concerned – the STL has to fulfill the specification, implementation and verification requirements described in the related SW chapter of the standard, i.e. IEC 61508-3 or ISO 26262-6. Those requirements consist of, amongst other factors, avoiding specific SW constructs (e.g., no multiple use of variable names) and providing detailed evidence of the structural coverage metrics reached (e.g., MC/DC coverage).

The following section demonstrates how it is impossible to claim real compliance with those requirements if the STL is designed with a black-box approach, i.e. without detailed information of the integrated circuit, and without the use of fault injection as a method to demonstrate the actual diagnostic coverage provided by the STL.

About black-box SW Test Libraries: the ARM Cortex-M3 example

To demonstrate how it is impossible to claim real compliance with the aforementioned requirements if the STL is designed with a black-box approach, we asked a skilled SW engineer, well trained in functional safety, – on the basis of the IEC 61508 2nd edition description of “Self-test by software” – to prepare an STL for a popular and simple processing unit such as the ARM Cortex-M3.
The SW engineer, using C language, prepared the STL as indicated by IEC 61508 2nd edition, i.e. testing all the ARM Cortex-M3 registers with “walking bit”, the ARM Cortex-M3 instruction decoder and address logic. The SW engineer was also requested to add the contribution of the watchdog. The SW was designed using state of the art certified compilers. The resulting STL consisted of a self-test library with a code size of 7,696 bits.

Then, a fault injection environment (based on YOGITECH’s Safety Verifier tool suite) was set up to demonstrate the diagnostic coverage achievable by this STL. The environment consisted of the gate-level netlist of ARM Cortex-M3 and memory models to store code and data during the execution of the self-test. More than 150,000 stuck-at faults were simulated (all the possible faults of ARM Cortex-M3) and the measured diagnostic coverage was 69.6%. The coverage was also measured with the registers’ walking bit test removed, resulting in a coverage of 59%. Different combinations of tests were tried, but the result was nevertheless in the range of 55%-70%, including the contribution of the watchdog.

This result shows that, even for a simple processing unit such as the ARM Cortex-M3, a skilled SW engineer without the support of a safety-oriented white-box methodology will definitively fall short of achieving the “Medium” coverage indicated by the standard. In particular, without any fault injection tool, it is impossible for the SW engineer to measure the results of his efforts. Furthermore, without a safety-oriented design exploration methodology, it is impossible for the SW engineer to direct his efforts in the right direction. It is easy to understand how this gap between the expected and the actual measured diagnostic coverage dramatically increases for complex CPU cores such as the ARM Cortex-R or Cortex-A series.

The following section describes how YOGITECH’s faultRobust Methodology is the appropriate safety-oriented white-box methodology to allow the overcoming of the limitations of the black-box approach.

About YOGITECH faultRobust Methodology (fRMethodology)

YOGITECH’s faultRobust Methodology (fRMethodology) is a patented white-box approach to perform and verify/validate functional safety analyses and the safety-oriented design exploration of integrated circuits, according to different functional safety standards. In essence, fRMethodology consists of:

- Dividing the component into elementary parts\(^1\) by using an automatic tool (YOGITECH’s Elementary Part Extractor) to guarantee the completeness of the analysis;
- Computing, by using an automatic tool (YOGITECH’s Safety Designer), the safety metrics by examining the fault models of each elementary part, attributing the failure rate, the dangerosity and then estimating the diagnostic coverage of the planned HW or SW safety mechanisms;
- Verifying/Validating, by using an automatic tool (YOGITECH’s Safety Verifier), the safety metrics by an extensive fault injection campaign, simulating permanent, transient and common cause faults.

The following section describes how an STL that is specified, implemented and verified with YOGITECH’s fRMethodology is able to completely fulfill the requirements of functional safety standards, providing tangible evidence of the claimed diagnostic coverage.

---

\(^1\) According to safety standards, an elementary part (EP) is the finest level of detail into which an integrated circuit can be sub-divided (ref. for example, ISO 26262-10, section 4.2). In YOGITECH fRMethodology, an EP is composed of a flip-flop and the logic cone contributing to its fan-in.
Using fRMethodology to specify, implement and verify STL

The flow to specify, implement and verify an STL is described in Figure 1.

![Diagram](image)

**Figure 1: fRMethodology flow for fRSTL**

It consists of: the processing of the integrated circuit database to partition it into elementary parts; executing a quantitative FMEDA in order to automatically compute the failure distribution of the elementary parts; estimating the dangerosity of each elementary parts, thereby defining the SW safety requirements describing the target diagnostic coverage for the STL; SW coding according to those safety requirements and following a strict development flow and then executing the safety verification by means of an extensive fault injection campaign aimed at verifying the estimated dangerosity and the diagnostic coverage achieved by the STL.

For example, in the case of a CPU, thanks to YOGITECH fRMethodology and YOGITECH’s Safety Designer tool, at first the safety engineer can quickly identify the key elementary parts that most contribute to the failure probability of the processing unit (see Figure 2, depending on CPU configuration); therefore, it can efficiently direct the efforts of the SW engineer by providing him with very detailed SW safety requirements. Then, the SW engineer prepares the STL according to those requirements. Eventually, thanks to YOGITECH’s Safety Verifier tool, faults are injected into the gate-level netlist to measure the diagnostic coverage achieved by the self-test – providing the same detailed information to the SW engineer in the case of remaining gaps. Furthermore, this generates tangible and objective evidence for the certification bodies. With this methodology, a diagnostic coverage of 90.5% was achieved with a code size of the STL of less than 10 Kbytes and a run time, at 100 MHz CPU operating frequency, of less than 1ms.

The following section describes the key features of YOGITECH faultRobust SW Test Libraries.
Key features of YOGITECH faultRobust STL (fRSTL)

YOGITECH faultRobust STLs (fRSTL) are the STLs produced by YOGITECH according to the flow described in the previous section. Their conceptual architecture is shown in Figure 3.

On top of their full compliance with the requirements of functional safety standards and the tangible evidence demonstrating the diagnostic coverage provided by the fRSTL, the key features are:

- Modularity ensured by the structure, consisting of a Test Interface and a set of Test Segments (TS). Each TS targets a specific function or a group of functions of the component; it provides pass/fail
information and self-checking signatures (CRC) and it may be interrupted at any time by the application SW;

- Flexibility, allowing the user to either run the full test suite or a subset;
- Easy integration, since the fRSTL is mostly written in C. Optimized assembler coding is used only to address specific sequences of instructions necessary to reach the required performance in terms of coverage;
- Low impact on the application SW, thanks to the optimization in size and run time allowed by the fRMethodology flow.

The following section describes how to use YOGITECH fRSTL for single and dual microcontroller designs.

**Using YOGITECH fRSTL in single, dual MCU or multi-cores**

YOGITECH fRSTL can be used in single and dual microcontroller (MCU) designs or multicores. If used in a single MCU, according to the few conditions of use detailed in the fRSTL Safety Manual, it provides diagnostic coverage compliant with permanent faults up to IEC 61508 SIL2 or ISO 2626 ASILB safety integrity levels. It can be executed either at power-on, periodically or before/after critical actions. Its flexible structure, organized in separated interruptible test segments, and its very short execution time allow the fRSTL to be used in every type of application, with a process safety time or fault-tolerant time interval ranging from a few to thousands of ms. YOGITECH fRSTL is typically combined with additional measures provided by the user to cover transient faults, such as data coding or algorithm time redundancy.

Since it has been developed according to the highest systematic capability (aligned with the systematic faults avoidance requirements of IEC 61508 SIL3 and ISO 26262 ASILD’s safety integrity levels), and since it guarantees a diagnostic coverage equal to or greater than 90% for each homogenous HW element, YOGITECH fRSTL can also be used in dual MCU designs targeting IEC 61508 SIL3 and ISO 26262 ASILD HW architectural metrics.

*Figure 4: fRSTL on dual MCU design*

As shown in Figure 4, YOGITECH fRSTL is typically executed in the two parallel MCUs either at power-on, periodically or before/after critical actions. To guarantee the highest level of diagnostic coverage for both permanent and transient faults, intermediate results (e.g. for each TS) are exchanged by means of a safety
protocol, and then cross-compared between the MCUs, together with intermediate results of the safety function. The same concept can be used for multicores. In this case, YOGITECH fRSTL is executed in the two parallel CPUs and — to guarantee the highest level of avoidance of common cause faults, especially for shared logic — the results are sent with a dedicated safety protocol to an independent judging element, also having the role of cross-comparing the intermediate results of the safety function. YOGITECH has unique market experience of safety-related single, dual-MCU or multi-CPU, and provides assistance to its customers on integrating fRSTL into their safety-relevant systems.

Summary
This white paper has demonstrated that an STL designed simply according to a user or reference manual, and verified without the support of a gate-level fault injection flow, will never be able to fulfill the requirements of functional safety standards. On the other hand, it has demonstrated that YOGITECH fRMethoodology is the appropriate white-box, safety-oriented methodology to guarantee meeting these safety requirements with an optimized, flexible and reusable fRSTL.

Availability of YOGITECH faultRobust STL products
The YOGITECH fRSTL portfolio includes fRSTL for all Cortex-M series and fRSTL for STM 32 MCU by ST Microelectronics. The Roadmap includes fRSTL for Cortex-A series. fRSTL can be prepared for other CPUs or other microcontrollers — please contact YOGITECH for further information.

References

Glossary
STL SW Test Library
fRSTL YOGITECH faultRobust STL
MCU Microcontroller
EP Elementary Part
TS Test Segment
About YOGITECH

Yogitech, founded in 2000, is a leading provider of services and solutions to silicon vendors and system integrators to help them meet their functional safety challenges. Yogitech’s faultRobust technology features different product lines - fRMethodology, fRTools, fRTrainings, fRIPs and fRSTL - coherently integrated into the mission to be the one-stop shop for functional safety. Yogitech’s customer portfolio includes major semiconductor vendors and major system integrators in different safety application domains, such as the automotive, industrial, medical and railways. Yogitech is member of the ISO 26262 Working Group, and the lead author of Part 10 – Annex A "ISO26262 and microcontrollers”. In addition, it is an active member of the newly-formed ISO 26262 Semiconductor Working Group.

YOGITECH S.p.A.
Via Lenin 132/P - 56017 San Giuliano Terme
Loc. San Martino Ulmiano (PI)
ITALY
TEL: +39 050 86351
FAX: + 39 050 861870
www.yogitech.com
contactus@yogitech.com

Date of Publication:
October, 2014