D2.5 Safety-critical applications regulations compliance handbook

Deliverable Id:	**D2.5**
Deliverable name:	**Safety-critical applications regulations compliance handbook**
Status:	**Final**
Dissemination level:	**Public**
Due date of deliverable:	**2023-02-28 (M30)**
Actual submission date:	**2023-02-23**
Work package:	**WP2 "Specifications & Methodology"**
Organization name of lead contractor for this deliverable:	ETH Zürich
Authors:	Andrea Cossettini, ETH Frank K. Gürkaynak, ETH Luca Bertaccini, ETH Michael Rogenmoser, ETH David Faura, Thales Jérôme Quévremont, Thales Ruben Lorenzo, BSC Sergi Alcaide, BSC Jaume Abella, BSC Carles Hernández, UPV Ilya Tuzov, UPV Ernst Wehlage, PLC2 Alexander Flick, PLC2 Artur Kaufmann, BEEA
Reviewers:	Jaume Abella, BSC Ramon Canal, BSC Martin Matschnig, Siemens

Abstract:
D2.5 "Safety-critical applications regulations compliance handbook" presents a guidance to the qualification processes of FRACTAL systems. Specifically, the document presents an introduction to safety standards and discusses their impact on the use of Artificial Intelligence (AI), different application domains (industrial, automotive, and medtech) and different platforms (Versal, PULP, and Noel-V).

Contents

Acknowledgement

ECSEL Joint Undertaking
Electronic Components and Systems for European Leadership

This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 877056. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Italy, Austria, Germany, Finland and Switzerland.

1 History

Version	Date	Modification reason	Modified by
0.0	2022-10-21	Agreed template	ETH
0.1	2023-01-20	Complete version	Authors
0.2	2023-02-03	Reviewed version	Reviewers
0.3	2023-02-17	Reviewer's remarks solved	Authors
1.0	2023-02-23	Final cleanup, delivered version	ETH

Table 1 – Document history

To cope with the high number of contributors, this document has been edited online. The Microsoft Sharepoint solution has been selected to keep information under EU legislation. This solution offers a reduced feature set compared to a "regular" Word editor. For instance, we have not been able to build a table of references and have instead used footnotes.

2 Summary

Task T2.3 is described in FRACTAL DoA as "This task focuses on the continuous monitoring and study of regulations to allow cognitive fractal systems be used for safety-critical applications. The aim is to produce simplified documentation (handbook type) that will allow value chain actors to have a macroscopic understanding of the regulatory framework and qualification process for safety-critical applications. This task is central and tightly coupled with the tasks of WP7 and WP8."

Accordingly, D2.5 "Safety-critical applications regulations compliance handbook" presents an overview to the regulatory framework concerning FRACTAL systems and building blocks. In order to offer guidance to the qualification processes, the document presents an introduction to safety standards and discusses their impact on the use of Artificial Intelligence (AI), different application domains (industrial, automotive, and medtech) and different platforms (Versal, PULP, Noel-V).

D2.2 "Methodologic Framework (a)" presented the development methodology related to several topics and included as well safety-critical applications. Since D2.4 "Methodologic Framework (b)" is an update of D2.2 (thereby replacing it) and is prepared concurrently to D2.5, the contents of D2.2 related to safety-critical applications have been moved into D2.5. Specifically, the former sections 6.1.1, 6.1.2, and 6.3 of D2.2 are now part of D2.5 (Sect. 4.1, Sect. 7, and Sect. 6.1, respectively).

A list of abbreviations is available at the end of the document.

3 Introduction

Safety standards and regulatory frameworks are key aspects for FRACTAL nodes. In fact, the cognitivity and re-configurability properties of FRACTAL nodes allow them to automatically adapt and change mode of operation according to the environment conditions. Therefore, their intended mode of operation is not known a-priori, challenging their certification before market release.

The goal of this deliverable is to offer the reader a macroscopic understanding of the regulatory framework and qualification process for safety-critical applications. We define the steps to be taken to possibly certify a FRACTAL system.

The document is structured into four main sections. Section 4 presents an introduction to the regulatory framework for the certification of safety-critical products, also taking into account AI-based systems and components. Section 5 discusses domain specific aspects, with a focus on industrial, automotive, and medtech industry. Section 6 presents platform-specific considerations, analyzing three main platforms that are used in the context of FRACTAL: Versal, PULP, and Noel-V. Finally, Section 7 offers a guidance for the practical application of the aforementioned safety standards within FRACTAL.

4 Regulatory framework guidance for the certification of safety-critical and AI products

4.1 Introduction to safety standards

IEC 61508[1] is the main European standard for functional safety. It provides a generic approach to all activities related to the safety lifecycle of safety-related Electrical/Electronic/Programmable Electronic (E/E/PE) systems that will be used to perform safety functions.

4.1.1 Functional safety

According to the IEC 61508 standard, functional safety[2] is the subset of the overall safety relating to equipment and its control system which depends on the correct operation of its safety related system which implements the required safety function. Functional safety ensures that there are no unacceptable risks and addresses the ability of safety-related systems to perform their safety functions as intended.

4.1.2 Objectives and application domain of the IEC 61508 standard

IEC 61508 standard was first published in the period 1998-2000. The standard was updated and improved with a second version in 2011. The general objective of this standard is to permit the development of E/E/PE safety related systems that will perform safety functions in accordance with the specification. For this, the standard proposes an operational approach to harness the E/E/PE safety-related system, starting from the study of the safety requirements and taking into account all stages of the system lifecycle.

The first intention of the working group was to produce a generic standard to be used as the basis for drafting other product and application sector international standards. However, in practice, IEC 61508 is used directly by industries.

4.1.3 General structure of the standard

In order to cover all aspects related to E/E/PE systems, the general structure of standard 61508 is organized in 7 parts (the references to the different parts can be found in the referenced IEC document[3]). The parts 1, 2, 3 and 4 are normative, while the parts 5, 6 and 7 are only informative, offering advice and guidance to apply the normative parts. Part 1 sets the requirements for the certification documentation and

[1] IEC 61508-1 Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 1: General requirements, 2011
[2] MTL application note, Functional safety, An introduction to Functional safety and the IEC 61508 series, 2002
[3] IEC Functional safety Essential to overall safety, 2019; https://www.iec.ch/basecamp/functional-safety-essential-overall-safety

the way to be compliant with the standard. It also defines the technical requirements and the associated management and assessment for achieving safety throughout the entire lifecycle of the system (Figure 1). The parts 2 and 3 cover the requirements for the development of E/E/PE hardware and for the software development while the part 4 provides the definitions used in the standard.

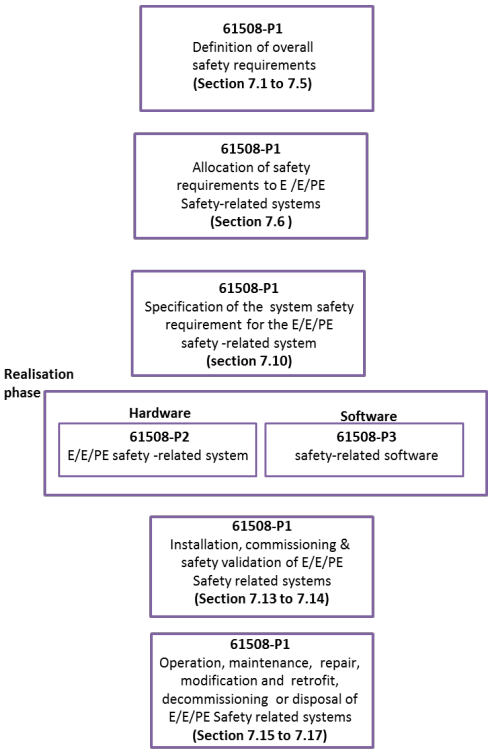

Figure 1 – IEC 61508 safety life cycle model

4.1.4 Risk reduction

The safety assessment in the IEC 61508 standard is based on risk analysis and risk reduction. In the risk analysis, hazardous events are identified and the necessary risk reduction mechanisms are determined. After the specification of the risk reduction, the safety requirements will be assigned an associated Safety Integrity Level (SIL) for each safety function and they will be implemented into one or more safety-related systems to mitigate the identified risk. The SIL[4] indicates a level of safety integrity (between 1 and 4) and its value depends on the level of risk reduction required by the analysis. The SIL may be defined as a measurement of operational safety that determines the recommendations related to the integrity of the safety features to be assigned to E/E/PE systems.

[4] Felix Redmill, "Understanding safety integrity levels", Measurement + Control, Volume 32, September 1999

The standard considers that the risk values are always approximate and the actual reduction obtained by the risk reduction mechanisms can never be determined with precision and cannot be zero (see Figure 2).

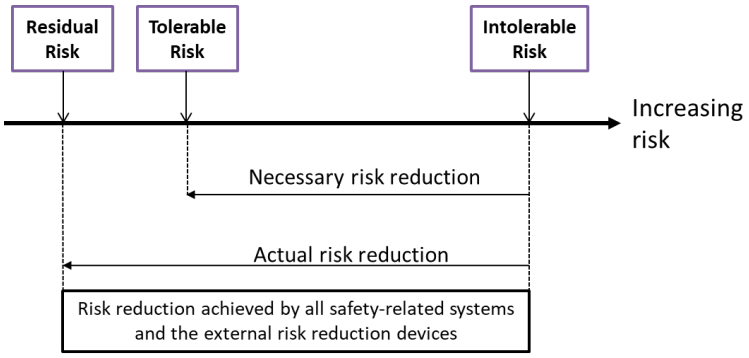

Figure 2 – Risk reduction principle

The risk reduction is linked to the development of the safety functions following the safety standard and it is well described in[5]. It requires the following steps:

- Identify and analyze risks;
- Determine the tolerability of each risk;
- Determine the risk reduction necessary for each intolerable risk;
- Specify the security requirements for each risk reduction and their SIL;
- Design and implement the safety-related function to meet security requirements;
- Validate the safety functions.

4.1.5 IEC 61508 a stand-alone standard and a basis for other standards

IEC 61508 can be a stand-alone standard. It provides suppliers and users of safety equipment with a common framework for the design of products and systems for safety-related applications. All parts of the IEC 61508 are suitable for direct use by the industry.

IEC 61508 parts 1, 2, 3 and 4 are the basic IEC publications in the field of functional safety. One of the responsibilities of IEC technical committees is to base, wherever possible, the drafting of their own industrial or product standards on these four parts whenever E/E/PE safety related systems are within their scope. IEC 61508 is also the basis for other industry standards such as automation[6], railway[7] and automotive domains[8] as shown in Figure 3.

[5] MTL application note, Functional safety, An introduction to Functional safety and the IEC 61508 series, 2002.
[6] IEC 61511-SER Functional safety – Safety instrumented systems for the process industry sector, 2004.
7 CENELEC EN 50126-1 Railway applications - The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS), 2010.
[8] ISO 26262-2 Road vehicles — Functional safety — Part 2: Management of functional safety, 2018.

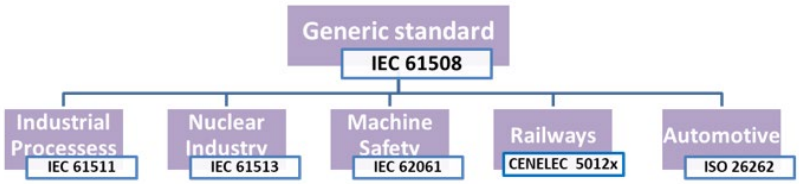

Figure 3 – Industry standards based on IEC 61508

IEC 61508 has a strong impact on the development of E/E/PE systems and multi-sector products concerned with safety. However, it should be noted that specific industry or product standards usually refer only to the specifications of the IEC 61508. Therefore, users will always need to consult IEC 61508. For the application domains addressed in FRACTAL, IEC 61508 forms the basis for:

- ISO 26262 in the automotive sector;
- CENELEC EN 50126, 50128, 50129 and 50159 in the railway sector[9].

4.2 AI-based systems and components

AI-based software is, by construction, at odds with safety-related development processes. Those processes generally build on an architectural design produced to address specific safety requirements, which is progressively refined propagating requirements until its items are simple enough so that they can be realized by specific hardware components and software units. These components or units can be developed, verified and validated on their own prior to integration. Software components are built as control algorithms with their own specifications, whose operation is intended to be valid by construction as long as input data is within the specified range. While those software components are intended to operate on data, such data is not part of the design of the components themselves. Data is used in the development process for validation (testing) activities only. Therefore, the design is not determined by data and its correctness can be assessed against specifications.

AI-based software, instead, is generally built empirically starting from specific implementations (e.g., building on appropriate libraries), without adhering to any specific architectural design, without responding to any safety requirement, and hence without any specification that allows explaining what their functionality is and how hypothetical safety requirements could ever be traced, as it is needed in a safety-related development process. Moreover, AI-based software design is determined by data, hence making data part of the design, which is against current practice for safety-related systems and it is not considered as a valid option by existing safety standards. Last but not least, functional safety standards impose a development process where software is intended to be correct by construction and only some

[9] RF 0015 Reference Document For The Certification Of The Safety Integrity Level Of Products Or Systems According To EN 50126, EN 50128, EN 50129, EN 50657, ISO 26262 and IEC EN 61508 standards, Certifer 2019.

residual risk exists. Instead, AI-based software is stochastic in nature, and subject to accuracy and confidence values. Hence, it can provide erroneous outputs, which is simply not accepted in existing functional safety standards.

Despite all the challenges posed by AI-based software for its use in safety-related development processes, AI-based software has been shown to be the only realistic means to perform specific tasks, such as those related to object detection, with acceptable accuracy and affordable computation costs (despite high). Therefore, AI-based software is expected to be part of the development process and products with safety requirements. In fact, it has already been used in those contexts subject to specific limitations.

Next, we detail some specific approaches followed for the use of AI-based software in safety-related development processes. We provide some examples and introduce some ongoing initiatives towards the broad adoption of AI-based software in safety-related development processes. In particular, we consider the following two scenarios for AI-based software, which we cover next:

- Used during the process, but not being part of the product;
- Used as part of the final product.

4.2.1 AI-based software used during the development process

AI-based software can be used for the development of safety-related products, but not being part of those products. In that case, AI-based software becomes a tool and, as such, it falls in the scope of qualification processes rather than certification processes. Still, the nature of AI-based software is against the needs of safety-related development processes due to the difficulties to qualify software that has not been developed against appropriate specifications. AI-based software is data-dependent in nature, it does not necessarily provide correct outcomes, and its adherence to specific requirements needs to be assessed a posteriori rather than being considered by construction, as it should be.

The generally accepted approach to use AI-based software in the development process of safety-related products (regardless of whether those products include AI-based software or not) consists of placing any safety requirement on non-AI-based software or processes so that AI-based software becomes a companion tool. For instance, some authors[10] have shown that AI-based software can be used to optimize a railway signaling system as long as the final result can be formally verified by non-AI-based tools that check that all safety requirements are met by the final configuration of the signaling system. Note that given that such formal verification tool is qualified, it is guaranteed that the solution provided adheres to its safety

[10] J. Perez, J. L. Flores, C. Blum, J. Cerquides and A. Abuin, "Optimization Techniques and Formal Verification for the Software Design of Boolean Algebra Based Safety-Critical Systems," in IEEE Transactions on Industrial Informatics, vol. 18, no. 1, pp. 620-630, Jan. 2022, doi: 10.1109/TII.2021.3074394.

requirements regardless of whether the solution has been generated following safety-amenable processes, manually, by means of random processes, or using AI-based software.

AI-based software can also be used to devise test cases following the same strategy: the sufficiency and completeness of the test cases need to be assessed a posteriori by non-AI-based means. For instance, some authors[11] have used AI-based software to derive a small number of test cases that achieves 100% MC/DC code coverage for a specific safety-related product. Again, the source of those tests has no impact on the safety of the product as long as it can be checked a posteriori that 100% MC/DC coverage is achieved using non-AI-based tools that adhere to safety-relevant qualification methods.

4.2.2 AI-based software as part of the product

AI-based software can be included as part of safety-relevant products under some conditions within current certification practices. The usual way to do it consists of decomposing the component where the AI-based functionality is included separating the function from the safety monitoring, so that the monitor inherits the safety requirements and the function inherits none. This is doable as long as the monitor itself can meet the safety requirements of the system, which is not always the case. For instance, if the system is fail-safe, hence meaning that a safe state exists, then the monitor can take care of detecting faults and transferring the system to a safe state regardless of the behavior of the AI-based software. Thus, the AI-based software can only impact availability -if it fails often- but not safety. However, in the context of fail-operational systems, the monitor may be unable to preserve the safety of the system by itself and, hence, the AI-based software cannot be relieved from all its safety requirements. Current practice does not allow using such software for that type of product. An example of the latter scenario would be a fully autonomous car where AI-based software controls steering, acceleration and braking. If, eventually, such software fails while driving at 120km/h, and the car does not even have a steering wheel to transfer the control to a hypothetical driver, a monitoring system detecting a failure in the steering may be unable to transfer the vehicle to a safe state on its own given that some driving decisions may be needed to reach such a safe state. For instance, we may want to safely lower the speed of the car, take it out of the driving lanes and stop it, but this takes some time during which steering is still needed.

Whether a safe state exists and hence, whether a safety monitor is viable, is fully system dependent. However, if it exists, functional safety standards already include strategies to decompose safety requirements that would allow using AI-based software in safety-critical systems. In the case of automotive, for instance, ISO

[11] J. Čegiň and K. Rástočný, "Test Data Generation for MC/DC Criterion using Reinforcement Learning," 2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 2020, pp. 354-357, doi: 10.1109/ICSTW50294.2020.00063.

26262 includes the so-called "ASIL decomposition" (ASIL stands for Automotive Safety Integrity Level). Figure 4 illustrates several possible decompositions. Note that ASIL spans from D (highest integrity level) to A (lowest integrity) level, and also includes QM (Quality Managed) components that are those with no safety requirement.

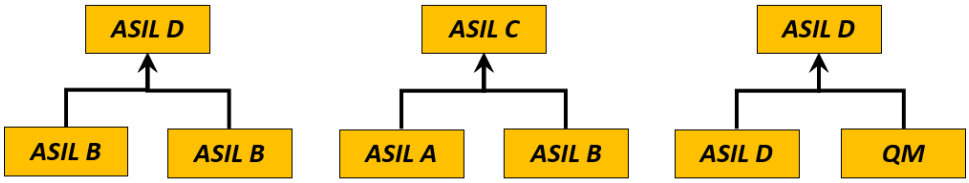

Figure 4 – Examples of ASIL decomposition in the context of ISO 26262

In general, a component with a given ASIL can be decomposed into multiple components that jointly achieve such ASIL as long as they are sufficiently independent. This means that the components are not subject to common cause failures (i.e., simultaneous failure due to a single fault). In Figure 4, on the left, we see how an ASIL D component is decomposed into two ASIL B components that implement the same functionality redundantly as long as they are sufficiently diverse. Similarly, it can be done with an ASIL A and an ASIL B components to jointly achieve ASIL C. Finally, the case of relevance for our discussion is the one on the right, where ASIL D is achieved by decomposing the item into a monitoring item inheriting the safety requirements (in this case ASIL D) and a functional item inheriting no safety requirement (hence QM), which in our case would be the AI-based functionality.

An example of such decomposition so that AI-based software is relieved of any safety requirement was developed in the context of railway interlocking systems more than 30 years ago[12]. In that case, a safe state exists where all trains are made to stop. Since the distance between two trains is always large enough so that they do not meet each other without finding appropriate signaling first, all of them can be made to stop switching all signaling to a given default state. The monitor in that case is in charge of monitoring that the trains' locations and current signaling status do not lead to any hazard with some form of formal verification against predefined rules. Note that if the AI-based software managing signaling fails often, the safety monitor will transfer the system to a safe state often, hence stopping all trains in the railways, which is against availability, but that will not be a safety concern.

Another example, closer to today's reality, is the use of AI-based software in Advanced Driving Assistance Systems (ADAS) in cars, such as for instance, lane keeping assistance. The lane keeping system, as of today, can always transfer the control to the driver upon a failure of the system, given that a non-AI-based monitor

[12] Peter Klein, The Safety-Bag Expert System in the Electronic Railway Interlocking System ELEKTRA, Editor(s): Gian Piero Zarri, Operational Expert System Applications in Europe, Pergamon, 1991, Pages 1-15, ISBN 9780080414386, https://doi.org/10.1016/B978-0-08-041438-6.50005-9.

is capable of detecting the malfunction of the lane keeping system timely and transfer the control to the driver.

4.2.3 Ongoing activities and future prospects

As explained, AI-based software and functional safety standards are not yet compatible, and initiatives are ongoing in both sides of the table to reconcile apparently incompatible principles. The activity on the functional safety side is very abundant and a large number of general and domain-specific relevant standards are under development or have been recently released. However, there is a lack of a record of their practical use, and ultimately, they do not solve the issues posed by current AI software, but instead, provide protocols and approaches to follow with the aim of enabling AI-based software inherit safety requirements in the future. Among those standards we have identified the following: ISO/PAS 21448, ANSI/UL 4600, ISO/TR 4804, ISO/IEC TR 5469, ISO/AWI PAS 8800, ISO/AWI TS 5083, and ISO/IEC AWI TS 6254.

On the AI side, activities towards enabling a broader use of AI-based software, including safety-relevant domains, are also abundant. There is significant progress in identifying some properties on which to build safety principles for a wide variety of applications. Those properties include explainability, transparency, and traceability, to name a few particularly relevant for safety-related products.

In the convergence of both areas, we can also find research initiatives such as the recently started Horizon Europe SAFEXPLAIN project (https://safexplain.eu/). SAFEXPLAIN aims at developing deep learning solutions and safety guidelines enabling the use of AI-based software inheriting safety requirements even for the most stringent safety integrity levels in domains such as automotive, railway and space.

Last but not least, AI-based systems can be regarded as static or adaptive depending on whether they are trained once and forever, or whether they have learning capabilities. Static ones are expected to be the first entry point since their design is fixed and higher control can be exercised over it. Systems with learning capabilities can evolve becoming “one of a kind”. In that context, not only the initial system needs being certified, but also the learning mechanisms by, for instance, only allowing to choose a predefined (and pre-certified) set of configurations.

5 Domain specific considerations

5.1 Industrial/machinery

The industrial domain applies the standard IEC 62061, which derives from IEC 61508 and it focuses on industrial machinery. As the standard IEC 62061 as well as IEC 61058 only consider functional safety of safety-related electrical, electronic, and programmable electronic control systems, the machinery as a system requires more standards for evaluation. For the design and risk assessment of machines, ISO 12100 (as a type-A standard) must be consulted in conjunction with ISO 13849-1 and -2. Furthermore, the type-C standard (safety standards) for a specific machine or a group of machines must be consulted in the machine design and the risk assessment of ISO 12100 as well. For instance, part 4 of the DIN EN ISO 3691-4 (Industrial trucks – Safety requirements and verification) describes driverless industrial trucks and their systems. Consequently, it was applied for the shuttle system in UC8, as the shuttles are identified as automated guided vehicles in a controlled environment.

The considerations in this subsection refer to the application in the specific scenario of UC8, which can't be applied in general to the standards and it would lead to an inadmissible product design. The reference concept for the safety services of UC8 can be read in D4.4 – FRACTAL SAFETY SERVICES with assignment to the carried-out components from the FRACTAL project.

5.1.1 Workflow for the application of the standards

As shown in Figure 5, all considerations start during the design process of a product/ machine. This means that all risks must be identified, classified and documented including the measures taken by the manufacturer for the CE marking of their machine. This process is mandatory for every manufacturer in Europe. The classification can be achieved with a scoring system called risk potential, which depends on the parameters degree of possible harm, likelihood of occurrence, frequency of exposure and number of persons at risk. High values mean high risk in terms of high probability of injury and severe harm.

Regarding UC8 as example for a typical workflow, the application of following standards can be justified by means of system size and complexity (red boxes in Figure 5). Starting from the top with ISO 12100 for the design process and evaluation of potential risks as base of the risk assessment, the relevant part was applied from ISO 13849-1 and -2, where the identified safety functions must follow the compliancy of these standards. The main reason for this decision is the separation of the targeted safety functions in UC8. In the state-of-the-art application, one safety function handles the access control. This access control observes all entry points in the system and handles the requests to gain access.

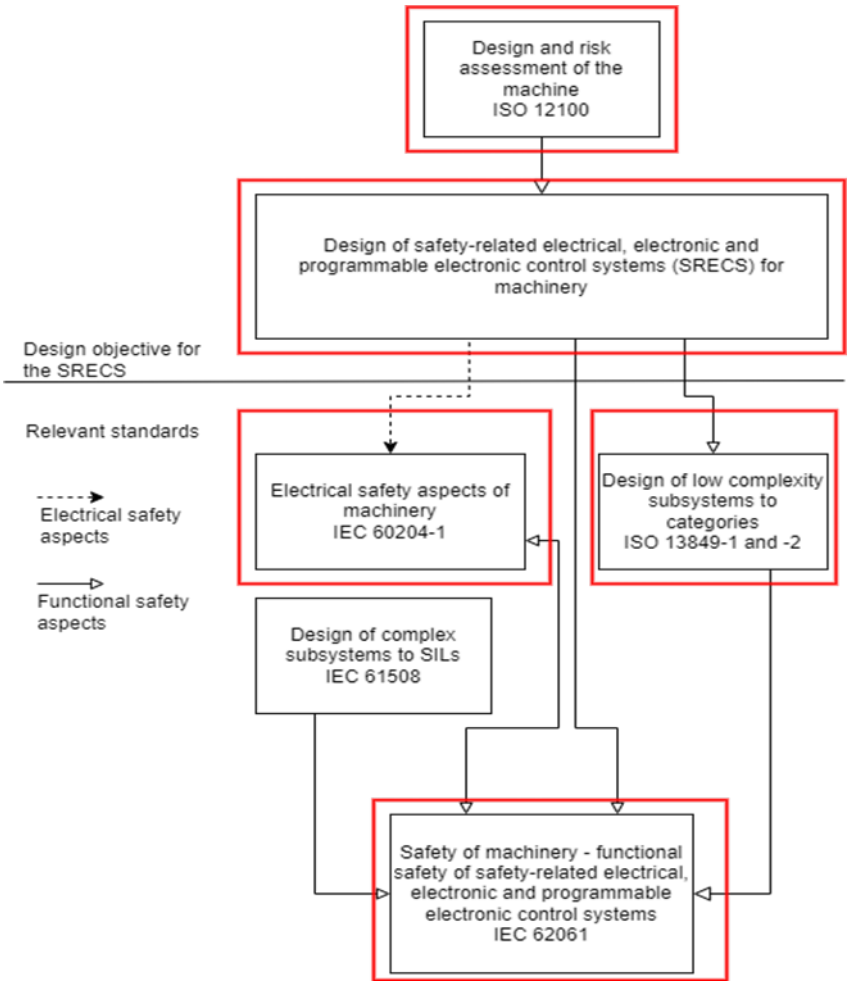

Figure 5 – Relationship of IEC 62061 to other relevant standards

This feature is coupled to a failsafe-PLC in each shuttle, where shuttles are placed in a safe state by selectively shutting them down when access is requested. In the new solution, the access to the system is not anymore connected to the shuttles in the system and the safety functions become less complex by that context. Additionally, the standard IEC 60204-1 must be considered for the electrical aspects of the machine, which is applied for component sizing, general wiring and fuse protection rules.

5.1.2 Performance level of safety functions

Through the application of the ISO 13849-1 and -2 standards, it is mandatory to identify the safety functions for the machine through the evaluation of each potential risk and the application of mechanical, technical and informative measures. Performance level (PL) values are defined from "a" to "e", as shown in Figure 6, and define the required PL, where "e" inherits a high risk in terms of severity, probability and frequency and "a" a low risk.

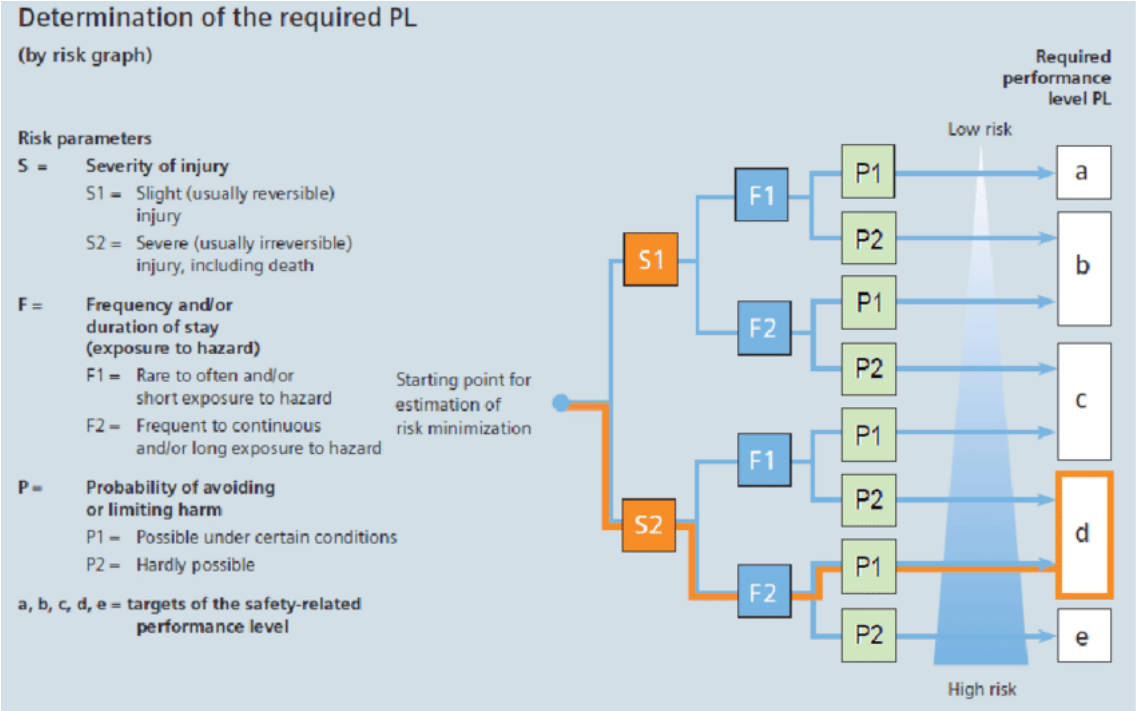

Figure 6 – The risk elements evaluation and PL requirements determination according to ISO 13849-1[13]

In context of UC8, the consideration of two main functions can be mentioned here:

1. Safe wireless communication between edge nodes;
2. Realization of a safety control service on a platform alongside other components.

The first point is used to send safety relevant information from "outside" of the system to the shuttles "inside" and gain flexibility regarding disturbing entities, like maintenance staff entering the system and affect negative impact on the flow from operational perspective. The second point is an approach to apply all functionalities on one platform, which inherits the typical control services, other components and additionally the safety control services to replace a classical failsafe-PLC in a subsystem like a shuttle.

The required safety functions will be realized in the safety control services of the shuttles and they can be defined as the distance estimation to a detected person in the system by a camera module and the speed monitoring of the travel motors. These were the results from the considerations of the risk assessment for the application of the FRACTAL components in the shuttle system. The whole approach depends on

[13] Safety integrity level (SIL) versus full quantitative risk value - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/The-risk-elements-evaluation-and-PL-requirements-determination-according-to-ISO-13849-1_fig1_291295542 [accessed 17 Jan, 2023]

platform specific considerations described in Section 6 and the regulatory framework guidance in Section 4. For preparation and verification of the worked-out functions, tools like the software assistant "SISTEMA[14]" can be consulted as preparation for the certification process in Germany.

5.2 Automotive

ISO 26262 is the application of ISO 61508 in the automotive field. It inherits its concepts and specializes them in the specific context as described in Section 4.1.5. Sections 4.2.2 and 4.2.3 described how the theoretical application of the AI software within the product is still an open research field. However, the application is theoretically possible thanks to the principles of decomposition which are based on the ASIL levels (Automotive Safety Integrity Level) which are assigned to the subsystems of the architecture. The concept of ASIL is therefore fundamental in this context, but it is only one aspect of the ISO to which focuses our attention. The development of security systems that include AI software is, in fact, composed of a series of techniques and processes which cannot be ignored and which therefore define the product's security life cycle, within which the concept of ASIL becomes the pivot point.

5.2.1 ISO 26262 functional safety standard for the automotive sector

Safety-critical automotive applications have a high demand for functional safety and reliability. So, Functional Safety becomes a fundamental requirement in the automotive systems to guarantee a tolerable level of risk.

ISO 26262: Road Vehicles—Functional Safety is the automotive industry standard designed for safety-related systems for series production passenger vehicles that are equipped with one or more E/E/PE subsystems.

According to ISO 26262, functional safety is defined as the "absence of unreasonable risk due to hazards caused by malfunctioning behavior of electrical/electronic systems".

These malfunctions can be classified into two types of failures:

- **Systematic failures**: "failure related in a deterministic way to a certain cause, that can only be eliminated by a change of the design or of the manufacturing process, operational procedures, documentation or other relevant factors" (ISO 26262- part1). Activities to eliminate failure can be addressed by tracking and traceability in the development process.
- **Random failures**: "failure that can occur unpredictably during the lifetime of a hardware element and that follows a probability distribution" (ISO 26262-part1). Random failures are typically addressed during the design and

[14] https://www.dguv.de/ifa/praxishilfen/practical-solutions-machine-safety/software-sistema/index.jsp

verification with the introduction of technical features called safety mechanisms (i.e. "a technical solution implemented by E/E functions or elements, or by other technologies, to detect and mitigate or tolerate faults or control or avoid failures in order to maintain intended functionality or achieve or maintain a safe state" [ISO 26262- part1]).

The ISO 26262 safety lifecycle (Figure 7) addresses the main safety activities during the concept phase, product development, production, operation, service and decommissioning.

Figure 7 – ISO 26262 lifecycle

The key management tasks that are performed throughout the lifecycle are the following:

- Planning and coordinating the safety activities;
- Monitoring the progress of the safety activities;
- Evaluate functional safety by performing confirmation measures.

The main principles of ISO 26262 are:

- The standard is designed for distributed development: all participants of the supply chain are now called to support and enable functional safety and reliability requirements. The responsibility of addressing functional safety has been distributed between OEM (car maker) and Tiers (automotive suppliers);
- ISO 26262 safety lifecycle is based on the V-model cycle (SLC) approach;
- ISO 26262 provides a risk-based approach to assess Automotive Safety Integrity Level. ASIL Level classification represents a framework for companies to develop functional safety systems;
- A quantitative approach to risk assessment: ISO 26262 defines metrics (SPFM, LFM, PMHF) for random HW failure evaluation.

5.2.1.1 ISO 26262 Safety Lifecycle

V-model is a cascade model from project definition to system production (Figure 7). It provides a guide for designing and implementing the project.

The goals of the V-model are minimization of the risks, improvement of quality, reduction of total cost and better communication between all stakeholders.

The V-model can be tailored, according to the type of system to be developed:

- New development: the overall safety lifecycle shall be implemented;
- Modification: an impact analysis can be performed to tailor the safety lifecycle.

For old systems it is possible to partially skip the safety lifecycle showing that the system under study is a carryover from other existing vehicles, through the proven in use argument.

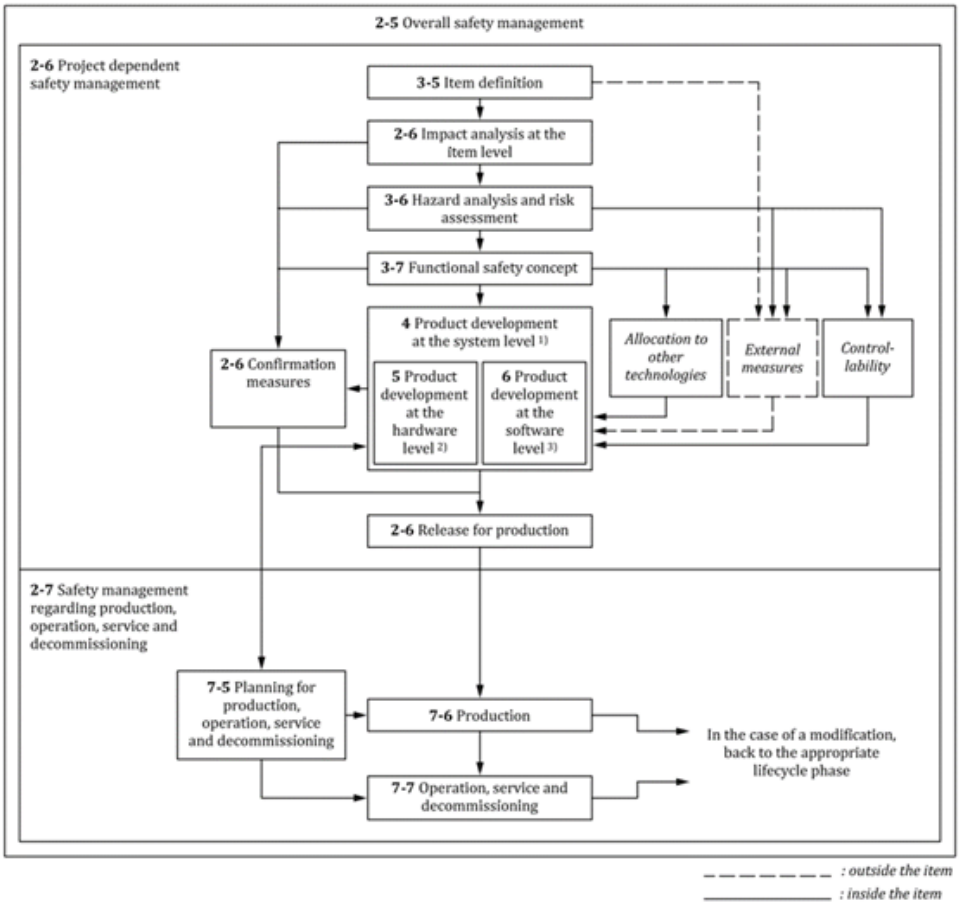

Figure 8 – ISO 26262 Safety Lifecycle

The development model shown in Figure 8 can be divided into the following phases:

- *Concept phase:* during the initial phase, analyses are performed to describe user needs and requirements documents are created to describe ASIL and

safety goals. Furthermore, some tests are designed. So, the first activity to be considered along the ISO 26262 Safety Lifecycle is the Hazard Analysis and Risk Assessment (HARA), to address safety goals and various design and performance requirements in the early stages after item definition.
The HARA method aims at identifying and categorizing hazardous events of items, and at specifying safety goals and ASIL levels related to the prevention or mitigation of the associated hazards to avoid unreasonable risk.
ASIL levels assigned to Safety Goals are functions of Severity, Controllability, Exposure as depicted in Figure 9:

Figure 9 – ASIL level

- *System and Architecture design*: during this stage, a specification set is filled in with detailed components, and a high-level architecture is designed, for describing the links among all the components. Integration tests are designed in this stage
- *Module design*: the development goes on through a low-level design phase for all elements, including HW and SW development of requirements and architecture, in order to design single module and unit tests
- *Implementation/Coding*: this stage is the bottom of the V-model; all previous specifications are converted into codes (for SW/FW) and into PCB layout design (for HW) and the system is prepared for testing.
- *Unit testing*: during this phase the unit is tested for checking and eliminating bugs and faults. In software field software design, coding (and code optimization) and software integration compose the software-in-the-loop test.
- *Integration testing*: this stage verifies the functionality across the components of the system and their integration. Software integration and hardware/software integration compose the processor-in-the-loop test.
- *System testing*: during this phase performance of complete system is evaluated. Hardware/software integration and vehicle integration compose the hardware-in the-loop test.
- *Production, Operation and Maintenance*: The system is ready for the production, and issues against functional safety during production phase shall be addressed. Moreover, during operation phase, maintenance is implemented to repair possible issues and upgrade the system.

ISO 26262 requires evidence that the mandatory activities were performed as required. In particular, evidences are based on nearly 130 work products, that can

be represented by a collection of documents, a single document, a process, an activity, a signature or an audit.

Work products format should be appropriate to the work product's content:

- Data files, models, source code, etc.
- May include currently existing documents
- Several work products may be in a collection of documents, a single document, a process, an activity, a signature or an audit

Work Products reviews are required by ISO 26262 according to the ASIL level assigned to the project.

5.3 Medtech

The essential performance of an active medical device or system has a direct impact on patient safety and users. Consequently, safety assessments address potential hazards that might arise during its clinical use. Functional safety guarantees that the function of a device is maintained, and the system can switch into a safe state in the event of a fault. Thus, it plays a critical role for the manufacturers, importers and distributors of medical devices.

Standards are an integral part of product design and development, and they are certainly important in medical applications. Their application comes from the need to translate the regulatory frameworks request into technical requirements to be applied to medical devices products.

5.3.1 Regulatory Framework for Medical devices

The medical device industry is highly regulated worldwide.

Key regulatory standards for medical devices include:

- EU Medical Device Regulation (MDR) — EU standard which replaces Medical Devices Directive in 2020.
- FDA regulations — U.S. standards for medical device compliance.

They establish the regulation manufacturers or distributors (e.g., importers) of medical devices must comply with to place their products on the market.

While for the US Market the applicable regulatory framework is regulated by the 21 CFR 820.30, in the European Union it is established by the MDR, Medical Device Regulation (EU) 2017/745.

The full applicability of MDR Regulation 2017/745 takes effect starting from 26 May 2021, however a derogation period is foreseen in which some of the devices compliant with the directives can continue to be legitimately placed on the market until 26 May 2024.

The MDR regulation identifies the following obligations for manufacturers:

- have risk management systems in place;
- have quality management systems in place.

Hence, the identification of a regulatory technical framework, Figure 10, for the management of quality processes and functional safety of products and processes.

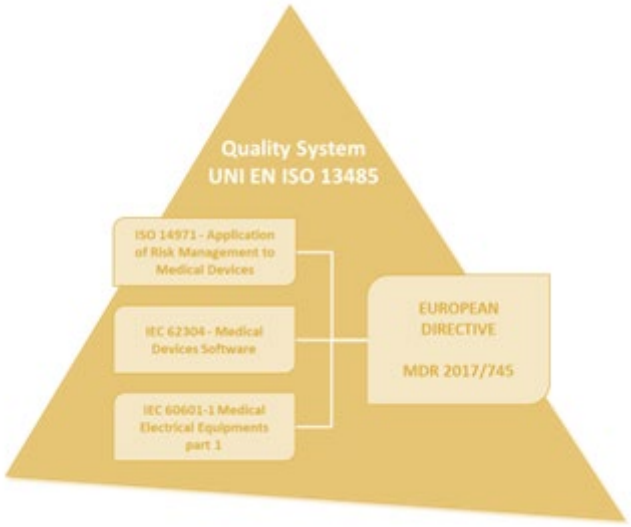

Figure 10 – EU Medical Device Standard Framework

Both MDR and CFR focuses on main aspects in the development of a medical device:

- Design Control;
- Risk Management.

5.3.1.1 Design Control

Design controls for medical devices are regulated by the FDA under 21 CFR 820.30. They must be implemented by manufacturers of class II or III medical devices (and some class I devices). ISO 13485 implements a set of very similar regulations (nearly exactly the same, actually).

The design control process, according to 21 CFR 820.30, follows a set of practices and procedures that help medical product developers:

- Manage quality;
- Ensure each product meets all requirements;
- Prevent potential issues or recalls in the future.

Medical design control stages from both the FDA and the ISO consist of:

- Design & development planning;
- Design inputs;
- Design outputs;
- Design review;
- Design verification;

- Design validation;
- Design transfer;
- Design changes;
- Design history file.

The regulations define each stage in a linear process. But each requirement actually is part of a dynamic process that can change and repeat. This is known as the design and development planning model (Figure 11).

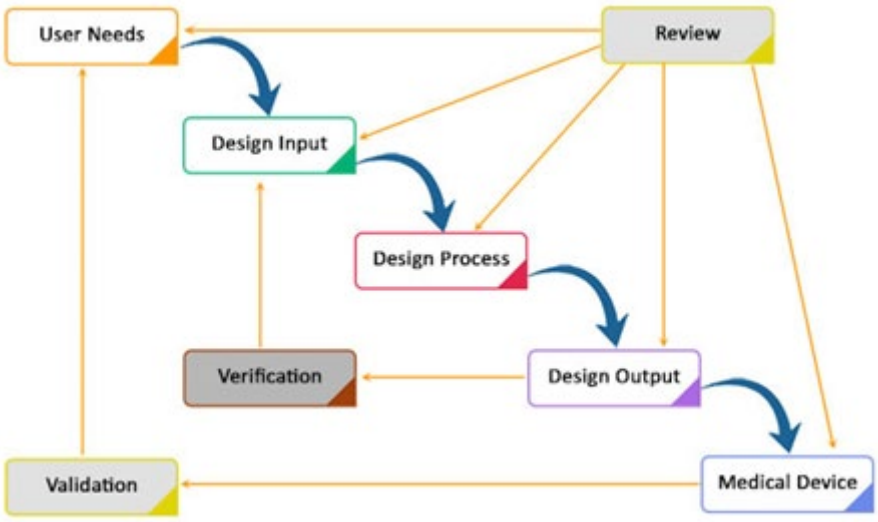

Figure 11 – Design and development planning model

ISO 13485 specifies requirements for a quality management system where an organization needs to demonstrate its ability to provide medical devices and related services that consistently meet customer and applicable regulatory requirements. ISO 13485 can also be used by suppliers or external parties that provide product, including quality management system-related services to such organizations.

Requirements of ISO 13485 are applicable to organizations regardless of their size and regardless of their type except where explicitly stated.

If applicable, regulatory requirements permit exclusions of design and development controls, this can be used as a justification for their exclusion from the quality management system. These regulatory requirements can provide alternative approaches that are to be addressed in the quality management system. It is the responsibility of the organization to ensure that claims of conformity to ISO 13485 reflect any exclusion of design and development controls.

5.3.1.2 Medical Device Risk Management

Risk Management has always played a key role in the medical devices sector, representing not only the first step for the implementation of product requirements, but also a constant flow in the life cycle of the product itself, as highlighted in Figure 12.

Figure 12 – Product Life Cycle

While the 21 CFR 820.30 requires a generic application of a risk management process, risk management is a key requirement for ISO 13485. It is addressed by the ISO 14971 risk management standard.

ISO 14971 satisfies the risk management requirement for IEC 60601-1 for medical electrical equipment and systems. It is a helpful tool for manufacturers in identifying and controlling the risks associated with their medical devices, but also evaluating interactions with other devices.

When it comes to medical device manufacturing, patient safety greatly depends on the quality and consistency of medical products. Both normative and legal requirements focus on the principle of single-fault safety, which means that a single first(-occurring) fault must not cause any hazards for either users or patients or result in unacceptable risk levels.

Basically, single faults can occur anytime and anywhere—throughout the control circuit, its parts, and components and in the software. They cannot be predicted. To keep health risks for users and patients to a minimum, high requirements are imposed on the essential performance and safety of medical devices by both the applicable laws and standards.

The process risk management (Figure 13) includes:

- Risk Management Planning;
- Risk Analysis;
- Risk Evaluation;
- Risk Controls;
- Overall Residual Risk Acceptability;
- Risk Management Review;
- Production & Post-Production Information.

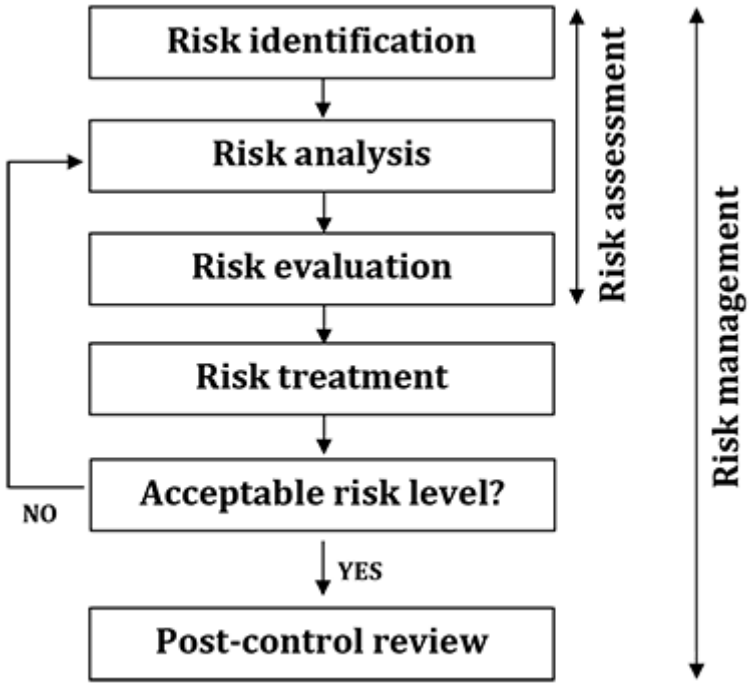

Figure 13 – Risk Management Process Flow Chart

6 Platform specific considerations

6.1 Versal ACAP

The Versal ACAP device family has been developed with safety focus. It already contains features to support different standard requirements from, e.g., ISO 26262, IEC 61508 and ISO 13849. For instance, the underlying building blocks exhibit various error detection and mitigation features from Error Correction Code (ECC) protection to self-test and more. These basic features are further supported by control level test and detection through system test libraries (STL) that can be deployed in various life-cycle stages from powerup to mission mode run-time.

The heterogeneous nature of the devices supports multiple ways of grouping features together. The typical split of safety related elements from more application focused components is an immanent part of the system design and needs to be considered along the guidance by the vendor. The release of reference designs and more detailed information is still pending as of the preparation of this report.

The following derives the approach to safety centric designs along the basic considerations for FRACTAL node implementation. The core information and relevant documentation is accessible only through the vendor directly as outlined in Section 6.1.1. While device features are documented under different focus within other FRACTAL deliverables, the relevant definitions (Section 6.1.2) are listed to enable an informed discussion of the safety approach as recommended by the vendor. Potential variants of partitioning are documented for separation along domains (Section 6.1.3). The application of these concepts for the FRACTAL node platform is outlined in Section 6.1.4.

6.1.1 Versal ACAP functional safety origin

Functional safety of the Xilinx Versal ACAP device families are based on the concepts that are already successfully deployed in Zynq Ultrascale+ MPSoC and inherit the specific core certifications. Due to the time for such a new introduction to be fully rolled out, the qualification and publishing of all the features has not been documented up to now and Xilinx's plans extend well into 2024. Section 6.1.5 reports the available information with the proposed methodology for the FRACTAL project.

For more accurate and up to date information, PLC2 and related partners will track the Xilinx's Functional Safety Lounge. This collection, available under NDA through https://www.xilinx.com/products/technology/functional-safety.html, provides safety related guidelines and tools such as:

- Certified Hardware and Software Design Tools
- Functional Safety Certificate and Reports
- Functional Safety Package
- Certified Methodologies

- Certified IPs
- Reliability Reports
- Xilinx Certification Papers

This Xilinx Functional Safety Lounge provides such information for all device families. Specific information on Versal ACAP is still only starting to become available.

The proceeding as described here applies the device specific information to a defined platform setup for FRACTAL that is held against certification requirements for specifically IEC 61508 and ISO 26262. To achieve this, the safety related items are listed with relation to safety impact.

6.1.2 Versal ACAP safety features

Versal ACAP has been developed to support the Functional Safety Standards IEC 61508, ISO 26262, SIL 3 in IEC 61508, ASIL D in ISO 26262 using decomposition while customers can use further artifacts from IEC 61508 and ISO 26262.

For the physical device, there exist 3 isolated domains in Versal:

- FPD (Full power domain)
- LPD (low power domain)
- PL Domain (programmable logic)

Additionally, domains are defined in Versal which need to be used in a shared context:

- PMC (Platform management controller)
- DDR (DDRAM memory controller)
- NoC (Network on Chip)

The Processing system provides a set CPU that are the potential access contenders:

- Cortex-A72 in the FPD (dual core)
- Cortex-R5 in the LPD (dual core)
- PMC: (hardened Xilinx MicroBlaze processors) in the LPD for boot and platform management

A Versal design typically scales across these groups. Resource sharing must be performed through well-structured intercommunication capabilities that are provided by the architecture. This also requires customer use-case dependent isolation where specific protection mechanisms need to be set up.

A typical isolation concept may be structured along the Versal power domains (supply voltages) with the respective blocks listed here:

- PS FPD (APU, CCI)
- PS LPD (RPU, OCM, dedicated Peripherals)
- NoC (Network on Chip for memory and streaming interfacing)
- PL+AIE+CPM (programmable hardware, programmable engines and I/O)

- PMC (platform management: boot, power, initialization)
- Battery (RTC, BBRAM)

The following section deploys these domains regarding the units and dependent configurations in terms of Safety Channels as positioned by AMD-Xilinx.

6.1.3 Systematic Versal Functional Safety

6.1.3.1 Versal Safety Components

The first safety component group is the LPD which includes the Cortex-R5 processors and is referred to as the "Real Time Channel". This grouping of functional blocks as shown in Figure 14 involves the following blocks:

- ARM Cortex R5 Cluster (2 R5 CPU's, TCM 128KB/Core, 32KB I/D Cache)
- Split mode: independent R5 cores running independent code
- Lock-step mode: both cores running same code, with temporal and physical diversity
- Logic Built-In-Self-Test (LBIST)
- Global Interrupt Controller (GIC) for the RPU
- On Chip Memory (OCM) 256KB can be used in addition with TCM
- Dedicated I/O: GbE, CAN FD, SPI, I2C, GPIO, UART, WDT, TTC
- Processing System Manager (PSM), controls PS power Islands
- Dedicated Direct Memory Access controller (LPD DMA)
- Network on Chip (NoC) Port for DDRAM access
- Xilinx Memory Protection Unit (XMPU), customizable isolation

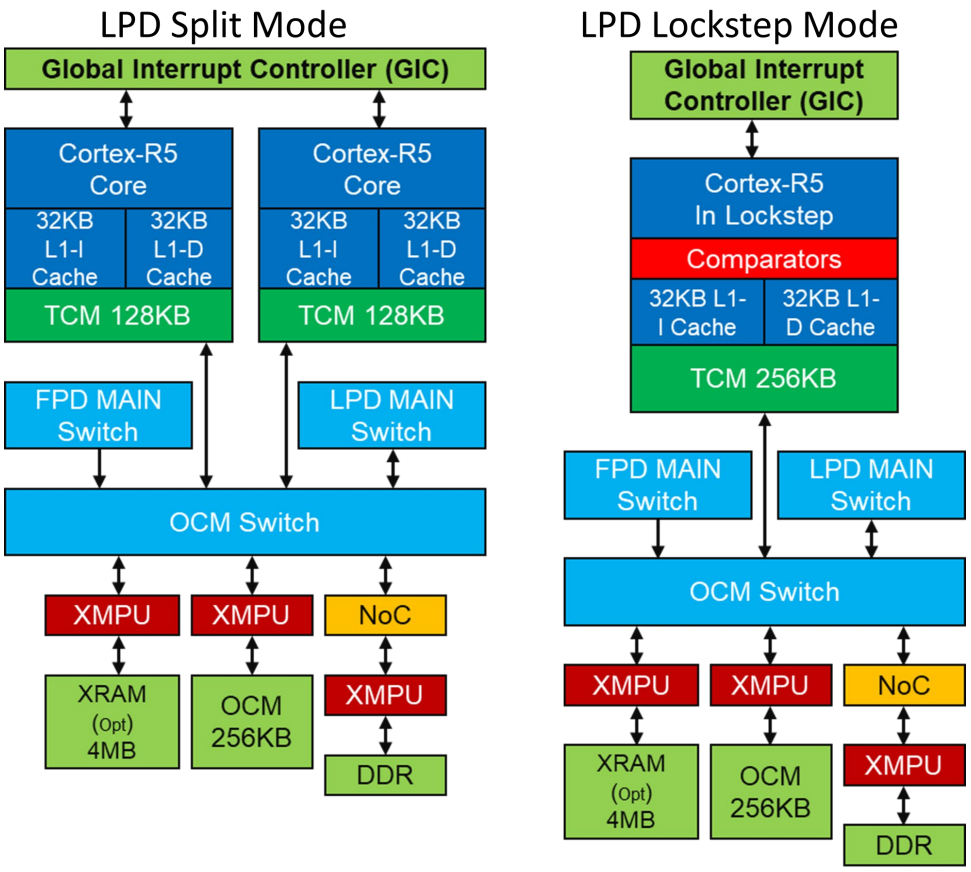

Figure 14 – Low Power Domain Components[15]

For random HW fault integrity, ASIL C / SIL 2 is the target for the LPD domain and for systematic safety SC 3 / ASIL D can be achieved for this domain with appropriate requirements of initialization and run-time protection. When using shared resources like PL or AIE accelerators these needs to be developed in responsibility of the user defined criteria as they exist outside of the LPD.

The second safety component group is the FPD ("Application Channel"), depicted in Figure 15, which includes the Cortex-A72 processors, Cache coherency management and the NoC:

- ARM Cortex A72 Cluster (2 A72 CPU's, 32KB I/D L1 Cache, 1MB L2 Cache)
- Neon + FPU
- Cache Coherent Interconnect (CCI)
- System Memory Management Unit (SMMU)
- Windowed Watchdog Timers
- Network on Chip (NoC) Port

[15] FSWG'20, Session 6: Versal Functional Safety Architecture, available through Functional Safety Lounge, courtesy of AMD-Xilinx.

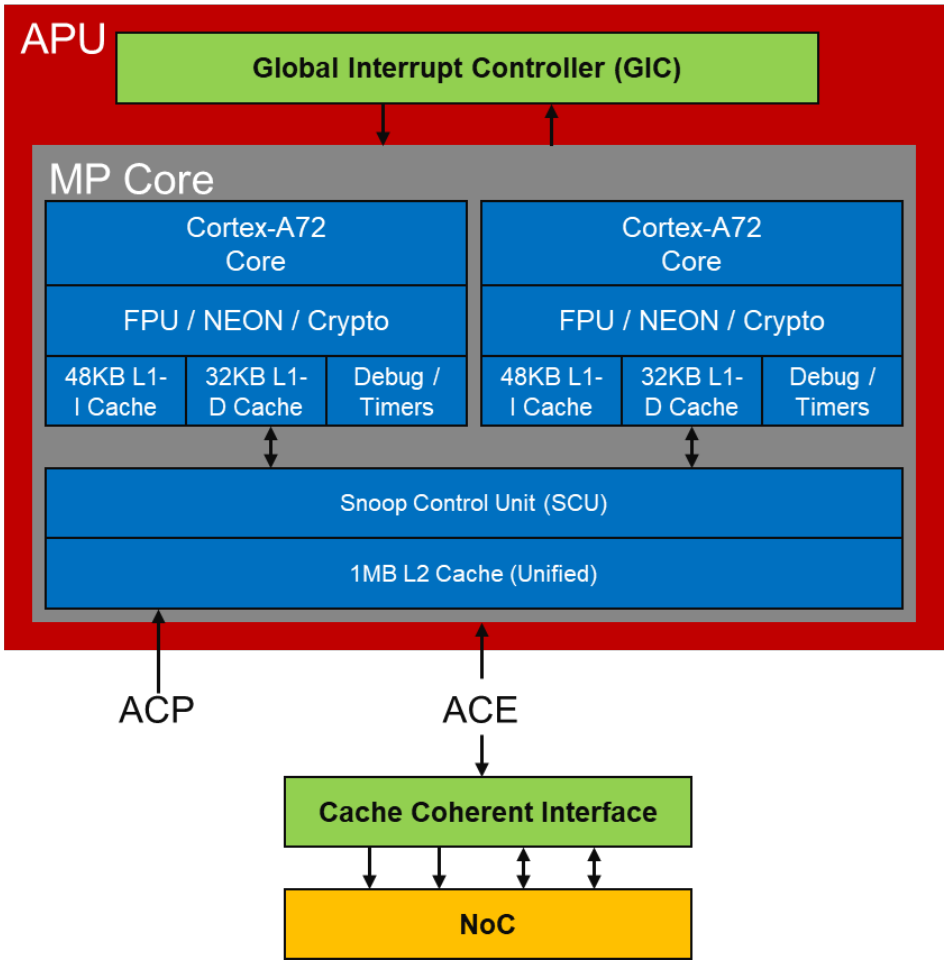

Figure 15 – Full Power Domain Components[16]

For random safety, the target is to support SIL 2 / ASIL B[17] for the FPD domain and for systematic safety SC 3 / ASIL D can be achieved for this domain with appropriate requirements of initialization and run-time protection.

For systematic safety, SC 3 / ASIL D can also be achieved for the PL domain modules programmed in the PL. For random safety, the target is to support SIL 2 / ASIL B for the DDRAM controller part and for systematic safety SC 3 / ASIL D can be achieved.

6.1.3.2 Versal Functional Isolation Techniques

The aforementioned general grouping along the power domains is supported by physical units within the respective domains to isolate access to defined components.

[16] FSWG'20, Session 6: Versal Functional Safety Architecture, available through Functional Safety Lounge, courtesy of AMD-Xilinx.
[17] Actual target in the certification process at AMD-Xilinx

The Versal ACAP platform includes several of these protection units of two distinct types:

- XPPU (Xilinx peripheral protection unit)
- XMPU (Xilinx memory protection unit)

The XPPU provides peripheral protection for

- PMC APB programming Interface
- NPI - NoC Programming Interface
- LPD APB, AXI Programming Interface
- FPD AXI Programming Interface
- LPD Peripherals
- PMC Peripherals

The XMPU provides memory protection for the memory access of

- DDRAM
- XRAM (Accelerator RAM)
- OCM
- PCM RAM

and can differentiate up to 16 regions of an address range for each XMPU.

Isolation on the transaction level is provided by ARM Trustzone layer when using the Linux OS on the Cortex-A72 separating secure and non-secure accesses along with OS process privilege levels.

Further memory protection also exists with MMUs: The Cortex-A72 cores includes a MMU and a System-MMU, which is integrated in the FPD to allow for hypervisor OS management.

6.1.3.3 Versal Safety Channel Architecture

A Functional Safety Channel is a set of hardware and software resources that implements the entire Safety Function. Any Functional Safety Channel has a sensor, logic solver and an actuator.

A particular safety channel can be defined in the LPD with the RPU processor (Figure 16), where these R5 CPUs can be initialized to run in the lock-step or split mode for the solver task. Sensor inputs may be sourced from peripherals in the LPD or PL and actuator outputs may be sent to peripherals in the LPD or PL.

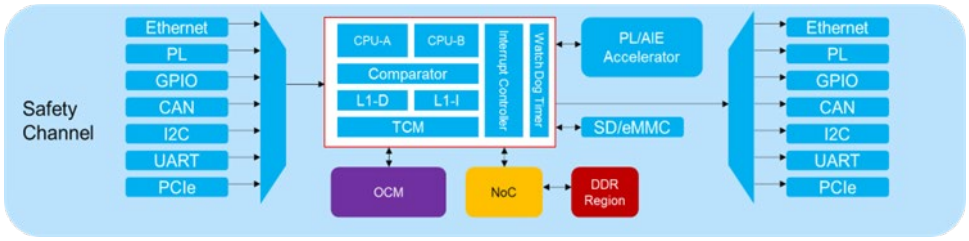

Figure 16 – LPD Centric Safety Channel: Realtime Channel

Along Figure 17 another safety channel can be defined in the FPD with the APU processor using the Cortex-A72 running a SMP OS like Linux or AMP like bare metal or FreeRTOS with a logic solver task. Sensor inputs may be sourced from peripherals in the LPD or PL and actuator outputs may be sent to peripherals in the LPD or PL.

The FPD centric safety channel certification is still pending release by AMD-Xilinx (Q4/2023).

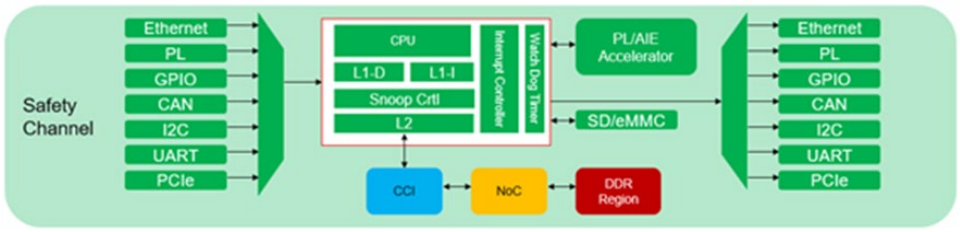

Figure 17 – FPD Centric Safety Channel: Application Channel

A PL centric safety channel can be defined in a PL implementation which holds a MicroBlaze soft-IP processor running bare metal or FreeRTOS with a logic solver task as in Figure 18. Even a hardware redundant core implementation is available for the MicroBlaze system. Again, sensor inputs may be sourced from peripherals in the LPD or PL and Actuator Outputs may be sent to peripherals in the LPD or PL.

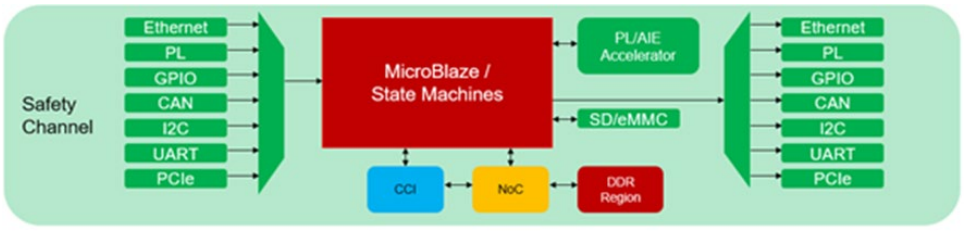

Figure 18 – PL Centric Safety Channel: Acceleration Channel

Such an "Accelerator Channel" may also be defined without the Microblaze but a solver implementation in PL or the AI Engines that requires isolated accesses to resources.

The definition of the channel architecture allows for multiple domain safety channel setups. These setups hold more than one safety channel as described above with LPD, FPD and PL based solvers (see Figure 19).

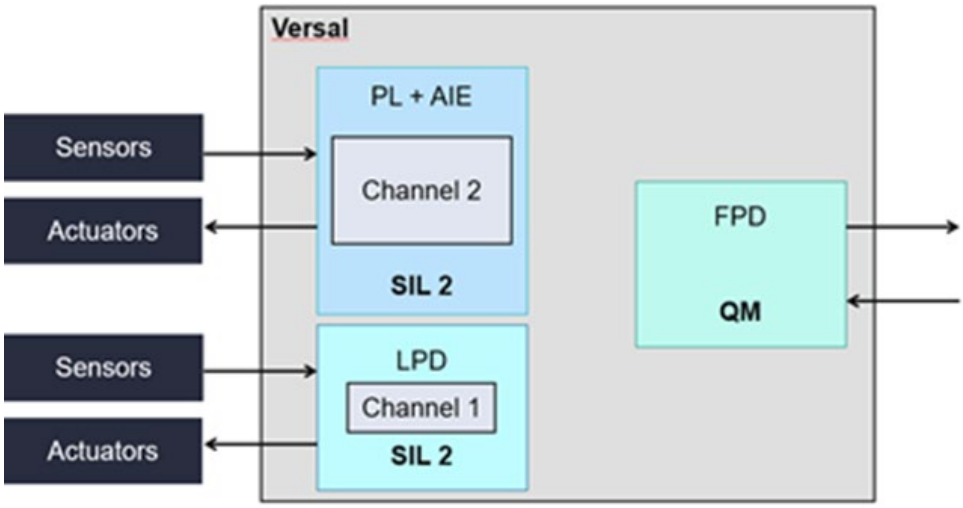

Figure 19 – Example: A Dual Safety Channel in Versal

Also, heterogeneous safety channels solvers can be created with solver architectures of co-processing i.e., APU + RPU co-processing or using the AIE as accelerators via NoC. A barrier between multiple safety channels can exist physically like a FPD / LPD separation or can exist in temporal diversity which is available when running two independent OS scheduled with a hypervisor running on the APU in the FPD.

6.1.3.4 Versal Diagnostics

Diagnostics are required for safety enabled systems and are used to detect a fault on Functional Safety. A safety requirement is guaranteed based on the diagnostic coverage. Parts of the Versal ACAP diagnostics implementation is available in hardware and software:

- Internal Hardware (intrinsic to the element)
- Packaged Software (embedded test libraries, test applications)
- Architecture (External Redundancy)

The LPD Diagnostics provide the following services:

- Lock-step CPU (Cortex-R5)
- ECC for TCM, OCM, Caches, DDRAM
- Windowed Watchdog Timers
- Temperature & Voltage Monitoring Satellites
- Bus switch Timeout, Parity, Port Isolation
- Protection Units (XPPU, XMPU)
- Software Test Library
- Check Register State (Parameters)

- Check Peripheral function
- Check the checkers (Fault injection)
- End to End Data Integrity (a.k.a. Black Channel) – Customer Driven

The FPD Diagnostics provide the following services:

- Hypervisor capable CPU (Cortex-A72)
- ECC for OCM, Caches, DDRAM
- Watchdog timer
- Temperature Monitoring Satellite
- Software test libraries
- Check Register State (Parameters)
- Check the Checkers (Fault injection)
- End to End Data Integrity (a.k.a. Black Channel) – Customer Driven

The AIE Diagnostics provide the following services:

- MBIST (for program and data memory)
- ECC for program memory and data memory
- TMR (triple mode redundancy) for critical system registers

Versal test libraries (STL) are provided for safety requirements with ASIL C for OS running in the LPD using the Cortex-R5 in the RPU or a MicroBlaze in the platform management (PMC). When using the STL test libraries running on the AIE engines the safety standard ASIL B can be achieved. Versal STLs support for the processing units

1. RPU (R5 core)
2. PSM (MicroBlaze), Xilinx provided firmware
3. PMC (MicroBlaze),
4. AIE (core functional tests including its interfaces)

6.1.3.5 Versal Tool Chain

Xilinx provides Vivado for the hardware design flow and Vitis for the software design flow. Vivado has been certified by the TÜV Süd in earlier releases (currently 2021.2) and it is used in the certification for Functional Safety Applications according to the standards ISO 26262 and IEC 61508 with some time lag in version. The Vitis environment provides the Xilinx toolchains for the embedded flow and acceleration flow mainly based on C/C++ language programming.

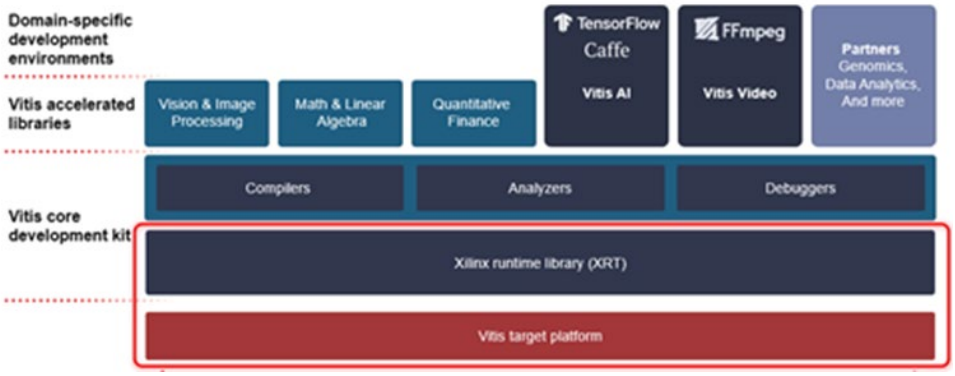

Figure 20 – Vitis Unified Software Platform

The Vitis release 2020.2 is the first release where the certification process is ongoing for specific firmware, like the platform management. The NoC Compiler requires to define separation of memory and port accesses and also for the isolation methodology in hardware (see Figure 20).

Versal adds additional features that may need to be considered for safety certification like the AI Engines and the NoC. The toolchain therefore adds the NoC Compiler to define separation of memory and port accesses. This defines a further level of isolation through generating exclusive groups or a separate hardware routing.

The in-depth analysis and documentation of safety enabled designs also requires formal tools. The Functional Safety Lounge also informs FMEDA Tools (Failure Modes, Effects and Diagnostic Analysis)

1. Functional safety metric computation
2. FMEDA data integrity checks
3. FMEDA in certification process at Xilinx
    a. yields FIT rates
    b. published through safety manuals
4. FMEDA tools at the development site (optional)

There are various tool methods that can be deployed to fulfil the proposed Methodology for the FRACTAL project so a definition of the actual scope must be created.

6.1.4 Versal ACAP Certification in FRACTAL

With all the safety related features that are available, there are typical FMEDA workflow phases that are described in the Xilinx ecosystem. The task at hand is to apply this to the proposed systematic approach in Section 6.1.3. Showing that this process has been followed with sound coverage should enable a certification of a FRACTAL node design.

Following this concept towards a potentially certifiable platform for FRACTAL use-cases commands a hardware partitioned architecture that supports a defined isolation scheme through safety channels, effectively fulfilling different classes of safety.

6.1.5 Systematic Implementation for VERSAL platform

The main scope of work on the Versal node platform safety design aligns the existing tool flow from the vendor Xilinx' ecosystem with the FRACTAL framework as forwarded in Section 4.

PLC2 provides a base design for the Versal development platforms (VCK190). This development kit is part of the certification evaluation process at Xilinx. Based on currently available information this is a starting point to add from further publication. This helps to simplify the isolation methodology application and the tool usage where a supported OS (Linux) is running on the APU. An initial platform demonstrates the control concepts of the Versal ACAP safety features to enable integration of the services envisioned in WP3 and WP4. FRACTAL node specific functionality is added and integration into the PL and on AIE engines may extend the scope of the defined safety channels. Additions on the application level should not impact the overall safety considerations. The basic Xilinx-side proposed workflow elements are listed here:

Design

1. H/W Partitioning
2. Failure modes identification
3. Safeness Estimation
4. Define requirements for phase STL and FSV

STL

5. Define SW requirements from FMEDA
6. STL definitions
    a. Design partitioning to support STL
    b. Implementation to deploy STL
7. Review the results from FSV

FSV (Functional Safety Verification)

8. Define Fault simulation environment
9. Fault simulation and analysis
10. Review process to STL phase

Functional Safety

11. Review and coordinate all the above phases
12. Methodology of maintenance and integration flow
13. Safeness metric generation

The actual protection and isolation concept is prepared to still allow platform operation in various modes with respect to power, function availability and more to support the adaptive node concepts. The objective is to provide a unified boot and initialization setup which can be used for all Versal based FRACTAL UCs. On top the development of this base platform (Figure 21) needs to also address the services architecture along the WP4 requirement.

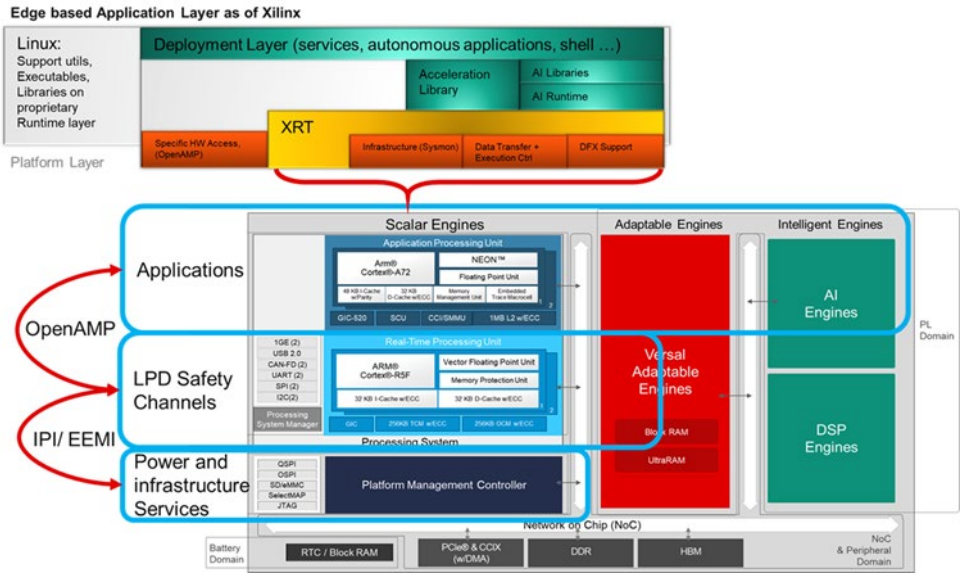

Figure 21 – Versal top level reference platform

The development of this base platform project is guided by the FRACTAL safety related methodology as of this document. The concrete implementation is growing along the Vendor-published information level along the project timelines.

6.2 PULP

6.2.1 Overview

In the context of FRACTAL, PULPissimo systems have been enhanced with fault-tolerant features to tackle potential safety-related issues. To tolerate Single Event Upsets (SEUs), the 32-bit RISC-V core has been replicated three times. Such redundancy has then been exploited to detect and correct potential SEUs. All three cores share a hierarchical instruction cache and a software-managed single-cycle Tightly-Coupled Data Memory (TCDM). Each core can access the TCDM through a logarithmic interconnect composed of a crossbar with round-robin priority.

We identified the cores, memory, and the on-chip interconnect as the most critical parts of the proposed PULP-based system. We tackled potential safety-related issues by implementing redundancy, error-correcting codes, and a watchdog timer supervising the whole system's behaviour.

Finally, we built and taped out a multicore PULPissimo SoC including all the fault-tolerant features mentioned above for further on-silicon verification and evaluation.

Such a system will lay the foundations for further efforts toward IEC 61508 and ISO 26262 standard compliance.

6.2.2 Safety components implemented in PULP

The initial steps to ensure functional safety within the PULP ecosystem is to tackle the critical components we identified in the architecture. The sections below outline the modification to key components within the multicore cluster system in order to increase their functional safety level in a hazardous environment. These modifications mainly aim to reduce the risk of failure and ensure continuous operation in the presence of soft errors (SEUs) throughout the entire system.

6.2.2.1 On-Demand Redundancy Grouping (ODRG)

To protect the processing cores throughout the system, we implemented ODRG as described in deliverable D4.4. To summarize, ODRG enables three cores to operate in a TMR lock-step configuration, making use of voting to ensure correct outputs. If an error occurs, the internal core state is saved to memory through the majority voters and it is reloaded back into the cores to continue processing from a verified safe state. Details are provided in Figure 22 and Figure 23.

If reliability features are not needed, the three cores can operate independently, allowing for up to 3x increased performance due to increased parallelization.

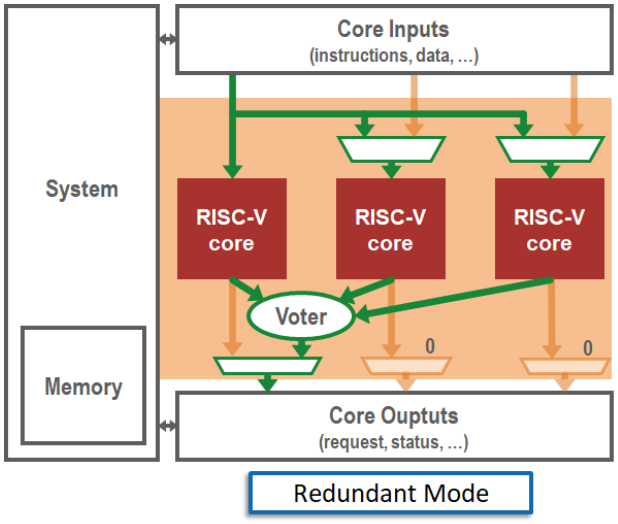

Figure 22 – On-Demand Redundancy: Redundant mode

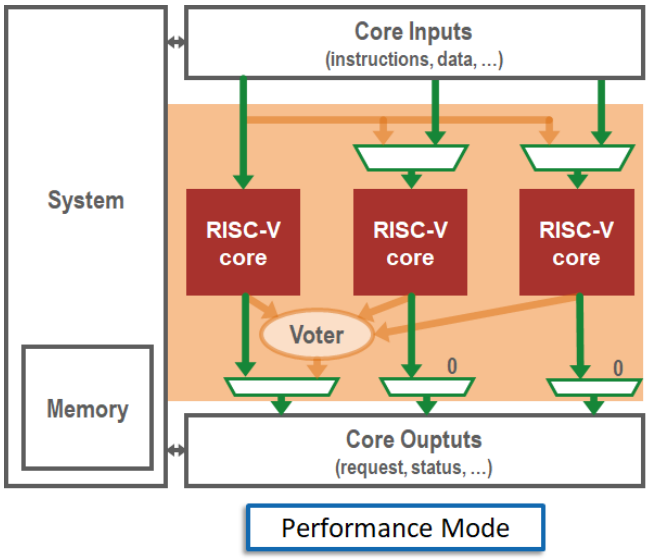

Figure 23 – On-Demand Redundancy: Performance mode

6.2.2.2 On-Chip Memory Protection

Memory protection within the PULP system is handled by ECC, as described in deliverable D4.4. With the efficient Hsiao codes, additional bits are added to each 32-bit data word, allowing for single error correction and double error detection within each word. Hardware is provided for efficient and fast byte-wise access within each data word. Additionally, a hardware scrubbing unit is implemented to continually scan each word within a memory bank for errors, correcting them once detected.

6.2.2.3 On-Chip Communication Protection

To further protect the integrity of the data within the system, the data bits of the on-chip communication are also protected with the ECC used for the on-chip memory. Through this reuse, further components within the system are protected from faults while simultaneously reducing the timing overhead required for encoding and decoding the ECC bits.

6.2.2.4 Error Tracking Unit

To keep track of errors throughout the system, as well as to configure the scrubbing intervals for each memory bank, an error tracking unit was implemented. This unit keeps track of the errors within each memory bank by counting the number of corrected errors, as well as the detected uncorrectable errors.

6.2.2.5 Watchdog Timer

While the essential and most vulnerable components are protected by the methods outlined above, the rest of the system may still suffer from faults. In case an error occurs in these components or an unrecoverable or uncaught error impacts the behaviour of the core system, a watchdog timer can be used to reset the SoC. This is also further detailed in deliverable D4.4.

6.2.3 Simulation and Testing

To verify the functionality of the various components and systems, various tests are performed in an RTL-level simulation of the full SoC. These tests focus on representative compute tasks, such as the CoreMark benchmark.

To verify the safety of the implemented fault tolerance methods, fault injection is performed on the RTL simulation using the simulator. These simulations were able to verify the corrective behaviour of the implemented tolerance mechanisms.

6.2.4 ASIC demonstrator

To test and verify the features outlined above, they were integrated into a PULPissimo SoC and implemented in a demonstrator ASIC in TSMC's 28nm technology called *Trikarenos*. The PULPissimo SoC contains a minimal 32-bit RISC-V host processor, which is triplicated for this implementation. ODRG allows these three cores to operate as a single, lock-stepped core with the required redundancy features or, if desired, as three independent cores.

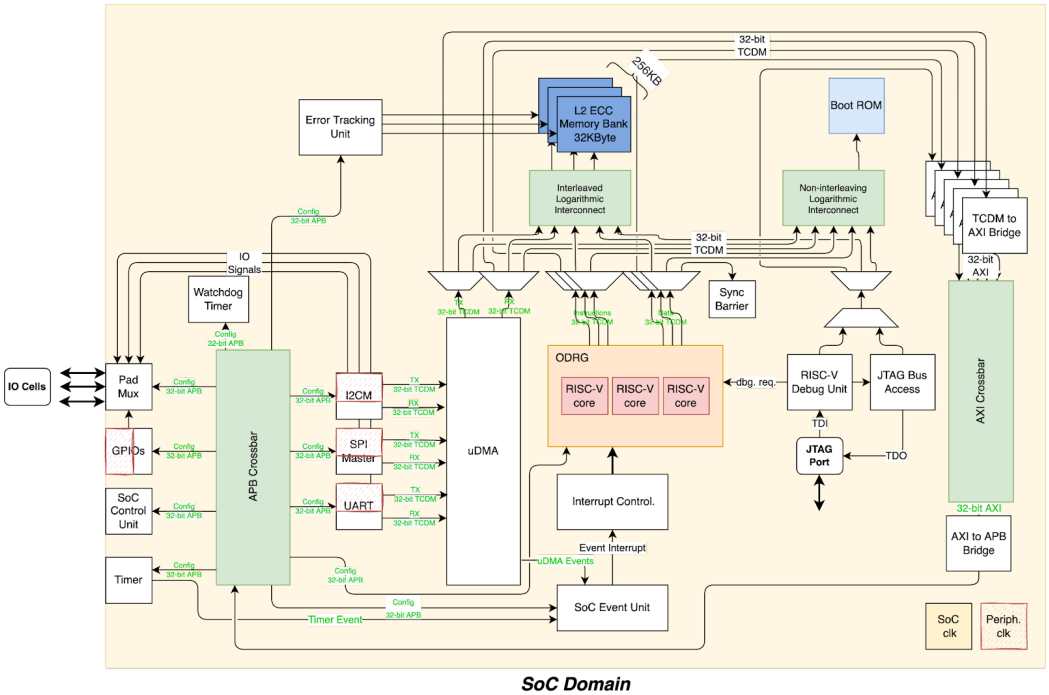

Figure 24 – Block Diagram of the PULPissimio system in Trikarenos

As shown in the block diagram of Figure 24, the PULPissimo system in Trikarenos implements ODRG, ECC memory, an Error Tracking Unit, as well as a watchdog timer. The system was taped out in TSMC's 28nm node (Figure 25), which has shown promising results regarding tolerance to hard errors and total ionizing dose, reducing the likelihood of permanent errors due to radiation.

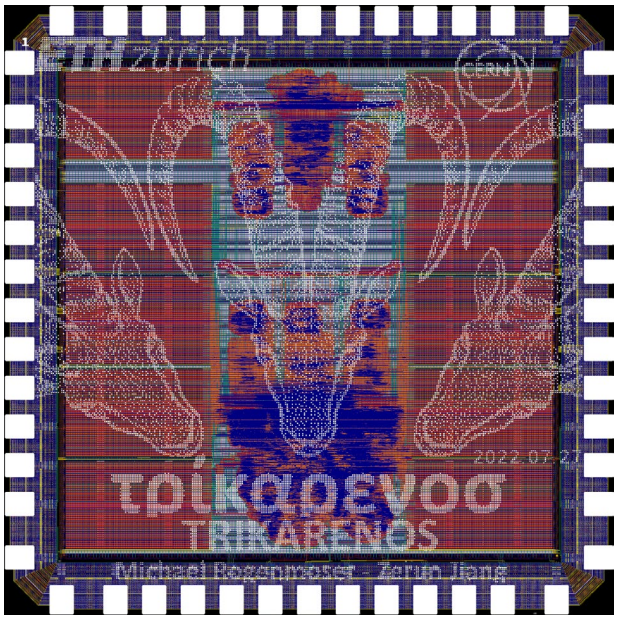

Figure 25 – Trikarenos: a prototype ASIC implementing the various safety features

6.2.4.1 Testing

To verify the functionality of the implemented fault-tolerance features in the final SoC, various testing features have been directly integrated into the ASIC design. First and foremost, standard testing methodologies have been integrated, including the RISC-V debug unit and PULP debug unit for simple verification and testing. Furthermore, scan chains were integrated into the design to ensure proper manufacturing.

To test the fault-tolerance specifics without requiring external fault sources, testing structures were integrated and modified for the purpose of injecting faults to observe corrective behaviour. Firstly, each of the cores is connected individually to a scan chain. This allows for targeted error insertion by making use of the scan chain, where the core's and SoC's internal state can be read out, modified, and reinserted appropriately. The remaining parts of the system, especially the SRAM memory banks, were modified to support this feature. Furthermore, various software structures were integrated to allow for targeted error insertion into memory.

Finally, we plan on testing the SoC under a radiation beam, ensuring correct behaviour in the presence of externally-induced faults. This allows for proper verification of the implemented features, as well as measuring detected and corrected errors through the error tracking unit.

Finally, a demonstration of the technology in space is planned, where Trikarenos will be integrated as a secondary payload in a dedicated CubeSat mission. This will allow for long-duration testing in a hazardous environment.

6.2.5 PULP-Based Systems and Safety Standards

To enhance PULP-based systems with fault-tolerant capabilities, we identified the main critical building blocks and applied the necessary extensions to detect and

correct faults, laying the foundations toward IEC 61508 and ISO 26262 standard compliance:

- Three cores operating in lock-step allow correcting SEUs;
- ECCs handle faults in the memory, and on-chip interconnect;
- The Watchdog timer supervises the system's behaviour detecting unrecoverable issues or issues appearing in non-protected parts of the system and triggering a reset that brings the chip back to a known state.

6.3 NOEL-V

NOEL-V is one of the RISC-V platforms supported by FRACTAL. NOEL-V and the SELENE SoC have been designed to target safety-related applications. In this section, we elaborate on the features this platform includes to favour the certification of safety applications using this SoC.

6.3.1 NOEL-V Core Protection Mechanisms

The NOEL-V cores implement the RV64GH instruction set architecture. These cores are provided with a GPL license and have been developed by Cobham Gaisler, a company that develops processors for the space domain. The open-source version of the NOEL-V cores do not include the Cobham's specific fault-tolerant support. This specific fault-tolerant support that is especially suitable for the space domain includes error correction codes at cache memories and register file. The protection of other processor components relies on the hardening of the technology cells for specific ASIC implementations which has been shown a very efficient approach for the space-domain. These protection mechanisms can be incorporated to the FRACTAL platform after a licensing agreement with Cobham Gaisler.

However, in the context of safety-related applications that do not operate in extremely harsh environments like the ones targeted in FRACTAL, protection mechanisms at the SoC level result very effective. SoC-level open-source protection mechanisms incorporated in the SELENE platform are covered in the next subsections.

The NOEL-V platform provides memory protection and resources spatial isolation. NOEL-V cores include a memory management unit following RISC-V specification for memory virtualization which is usually a requirement in safety-critical applications to protect the system from software and hardware errors and/or malicious attacks. Additionally, NOEL-V cores implement the RISC-V hypervisor extension which enables the effective deployment of software virtualization for this platform. Multicore virtualization enables the utilization of the NOEL-V in mixed-criticality applications in which several system functionalities can be consolidated in the same compute platform. NOEL-V supports Jailhouse and Xtratum Next Generation (XNG) hypervisors. These hypervisor layer enable the utilization of Linux OS for low criticality high-performance functionalities and low complex more easily analysable RTOS for the critical functionalities.

To improve protection of NOEL-V cores against common-cause faults (CCF), they have been instrumented with a Register File Randomization (RFR) mechanism. RFR dynamically modifies the physical register file access pattern in each NOEL-V core by means of a low-complexity hashing circuit, as depicted in Figure 26. This makes that same logical RF registers (x) are mapped onto different physical registers (p) in each CPU core. This improves NOEL-V robustness in two ways. First, it allows to detect and correct CCFs originated in the register file by modifying the effective physical location of registers in the different core replicas. Second, by periodically remapping logical registers to a different set of physical registers (by changing a random key), it also equalizes the utilization rate of physical registers.

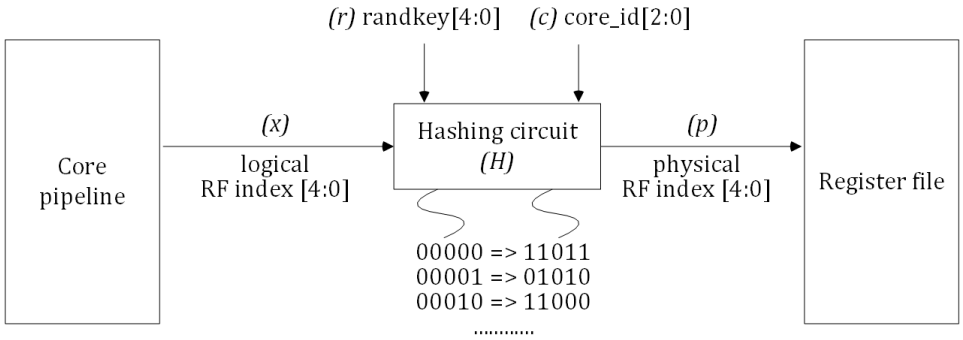

Figure 26 – Register File Randomization mechanism

The efficiency of RFR mechanism against CCFs has been evaluated by means of simulation-based fault injection (SBFI) experiments (using the DAVOS toolkit). Figure 27 summarizes the obtained SBFI results, obtained for the original and RFR-instrumented NOEL-V system, considering seven different workloads. On average, RFR improves the robustness of resulting system against CCFs by roughly an order of magnitude, providing up to 99% CCF coverage.

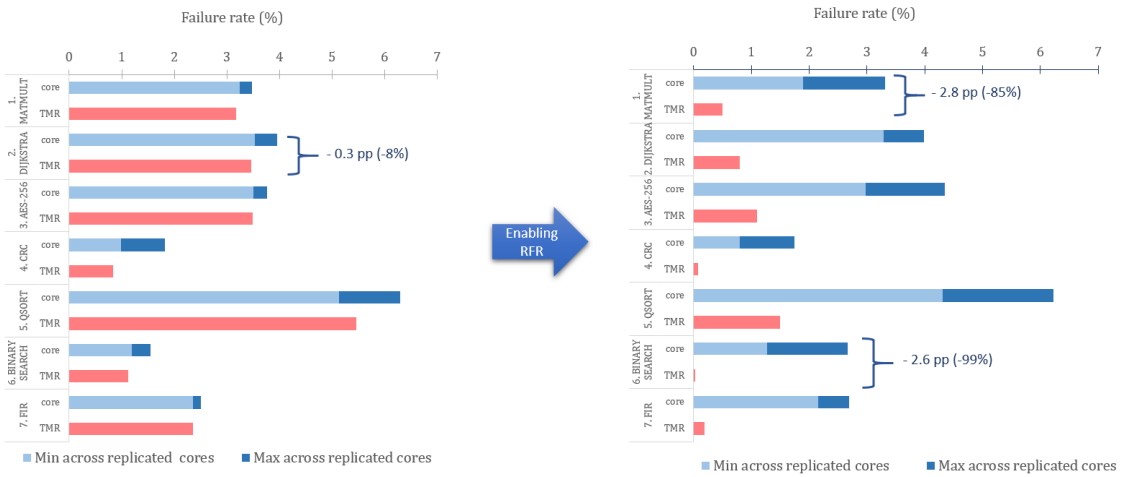

Figure 27 – Failure rate of individual NOEL-V cores and TMR assembly under staggered execution: RFR disabled (left barchart), RFR enabled (right barchart)

In addition, the simulation-based profiling of the register file has shown that RFR mechanism equalizes the read and write access rate of physical registers, as it is detailed in Figure 28. As the result, the maximum switching activity per register is reduced by roughly 5 times, reducing the electrical stress suffered by the most utilized registers and extending the expected register file lifetime by 4 to7 times.

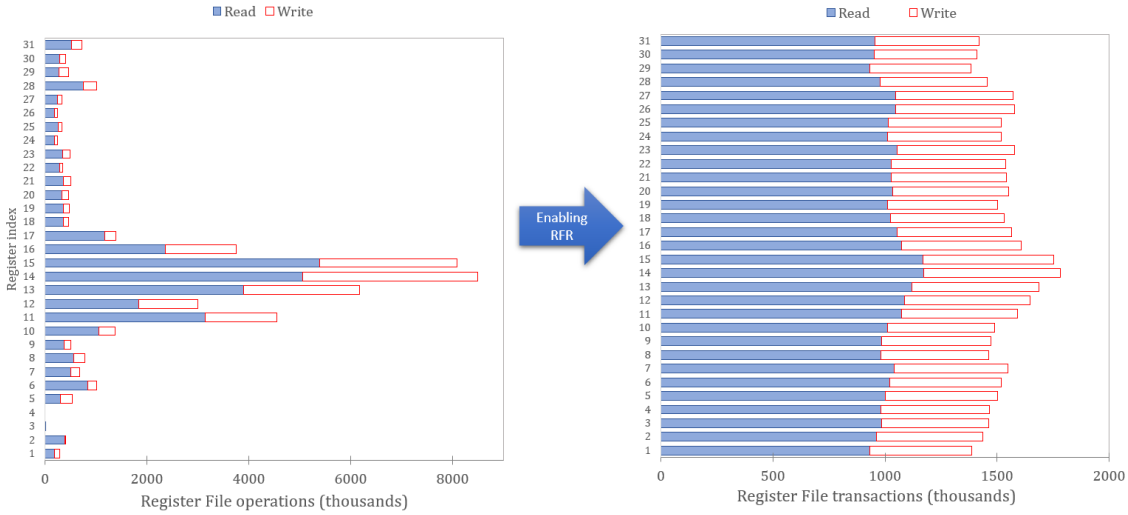

Figure 28 – Register file utilization measured by simulation-based profiling: RFR disabled (left barchart), RFR enabled (right barchart)

6.3.2 SoC-lLevel Architectural Protection

The main architectural level protection implemented in this platform is staggered redundant execution (SRE). SRE is usually a de-facto requirement of the highest criticality applications in many safety-related domains like automotive or railway. In these domains, SRE is usually implemented in the form of lock-step processors in which two identical cores are used to redundantly execute a single task in a synchronized manner such that one core (the head core) is always several instructions ahead of the trailing core. Unfortunately, in the context of complex applications in which significant compute power is needed and applications of different criticality coexist exclusively dedicating a core to the redundant execution wastes resources and limits the flexibility of use of the hardware platform.

In this context, the mechanism implemented in the SELENE SoC to enable SRE relies on the flexible utilization of the spare cores in the multicore and using a non-intrusive hardware/software mechanism to ensure the execution staggering is preserved. Ensuring the staggering becomes crucial to avoid common-cause transient fault like voltage drops or electromagnetic interference amongst others. With this approach voting the results of tasks execution at predefines application steps can be performed by software and/or hardware means. A voter IP is interconnected in this platform and can be instantiated for detecting and/or correcting errors checking discrepancies at the voted results or when results are not produced in time.

6.3.3 Robustness Evaluation

The NOEL-V platform robustness can be estimated at the RTL level with the DAVOS fault-injection tool (https://gitlab.com/selene-riscv-platform/DAVOS). Robustness evaluation is a mandatory step for products safety certification. While simulation-based fault-injection is not mandatory, since physical testing campaigns are also needed, its use is recommended in most of the functional safety standards (e.g. ISO 26262) to detect potential deviations at early design stages.

The robustness of FPGA-based platforms can also be tested using BAFFI (Bit-accurate FPGA-based Fault-injection tool). BAFFI is an extension to DAVOS carried out in the context of FRACTAL, and enables both: (1) robustness evaluation of designs targeting FPGA technologies, and (2) safety mechanisms diagnostic capabilities evaluation. Note that for large designs simulation-based experiments are too costly to perform thorough verification and validation processes.

For that reason, BAFFI is designed to enable fine-grained FI experiments, providing detailed robustness metrics of FPGA design as usually provided by SBFI, yet two to four orders of magnitude faster than SBFI. This level of detail is achieved by a custom netlist-to-bitstream mapping mechanism embedded in BAFFI (as depicted in Figure 29). This allows to selectively target any node (design component) in the design tree with an accuracy of up to an individual register, LUT, or BRAM cell. Such detailed experiments are set-up merely by specifying the hierarchical path of the targeted design node, in a completely non-intrusive way, i.e. without the need for design modification or floorplanning.

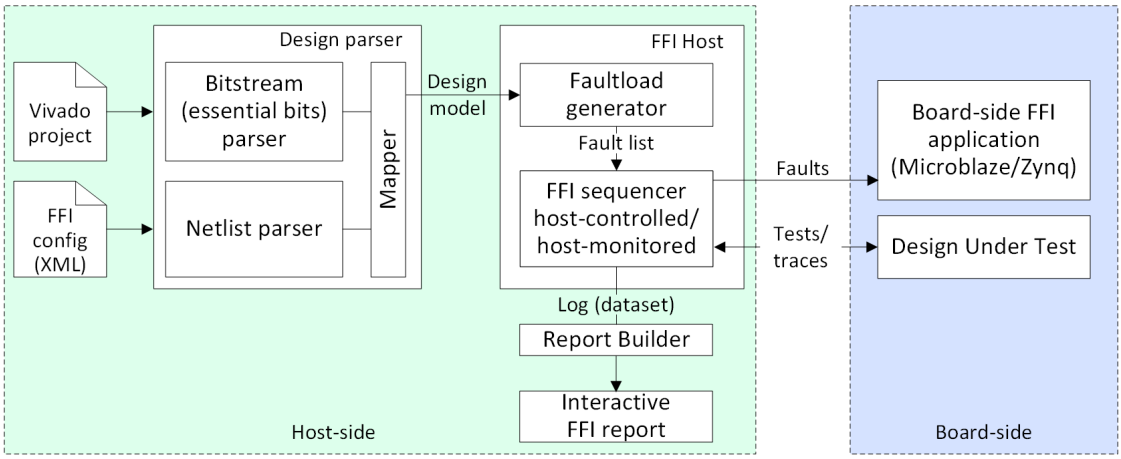

Figure 29 – Top-level architecture of the BAFFI tool

6.3.4 Hardware Monitors

The NOEL-V platform used in FRACTAL implements several hardware monitors to detect both functional and timing errors. A traditional watchdog functionality is implemented within the voter module to allow detecting system hangs and preventing crashes at coarse-grain granularity. At a finer granularity the NOEL-V platform used

in FRACTAL implements a statistics unit with the ability to monitor the duration of SoC requests (potentially detecting timing violations and deadlocks) and the tracking of contention in shared resources. This hardware timing monitors are very useful tool for the software timing verification step required to meet safety requirements.

6.3.5 Software timing and Freedom from Interference

In terms of software development functional safety standards call for two main requirements. The first one is the derivation of an upper-bound for the execution time of software tasks and the second one is the freedom from interference. The first requirement, asks for means to ease the computation of the worst-case execution time (WCET) and the second one calls for means to integrate different software functionalities in the same platform such that the scheduling of all the critical tasks can be guaranteed.

The statistics unit implemented in the NOELV platform serves both purposes. On the one hand, this unit provides accurate fine grain timing measurements to derivate WCET estimates with enough confidence. On the other hand, the contention monitoring capabilities enables the possibility of using quotas to shared resources or globally so that the maximum execution time inflation suffered by a task can be guaranteed. This latter part is required to ensure task can be timely scheduled.

7 Guidance to apply safety standards in FRACTAL

In the FRACTAL project, we have established a specific methodology derived from the functional safety standards that will give the system designer an early assurance of the reduction of systematic errors and the feasibility of functional, non-functional and safety specifications on some of the building blocks for use cases that require it. This confidence in the different building blocks of the system can be acquired and confirmed as they increase in maturity during or after FRACTAL. In order to guarantee the expected final safety level, the system "industrialization" project should adopt one of the recognized functional safety standards such as IEC 61508 in early stages.

In this section, we present a simplified methodology, derived from the functional safety standards, which can be used by the FRACTAL partners. This methodology is available for use cases that are interested in increasing the level of confidence in the feasibility and viability of their building blocks that will be introduced in the industrialization phase of a system with operational safety constraints.

7.1 Prerequisites of the methodology

The methodology assumes the use of a system description method from systems engineering such as the hierarchical decomposition of a system in functional blocks or in building blocks. Figure 30 shows an example of the hierarchical breakdown of a system into building blocks down to the basic building blocks which are considered as elementary building blocks.

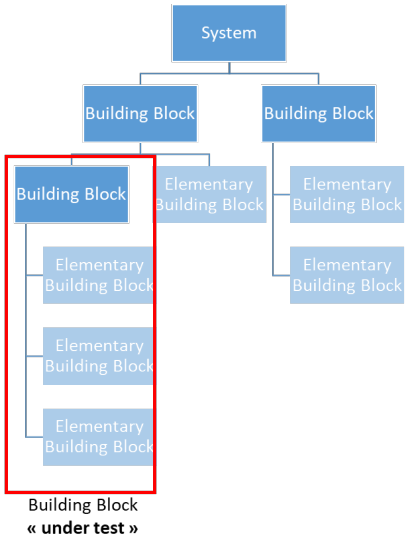

Figure 30 – Hierarchical breakdown of a system

7.2 Best practices for the FRACTAL project

Some best practices can be anticipated for safety-related building blocks and sub-systems and then reused during the actual product certification. In the following, we provide a short set of recommendations and best practices to anticipate future safety certification that can be applied in the project:

- Sort safety-critical UC requirements into three categories:
  - Functional requirements
  - Non-functional requirements
  - Safety requirements
- Capture these requirements in a deliverable related to the use case
- Test safety requirements in WP7/8
  - Capture the results
- Full traceability among the UC specification, development, test, verification and validation process

7.3 Safety by construction

While the design is intended to be built safe by construction, hardware random faults cannot be avoided, and hence, appropriate safety measures are incorporated during the architectural design to guarantee that those faults are properly managed by means of either fault-tolerance features or by transitioning timely to a safe state. For instance, in the context of this document, safety features relate to (i) avoiding common cause failures (CCFs) that might lead redundant components to the same error, and hence a failure despite redundancy, (ii) limiting and monitoring time overruns due to multicore interference to preserve freedom from interference, (iii) reliable (fault-tolerant) communication, and (iv) allowing the use of AI-based components by resorting to appropriate ASIL decompositions relieving AI-components from inheriting safety requirements.

7.4 Simplified safety guidelines for the FRACTAL project

These guidelines were suggested at the beginning of the FRACTAL project to anticipate safety certification.

1) Implementation of a hierarchical traceability system

2) Breakdown of the system into building blocks

3) Identification of the elementary building blocks

4) Hierarchical specification of the use case (HW/SW)

   - Specification of the functional requirements
   - Specification of the non-functional requirements

5) Hazard analysis / feared events analysis

6) Safety requirements specification

7) Identification of relevant buildings that will be under test during the project

- Select the buildings blocks that require an increased confidence level and a reduced number of systematic faults

8) Isolation/extraction of specifications related to the building blocks "under test"

9) Specification & realization & documentation of functional tests

- Functional tests of the elementary blocks
- Functional tests of software integration
- Functional tests of HW/SW integration

10) Specification & realization & documentation of non-functional tests

11) Specification & realization & documentation of the fault injection tests

- This goes beyond the tests usually used in non safety-critical domains.
- Faults are injected on the inputs of the functional blocks and shall not propagate to the outputs.
- Fault injection is a time-consuming practice and the choice of the hierarchical breakdown where to apply it is crucial.
- Fault injection can detect implementation errors, dysfunctional architecture and also incomplete or erroneous specifications.
- See Figure 31 below

12) Functional validation coverage of building blocks under test

- Verification of the traceability between the building blocks specification and the tests specification
- Verification of the adequacy of the hierarchical specification, the tests specification and the obtained test results
- Identify what has been validated
- Determine if the building blocks meet the requirements of the building block specification
- Analysis and identification of malfunctions and gaps

13) Technical recommendations regarding the "buildings blocks under test" for future projects and towards a future product

- What has been validated in the FRACTAL project
- What doesn't work or may cause problems
- Desirable future improvements
- What remains to be validated

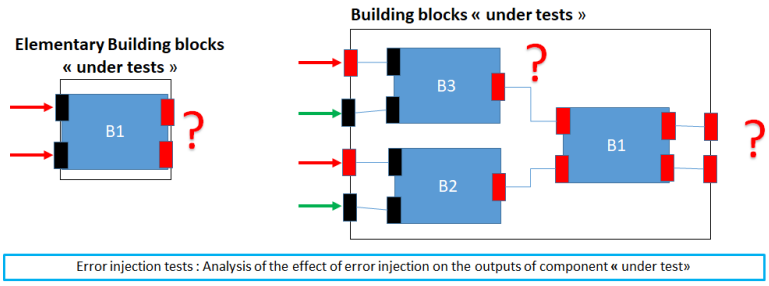

Figure 31 – Illustration of fault simulation: error propagation

7.5 Reusing building blocks

In the context of FRACTAL, products targeting safety-related application may rely on pre-existing building blocks. The main reason for this is that the cost of development from scratch can be prohibitive in relatively complex systems. For instance, creating a complex safe and secure system with capabilities comparable to mainstream computing systems (Microsoft Windows or GNU/Linux) would incur in prohibitive costs – estimated at 50 B$ for the Linux kernel[18].

Safety standards allow the reutilization building blocks. For instance, IEC 61508 allows using a proven-in-use argument (named Route 2S in IEC 61508-3). However, to achieve certification based on this argument, the product developer needs to provide a vast amount of detailed information of collected historic data (e.g. in IEC 61508-7, C.2.10.1) that sometimes is not available.

However, there are other means to achieve qualification of pre-existing software elements. For instance, the SIL2LinuxMP[19] project provides the safety qualification argument for the pre-existing software elements of a constrained Linux environment using IEC 61508, Route 3S. For that, one has to provide arguments explaining why the development process of those pre-existing software elements satisfies the high standards of IEC 61508[20].

Unfortunately, in the context of close-source libraries (e.g., CudaDNN) certification cannot be achieved by the application developer unless the owners of these libraries go through an out-of-context (an element in isolation) certification process and/or adapt their libraries to fit specific standards and are able to provide the required safety documentation. While in theory, the usage of close-source libraries not developed in conformance with safety standards is not actually precluded by certification standards (e.g., end-users can use black-box testing), this however, has

[18] Nicholas Mc Guire and Carles Hernandez, An open dependable platform for safety critical systems, Hipeac Magazine April 2020, https://www.hipeac.net/magazine/7154/
[19] https://sil2.osadl.org/
[20] Andreas Platschek, Nicholas Mc Guire, Lukas Bulwahn, Certifying Linux: Lessons Learned in Three Years of SIL2LinuxMP, Embedded World 2018.

severe implications for applicability. Thus, for these HW/SW products to be qualifiable for safety-related applications, suppliers should either adapt their products to comply with specific safety standard or allow end-user to go for alternative open-source libraries. For the latter, in order to make this approach attractive, open-source libraries must provide competitive performance with respect to the existing closed-source libraries[21].

[21] H. Tabani, et. al., Assessing the Adherence of an Industrial Autonomous Driving Framework to ISO 26262 Software Guidelines, Design Automation Conference 2019.

8 Conclusions

This deliverable presented a macroscopic overview of the regulatory framework and qualification process for safety-critical applications, thereby facilitating the definition of the steps for certification of FRACTAL systems.

The document started with an introduction to the relevant safety standards. Specifically, we addressed functional safety and the objectives of the IEC 61508 standard, presenting its general structure, the concept of risk reduction, and discussing how this standard also forms the basis for other standards. Subsequently, we detailed considerations for AI-based systems and components, clarifying how AI-based software and functional safety standards are not yet compatible and discussing the ongoing efforts for reconciliation.

After such introduction, the document presented domain specific considerations for the industrial, automotive, and medtech industry. Specific standards for each industry field have been presented and discussed within the scope of the bigger picture offered by IEC 61508. Next, platform specific considerations have also been presented, discussing about Versal, PULP, and Noel-V.

Finally, Section7 presented a guidance for the application of safety standards in FRACTAL.

9 List of Figures

10 List of Tables

11 List of Abbreviations

ACAP	Adaptive Compute Acceleration Platform (relates to VERSAL)
ADAS	Advanced Driving Assistance Systems
AI	Artificial Intelligence
AIE	AI Engine
AMP	Asymmetric Multi-Processing
ANSI	American National Standards Institute
APB	Advanced Peripheral Bus
APU	Accelerated Processing Unit
ASIC	Application Specific Integrated Circuits
ASIL	Automotive Safety Integrity Level
AWI	Approved new Work Item (relates to ISO)
AXI	Advanced eXtensible Interface
BBRAM	Battery Backed Random Access Memory (Battery Backed RAM)
BIST	Built-In Self-Test
BRAM	Block Random Access Memory (Block RAM)
CAN	Controller Area Network
CAN FD	Controller Area Network Flexible Data-Rate
CCF	Common Cause Failure
CCI	Cache Coherent Interconnect
CCIX	Cache Coherent Interconnect for Accelerators
CENELEC	Comité Européen de Normalisation Électrotechnique (English: European Committee for Electrotechnical Standardization)
CFR	Code of Federal Regulations
CPM	CCIX-PCIe Module (relates to Versal)
CPU	Control Processing Unit
DDR	Double Data Rate
DDRAM	Double data rate Dynamic Random Access Memory
DIN	Deutsches Institut fur Normung (German Institute for Standardisation)
DMA	Direct Memory Access
DNN	Deep Neural Network
DRAM	Dynamic Random Access Memory
DoA	Description of Action
ECC	Error Correction Code
E/E/PE	Electrical/Electronic/Programmable Electronic
EN	European Norm
EU	European Union
FDA	Food and Drug Administration
FIT	Failure In Time
FMEDA	Failure Modes Effects and Diagnostic Analysis

FPD	Full Power Domain
FPGA	Field Programmable Gate Arrays
FSV	Functional Safety Verification
FW	Firmware
GbE	Gigabit Ethernet
GIC	Global Interrupt Controller
GNU	GNU is Not Unix
GPIO	General Purpose Input/Output
GPL	General Public License
HARA	Hazard Analysis and Risk Assessment
HW	Hardware
I2C	Inter-Integrated Circuit
IC	Integrated Circuit
I/D Cache	Instruction/Data Cache
IEC	International Electrotechnical Commission
I/O	Input/Output
IP	Intellectual Property
ISA	Instruction Set Architecture
ISO	International Organization for Standardization
LBIST	Logic Built-In-Self-Test
LFM	Latent Fault Metric
LPD	Low Power Domain
LUT	Look-Up Table
MBIST	Memory Built-In Self-Test
MC/DC	Modified condition/decision coverage
MDR	Medical Device Regulation
MMU	Memory Management Unit
MPSoC	Multi-Processor SoC
NDA	Non-Disclosure Agreement
NoC	Network on Chip
NPI	NoC Programming Interface
OCM	On Chip Memory
ODRG	On-Demand Redundancy Grouping
OEM	Original Equipment Manufacturer
OS	Operating System
PAS	Publicly Available Specifications (relates to ISO)
PCB	Printed Circuit Board
PCIe	Peripheral Component Interconnect Express
PCM	Phase Change Memory
PL	Programmable Logic
PLC	Programmable Logic Controller
PMC	Platform Management Controller

PMHF	Probabilistic Metric for random Hardware Failures
PS	Processing System
PSM	Processing System Manager
PULP	Parallel Ultra Low Power
QM	Quality Management
RAM	Random Access Memory
RAMS	Reliability, Availability, Maintainability and Safety
RF	Register File
RFR	Register File Randomization
RISC	Reduced Instruction Set Computer
RPU	Real time Processing Unit
RTC	Real Time Clock
RTL	Register Transfer Level
RTOS	Real Time Operating System
SBFI	Simulation-Based Fault Injection
SC	SIL capable
SC 3	SIL 3 capable
SEU	Single Event Upsets
SIL	Safety Integrity Level
SLC	Safety Life Cycle
SMMU	System Memory Management Unit
SMP	Symmetric Multi-Processing
SoC	System-on-Chip
SPFM	Single Point Fault Metric
SPI	Synchronous Peripheral Interface
SRAM	Static Random-Access Memory (Static RAM)
SRE	Staggered Redundant Execution
STL	Software Test Library
SW	Software
TCDM	Tightly-Coupled Data Memory
TCM	Tightly Coupled Memory
TMR	Triple Mode Redundancy
TR	Technical Reports (relates to ISO)
TS	Technical Specifications (relates to ISO)
TSMC	Taiwan Semiconductor Manufacturing Company
TTC	Time-To-Completion constraint
TÜV	Technischer Überwachungsverein (German safety and standards institution)
UART	Universal Asynchronous Receive/Transmit
UC	Use case
UL	Underwriter Laboratories
U.S.	United States
WCET	Worst Case Execution Time

WDT	WatchDog Timer
WP	Work Package
XMPU	Xilinx Memory Protection Unit
XNG	Xtratum Next Generation
XPPU	Xilinx Peripheral Protection Unit
XRAM	Accelerator RAM

The short names of FRACTAL partners are not considered as abbreviations: ACP, AITEK, AVL, BEE, BSC, CAF, ETH, HALTIAN, IKER, LKS, MODIS, OFFC, PLC2, PROINTEC, QUA, ROT, RULEX, SIEG, SIEM, SML, THA, UNIGE, UNIMORE, UNIVAQ, UOULU, UPV, VIF, ZYLK.