Series of articles Functional Safety, Part 4

FUNCTIONALE SAFETY

Safe drive technology on the basis of EN61800-5-2

Motivation

Part 1 of this series of articles gives an overview of the standards of functional safety. A fundamental problem with using IEC 61508 is the fact that it should be general and applicable for almost any application scenario; however, one objective of IEC 61508 is also to serve as the basis for product/sector-specific standards.

Firstly, a brief clarification should be provided on how the applicable standard is identified.

When implementing functionally safe components of industrial automation, it is absolutely essential to intensively light the normative environment. The applicable product standard, EN 61800-5-2, is listed as a B standard under the EU Machinery Directive. To manage functional safety, which, amongst other things, requires topics such as responsibilities, employee qualification, and quality assurance measures, this product standard refers directly to IEC 61508-1. However, this part is not considered in detail in this article because it is a complex, separate subject area.

On the technical side, we will now examine the requirements for the hardware electronics and the device software. The most basic drive safety function safe torque off (STO) can, in principle, be implemented as purely hardware-based. EN 61800-5-2 provides good support for this. On the other hand, it becomes interesting when more complex drive safety functions are required.

In practice, such functions are implemented in software. For software, the drive standard refers to IEC 61508-3. As the safety integrity level (SIL) increases, very high demands are placed on software development, such as the methods used and the verifications to be performed including the tools used.

How are the relevant standards applied together?

Let us take a look at the typical block diagram for a safety-related drive, as can be found in the product standard. The standard refers to a powerdrive system (safety related) (PDS (SR)).

In addition to very product-specific blocks, such as the power unit and control unit, the above-mentioned common parts emerge with the diagnostic functions and the communications & IO Block.

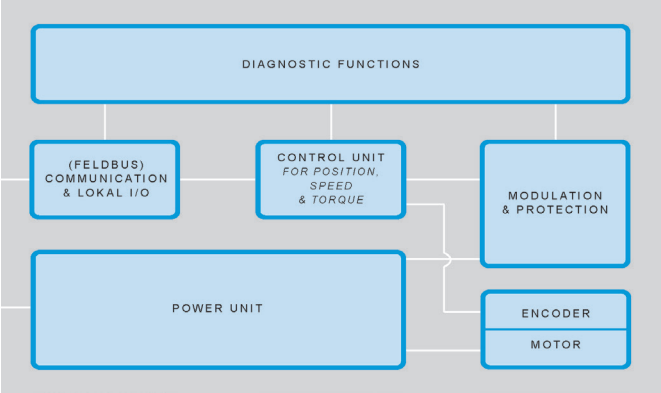

Figure 1: Block diagram of safety-related drive - PDS(SR)

The function block diagnostic functions implements the drive or motion-specific safety functions such as the safe operating stop (SOS), safely limited speed (SLS), and safe direction of rotation (SDI). The typical input information for this block is the speed or position information, which is captured as raw data by the encoder and is usually transferred to the drive via an encoder-specific communication interface. Additional input data is received from local inputs on the drive or via the fieldbus/Ethernet interfaces.

The safety functions (SF) are – to a large extent – monitoring functions implemented in software. In accordance with the selected SF, for example, the SLS continually monitors whether the rotational speed is below the selected limit value. Here, the standard requires a distinction between the reaction to detect limit violations and internally recognised errors. If the speed is registered as being above the limit, a meaningful reaction function corresponding to the application is triggered. In practice, these are usually the safety functions STO, SS1, or SS2.

It is interesting to note that the software-technical implementation of these drive safety functions is generic and can be easily reused in the form of a software library. Such a library can be integrated by the drive manufacturer during the development phase to save valuable development resources.

The symbolic relationship between the library, in this case the MESCO example's pre-defined secure drive functions

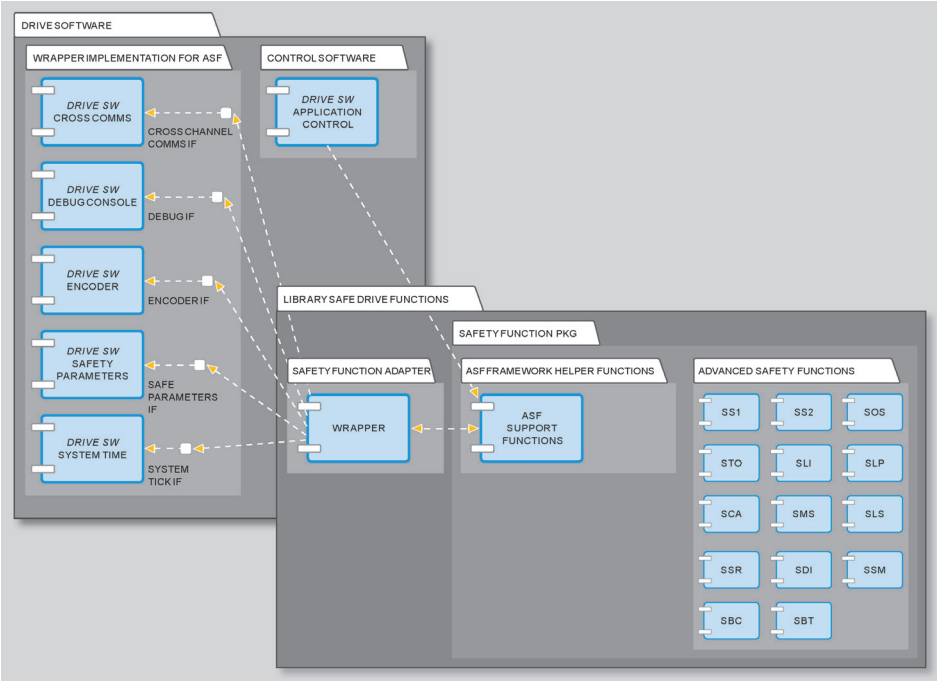

Figure 2: SW Architecture - Integration of the library

library, and the superimposed device firmware is illustrated in the following figure. The library is designed for two-channel operation and has two different interface types. On the one hand, the library needs information that is retrieved from the device firmware by *callback* functions enclosed in the wrapper class. Typical examples are the encoder and time information or the function interface for requesting or releasing the desired safety function activated by the device firmware.

This library currently includes SS1, SS2, SOS, SLI, SLP, SCA, SMS, SLS, SSR, SDI, SSM and SBC/SBT safety functions.

The drive manufacturers can implement any combination of safety functions on a common hardware platform. This can be used, for example, for individual price models

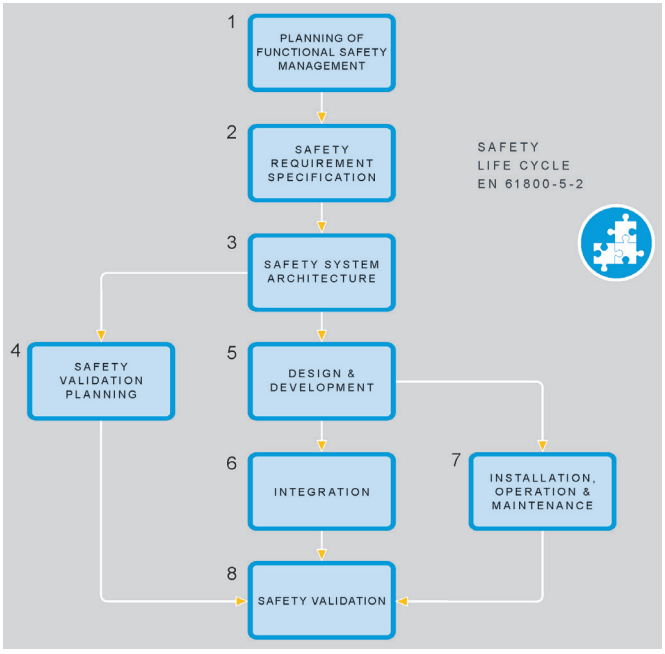

Figure 3: Safety life cycle according to EN 61800-5-2

depending on the unlocked range of functions. In order to fully understand the benefits of design packages in the development of a functionally safe component, it is helpful to look at the normatively required development cycle (Figure 3).

The drive standard defines eight phases that begin with planning functional safety management and, if the process is positive, conclude with safety validation. Core areas are the development phases safety requirement specification, safety system architecture, design & development, and the test phases integration and safety validation. Accompanying quality assurance measures are assigned to phase 4 – safety validation planning. Phase 7 is often implemented by product management in practice, as the primary result of this phase is the operating manual or the so-called safety manual.

A design package that includes only the pure technical documentation such as source code or circuit diagrams, bills of materials, and gerber data in a hardware design package would be just the tip of the iceberg. Therefore, MESCO's (safety) design packages provide appropriate support for each of these phases, making it easier for the designer of the device to accumulate the required documentation. Library integration starts in the phases that precede phase 5.

Support is of a different nature, such as requirement fragments, test specifications, source code, and the explicitly normatively required supporting evidence. Supporting evidence is, on the one hand, quite typical test protocols, but also evidence of the verifications performed for the respective phases. These are of great importance in validating the safety of the entire drive, as they do not initially have to be performed or documented in detail. This also saves valuable time and reduces costs. An advantage of this approach is that the user can make individual adjustments to the library; a rigid certification of the library would mean a loss of flexibility.

For the left branch of the V model – phases 2, 3, and 5 – the design package provides data for integration into the requirements tracking tool, for example Polarion or Doors.

For the safety requirements specification, a detailed description and parameter set such as (achievable) SIL, performance level, category, PFH, reaction time, fault reaction function, and recommendations for verification are displayed for each safety function. These parameters must be partially adapted in the context of the drive to be developed. The operating of the design package also provides support.

The requirements for the safety functions and the software architecture must be further specified at the next level, safety architecture. For this purpose, flow charts are provided for illustration, in which the relevant parameters are shown.

Again, exemplified by the safety function SLS in Figure 4:

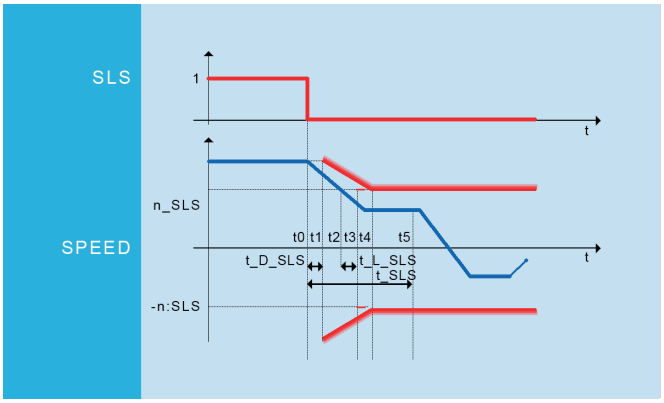

Figure 4: Timing diagram of the safety function SLS

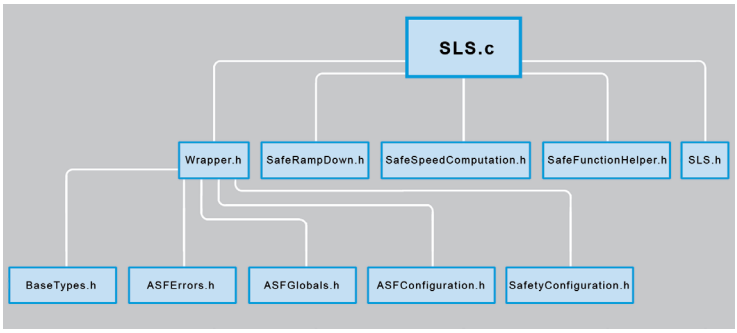

Figure 5: Dependency graph of the SLS safety function

Figure 5 illustrates that the behaviour and structure of the safety function modules have already been further refined and thus more comprehensible specifications for the subsequent implementation have been provided. The idea behind this gradual refinement of requirements is to avoid systematic errors in the development process.

Last but not least, the quality of the source code and the associated supporting evidence play a major role. Applied coding standards, implemented code reviews, static code analysis, and software module tests with certified test tools reinforce this.

In summary

With the design package library safety drive functions, Mesco has systematised the reusability of development parts and thus reacts to the increasing cost and deadline pressure during product development. With the help of the library in combination with the supplementary consulting and development services, the component manufacturers receive a wide range of support in the individual implementation of their functionally safe product developments.

Author
Armin Götzmann
Managing Director, MESCO Systems GmbH