GB

NATIONAL STANDARD OF THE

PEOPLE'S REPUBLIC OF CHINA

ICS 29.020
K 09

GB 28526-2012 / IEC 62061:2005

Electrical Safety of Machinery – Functional Safety of Safety-Related Electrical, Electronic and Programmable Electronic Control Systems

(IEC 62061:2005, Safety of Machinery – Functional Safety of Safety-related Electrical, Electronic and Programmable Electronic Control Systems, IDT)

Issued on: June 29, 2012 **Implemented on: May 1, 2013**

Issued by: General Administration of Quality Supervision, Inspection and Quarantine;

Standardization Administration of PRC.

Table of Contents

Foreword

The 5, 6.4, 6.6.3, 6.10, 6.12 in this Standard are mandatory, while the rest are recommended.

This Standard was drafted as per the rules specified in GB/T 1.1-2009.

Using translation method, this Standard equally adopts IEC 62061:2005 *Safety of Machinery – Safety Function of Safety-Related Electrical, Electronic and Programmable Electronic Control Systems*.

This Standard makes the following editorial changes:

--- Change the standard name into *Safety of Machinery – Safety Function of Safety-Related Electrical, Electronic and Programmable Electronic Control Systems.*

--- Delete the foreword of International Standard.

This Standard was proposed by China Machinery Industry Association.

This Standard shall be under the jurisdiction of National Technical Committee on Electrical Systems of Industrial Machinery of Standardization Administration of China (SAC/TC 231).

Drafting organizations of this Standard: China National Machine Tool Quality Supervision Testing Center, and Shenyang Institute of Computing Technology of Chinese academy of Sciences.

Participating drafting organizations of this Standard: Googol Technology (Shenzhen) Co., Ltd., Beijing Knd Cnc Technology Co., Ltd., Jinan Link Control Numerical Control Co., Ltd., Suzhou Centre Testing International Group Co., Ltd., and Zhejiang Kaida Machine Tool Co., Ltd.

Chief drafting staffs of this Standard: Huang Zuguang, Yin Zhenyu, Zhao Qinzhi, Yang Jingyan, Huang Lin, Yu Dong, Gong Xiaoyun, Zhang Chengrui, Yang Hongli, Zhu Ping, He Yujun, and Hu Yi.

Introduction

As a result of automation, demand for increased production and reduced operator physical effort, Safety-Related Electrical Control Systems (referred to as SRECS) of machines play an increasing role in the achievement of overall machine safety. Furthermore, the SRECS themselves increasingly employ complex electronic technology.

Previously, in the absence of standards, there has been a reluctance to accept SRECS in safety-related functions for significant machine hazards because of uncertainty regarding the performance of such technology.

This Standard is intended for use by machinery designers, control system manufacturers and integrators, and others involved in the specification, design and validation of a SRECS. It sets out an approach and provides requirements of achieve the necessary performance.

This Standard is machine sector specific within the framework of IEC 61508. It is intended to facilitate the specification of the performance of safety-related electrical control systems in relation to the significant hazards (see 3.8 of ISO 12100-1) of machines.

This Standard provides a machine sector specific framework for functional safety of a SRECS of machines. It only covers those aspects of the safety lifecycle that are related to safety requirements allocation through to safety validation. Requirements are provided for information for safe use of SRECS of machines that can also be relevant to later phases of the life of a SRECS.

There are many situations on machines where SRECS are employed as part of safety measures that have been provided to achieve risk reduction. A typical case is the use of an interlocking guard that, when it is opened to allow access to the danger zone, signals the electrical control system to stop hazardous machine operation. Also in automation, the electrical control system that is used to achieve correct operation of the machine process often contributes to safety by mitigating risks associated with hazards arising directly from control system failures. This Standard gives a methodology and requirements to:

--- Assign the required safety integrity level for each safety-related control function to be implemented by SRECS;

--- Enable the design of the SRECS appropriate to the assigned safety-related control functions(s);

--- Integrate safety-related subsystems designed in accordance with ISO 13849;

--- Validate the SRECS.

Electrical Safety of Machinery – Functional Safety of Safety-related Electrical, Electronic and Programmable Electronic Control Systems

1 Scope

This Standard specifies requirements and makes recommendations for the design, integration and validation of safety-related electrical, electronic and programmable electronic control systems (SRECS) for machines (see Notes 1 and 2).

This Standard is applicable to control systems used, either singly or in combination, to carry out safety-related control functions on machines that are not portable by hand while working, including a group of machines working together in a coordinated manner.

NOTE 1: In this Standard, the term “electrical control systems” is used to stand for “Electrical, Electronic, Programmable Electronic (E/EPE) control systems’ and “SRECS” is used to stand for “safety-related electrical, electronic and programmable electronic systems”.

NOTE 2: In this Standard, it is presumed that the design of complex programmable electronic subsystems or subsystem elements conforms to the relevant requirements of IEC 61508. This Standard provides a methodology for use, rather than development, of such subsystems and subsystem elements as part of a SRECS.

This Standard is an application standard and is not intended to limit or inhibit technological advancement. It does not cover all the requirements (e.g. guarding, no-electrical interlocking or non-electrical control) that are needed or required by other standards or regulations in order to safeguard persons from hazards. Each type of machine has unique requirements to be satisfied to provide adequate safety.

This Standard:

--- Is concerned only with functional safety requirements intended to reduce the risk of injury or damage to the health of persons in the immediate vicinity of the machine and those directly involved in the use of the machine.

--- Is restricted to risks arising directly from the hazards of the machine itself or from a group of machines working together in a coordinated manner:

NOTE 3: Requirements to mitigate risks arising from other hazards are provided in relevant section standards. For example, where a machine(s) is part of a process

NOTE 4: Continuous mode means that a safety-related control function is performed perpetually (continuously) i.e. the SRECS is continuously controlling the machine and a (dangerous) failure of its function can result in a hazard.

3.2.28 Probability of dangerous Failure per Hour

PFH_D

Average probability of dangerous failure within 1 h.

NOTE: PFH_D should not be confused with probability of failure on demand (PFD).

3.2.29 Target failure value

Intended PFHD to be achieved to meet a specific safety integrity requirement(s).

NOTE 1: Re-write GB/T 20438.1-2006, definition 3.5.13.

NOTE 2: Target failure value is specified in terms of the probability of dangerous failure per hour.

3.2.30 Fault

Abnormal condition that may cause a reduction in or loss of, the capability of a SRECS, a subsystem, or a subsystem element to perform a required function

NOTE: Re-write GB/T 20438.1-2006, definition 3.6.1.

3.2.31 Fault tolerance

Ability of a SRECS, a subsystem, or subsystem element to continue to perform a required function in the presence of faults or failures.

NOTE: Re-write GB/T 20438.1-2006, definition 3.6.3.

3.2.32 Function block

Smallest element of a SRCF whose failure can result in a failure of the SRCF.

NOTE 1: In this standard, a SRCF (F) may be seen as a logical AND of the function blocks (FB), i.e. $F = FB_1$ AND FB_2 AND FB_n.

NOTE 2: This definition of a function block differs from those used in GB/T 15969.3 and other standards.

3.2.33 Function block element

Part of a function block.

3.2.34 Mean Time To Failure

safe operation of the machine while the faulty part is repaired. In this case, if the faulty part is not repaired within the estimated maximum time as assumed in the calculation of the probability of random hardware failure (see 6.7.8), then a second fault reaction shall be performed to maintain a safe condition.

Where the SRECS is designed for online repair, isolation of a faulty part shall only be applicable where this does not increase the probability of dangerous random hardware failure of the SRECS above that specified in the SRS.

After the occurrence of faults that reduce the hardware fault tolerance to zero, the requirements of 6.3.2 apply.

NOTE: The mean time to restoration (see IEV 191-13-08) that is considered in the reliability model will need to take into account the diagnostic test interval, the repair time and any other delays prior to restoration.

6.3.2 Where a diagnostic function(s) is necessary to achieve the required probability of dangerous random hardware failure and the subsystem has a hardware fault tolerance of zero, then the fault detection and specified fault reaction shall be performed before the hazardous situation addressed by the SRCF can occur.

EXCEPTION to 6.3.2: In the case of a subsystem implementing a particular SRCF where the hardware fault tolerance is zero and the ratio of the diagnostic test rate to the demand rate exceeds 100, then the diagnostic test interval of that subsystem shall be such as to enable the subsystem to meet the requirement for the probability of dangerous random hardware failure.

6.3.3 Where performance of a fault reaction function as part of a SRCF that is specified as SIL 3 has resulted in the machine being stopped, subsequent normal operation of the machine via the SRECS (e.g. enabling re-start of the machine) shall not be possible until the fault has been repaired or rectified. For SRCFs with a specified safety integrity of less than SIL 3, the behavior of the machine after performance of a fault reaction function (e.g. restarting normal operation) shall depend on the specification of relevant fault reaction functions (see 5.2.3).

6.4 Requirements for systematic safety integrity of the SRECS

NOTE: These requirements are applicable at the ‘system level’ where subsystems are interconnected to realize a SRECS. For requirements relevant to subsystem realization, see 6.7.8.

6.4.1 Requirements for the avoidance of systematic hardware failures

6.4.1.1 The following measures shall be applied:

a) the SRECS shall be designed and implemented in accordance with the functional safety plan (see 4.2);

b) proper selection, combination, arrangements, assembly and installation of

EXAMPLE: The function of the SRECS can be simulated on a computer via a software behavioral model (see 6.11.3.4) where individual subsystems or subsystem elements each have their own simulated behavior, and the response of the circuit in which they are connected is examined by looking at the marginal data of each subsystem or subsystem element.

6.4.2 Requirements for the control of systematic faults

The following measures shall be applied:

a) use of de-energization: the SRECS shall be designed so that with loss of its electrical supply a safe state of the machine is achieved or maintained;

b) measures to control the effect of temporary subsystem failures: the SRECS shall be designed so that, for example:

--- voltage variation (e.g. interruptions, dips) to an individual subsystem or a part of a subsystem does not lead to a hazard (e.g. a voltage interruption that affects a motor circuit shall not cause an unexpected start-up when the supply is restored), and

NOTE 1: See also relevant requirements of GB 5226.1.

In particular: overvoltage or undervoltage should be detected early enough so that all outputs can be switched to a safe condition by the power-down routine or a switch-over to a second power unit; and/or

where necessary, overvoltage or undervoltage should be detected early enough so that the internal state can be saved in non-volatile memory, so that all outputs can be set to a safe condition by the power-down routine, or all outputs can be switched to a safe condition by the power-down routine or a switch-over to a second power unit.

--- the effects of electromagnetic interference from the physical environment or a subsystem(s) do not lead to a hazard;

c) measures to control the effects of errors and other effects arising from any data communication process, including transmission errors, repetitions, deletion, insertion, re-sequencing, corruption, delay and masquerade;

NOTE 2: Further information can be found in GB/T 18657, GB/T 24339.1, EN 50159-2, and GB/t 20438.2.

NOTE 3: The term ‘masquerade’ means that the true contents of a message are not correctly identified. For example, a message from a non-safety component is incorrectly identified as a message from a safety component.

d) when a dangerous fault occurs at an interface, the fault reaction function shall be performed before the hazard due to this fault can occur. When a fault that reduces the hardware fault tolerance to zero occurs, this fault reaction shall take

6.6.2).

6.6.1.3 Where the use of diagnostics is necessary to achieve the required safety integrity when a fault is detected, the SRECS shall perform the specified fault reaction function (see 5.2 and 6.3).

6.6.1.4 Where a SRECS or part of a SRECS (i.e. its subsystem(s)) is to implement both SRCFs and other functions, then all its hardware and software shall be treated as safety-related unless it can be shown that the implementation of the SRCFs and other functions is sufficiently independent (i.e. that the normal operation or failure of any other functions do not affect the SRCFs).

NOTE: Sufficient independence of implementation can be established by showing that the probability of a dependent failure between the non-safety and safety-related parts is equivalent to that of the safety integrity level of the SRECS.

6.6.1.5 For a SRECS or its subsystems that implements safety-related control functions of different safety integrity levels, its hardware and software shall be treated as requiring the highest safety integrity level unless it can be shown that the implementation of the safety-related control functions of the different safety integrity levels is sufficiently independent.

NOTE: Sufficient independence of implementation can be established by showing that the probability of a dependent failure between the parts implementing SRCFs of different integrity levels is equivalent to that of the safety integrity level achieved by the SRECS.

6.6.1.6 Interconnections (e.g. wiring, cabling) other than digital data communication shall be considered to be part of one of the subsystems to which they are connected (see also item d) of 6.4.2).

6.6.1.7 Where digital data communication is used as a part of a SRECS implementation it shall satisfy the relevant requirements of GB/T 20438.2 in accordance with the SIL target(s) of the SRCF(s).

6.6.1.8 The information for use of the SRECS shall specify those techniques and measures necessary during the design life of the SRECS to maintain the safety integrity level.

6.6.2 Design and development process

The design and development shall follow a clearly defined process that shall take into account all aspects covered by the process shown in Figure 2.

NOTE: The approach of this standard is to apply a structured design process to the SRECS beginning from the requirements that are specified in the Safety Requirements Specification. Figure 3 shows the workflow of the design process and the terminology that applies to the different levels.

6.6.2.1 System architecture design

b) the estimated rates of failure (due to random hardware failures) declared in any modes which could cause a dangerous failure of the SRECS;

NOTE 1: For electromechanical subsystems, the probability of failure should be estimated taking into account the number of operating cycles declared by the manufacturer and the duty cycle (see 5.2.3). This information should be based upon a B10 value (i.e. the expected time at which 10% of the population will fail). See also IEC 61810-2.

c) constraints on the subsystem for:

--- the environment and operating conditions which should be observed in order to maintain the validity of the estimated rates of failure due to random hardware failures; and

--- the lifetime of the subsystem which should not be exceeded in order to maintain the validity of the estimated rates of failure due to random hardware failures;

d) any test and/or maintenance requirements;

e) the diagnostic coverage and the diagnostic test interval (when required, see Note 2);

NOTE 2: Item e) relates to diagnostic functions that are external to the subsystem. This information is only required when credit is claimed in the reliability model of the SRECS for the action of the diagnostic functions performed in the subsystem.

f) any additional information (e.g. repair times) which is necessary to allow the derivation of a mean time to restoration (MTTR) following detection of a fault by the diagnostics.

NOTE 3: Items b) to f) are needed to allow the probability of failure per hour of the SRCF to be estimated.

g) the SILCL due to architectural constraints (see 6.7.6) or:

--- all information which is necessary to enable the derivation of the safe failure fraction (SFF) of the subsystem as applied in the SRECS; and

NOTE 4: The required information is the possible failure modes of the subsystem. Based on the failure modes of the subsystem, it can be decided whether the subsystem failure causes a safe or a dangerous failure of the SRECS.

NOTE 5: For details on estimation of the SFF see 6.7.7.

--- the hardware fault tolerance of the subsystem;

h) any limits on the application of the subsystem which should be observed in order

NOTE: For electromechanical subsystems, the probability of failure should be estimated taking into account the number of operating cycles declared by the manufacturer and the duty cycle of the application (see 5.2.3). This information should be based upon a B10 value (i.e. the expected time at which 10 % of the population will fail). See also IEC 61810-2.

d) constraints on the subsystem element for

--- the environment and operating conditions which should be observed in order to maintain the validity of the information given in item c); and

--- the lifetime of the subsystem element which should not be exceeded in order to maintain the validity of the information given in item (c);

e) any periodic proof test and/or maintenance requirements;

f) features that can contribute to diagnostics (e.g. mechanically linked contacts);

g) any additional information (e.g. repair times) which is necessary to allow the derivation of a mean time to restoration (MTTR) following detection of a fault by the diagnostics;

h) any limits on the application of the subsystem element which should be observed in order to avoid systematic failures;

i) hardware fault tolerance.

6.7.5 Determination of the safety performance of the subsystem

The safety performance of a subsystem is characterized by the SILCL determined by its architectural constraints (6.7.6), its SILCL due to systematic integrity (6.7.9) and its probability of dangerous random hardware failure (6.7.8).

NOTE 1: The SILCL of a subsystem sets a limit for the maximum safety integrity level that can be claimed for a safety-related control function using this subsystem.

NOTE 2: Information about all three aspects is necessary to determine the SIL achieved by the safety-related control system implementing the allocated SRCF.

6.7.6 Architectural constraints on hardware safety integrity of subsystems

6.7.6.1 In the context of hardware safety integrity, the highest safety integrity level that can be claimed for a SRCF is limited by the hardware fault tolerances and safe failure fractions of the subsystems that carry out that SRCF. Table 5 specifies the highest safety integrity level that can be claimed for a SRCF that uses a subsystem taking into account the hardware fault tolerance and safe failure fraction of that subsystem. The architectural constraints given in Table 5 shall be applied to each subsystem. With respect to these architectural constraints:

g) the repair times for detected faults where the subsystem is designed for online repair.

NOTE 3: The maximum repair time will constitute one part of the time to restoration (see IEV 191-10-05), including also the time taken to detect a fault and any time period during which repair is not possible (see GB/T 20438.6, Annex B for an example of how the mean time to restoration can be used to calculate the probability of failure). For situations where the repair can only be carried out during a specific period of time, while the machine is shut down and in a safe state, it is particularly important that full account is taken of the time period when no repair can be carried out, especially when this is relatively large.

NOTE 4: A simplified approach for the estimation of the probability of dangerous random hardware failure of subsystems is given in 6.7.8.2. Other methods are available and the most appropriate method will depend on the circumstances. Available methods include:

a) fault tree analysis (see B.6.6.5 of GB/T 20438.7 and GB/T 7829);

b) Markov models (see C.6.4 of GB/T 20438.7 and IEC 61165-13);

c) reliability block diagrams (see C.6.5 of GB/T 20438.7).

NOTE 5: Failures due to common cause effects and data communication processes can result from effects other than actual failures of hardware components (e.g. electromagnetic interference, software errors, etc.) See 6.7.9.

6.7.8.1.3 For subsystems or subsystem elements where the probability of failure is given in relation to a number of operating cycles, these values shall be transformed into time-related values by using the specified duty cycle for the relevant SRCFs (see 5.2.3).

6.7.8.1.4 The diagnostic test interval of any subsystem having a hardware fault tolerance of more than zero shall be such as to enable the subsystem to meet the requirement for the probability of random hardware failure (see 6.3.1).

NOTE: This diagnostic test interval should be such that a fault is detected before the occurrence of a subsequent fault that may lead to dangerous failure of the subsystem and exceeds the target failure measure.

6.7.8.1.5 The diagnostic test interval of any subsystem having a hardware fault tolerance of zero shall be such that the requirements of 6.3.2 are fulfilled.

6.7.8.1.6 Where a low complexity subsystem is designed according to GB/T 16855.1 and failure (PFH_D) given in Table 7 can be used to estimate the hardware safety integrity (see 6.6.3.2). validated according to GB/T 16855.2 and also meets the requirements for architectural constraints (see 6.7.6) and systematic safety integrity (see 6.7.9), the threshold values of probability of dangerous failure (PFH_D) given in Table 7 can be used to estimate the hardware safety integrity (see 6.6.3.2).

6.7.9.1.2 In addition, one or more of the following measures shall be applied taking into account the complexity of the subsystem:

a) hardware design review (e.g. by inspection or walk-through): to reveal by reviews and/or analysis discrepancies between the specification and implementation;

NOTE 1: In order to reveal discrepancies between the specification and implementation, any points of doubt or potential weak points concerning the realization, the implementation and the use of the product are documented so they may be resolved; taking into account that on an inspection procedure the author is passive and the inspector is active whilst on a walk-through procedure the author is active and the inspector is passive.

b) computer-aided design tools capable of simulation or analysis: perform the design procedure systematically and include appropriate automatic construction elements that are already available and tested;

NOTE 2: The integrity of these tools can be demonstrated by specific testing, or by an extensive history of satisfactory use, or by independent verification of their output for the particular subsystem that is being designed. See 6.11.3.4.

c) simulation: perform a systematic simulation of a subsystem design in terms of both the functional performance and the correct dimensioning of their components.

NOTE 3: The function of the subsystem can be simulated on a computer via a software behavioral model (see 6.11.3.4) where individual components of the circuit each have their own simulated behavior, and the response of the subsystem in which they are connected is examined by looking at the marginal data of each component.

6.7.9.2 Requirements for the control of systematic failures

6.7.9.2.1 The following measures shall be applied:

a) measures to control the effects of insulation breakdown, voltage variations and interruptions, overvoltage and undervoltage: subsystem behavior in response to insulation breakdown, voltage variations and interruptions, overvoltage and undervoltage conditions shall be pre-determined so that the subsystem can achieve or maintain a safe state of the SRECS;

NOTE 1: See also relevant requirements of GB 5226.1. In particular:

– overvoltage should be detected early enough so that all outputs can be switched to a safe condition by the power-down routine or a switch-over to a second power unit; and/or

– the control circuit voltage should be monitored and a power-down initiated, or a switch-over to a second power unit, if it is not within its specified range;

the same as those specified for the corresponding SRCF(s); or

NOTE 1: Architectural constraints on hardware safety integrity need not apply to the realization of diagnostic function(s).

--- where the probability of dangerous random hardware failure is of an order of magnitude greater than that specified for the SRCF, then a test shall be performed to determine whether diagnostic function(s) or diagnosing device(s) remain operational. It is assumed that such a test of the diagnostic function(s) or diagnosing device(s) be carried out at a minimum of 10 times during the interval between proof tests applied to the subsystem.

NOTE 2: A test of the diagnostic function(s) should as far as practicable cover 100 % of those parts implementing the diagnostic function(s).

NOTE 3: Where a diagnostic function is implemented by the logic solver of the SRECS it can be unnecessary to perform a separate test of the diagnostic function as its failure can be revealed as a failure of the SRCF.

NOTE 4: A test can be performed by either external means (e.g. test equipment) or internal dynamic checks (e.g., embedded within the logic solver) of the SRECS.

6.9 Hardware implementation of the SRECS

The SRECS shall be implemented in accordance with the documented SRECS design.

6.9.1 SRECS interconnection

6.9.1.1 The SRECS shall be interconnected so as to satisfy appropriate parts of the SRECS safety requirements specification and those requirements relevant to conductors, cabling and wiring practices in GB 5226.1.

6.9.1.2 Measures for avoiding and controlling failures of interconnecting conductors and cables shall be realized in accordance with 6.4.1 and 6.4.2.

6.10 Software safety requirements specification

6.10.1 General

Where software is to be used in any part of a SRECS implementing a safety-related control function(s), a software safety requirements specification shall be developed and documented.

6.10.2 Requirements

6.10.2.1 A software safety requirements specification shall be developed for each subsystem on the basis of the SRECS specification and architecture.

6.10.2.2 The specification of the requirements for software safety for each

d) verification and validation, including structural testing (white box) of the application software, functional testing (black box) of the integrated application program and interface testing (grey box) of the interaction with the SRECS and its application specific hardware configuration;

e) safe modification.

6.11.3.1.4 Testing shall be the main verification method used for the application software. Test planning shall address the following:

– the policy for verification of the integration of software and hardware;

– test cases and test results;

– types of tests to be performed;

– test equipment including tools, support software and configuration description;

– test criteria on which the completion of the test shall be judged;

– physical location(s) (e.g. factory or site);

– dependence on external functionality;

– the amount of test cases necessary; and

– completeness with respect to the related functions or requirements.

6.11.3.1.5 Where the application software is to implement both non-safety and safety-related control functions, then all of the application software shall be treated as safety-related, unless adequate independence between the functions can be demonstrated in the design.

6.11.3.1.6 The design shall include data integrity checks and reasonableness checks at the application layer (e.g. checks in communication links, bounds checking on sensor inputs, bounds checking on data parameters).

6.11.3.1.7 The application software design shall include self-monitoring of control flow and data flow unless such functions are included in the embedded software. On failure detection, appropriate actions shall be performed to achieve or maintain a safe state.

6.11.3.1.8 Where previously developed software library functions are to be used as part of the design, their suitability in satisfying the specification of requirements for software safety shall be justified. Suitability shall be based upon evidence of satisfactory operation in similar applications that have been demonstrated to have similar functionality, or shall be subject to the same verification and validation procedures as would be expected for any newly developed safety-related software. Constraints from the previous software environment (for example operating system

6.11.3.4 Requirements for support tools, user manual and application languages

6.11.3.4.1 A suitable set of tools, including configuration management, simulation, and test harness tools shall be selected. The availability of suitable tools (not necessarily those used during initial system development) to supply the relevant services over the lifetime of the SRECS shall be considered. The suitability of the tools shall be explained and documented.

NOTE: The selection of development tools depends on the nature of the software development activities, the embedded software and the software architecture. Verification and validation tools such as code analyzers, and simulators may be needed.

6.11.3.4.2 Wherever necessary a sub-set of the application programming language shall be defined.

6.11.3.4.3 Application software shall be designed taking into account constraints and known weaknesses included in the SRECS and subsystem(s) user manuals.

6.11.3.4.4 The application language selected shall either:

– be processed using a translator/compiler which shall be assessed to establish its fitness for purpose;

– be completely and unambiguously defined or restricted to unambiguously defined features;

– correspond to the characteristics of the application;

NOTE: An application's characteristics refer, for example to any performance constraints.

– contain features that facilitate the detection of programming mistakes; and

– support features that match the design method;

or, the deficiencies of the language used shall be documented in the software architecture design description and the fitness for purpose of the language shall be explained including additional measures necessary to address the identified shortcomings of the language.

6.11.3.4.5 The procedures for use of the application language shall specify good configuration practice, proscribe unsafe generic software features (for example, undefined language features, unstructured designs, etc.), identify checks that can be used to detect errors in the configuration and specify procedures for documentation of the application program. As a minimum, the following information shall be contained in the application program documentation:

a) legal entity (for example company, author(s), etc.);

– ensure each branch of any application software modules is exercised;

– ensure boundary data is exercised;

– ensure sequences are correctly implemented, including relevant synchronization conditions.

6.11.3.7.4 The results of the application software module testing shall be documented.

6.11.3.7.5 Where software has already been assessed or when a significant amount of positive operating experience is available, the amount of testing may be reduced.

6.11.3.8 Requirements for application software integration testing

NOTE: Testing that the software is correctly integrated is a verification activity.

6.11.3.8.1 The application software tests shall verify that all application software modules and components/subsystems interact correctly with each other and with the underlying embedded software to perform their intended function and do not perform unintended functions that could jeopardize any safety function.

6.11.3.8.2 The results of application software integration testing shall be documented, stating:

– the test results; and

– whether the objectives of the test criteria have been met.

6.11.3.8.3 If there is a failure, the reasons for the failure and corrective action taken shall be included in the test results documentation.

6.11.3.8.4 During application software integration, any modification or change to the software shall be subject to a safety impact analysis that shall determine:

– all software modules impacted; and

– all necessary re-verification and re-design activities.

6.12 Safety-related electrical control system integration and testing

NOTE: SRECS integration is usually carried out prior to installation but, in some cases, the SRECS integration cannot be carried out until after installation (for example, when the application software development is not finalized until after installation).

6.12.1 General requirements

6.12.1.1 The SRECS shall be integrated according to the specified SRECS design. As part of the integration of all subsystems and subsystem elements into the SRECS, the SRECS shall be tested according to the specified integration tests. These tests

6.13.2.2 Appropriate records of the installation of the SRECS shall be produced, stating any test results. If there is a failure, the reasons for the failure shall be recorded.

7 Information for Use of the SRECS

7.1 Objective

Information on the SRECS shall be provided to enable the user to develop procedures to ensure that the required functional safety of the SRECS is maintained during use and maintenance of the machine.

7.2 Documentation for installation, use and maintenance

NOTE 1: See also Clause 6 of GB/T 15706.2 that provides general information that should be considered during drafting of accompanying documents.

NOTE 2: One or more items of the documentation described in this subclause may have been developed in order to satisfy other aspects of this standard.

The documentation shall provide information for installation, use and maintenance of the SRECS. This shall include:

a) a comprehensive description of the equipment, installation and mounting.

b) a statement of the intended use of the SRECS and any measures that can be necessary to prevent reasonably foreseeable misuse.

c) information on the physical environment (e.g. lighting, vibration, noise levels, atmospheric contaminants) where appropriate.

d) overview (block) diagram(s) where appropriate.

e) circuit diagram(s).

f) proof test interval or lifetime.

g) a description (including interconnection diagrams) of the interaction (if any) between the SRECS function (s) and the machine electrical control system function(s).

h) a description of the necessary measures to ensure the separation of the SRECS function(s) from the machine electrical control system function(s).

i) a description of the safeguarding and of the means provided to maintain safety where it is necessary to suspend the SRCF(s) (e.g. for manual programming, program verification).

8.2.1 The validation of the SRECS shall be carried out in accordance with a prepared plan (see 4.2).

NOTE 1: In some cases, the safety validation cannot be completed until after installation (for example, when the application software development is not finalized until after installation).

NOTE 2: Validation of a programmable SRECS comprises validation of both hardware and software. The requirements for validation of software are contained in 6.11.3.

8.2.2 Each SRCF specified in the SRECS requirements specification (see 5.2), and all the SRECS operation and maintenance procedures shall be validated by test and/or analysis.

8.2.3 Appropriate documentation of the SRECS safety validation testing shall be produced, which shall state for each SRCF:

a) the version of the SRECS safety validation plan being used and the version of the SRECS tested;

b) the SRCF under test (or analysis), along with the specific reference to the requirement specified during the SRECS safety validation planning;

c) tools and equipment used, along with calibration data;

d) the results of each test;

e) discrepancies between expected and actual results.

8.2.4 When discrepancies occur, corrective action and re-testing shall be carried out as necessary and documented.

8.3 Validation of SRECS systematic safety integrity

8.3.1 The following shall be applied:

a) functional testing to reveal failures during the specification, design and integration phases, and to avoid failures during validation of SRECS software and hardware shall be applied. This shall include verification (e.g., by inspection or test) to assess whether the SRECS is protected against adverse environmental influences and shall be based upon the safety requirements specification;

NOTE 1: See also 6.12.2.1.

b) interference immunity testing to ensure that the SRECS is able to satisfy 5.2.3. Testing for immunity to electromagnetic interference need not be performed on SRECS subsystems or subsystem elements where adequate immunity of the SRECS for its intended application can be shown by analysis;

NOTE 2: The SRECS should, wherever practicable, be loaded with a typical application

highest permissible marginal values. The most essential responses of the SRECS are inspected and compared with the safety requirements specification.

d) field experience: the use of field experience from different applications as one of the measures to avoid faults during SRECS validation.

NOTE 3: See also 6.12.2.

9 Modification

9.1 Objective

[Translator: No content in this Clause]

9.2 Modification procedure

This Clause specifies the modification procedure(s) to be applied when modifying the SRECS during design, integration and validation (e.g. during SRECS installation and commissioning).

9.2.1 The request for a modification of the SRECS can arise from, for example:

– safety requirements specification changed;

– conditions of actual use;

– incident/accident experience;

– change of material processed;

– modifications of the machine or of its operating modes.

NOTE: Interventions (e.g. adjustment, setting, repairs) on the SRECS made in accordance with the information for use or instruction manual for the SRECS are not considered to be a modification in the context of this Clause.

9.2.2 The reason(s) for the request for a modification of the SRECS shall be documented.

9.2.3 The effect of the requested modification shall be analyzed to establish the effect on the functional safety of the SRECS.

9.2.4 The modification impact analysis and the effect on the functional safety of the SRECS shall be documented.

9.2.5 All accepted modifications that have an effect on the SRECS shall initiate a return to an appropriate design phase for its hardware and/or for its software (e.g. specification, design, integration, installation, commissioning, and validation). All subsequent phases shall then be carried out in accordance with the procedures

13) authorization to carry out the required change request activity shall be dependent on the results of the impact analysis.

9.3.3 The documentation of the change control process shall contain at least

a) a plan of each modification process;

b) a documentation of each of the above mentioned organizational requirements and procedures;

c) a documentation of the decision making process and each SRECS-relevant decision made;

d) a chronological documentation (logbook) of the change request procedures including

- identified hazards which may be affected;
- description of the change request (hardware and/or software);
- reason(s) for the change request (see also 9.2.1);
- decision made (and authorization for each decision);
- the impact analysis;
- re-verification (of each phase) and revalidation;
- all documents affected by the change request activities;
- all activities which were carried out during the change process and the persons/entities who were responsible for them;

e) documentation of the following information to permit a subsequent audit:

- configuration status;
- release status;
- the justification for and approval of all modifications;
- the details of the modification.

10 Documentation

10.1 The documentation shall:

– be accurate and concise;

– be easy to understand by those persons having to make use of it;

software elements developed by the designer for previous projects, or commercially available software (e.g. modules for calculations, algorithms for data sorting).

When dealing with this type of software, and especially in the case of commercial software elements, the designer does not always have access to all the elements needed to satisfy the previous requirements (e.g. what tests have been carried out, is the design documentation available). Specific co-ordination with the analyst can therefore be necessary at the earliest possible moment.

The designer should indicate the use of pre-existent software to the analyst, and the designer should demonstrate that pre-existent software has the same level as the other software elements. Such a demonstration should be done:

a) either by using the same verification activities on the pre-existent software as on the rest of the software; and/or

b) through practical experience where the pre-existent software has functioned on a similar system in a comparable executable environment (e.g. it may be necessary to evaluate the consequences of a change of the compiler or of a different software architecture format).

NOTE 1: The goal of indicating the use of pre-existent software is to open up consultation with the analyst as early as possible about any eventual difficulties that this type of software might cause. The integration of pre-existent source modules can be the cause of certain anomalies or unsafe behavior if they were not developed with the same rigor as the rest of the software.

Pre-existent software should be identified using the same configuration management and version control principles that are applied to the rest of the software.

NOTE 2: Configuration management and version control should be exercised over all the software components, regardless of their origin.

C.2.4 Software design

Description of the software design should include a description of:

– the software architecture that defines the structure decided to satisfy specifications;

– inputs and outputs (e.g. in the form of an internal and external data dictionary), for all the modules making up the software architecture;

– the interrupts;

– the global data;

– each software module (inputs/outputs, algorithm, design particularities, etc.);

– module or data libraries used;

– pre-existent software used.

NOTE 1: The purpose is to be able to trace the historical development of each article: what modifications have been made, why, and when.

Software configuration management should allow a precise and unique software version identification to be obtained. Configuration management should associate all the articles (and their version) needed to demonstrate the functional safety.

All articles in the software configuration should be covered by the configuration management procedure before being tested or being requested by the analyst for final software version evaluation.

NOTE 2: The objective here is to ensure that the evaluation procedure be performed on software with all elements in a precise state. Any subsequent change may necessitate revision of the software so that it can be identifiable by the analyst.

Procedures for the archiving of software and its associated data should be established (methods for storing backups and archives).

NOTE 3: These backups and archives can be used to maintain and modify software during its functional lifetime.

C.3.5 Software modifications management

Any software modification which has an impact on the functional safety of the SRECS should be subject to the rules established for modification and configuration management such that the development process be recommenced at the highest "upstream" point needed to take the modification into account without diminishing the functional safety.

NOTE: In particular, the documentation should also be updated, and all necessary verification activities carried out. This guarantees that the software will keep all its initial properties after any modification.

C.4 Development tools

Tools used during the development procedure (compiler, linker, tests, etc.) should be identified (name, reference, version, etc.) in the documentation associated with the software version (e.g. in the version control documentation).

NOTE: Different versions of tools do not necessarily produce the same results. Precise identification of tools thus directly demonstrates the continuity of the process of generation of an executable version in the event that a version is modified.

C.5 Reproduction, delivery

C.5.1 Executable code production

Any option or change in the generation, during the software production should be recorded (e.g. in the version sheet) so that it is possible to say how and when the software was generated.

C.5.2 Software installation and exploitation

Analysis activities and software design verification should verify the conformity to specifications.

NOTE 1: The purpose is to ensure that the software specification and design (both detailed and preliminary) are coherent.

An external validation review (with the analyst) should be held at the end of the validation phase.

NOTE 2: This can be used to ascertain whether or not the element satisfies the specifications.

The result of each review should be documented and archived. It should include a list of all actions decided on in the review process, and the review conclusion (decision on whether or not to move on to the next activity). The activities defined in the review should be monitored and treated.

C.9 Software testing

C.9.1 General validation

Before writing the first test sheets, it is important to establish a test strategy in a test plan. This strategy indicates the approach adopted, the objectives that have been set in terms of test coverage, the environments and specific techniques used, the success criteria to be applied, etc.

The test objectives should be adapted to the type of software, and to the specific factors. These criteria determine the types of test to be undertaken – functional tests, limit tests, out of limit tests, performance tests, load tests, external equipment failure tests, configuration tests – as well as the range of objects to be covered by the tests (functional mode tests, safety-related control function tests, tests of each element in the specification, etc.).

Verification of a new software version should include non-regression tests.

NOTE: Non-regression tests are used to ensure that the modifications performed on the software have not modified the behaviour of the software in any unexpected way.

C.9.2 Software specification verification: validation tests

The purpose of these verifications is to detect errors associated with the software in the target system environment. Errors detected by this type of verification include: any incorrect mechanism to treat interruptions, insufficient respect of running time requirements, incorrect response from the software operating in transient mode (start-up, input flow, switching in a degraded mode, etc.), conflicts of access to different resources or organizational problems in the memory, inability of integrated tests to detect faults, software/hardware interface errors, stack overflows. Validation tests are the principal component of software specification verification.

The test coverage should be made explicit in a traceability matrix and ensure that:

– each element of the specification, including safety mechanisms, is covered by a validation test; and

– the real-time behaviour of the software in any operational mode can be verified.

Furthermore, the validation should be carried out in conditions representative of the operational

This is an excerpt of the PDF (Some pages are marked off intentionally)

------ The End ------