Design of Microcontrollers for Safety Critical Operation

Karl Greb and Anthony Seely
Microcontroller
Texas Instruments

Agenda

- Introduction to functional safety
- IEC 61508 standard
- Preview of ISO 26262 standard
- Functional safety lessons learned
- Summary and closing thoughts

Introduction to Functional Safety

What is Functional Safety?

- IEC 61508 Definition:
  - *Safety* is the freedom from unacceptable risk of physical injury or of damage to the health of people, either directly, or indirectly as a result of damage to property or to the environment.
  - *Functional Safety* is part of the overall safety that depends on a system or equipment operating correctly in response to its inputs.

- ISO 26262 Definition:
  - Absence of unacceptable risk due to hazards caused by mal-functional behavior of electrical and/or electronic systems

State of the Industry

- TI has been shipping microcontrollers into automotive safety critical products for over 20 years.
- Starting in 2006, major shift in customer requirements in automotive safety applications
  - Increased awareness of functional safety standards
  - Requirements to support customer assessment of systems incorporating TI product
  - Requirements for development of new products in flows compliant to IEC 61508 and later ISO 26262.
- This trend continues today in automotive
  - Application of functional safety techniques moving from system level→ component level → IP component level
  - System level safety measures are being implemented on-chip as we move to more complex system on chip devices.

Functional Safety Basic Concepts

- All systems will have some inherent, quantifiable failure rate. It is not possible to develop a system with zero failure rate.
- For each application, there is some tolerable failure rate which does not lead to unacceptable risk.
- Acceptable failure rates vary per application, based on the potential for direct or indirect physical injury in the event of system malfunction.
- Categories can be developed to quantify similar levels of risk. These are known as *Safety Integrity Levels*, or *SILs*.

Safety Failures and their Causes

- Failures in a functional safety system can be broadly classified into two categories: systematic and random failures

- Systematic Failures
  - Result from a failure in design or manufacturing
  - Often a result of failure to follow best practices
  - Rate of systematic failures can be reduced through continual and rigorous process improvement

- Random Failures
  - Result from random defects inherent to process or usage condition
  - Rate of random failures cannot generally be reduced; focus must be on the detection and handling of random failures in the application.

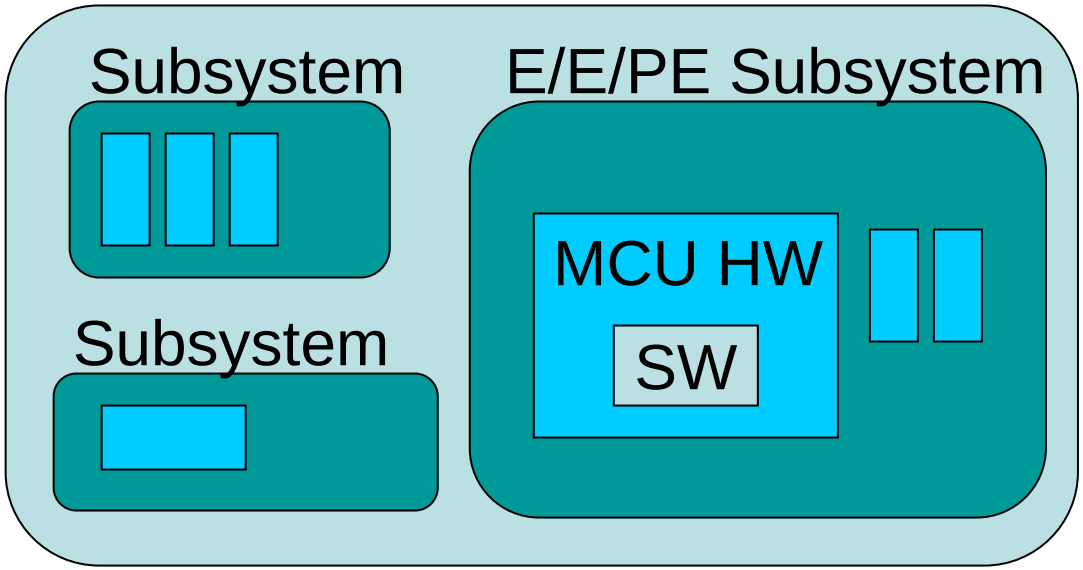
Lifecycle and Requirements

- To provide adequate levels of functional safety, the entire lifecycle of a product must be addressed.
  - Concept
  - Design
  - Prototyping
  - Release to manufacturing
  - Field implementation
  - Removal from field usage at end of life

- For many markets the safety lifecycle can be more than 20 years.

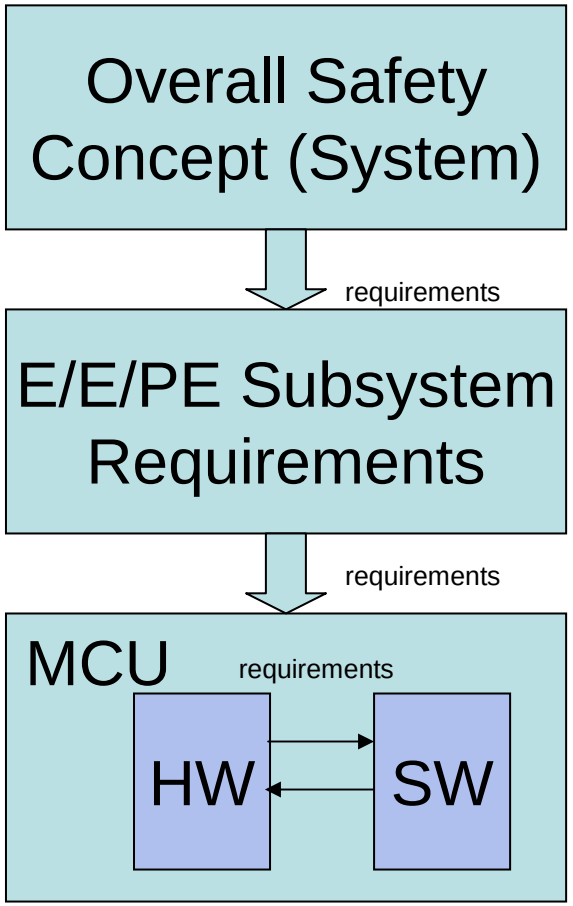
Microcontroller in Safety Operation

- System components are generally classed Electrical, Electronic, or Programmable Electronic (E/E/PE). A microcontroller is a complex PE component.
- MCU HW and SW functions may be safety critical. Safety of MCU subsystem cannot be realized without respecting both HW and SW requirements.
- System integrator has final responsibility for overall MCU system safety



Safety Requirements Derivation

- Safety concept starts at system level
  - Definition of tolerable risk/safety integrity levels
  - Definition of safety function
  - Allocation of safety goals per function
- System level requirements flow down to functional subsystems
- Final components, such as an MCU, have HW and SW requirements for safety
  - SW requirements may originate
    - As allocate system safety functions
    - Integrity checks on HW
    - Integrity checks on SW



Standards for Functional Safety

- Many standards exist for application of functional safety principles to general or specific markets.
  - IEC 61508 – general market
  - IEC 60730 – white goods
  - ISO 26262 – automotive

- These standards provide:
  - Processes to assess risk for safety critical systems and assign safety goals
  - Best practice development process requirements in order to reduce systematic failures
  - Frameworks for quantitative analysis of random failure rates and effectiveness of diagnostic measures to detect random failure
  - Ongoing procedures to ensure functional safety after product deployment.

IEC 61508 Overview

What is IEC 61508?

- Consensus standard developed by International Electrotechnical Commission (IEC) for general market functional safety applications.
- Primarily designed for application at system level, but has been applied to product level and component level
- Provides measures for management and reduction of both systematic and random failures
- Addresses E/E/PE elements including hardware and software
- Many points are variable/open to different interpretation by different readers. Care is needed to align interpretations of all concerned parties.

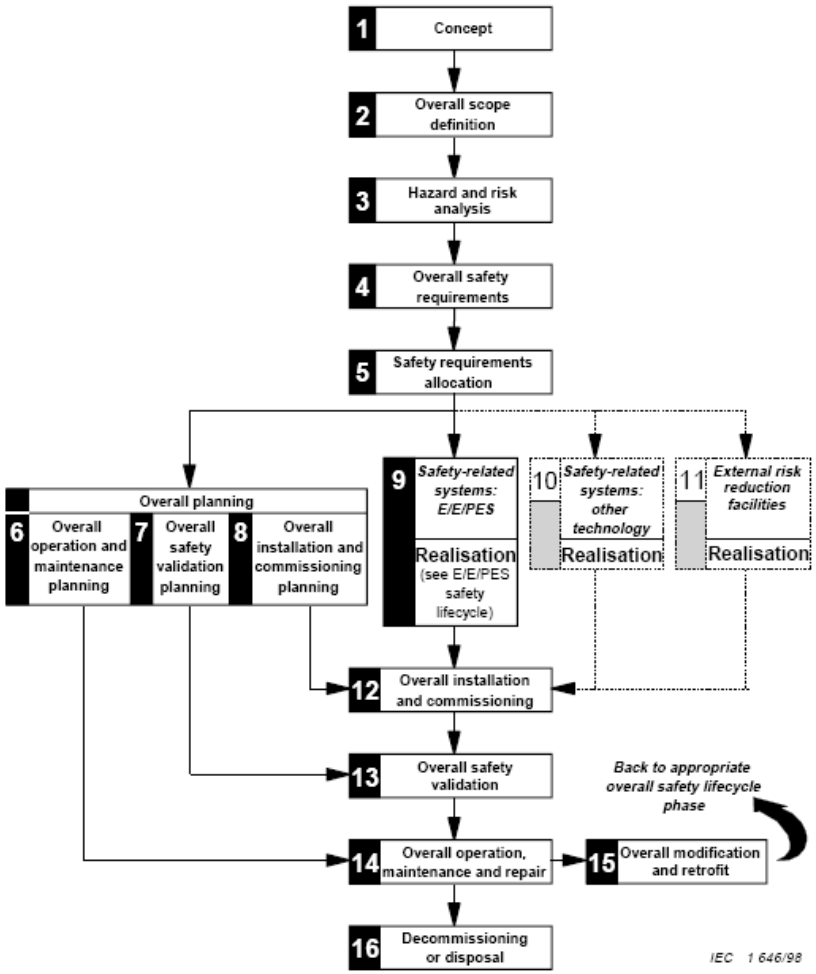
IEC 61508 Augmentation

- A general market standard cannot reflect the “state of the industry” in terms of accepted usage cases, risk, etc.

- Many industries have developed industry specific standards to augment or replace IEC 61508
  - IEC 61511: Industrial Process Control
  - IEC 61513/62138: Nuclear
  - EN 50128: Railway
  - IEC 62061: Machine Tooling/Industrial

IEC 61508 Safety Life Cycle

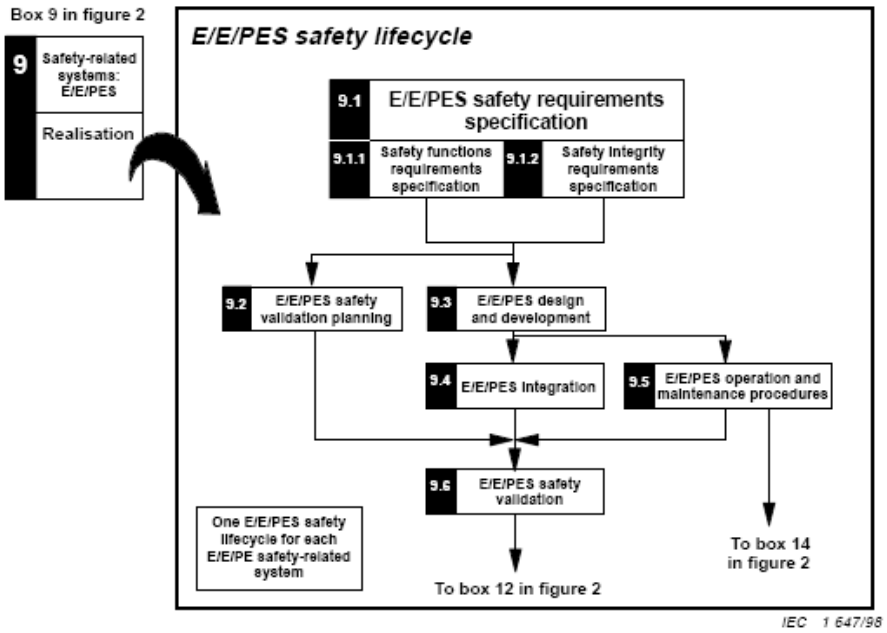
IEC 61508 -1 Figure 2



- Structured flow used to prevent systematic failures.
- Safety activities do not end after product design, but continue throughout product lifetime.
- Adaptation of the IEC 61508 life cycle to existing product development flows can be challenging.

ARM techcon³
DESIGN TO THE POWER OF THREE

IEC 61508 Development Flow



IEC 61508 -1 Figure 3

- Safety designs require additional work during the pre-design, design, and validation phases.
- Standard has requirements and recommendations to be completed during each phase.
- It is generally not possible to retrofit these stages to a product which is already developed.

ARM techcon³
DESIGN TO THE POWER OF THREE

IEC 61508 Failure Rate Definitions

- Failure rate is represented with the Greek character lambda, λ , and can be broken into many categories.
  - λ_S: rate of "safe" failures which do not affect safety function
    - λ_{SD} : safe, detected failure rate
    - λ_{SU} : safe, undetected failure rate
  - λ_D: rate of "dangerous" failures which compromise the safety function
    - λ_{DD} : dangerous, detected failure rate
    - λ_{DU} : dangerous, undetected failure rate

- Failure rate is often expressed in "FITs". One FIT (Failure In Time) = 1 failure per billion hours of operation (1×10^{-9} failures/hour)

Metrics for Probability of Failure

- Probability of Failure on Demand (PFD)
  - PFD = λ_{DD} * (RT) + λ_{DU} * (TI)
  - RT = repair time, TI = test interval

- Average Probability of Failure on Demand (PFD_{AVG})
  - Assumes low demand for safety function

$$PFD_{AVG}(TI) = \frac{1}{TI}\int_{o}^{TI}(PFD)dt$$

- Probability of Failure per Hour (PFH)
  - PFH = λ_{DU}
  - Assumes high demand or continuous demand for safety function

- Note: Values based on 1oo1 architecture; algorithms differ slightly for other architectures (see -6)

SIL Requirements

Safety integrity level	Low demand mode of operation (Average probability of failure to perform its design function on demand)
4	$\geq 10^{-5}$ to $< 10^{-4}$
3	$\geq 10^{-4}$ to $< 10^{-3}$
2	$\geq 10^{-3}$ to $< 10^{-2}$
1	$\geq 10^{-2}$ to $< 10^{-1}$
NOTE – See notes 3 to 9 below for details on interpreting this table.	

IEC 61508 -1 Table 2

Safety integrity level	High demand or continuous mode of operation (Probability of a dangerous failure per hour)
4	$\geq 10^{-9}$ to $< 10^{-8}$
3	$\geq 10^{-8}$ to $< 10^{-7}$
2	$\geq 10^{-7}$ to $< 10^{-6}$
1	$\geq 10^{-6}$ to $< 10^{-5}$
NOTE – See notes 3 to 9 below for details on interpreting this table.	

IEC 61508 -1 Table 3

- Failure rates required to achieve a particular SIL are dependent on the probability that the safety function will be used.
- A function which is more frequently required has a stronger demand on achieved failure rate.
- Most automotive applications are considered high or continuous demand.

Safe Failure Fraction

- Safe Failure Fraction (SFF) is defined as the ratio of safe and dangerous (but detected) failures in a system as compared to the total failure rate.

$$SFF = \left(\frac{\lambda_{DD} + \lambda_{SD} + \lambda_{SU}}{\lambda_{DD} + \lambda_{DD} + \lambda_{SD} + \lambda_{SU}} \right) = \left(\frac{\lambda_{DD} + \lambda_{SD} + \lambda_{SU}}{\lambda} \right)$$

- SFF is a metric best applied at the system level per safety function. SFF is generally not applied at the component level.

SFF Requirements Per SIL

Table 3 – Hardware safety integrity: architectural constraints on type B safety-related subsystems

Safe failure fraction	Hardware fault tolerance (see note 2)		
	0	1	2
< 60 %	Not allowed	SIL1	SIL2
60 % – < 90 %	SIL1	SIL2	SIL3
90 % – < 99 %	SIL2	SIL3	SIL4
≥ 99 %	SIL3	SIL4	SIL4
NOTE 1 See 7.4.3.1.1 to 7.4.3.1.4 for details on interpreting this table.			
NOTE 2 A hardware fault tolerance of N means that N + 1 faults could cause a loss of the safety function.			
NOTE 3 See annex C for details of how to calculate safe failure fraction.			

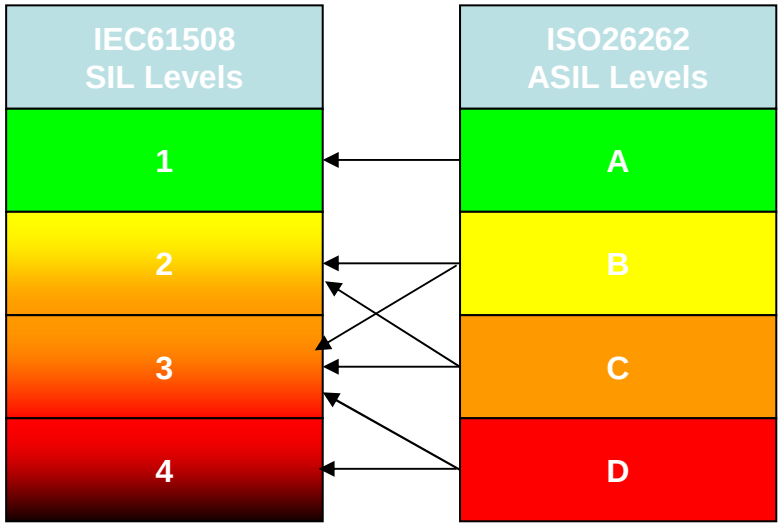
IEC 61508 -2 Table 3

- SFF requirements per SIL target are dependent on hardware fault tolerance type.
  - Type “A” – all failure mechanisms are known
  - Type “B” – all failure mechanisms are not known
- State of industry is that any IC built in smaller than 1 micron process geometry is type “B”
- Hardware Fault Tolerance (HFT) is the ability of a system to continue safe operation after a dangerous failure.
  - HFT = 0 is a single channel system
  - HFT = 1 is at least a dual channel system
- The higher the fault tolerance, the lower the SFF required to meet the target SIL.

ISO 26262 Preview

ISO 26262

- ISO 26262 (Road Vehicles) is under development to meet automotive industry specific needs as an IEC 61508 replacement.
- Standard is in definition (ISO-TC22-SC3-WG16).
  - Draft available June 2009
  - Expected release in 2010
  - TI is participating in the US working group.



- ISO26262 defines 4 safety integrity levels (ASIL-A,B,C,D)
- There is no direct correlation between IEC61508 SIL and ISO 26262 ASIL levels

ISO 26262 Key Differences from IEC 61508

- Standard is aligned to automotive industry use cases and definitions of acceptable risks.

- Development deliverables are clearly defined in ISO 26262 as “work products”. This greatly simplifies definition of compliant development flows and narrows possible interpretations.

- Narrowed and tightened definition of safety metrics
  - No Low Demand of Operation – all systems are effectively high or continuous demand
  - No Hardware Fault Tolerance Type – it is assumed impossible to comprehend all possible faults in system (Type B)
  - No reduction in safety metric requirements for hardware fault tolerance (HFT).

Safety Failure Targets Compared

ISO 26262 -5 (Draft)

Table G.1 — Random hardware failure target values

ASIL Level	Random hardware failure target values
D	$< 10^{-8}\ h^{-1}$
C	$< 10^{-7}\ h^{-1}$
B	$< 10^{-7}\ h^{-1}$
A	$< 10^{-6}\ h^{-1}$

IEC 61508 -1

Table 3 – Safety integrity levels: target failure measures for a safety function operating in high demand or continuous mode of operation

Safety integrity level	High demand or continuous mode of operation (Probability of a dangerous failure per hour)
4	$\geq 10^{-9}$ to $< 10^{-8}$
3	$\geq 10^{-8}$ to $< 10^{-7}$
2	$\geq 10^{-7}$ to $< 10^{-6}$
1	$\geq 10^{-6}$ to $< 10^{-5}$
NOTE – See notes 3 to 9 below for details on interpreting this table.	

- ASIL-A has 1000 FIT system budget – equivalent to SIL 2
- ASIL-B has 100 FIT system budget – equivalent to SIL 3
- ASIL-C has 100 FIT system budget – equivalent to SIL 3
- ASIL-D has 10 FIT system budget – equivalent to SIL 4

Single Point and Latent Faults

ISO 26262 -5 (Draft)

Table E.1 — Single point faults metric and latent faults metric target values

	ASIL B	ASIL C	ASIL D
Single point faults metric	> 90 %	> 97 %	> 99 %
Latent faults metric	> 60 %	> 80 %	> 90 %

- ISO 26262 removes the concept of safe failure fraction and replaces with the single point faults metric and latent faults metric.
- A single point fault is any non-detected fault which leads directly to violation of a safety goal.
- A latent fault is a undetected, dangerous failure caused by the presence of multiple faults which independently may not cause dangerous failures.

Functional Safety Lessons Learned

A Journey to IEC 61508 Compliance

- TI has been shipping microcontrollers into automotive safety critical products for over 20 years.
- In 2007 TI started development of the first TMS570 Cortex R4F based device with explicit compliance to the IEC 61508 standard.
- This section describes some of our key lessons learned in the process.

When in Doubt, Consult Experts

- Application of functional safety at MCU level is a complex topic.
- Without expert advice and training, it can be difficult to start new safety critical development.
- TI engaged with two third parties to jump start IEC 61508 standard compliant development.

exida

Your **Functional Safety** Partner of **Choice**

exida is a unique organization rich with Functional Safety support, products, services, experience, expertise, and an unending quest to exceed customer expectations. Fully integrated with global Functional Safety requirements and standards, exida delivers best-in-class global Functional Safety products, services, and solutions to both automation hardware manufacturers and process market end users

FACHHOCHSCHULE REGENSBURG
UNIVERSITY OF APPLIED SCIENCES

Prof. Dr. Jürgen Mottok
University of Applied Sciences
Head of the Laboratory for Safe and Secure Systems (LaS3)
Member of the advisory board of the Bavarian IT-Security and Safety Cluster

Prof. Georg Scharfenberg
University of Applied Sciences
Dean of Faculty Electronics and Information Engineering
Head of the laboratory for microcomputers
Member of the Bavarian IT-Security and Safety Cluster

ARM techcon³
DESIGN TO THE POWER OF THREE

Gap Analysis of Development Process

- The first step in developing for functional safety is an honest gap analysis of your existing development process to the safety standard in question.

- Frequently seen development gaps include:
  - Lack of Functional Safety Management (FSM) plan
  - Lack of Safety Requirements Specification (SRS)
  - Safety architecture not clearly defined
  - No quantitative analysis of safety architecture effectiveness
  - Lack of safety validation plan
  - Consideration of impact to safety in change management flow

How Good is my Current Architecture?

- Most hardware safety designs are based on a qualitative analysis techniques such as FMEA – Failure Mode and Effect Analysis.
- A qualitative analysis alone is not sufficient to determine compliance to either IEC 61508 or ISO 26262 safety standards.
- Quantitative analysis techniques, such as FMEDA – Failure Modes Effects and Diagnostic Analysis – are needed to determine effectiveness of the safety architecture.
- These methods should be used as an iterative tool to evaluate and improve the product safety architecture until safety goals are met.
- Availability of component level failure rate data greatly simplifies the job of the system integrator.

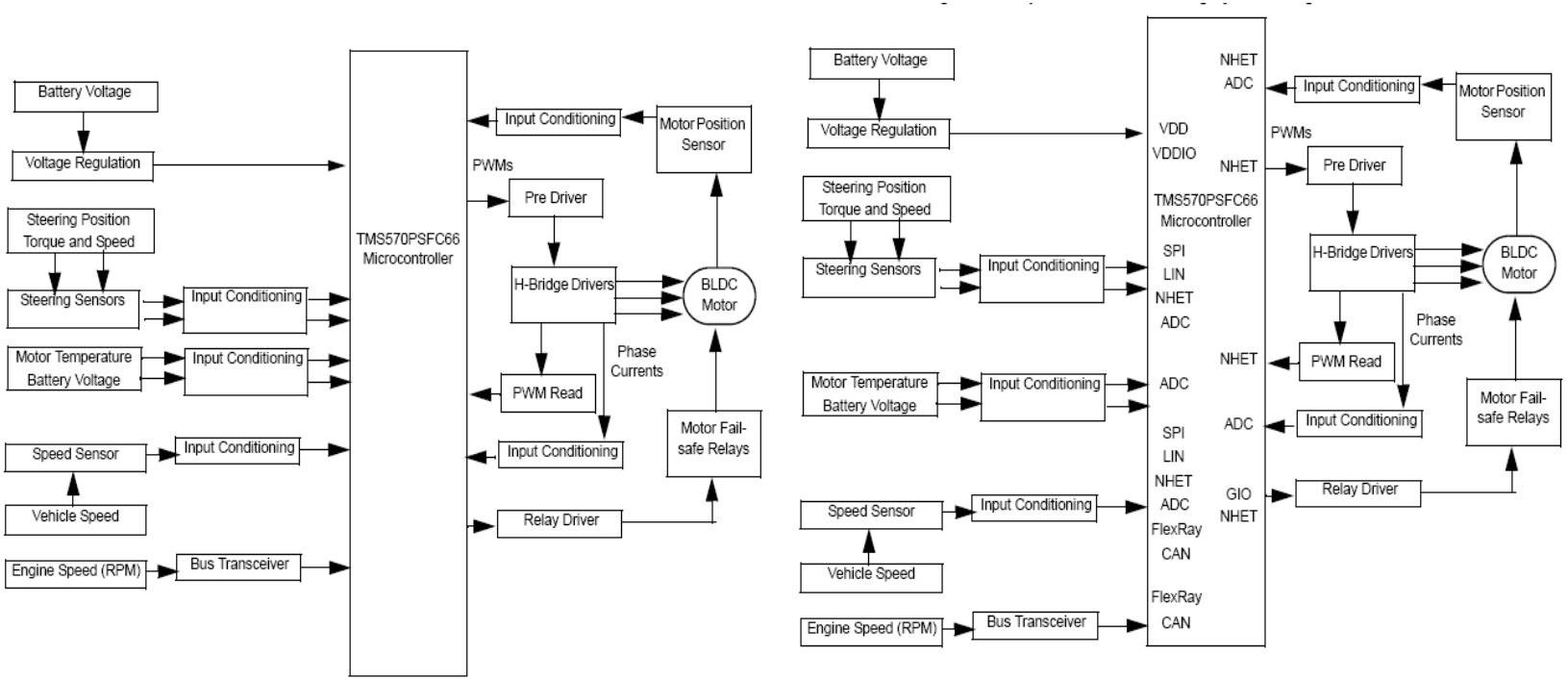
Determination of Failure Rate

- Determination of failure rates is challenging. Safety failure rates differ from those reported in most semiconductor quality metrics.
- ***There is no single accepted set of faults or fault models which is accepted industry wide. You must be flexible and align with the faults recognized by your end customer!***
- Hard fault rates were derived from models using the IEC 62380 standard for semiconductor fault modeling and application of an automotive engine bay environmental profile.
- Soft fault rates were derived from experimental particle bombardment tests on silicon in conjunction with Los Alamos National Labs.

What Functions are Safety Critical?

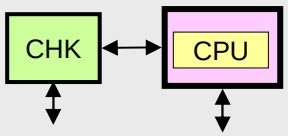
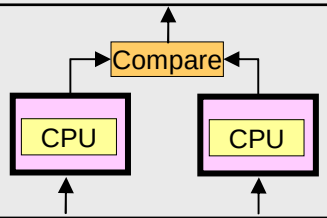
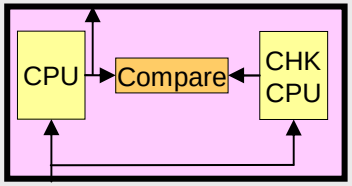
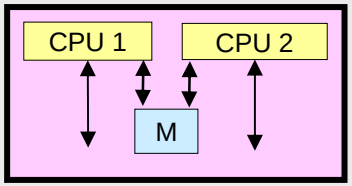
Theoretical Steering System

Potential Realizations of Steering System

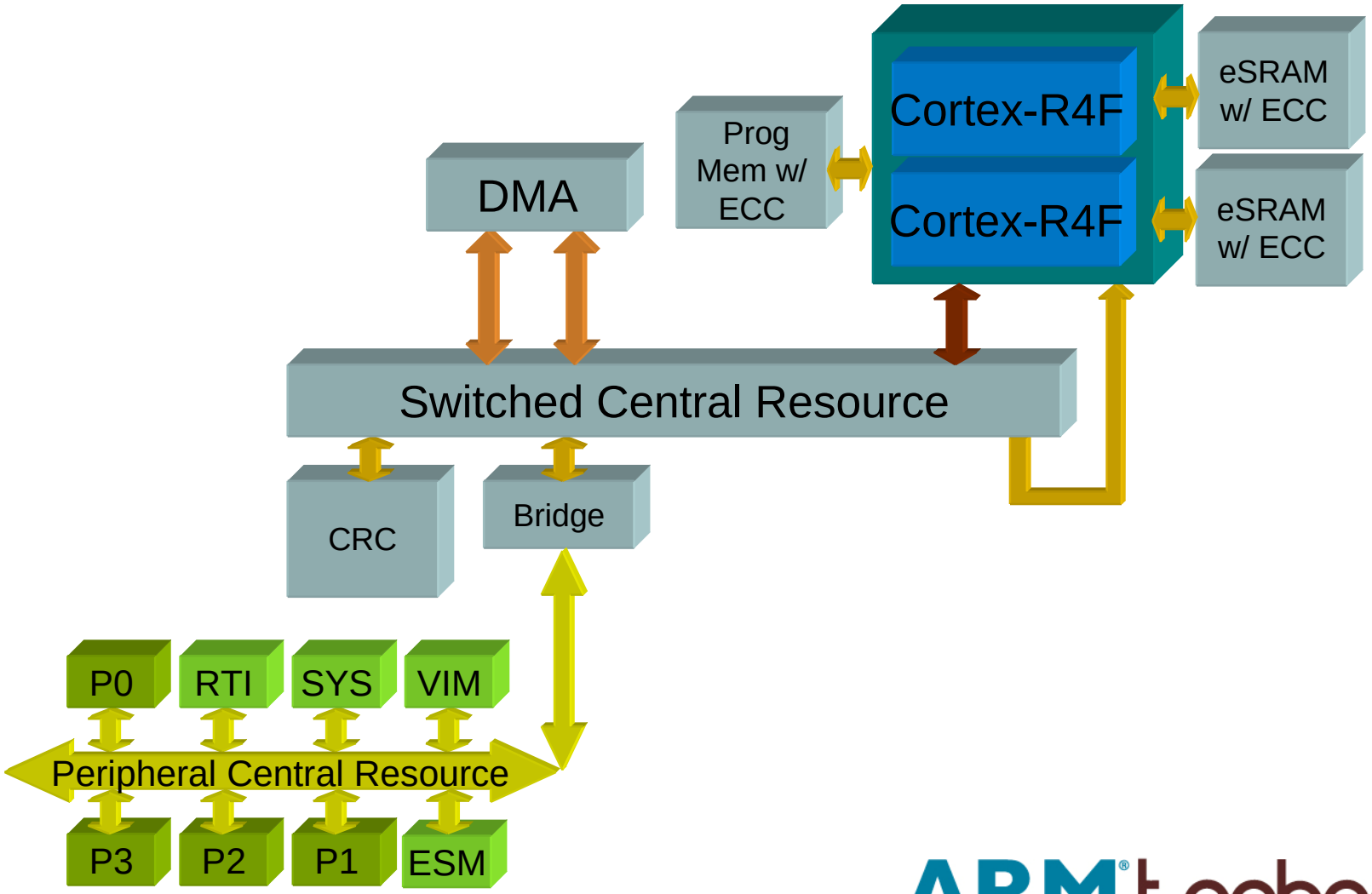


Variability in sensors, actuators, and overall customer system concept mean that no peripherals can be ruled non safety critical by semiconductor vendor providing catalog products. Customers must be presented enough detailed information to determine expected safety metrics for their particular configuration.

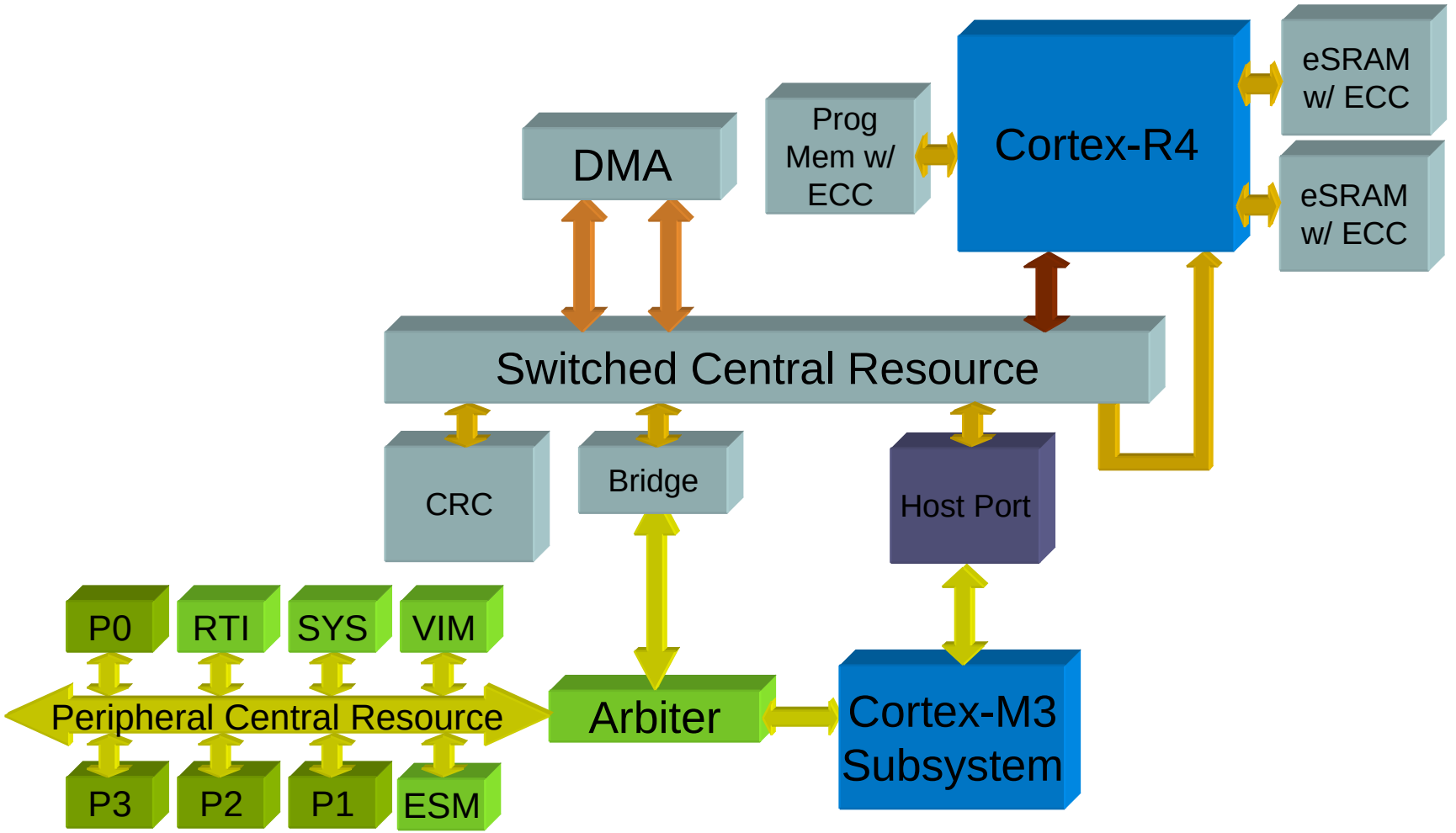
How Should I Protect Processing Functions?

Method	Diagram	Advantages	Disadvantages
Single 32b device with 8/16b checker device		▪ Relatively low cost ▪ Safety through diverse hardware	▪ SIL may be limited by processing capacity of simple checker micro ▪ Processing power limited by frequent on-line diagnostics
Dual devices with external compare of safety outputs and optional SW message passing		▪ SIL3 generally possible ▪ Can double performance for non-safety critical tasks ▪ Simplicity of sourcing ▪ Potential for redundancy	▪ Increased complexity for safety SW synchronization ▪ Additional cost, board space
Device with internal safety logic (CPU) in lock-step		▪ SIL3 generally possible ▪ Reduction in board space ▪ Reduced S/W complexity	▪ Customized implementation ▪ Same performance as single CPU
Single device dual CPU (not in lockstep) with internal self test.		▪ SIL3 generally possible ▪ Multi-core performance for non-safety critical tasks	▪ Customized implementation ▪ Increased complexity for safety SW synchronization

Simplified TMS570 Lockstep System



Simplified TMS570 Dual Core AMP System



What HW Documentation is Needed?

- Safety Manual
  - A TRM or user guide for the HW safety architecture
  - Guides on suitability for usage to achieve safety goals
  - Requirements and recommendations for use of HW/SW diagnostics to achieve safety goals.
  - Guidelines for usage, repair, training of operator, and proper disposal
- FMEDA Report
  - Quantitative assessment of failure rates per module
  - Often used for definition of system safety architecture
  - Failure rates assume system complies to all requirements found in the safety manual
- Independent Assessment Report (Optional)
  - 3rd party audit of compliance to functional safety standard
  - Not required by all customers but useful in evaluating suppliers.

Safety Challenges for MCU Software

- Software has no random failure modes or wear-out mechanisms. All defects are introduced by mistakes in requirements capture, or development.

- Probability of failure does not increase with time, instead it is related to probability that the inputs / set of conditions necessary to trigger the failure occurs.

- Methods to control software safety integrity are similar to methods used to control systematic hardware failure rates. These focus on design, construction, and validation of the product.

Programming Language Selection

- How to select a language or compiler?
  - Conformance to national or international standard
  - Unambiguous definition
  - Strongly typed
  - Supports detection of programming errors
  - Supports limited language subsets
- C is explicitly not recommended by IEC 61508-7 for SIL3,SIL4
  - Not strongly typed
  - Compiler specific and implementation specific behavior
  - Easy to introduce programming errors
- A C language subset is highly recommended for all SIL
- Other recommended languages include Modula 2, Pascal, Fortran, ADA

MISRA-C C Language Subset

- Guidelines released in 1998, updated in 2004
- Widely adopted by automotive, aerospace, medical, industrial markets
- Rules (required) & advisories covering 21 areas:
  - Areas addressed include: language extensions, pointer type conversions, control flow, and standard libraries
  - Total of 122 rules + 20 advisories
  - Example rule: *(6.2) "Signed and unsigned char shall only be used for the storage and use of numeric values."*
- Some compliance checks can be automated
  - TI's TMS470/TMS570 C/C++ compiler will check 70% of the rules
  - Remaining rules require manual checks

Object Oriented Languages

- Economic pressure to design increasingly complex systems more quickly motivates the use of OOT in the embedded market
- Use of OOT in safety systems raises special concerns.
- Issues raised by CAST (aviation) regarding C++ include:
  - Indeterminate execution time caused by dynamic memory management related to object instantiation / removal
  - Difficulty meeting traceability requirements (test cases/results) and collecting structural code coverage metrics introduced by object inheritance.
  - Dead/deactivated code – overridden methods
  - Reuse of class libraries not originally developed for safety applications

Object Oriented Languages for Safety

- MISRA published a set of guidelines for the use of C++ in safe critical systems in 2008. Similar to MISRA-C these guidelines outline a safe subset of the C++ language.

- Java Community has developed “The Real-Time Specification for Java” JSR-1, and a working group exists with the aim of creating a subset appropriate for safety critical systems (JSR-302 under development).

Can an MCU Simplify System SW Safety?

- MINIMIZE the number of additional requirements for software system design that originate as tests/diagnostics required for the MCU to achieve a specific HW SIL level

- PROVIDE tools appropriate to the safety applications – compilers, trace/debug tools, simulation models, and safety libraries

- PARTNER with OS/Middleware vendors that specialize in delivering COTS components designed for safety critical systems.

Software Requirements to Achieve HW SIL

- MCU Safety Manual / FMEDA require software to initiate / perform specific diagnostics for HW SIL

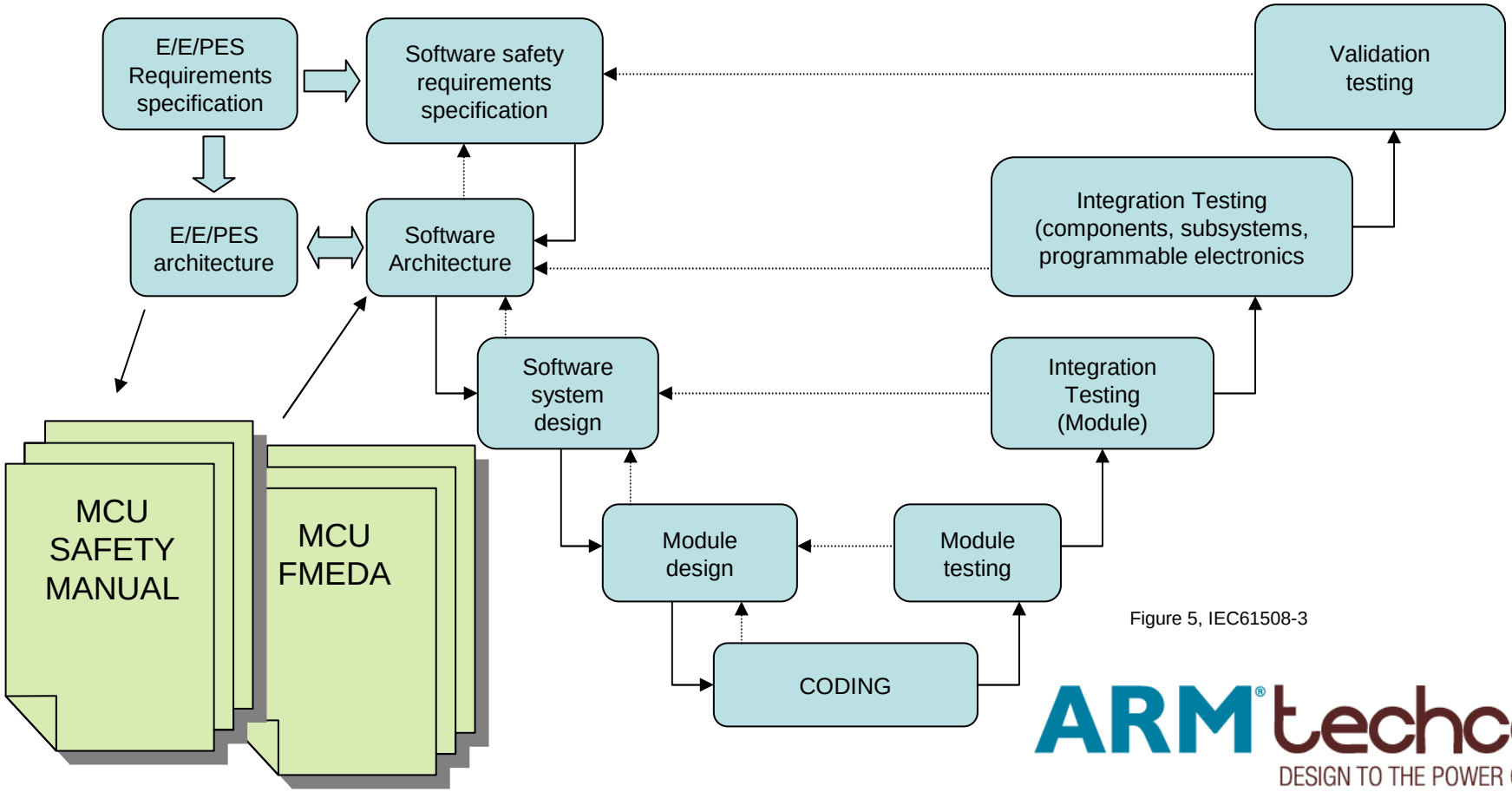


Figure 5, IEC61508-3

Minimizing SW Requirements to Achieve Desired SIL on HW

	TMS570 Safety MCU	Non-Safety General Market MCU
CPU	Continuous checking by HW compare (lockstep CPU) or traditional SW methods	Redundant comparison with single CPU or Diverse/reciprocal comparisons between multiple CPUs
	CPU Logic BIST (self-test by HW)	CPU Self Test by software
CPU <> Interconnect	Continuous checking and multi-bit HW redundancy	Inspection with test patterns
FLASH / ROM	Background CRC automated by HW	CRC computation by CPU
	Continuous checking and multi-bit HW redundancy	
RAM	Continuous checking and multi-bit HW redundancy	RAM tests by SW – generally not production quality algorithms
	RAM Production Tests (BIST)	

- Diagnostics in hardware reduce fault detection time and reduce software development, simplifying system integration.

Summary and Closing Thoughts

Summary

- Functional Safety must start at system level and incorporate all system components.
- Development of complex system on chip devices for safety application requires component vendors to take an active design role to ensure system safety.
- Continuously tightening requirements mean that system implementation general purpose components becomes more complex and costly. Trend toward commercial, off-the-shelf (COTS) components designed for safety critical systems.
- There are significant opportunities for vendors who can provide COTS HW and SW components explicitly developed for use in functional safety applications.
- Component level compliance to IEC 61508 or ISO 26262 is possible with effort and the right device architecture.

TMS570PSFC66 Achieves SIL3 Certification

exida has assessed and certified the TMS570PSFC66 Cortex R4F based MCU to be IEC 61508 SIL3 capable

"…exida worked closely with TI and performed a detailed review of the TI development processes and conducted a reliability analysis and found that the TMS570PSFC66 device meets all of the relevant SIL3 requirements of IEC 61508"

- Dr. Bill Goble, CEO, exida Consulting LLC

ARM techcon3
DESIGN TO THE POWER OF THREE

Open Question and Discussion Period

Recommended Further Reading

- Books/Papers
  - “Safety Instrumented Systems Verification – Practical Probabilistic Calculations” by William M. Goble and Harry Cheddie, ISA 2005.
  - “Software Safety and Reliability” by Debra S. Herrmann, IEEE Computer Society 1999.
  - “Safeware – System Safety and Computers” by Nancy G. Leveson, Addison-Wesley 1995.
  - “Guide to the Software Engineering Body of Knowledge”, IEEE Computer Society, 2004.
  - “Object Oriented Technology in Civil Aviation Projects: Certification Concerns”, CAST 2000.
- Standards
  - “IEC 61508 – Functional safety of electrical/electronic/programmable electronic safety-related systems”, IEC 2000.
  - “ISO 26262 – Road vehicles – Functional Safety”, ISO 2009 (draft).
  - “MISRA-C:2004 – Guidelines for the use of the C language in critical systems”, MISRA, 2004.