Safety and security for automotive SoC design

Chris Turner
Director of advanced technology marketing, CPU group

Seoul, June 28th 2016
Taipei, July 1st 2016

©ARM 2016
Innovation in automotive computing

Emissions & efficiency
Advanced powertrain ECU

Electrification
Hybrid and all-electric

Infotainment
Smartphone connectivity

Connected car
Telematics, eCall, LTE, V2X

In-car networks
CAN, LIN, Ethernet

Reducing accidents
Passive and active safety

Smart cities
Intelligent traffic systems

Assisted driving
ADAS to highly assisted

Autonomous vehicles
Cloud services
Functional safety

- Systems that must function correctly in order to avoid hazardous situations
  - Faults must be detected and controlled

- Safety-critical
  - Braking, steering, acceleration, chassis control, air bag, seat belt tension…
  - Driver relies on these systems to function correctly all the time
  - Probably ASIL D

- Safety ‘nominal’
  - Lane departure, speedometer, rear camera…
  - So long as the driver is made aware if the system is not working
  - Probably ASIL B
Applicable standards

- A number of functional safety standards exist
  - ISO 26262 – Road vehicles
  - IEC 61508 – Electrical, electronic, programmable electronic systems
  - DO 254 – Electronics that fly: e.g. airplanes & helicopters

- Standards always represent an industry consensus
  - Long lead times for standards development (5-10 years)
  - Often lagging behind true state-of-the-art

- Safety Integrity Levels (low to high)
  - SIL 1 to SIL 3
    - Typically SIL 1 or SIL 3
  - ASIL A to ASIL D
    - Typically ASIL B (e.g. parking) or ASIL D (e.g. braking)
ISO26262 – From IP designs to systems

- **IP supplier**
- **IP integrator, e.g. MCU designer**
- **Tier-1 designer**
- **Automotive OEM**

**ISO 26262**

- Applicable requirement
- Not applicable requirements

**Requirements, assumptions**

**Supporting documentation (evidence)**

- From IP designs to systems
Faults

- Systematic faults
  - Hardware errata
  - Software bugs
  - Incorrect specification
  - Incomplete requirements

- Random faults
  - Caused by hard errors, e.g. a failed transistor or metal connection
  - Caused by soft errors, e.g. alpha particle switches a RAM bit
  - Permanent faults that persist, or may be recoverable if they’re managed
  - Transient faults that appear but may then go away of their own accord. However, these could cause a system to operate incorrectly
  - Latent faults that exist but do not impact the system for some while, e.g. a RAM error in a bit which isn’t accessed until some time after it occurs

Source: ASML
Functional safety engineering

**Random faults**
- **Fault models**
  - Sources of errors
  - Possible failures
  - Fault detection design
  - Fault control design
    - Error reporting
- **Fault metrics**
  - Permanent faults
  - Transient faults
  - FMEA
  - SIL, e.g. for ASIL B or D
    - 90 or 99% SPFM
    - 60 or 90% LFM
    - Within FTTI
- **Process**
  - Requirements (traceable)
  - Planning
  - Training
  - Design & Verification
  - Tools
  - Review & Assessment
  - Audit
- **Development**
  - Safety lifecycle
  - Traceability
    - Project report
  - Documentation
    - Safety manual
    - AoU
    - FMEA
    - DIR

**Systematic faults**
- **Design (HW and SW)**
  - LBIST and MBIST
  - STL aka SWBIST
  - ECC
  - Exceptions
  - Watchdogs
  - DCLS
  - Redundant execution
  - Diversity
  - Fail operational
    - At system level
    - Fail silent at SoC level
- **Deployment**
  - QA
  - Errata
- **Audit & Assessment**
  - Internal and external
  - A&A independence
  - Level of detail
    - ASIL B
    - ASIL D
- **Top-level requirements**
  - Hazards
  - Risks
  - Safety goal
    - Layered safety requirements
  - Required SIL/ASIL
    - Layered system components

Considered for ASIL C and D. Redundancy will be required.

**Considered for ASIL C and D. Redundancy will be required.**
Automotive safety integrity levels

- Fault metrics
  - Measurement of possible faults that are detectable, and mitigated locally if possible

- Single Point Fault Metric
  - Immediately effective faults

- Latent Fault Metric
  - Initially silent faults, e.g. in memory bits

- QM: Quality Managed

<table>
<thead>
<tr>
<th>Safety</th>
<th>SPFM</th>
<th>LFM</th>
</tr>
</thead>
<tbody>
<tr>
<td>QM</td>
<td>Design assurance</td>
<td></td>
</tr>
<tr>
<td>ASIL A</td>
<td>Nominal</td>
<td></td>
</tr>
<tr>
<td>ASIL B</td>
<td>90%</td>
<td>60%</td>
</tr>
<tr>
<td>ASIL C</td>
<td>97%</td>
<td>80%</td>
</tr>
<tr>
<td>ASIL D</td>
<td>99%</td>
<td>90%</td>
</tr>
</tbody>
</table>

*Expect to extend to ASIL B post 2018*
Fault mitigation, i.e. detection and control

- BIST for ASIL A and B
  - Testing: SW BIST or Logic BIST
  - Add features to improve coverage and speed up test time
  - But transient faults can be missed

- Diversity for ASIL C and D
  - Redundant HW
  - Redundant SW
Fault mitigation, i.e. detection and control 2

- BIST for ASIL A and B

  - Testing: SW BIST or Logic BIST
  - Add features to improve coverage and speed up test time
  - But transient faults can be missed

- Diversity for ASIL C and D

  - Transient fault could cause compute error, leaving wrong data. Subsequent test could still show everything working OK
  - Transient fault flagged ‘immediately’ by checking between diverse hardware
  - Transient fault flagged within FTTI by difference between two software results
Different solutions

- **Diverse systems**
  - Dual or triple systems
  - Diverse implementations
  - Random and systematic
  - Can be fail-operational

- **Redundant hardware**
  - Dual Core Lock Step, or
  - Dual asynchronous clusters
  - Memory ECC
  - Doubles/adds area & power
  - Need to test the checkers

- **Redundant execution**
  - Temporal redundancy
  - Can halve performance
  - But 1:1 duty cycle may not be needed
  - Separated safety island required as a checker
Redundant hardware in MPCore

- Dual core lock step, Cortex-R
  - Complemented with memory ECC
- Processor RTL is divided apart
  - Performance impacted

- Dual asynchronous cluster, Cortex-A
  - Memory ECC expected
  - No DCLS in-cluster
- Redundant execution
  - Spatial separation
  - Some temporal separation by software
  - Mitigation of common cause faults in L3?
Typical fault detection and control mechanisms

- **Processor implementation**
  - ECC or parity on memories
  - Soft and hard error management
  - ECC protected bus ports
  - Dual Core Lock-Step with delay
  - Error reporting interface
  - Timing protection
  - Logic BIST
  - Memory BIST
  - Software BIST

- **ARM architecture**
  - Memory protection unit
  - Hypervisor for software separation
  - Exception handling
# Functional safety for ARM Cortex processors

## Safety documentation package

- **Support for up to ASIL D systematic capability**
  - Safety Manual
  - FMEA Report
  - Development Interface Report
  - Third-party functional safety assessment report

## Product features supporting functional safety

### ARMv8-R
- ARMv8-R virtualization
- More hardware features
- Bus ECC
- TCM ECC

### Cortex-R5
- Dual-core lock-step
- Cache ECC
- TCM ECC interface

### Cortex-M7
- ECC interface

### ARMv8-M
- Cache parity / ECC

### Cortex-A processors*
- Memory protection unit
- Exception handling

### Cortex-M4
- Cortex-M3
- Cortex-M0+

## Increasing feature set provided by ARM for fault detection and control

* except Cortex-A73
Silicon IP for functional safety

Safety management
Requirements management
Quality

Errata management
Training
Documentation

Fault detection & control
Memory protection
Error correction
Redundancy

Error reporting
Fault containment
Fault injection

Safety Package contains
Safety Manual
Failure Modes Effects Analysis

Development Interface Report or agreement

Processes

Design & Verification

Safety Package
Functional safety package

- Safety manual
  - Describes design and verification process
  - Fault detection & control features
  - Verification summary
- FMEA report
  - Evidence of safety analysis on the ARM IP
  - Aids partners with their own SoC level FMEA
- Development Interface Report
  - Defines interworking relationship with ARM
  - Replaces bespoke dev. interface agreement (DIA)
- Other
  - Future products may have additional IP
  - E.g. Software test library
Safety analysis through FMEA

- FMEA – failure modes and effects analysis – is a systematic safety analysis method
- Allows analysis of effects of faults at given design hierarchy
- Used throughout safety-related designs
  - IP level analysis
  - SoC level analysis
  - ECU level analysis
  - Vehicle level analysis
What’s in an FMEA?

- Design is subdivided into smaller elements
  - Number of hierarchy levels depends on complexity of designs

- Faults within each element, and effects of faults locally and globally are analysed

- FMEA records are typically presented in a tabular format

### Instruction fetch unit (IFU)

![IFU Diagram]

Example fault consideration within an element (IFU)

- **Fault:** e.g. flipped bit in program counter
- **Error:** e.g. incorrect program counter value
- **Failure:** e.g. incorrect instruction execution

### FMEA excerpt with potential effects

<table>
<thead>
<tr>
<th>Failure mode ID</th>
<th>Component level</th>
<th>Block level</th>
<th>Safety related</th>
<th>Failure mode description</th>
<th>Potential end effect at CPU boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td>FMEDA713</td>
<td>cortexa53</td>
<td>L1 duplicate tag RAMs</td>
<td>Y</td>
<td>Failure in reading from L1 duplicate tag RAMs</td>
<td>Performance</td>
</tr>
<tr>
<td>FMEDA710</td>
<td>cortexa53</td>
<td>L1 duplicate tag RAMs</td>
<td>Y</td>
<td>Failure in reading from L1 duplicate tag RAMs</td>
<td>Security Violation</td>
</tr>
<tr>
<td>FMEDA568</td>
<td>cortexa53</td>
<td>L1 duplicate tag RAMs</td>
<td>Y</td>
<td>Failure in reading from L1 duplicate tag RAMs</td>
<td>Livelock</td>
</tr>
<tr>
<td>FMEDA567</td>
<td>cortexa53</td>
<td>L1 duplicate tag RAMs</td>
<td>Y</td>
<td>Failure in reading from L1 duplicate tag RAMs</td>
<td>Modified Instruction Execution</td>
</tr>
<tr>
<td>FMEDA543</td>
<td>cortexa53</td>
<td>L1 duplicate tag RAMs</td>
<td>Y</td>
<td>Failure in reading from L1 duplicate tag RAMs</td>
<td>Data corruption</td>
</tr>
</tbody>
</table>
AEC Q100 and ISO/TS 16949

- I’m often asked about these in the context of synthesisable (soft) IP…

- ISO/TS 16949
  - A technical specification, in conjunction with ISO 9001:2008 for quality management systems for automotive-related products
  - Applies to an organisation’s sites where there is manufacturing

- AEC-Q100
  - A high-level test standard for automotive grade electronics
  - Product grade 0 up to 150°C, grade 1 to 125°C, grade 2 to 105°C etc.

- Long term reliability, e.g. failures in time due to electron migration
Automotive security threats

- Theft
  - Intrusion

- Privacy
  - Data protection
  - Location, speed, direction

- Safety
  - Malicious hacking
  - Unauthorised upgrades
Safe systems must be secured

- Cars offer multiple attack surfaces
  - From outside – wireless
  - From inside – On-board diagnostics port

- Everything in a vehicle is connected
  - Conventional CAN and LIN bus
  - Real-time Ethernet

- Debug, software provisioning and updating has to be facilitated

- Vehicle architectures are evolving to include gateways with security
  - And protocol conversions
ARM technology for automotive

Performance and Architecture leadership for the car of the future

Best performance within tight Thermal and Space constraints

Safe and Secure with functional safety support and ARM TrustZone®

Rich ARM Partnership with 1000+ Ecosystem

Scalable solutions throughout the vehicle

Diverse and competitive supply chain to the automotive industry