Agenda

- About Case Study
- Roadmap – AGL
- General - AGL
- Functional Safety - AGL
- Functional Safety Analysis - AGL
Audience, Takeaways

**Areas / Intended Audience**
- Functional safety – ISO26262
- IC, HUD use cases
- Software Development - Automotive
- GNU/Linux Subsystem

**Takeaways**
- Basics of FS feasibility in AGL
- Basics of FS process for AGL
- FS specific Design strategies for IC & HUD SW

**Consolidation**
- QnA
- Further interests
Background / Key Motivation / Interest

Quoting from “https://www.automotivelinux.org/about”

“Automotive Grade Linux (AGL) is a Linux Foundation Workgroup dedicated to creating open source software solutions for automotive applications. Although the initial target for AGL is In-Vehicle-Infotainment (IVI) systems, additional use cases such as “instrument clusters” and telematics systems will eventually be supported.”

This case study checks the feasibility of implementing Instrument cluster + Head up display use cases in AGL where functional safety is a requirement.
Architecture Approaches – Safety Perspective

Hardware Platform

ASIL B IC Features

Non ASIL IC Features

RTOS (ASIL B)

Safety Partition

Non Safe Partition

Hardware Platform

Hypervisor (ASIL B)

Safety Partition

Non Safe Partition

Opensource ASIL B Hypervisors? 
Opensource ASIL B RTOS? 
Performance? 
Complexity? 
Cost?
Roadmap – in AGL

- IC, HUD MW (ASIL B)
- Fastboot in AGL
- eCockpit in AGL
- Functional safety in AGL
<table>
<thead>
<tr>
<th>General – from AGL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BSP and SOC</strong></td>
</tr>
<tr>
<td><strong>Version</strong></td>
</tr>
<tr>
<td><strong>Kernel</strong></td>
</tr>
</tbody>
</table>
Functional Safety – Analysis in AGL

**Functional Safety**: Absence of unacceptable risk due to hazards caused by malfunction behavior of systems

Risk = Exposure * Effect * Probability

High Risk

Low Risk
Current Software Architecture - AGL

Source: AGL Specification 1.0
Derived - Software Architecture with Safety Stack – in AGL
Way To Functional Safety Compliance – in AGL Arch

- Identify existing components in AGL for IC,HUD use cases
- Other components for IC,HUD (to be developed)
- Safety V/S Non-safety Partitioning

Safety Lifecycle

Freedom From Interference (FFI)
Existing components and Tools used – in AGL

- **Kernel (v3.10)**
  - Task management
  - Memory Management
  - Protection

- **Device Drivers**

- **Libraries**
  - GLIBC (v2.20)
  - POSIX
  - ALSA (v1.0.28)
  - DRM (v2.4)
  - KMS (v1.4.0)

- **Other Tools used**
  - gcc for arm Compiler (v4.9.1)
  - DOORS/Microsoft Office Excel for SRS.
  - Enterprise Architect 12.0 for SAD
  - Enterprise Architect 12.0 for SUD
  - Source code editor (Vim)
  - Static analyzing tool (QAC 8.1)
  - Unit testing tool (TESSY 2.3)
  - Version control tool (SVN)
Other components for IC, HUD use cases – in AGL

- Instrument Cluster Middleware
- HUD Middleware
- Interface Layer
- Safety draw
- Safety sound
- Safety critical applications
- ASIL Compliant HMI Tool (Third party – Option 2)
Derived - Software Architecture with Safety Stack – in AGL
ASIL B Highlighted – Option 1
Derived - Software Architecture with Safety Stack – in AGL
ASIL B Highlighted – Option 2
Safety Software Architecture (Partitioning) – in AGL
Safety Software Architecture – Freedom From Interference

Shared Hardware resources
(CPU, Memory, Peripherals etc)

Shared Software resources
(Kernel, drivers, libraries etc)

- Limited interaction
- Static allocation
- Duplication
- Grouping
- Protection
- Monitoring
- Minimization of code etc..
Assumptions

Assume Requirements

Assumptions on design external to SEooC

SEooC Requirements

SEooC Design
SEooC – S/W Development

4. Product Development
- Specification of Technical Safety Requirements
- System Design
- Item Integration & Testing
- Assumptions on System level
- 8. Change management

6. Product Development (Software Level)
- Initiation
- SW Safety Requirements
- Software Arch Design
- Unit Design & Implementation
- Unit Testing
- Integration & Testing
- Verification of SW Safety Requirements
- Establish Validity of Assumptions

3. Concept Phase
- 8. Change management

4. Product Development (System Level)

6. Product Development (Software Level)
SEooC - Component Integration

- SEooC Component Development
- SEooC Component
- Validate Assumptions
- Impact Analysis
- New Context
- Integrated Component
- Change Management

Flow:
1. SEooC Component Development leads to SEooC Component.
2. SEooC Component leads to Validate Assumptions.
3. Validate Assumptions leads to Impact Analysis.
4. Impact Analysis leads to Fail or Success.
5. Success leads to Integrated Component.
6. Fail leads to Change Management.
7. Change Management leads to SEooC Component Development.

Decision Points:
- Success
- Fail
SEooC – The Process (V Model)

Initiation
- Assumptions
- Specification of S/W Safety Requirements
- DFMEA
- Design Traceability
- Software Unit Design & Implementation
- Software Testing
- Test Report, Coverage Reports, Review Reports

Design Architectural Design
- Software Integration and Functional testing
- Integration Report, Review Report

Verification of S/W Safety Requirements
- Verification of S/W Safety Requirements
- Integration Report, Review Report

Impact Analysis Report
- Acceptance
- Item Development, Updated Work Products, QA Report

Acceptance
- Kick off, Safety Plan, Plan Documents, Tool Evaluation and Qualification Reports
- Software Safety Requirements, Safety Req Analysis
- Design, DFA, FMEA Review Reports
- Software code, Static Analysis Reports

Acceptance
- Kick off, Safety Plan, Plan Documents, Tool Evaluation and Qualification Reports
- Software Safety Requirements, Safety Req Analysis
- Design, DFA, FMEA Review Reports
- Software code, Static Analysis Reports
NOTE: For detailed information about process, Refer ISO26262 Part 6
SEooC – Tool Classification

Identify Tool Use cases

Identify relevant failure modes

Determine Tool Impact

No Impact

TI 1

High Confidence in tool

TD 1

No qualification required

Determine Tool error detection

Impact

TI 2

Med Confidence in tool

TD 2

TI 1

No/Low Confidence in tool

TD 3

High Confidence in tool

Tool Error Detection

<table>
<thead>
<tr>
<th>Tool Impact</th>
<th>TD1</th>
<th>TD2</th>
<th>TD3</th>
</tr>
</thead>
<tbody>
<tr>
<td>TI 1</td>
<td>TCL 1</td>
<td>TCL 1</td>
<td>TCL 1</td>
</tr>
<tr>
<td>TI 2</td>
<td>TCL 1</td>
<td>TCL 2</td>
<td>TCL 3</td>
</tr>
</tbody>
</table>
# SEooC – Tool Qualification

<table>
<thead>
<tr>
<th>Method</th>
<th>TCL 1</th>
<th>TCL 2</th>
<th>TCL 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Increased confidence from use</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No Qualification Required</td>
<td>++</td>
<td>++</td>
<td></td>
</tr>
<tr>
<td>Evaluation of the development process</td>
<td>++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Validation of the software tool</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Development in compliance with a safety standard</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ASIL</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>B</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>C</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>D</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>++</td>
</tr>
</tbody>
</table>
Conclusion - Feasibility

- Development
- Testing
- Review
- Certification

- The Team
- Hardware
- New Releases

- Risk
- Dependencies
- Effort
- Timeline

Safety Life cycle Process

Technically

YES

© 2016 Tata Elxsi | Confidential
1. https://www.automotivelinux.org
3. ISO26262:2011 Standard
Thank You

Renjith G | Shilu SL
renjithg@tataelxsi.co.in | shilu@tataelxsi.co.in
+91 471 666 1138 | +91 471 666 1333