OPEN SESAME!
WHY FUNCTIONAL SAFETY IS THE MASTER KEY TO OPEN THE DOOR FOR LINUX INTO AUTOMOTIVE SYSTEMS

Authors: Nico Peper, Jan-Christian Arnold, Hans-Leo Ross
Agenda

1. Why Automotive Software is special
2. Short introduction to Automotive Safety
3. Introducing an example: Linux powering Vehicle to Infrastructure communication (V2X)
4. Implementing ASIL B on a V2X traffic light
5. Safety relevant open questions in a V2X context
6. Conclusion
WHY AUTOMOTIVE SOFTWARE IS SPECIAL & SHORT INTRODUCTION TO AUTOMOTIVE SAFETY
Open Sesame! Functional Safety for Linux in Automotive

Why Automotive Software is special

- Most automotive systems are powered purely by closed source software
- Licence obligations, like given in GPL V3, are often a blocking point

- Automotive base requirement is a 15 year product live cycle

- To achieve high volume low cost scalability process orientation is the automotive fundamental approach

- Safety -> Functional safety -> ISO26262
Short introduction to automotive safety

Potential damage caused by the operation of the vehicle or additional functions \( (S = \text{extent of damage}) \)

- Probability of hazardous situation \(-E = \text{probability.}\) (Situation \(x\) risk)
- Aversion of dangerous events
  - \(C = \text{Controllability}\)
  - Measures of other technology
  - External measures
  - Driver warning, if regulated by law

**Safety Integrity Measures according ISO 26262**

- only risk due malfunctions of EE system considered!

Area of risk and tolerable risk (Source: various different publications)
INTRODUCING AN EXAMPLE: LINUX POWERING VEHICLE TO INFRASTRUCTURE COMMUNICATION (V2X)
V2X enables the direct communication between cars (V2V) and infrastructure (V2X)

- It's an enabler for advanced driving assistance and autonomous driving functions

- US government is actively pushing V2X systems by legislation. Standard will be released soon

- Today's V2X communication stacks are POSIX based implementations

- Systems need to handle complex and non-deterministic systems and algorithms (Fuzzy, neuronal networks etc.)

→ Linux will make a clear push to autonomous driving systems by operating V2X
Linux powering V2X communication

- V2X communication stacks are implemented to be operated on Linux
- Linux will therefore make a clear push to the area of autonomous driving
Linux powering V2X communication

V2X communication stacks are implemented to be operated on Linux. Linux will therefore make a clear push to the area of autonomous driving.

Base layout for connected traffic light control system

- Embedded Linux based multicore Processor (i.e. i.mx7)
- Light Control
- Power Stage (Relais)
- V2X Interface
- V2X Modem
Open Sesame! Functional Safety for Linux in Automotive

Linux powering V2X communication

OSS vs. closed source
Hard real time behaviour
Safety conformity
Open Sesame! Functional Safety for Linux in Automotive

Linux powering V2X communication
IMPLEMENTING ASIL B ON A V2X TRAFFIC LIGHT
Open Sesame! Functional Safety for Linux in Automotive
Implementing ASIL B on a V2X traffic light
Implementing ASIL B on a V2X traffic light

- In contrast trying to reach ASIL B by handling the system based on processes a system within system functionality view within given context needs to be introduced.

- Hazard risks are safely to be reduced by determine their severity (expected loss) and frequency / probability.

- The system functionality must be reliable and in case of system errors a sufficient safety level (safe state) needs to be achieved.

- State of the art safety measures are needed to assure safe operation of vehicles!

\[
R = \sum_{\text{For all accidents}} (\text{probability of the accident occurring}) \times (\text{expected loss in case of the accident})
\]
An IT system may be represented by its basic functionalities:
- Data and system control
- Data flow
- Data processing

To achieve safety integrity of the overall system these basic functionalities need to be monitored and validated.
Implementing ASIL B on an V2X traffic light

- Example
  - Data exchange between car and traffic light via radio (i.e. dedicated short range communication)

- Safety target:
  - Ensure data flow integrity
  - Ensure safe modes & conditions in driving situations

- Monitoring mechanisms
  - Checksum & message Counters
  - Cyclic Redundancy Checks
  - Keyed-Hash Message Authentication Code
Open Sesame! Functional Safety for Linux in Automotive
Implementing ASIL B on an V2X traffic light

Data Flow

Embedded Linux based multicore Processor (i.e. i.mx7)

Light Control

- Power Stage (Relais)
- I/O connection

L1 Trafficlight Phase and Timing calculation

V2X Modem

V2X Interface

CRC calculation
CRC validation

Data flow
Example

- Process traffic light status for the V2X communication stream

Safety target:

- Ensure data integrity by avoiding calculation based errors, i.e.:
  - bit errors
  - rounding or overflow errors

Monitoring of data processing

- Redundant calculation including diverse calculation based on different data structures / types etc.
Implementing ASIL B on a V2X traffic light

- **Data Flow**
  - **CRC calculation**
  - **CRC validation**

- **Embedded Linux based multicore Processor (i.e. l.mx7)**
  - **Light Control**
    - **Power Stage (Relais)**
    - **I/O connection**
  - **L1 Traffic light Phase and Timing calculation**
  - **L2 Calculation with data type difference**

- **Data Processing**
  - **Redundant Data processing**

- **V2X Modem**
  - **V2X Interface**
Implementing ASIL B on an V2X traffic light

- **Example**
  - Operating traffic light colours and status (i.e. system control state machine)

- **Safety target**
  - System control integrity is ensured

- **Monitoring**
  - Monitoring of forbidden states
Implementing ASIL B on a V2X traffic light

Data / System Control

Embedded Linux based multicore Processor (i.e. i.mx7)

Light Control

Power Stage (Relais)

I/O connection

L1 Trafficlight Phase and Timing calculation

L2 Calculation with data type difference

CRC calculation

CRC validation

Data control

Redundant Data processing

Safety Data control

V2X Interface

V2X Modem

Data flow

Bosch Engineering | Nico Peper, Jan-Christian Arnold, Hans-Leo Ross | October 2017
© Bosch Engineering GmbH 2017. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.
Open Sesame! Functional Safety for Linux in Automotive
Implementing ASIL B on an V2X traffic light

Example
- Ensure correct real time behaviour of the traffic light

Safety target
- Traffic lights are operated conforming the real system state

Monitoring
- A safely operated entity is needed to ensure the system operation and the traffic light feedback (deadline monitoring)
Open Sesame! Functional Safety for Linux in Automotive
Implementing ASIL B on an V2X traffic light

Data

System Control (real time integrity)

Hardware feedback
Open Sesame! Functional Safety for Linux in Automotive
Implementing ASIL B on an V2X traffic light

Data input

Layer 1
Control traffic light
Provide traffic light status

Layer 2
Functional monitoring
Redundant calculation

Layer 3
System feedback
Deadline monitoring

Embedded Linux System

Safe Data output

Shut-Off to safe state

Micro controller
Safety Relevant Open Questions in a V2X Context

- Certification of complex systems (i.e. SIL2LinuxMP)
- Feedback loop including in vehicle ADAS
- Safe state for complete complex systems (i.e. traffic light & vehicles)
Conclusion

- Embedded Linux is making a clear push to automotive core systems
- Presented challenges concerning embedded Linux (i.e. licence obligations, real time behaviour and ASIL conformity) can be tackled
- By introducing the functional view to system safety Linux isn’t a blocking point for functional safety
- The three layered safety concept is proven in use in various product families and can be fully applied to embedded Linux systems
- Challenges exist! They still need to be solved
  - This should be a shared effort of the automotive industry and the Linux community
Open Sesame! Functional Safety for Linux in Automotive

Contact

Nico Peper
Product line owner embedded Linux based IOT systems
+49 (151)1680 5265
nico.peper@de.bosch.com

Jan-Christian Arnold
Expert embedded Linux based IOT systems
+49 (7062)6357
jan-christian.arnold@de.bosch.com

Hans-Leo Ross
Senior Consultant Safety
+49 (173) 314 1579
hans-leo.ross@de.bosch.com