Safety-Related Multi-Core ECUs with AUTOSAR

Niko Böhm, Software Architect Operating Systems
23.01.2014
Safety-Related Multi-Core ECUs with AUTOSAR

Agenda

• About EB Automotive
• The Importance of Functional Safety
• EB tresos Safety Products
• New Challenges: Multi-Core Architectures for Safety Applications
Agenda

• About EB Automotive
• The Importance of Functional Safety
• EB tresos Safety Products
• New Challenges: Multi-Core Architectures for Safety Applications
About EB Automotive

EB solution for the automotive world

- **Infotainment software and services**
  - EB street director – a navigation software platform for automotive and consumer markets
  - EB GUIDE – a development platform for multimodal HMIs for infotainment systems, digital instrument clusters and head-up display
  - Global software integration and engineering services

- **Driver Assistance software and services**
  - EB Assist ADTF – a software development environment for driver assistance functions
  - Electronic horizon and test drive recording solutions
  - Driver Assistance modules and algorithms

- **ECU software and services**
  - EB tresos – a family of integrated software tools used in ECUs, based on AUTOSAR standards
  - Complete solutions for: basic software configuration, debugging, cluster emulation and bus analysis products and services
  - Single-Core and Multi-Core Operating Systems, **Functional Safety Solutions**
Agenda

• About EB Automotive
• The Importance of Functional Safety
• EB tresos Safety Products
• New Challenges: Multi-Core Architectures for Safety Applications
From IEC 62304 Standard:

“There is no known method to guarantee 100% safety for any kind of software. There are three major principles which promote safety for medical device software:

1. Risk management
2. Quality management
3. Software engineering”
From IEC 62304 Standard:

“There is no known method to guarantee 100 % safety for any kind of software. There are **three major principles** which promote safety for medical device software:

1. *Risk management*
2. *Quality management*
3. *Software engineering*”
Risk Control as part of risk management

• The **goal** of risk control is to reduce the risk to an “acceptable level”.

• **Methods** of risk control *(in order of preference!):*

  1. Eliminate the risk *(by design)*
     Easiest by omitting the functionality which creates the risk.

  2. Implement protective measures
     E.g., fault detection, prevention of fault propagation *(freedom from interference)*, ...

  3. Provide adequate information
     E.g., warnings, guidance in the safety manual, ...
Risk Control as part of risk management

- **The goal** of risk control is to reduce the risk to an “acceptable level”.
- **Methods** of risk control *(in order of preference!)*:

1. **Eliminate the risk (by design)**
   Easiest by omitting the functionality which creates the risk.

2. **Implement protective measures**
   E.g., fault detection, prevention of fault propagation *(freedom from interference)*, ...

3. **Provide adequate information**
   E.g., warnings, guidance in the safety manual, ...
Freedom from Interference (ISO 26262)

**Memory**
- Unintended writing to memory of another partition
- Register/Configuration corruption due to unintended writing to processor registers

**CPU Time**
- Blocking of partitions
- Wrong allocation of processor execution time

**Communication**
- Loss of communication
- Insertions of messages
Agenda

- About EB Automotive
- The Importance of Functional Safety
- EB tresos Safety Products
- New Challenges: Multi-Core Architectures for Safety Applications
EB tresos Safety Products for AUTOSAR

**EB tresos Safety OS**
- Protected microkernel operating system

**EB tresos Safety TimE Protection**
- Alive supervision
- Deadline monitoring
- Control flow monitoring

**EB tresos Safety E2E Protection**
- Inter-ECU communication protection

**EB tresos Safety RTE**
- Safe handling of RTE-service between SW in different memory partitions
EB tresos Safety Architecture*

Safety RTE for protected communication between Memory Partitions

Safety OS:
- Data Protection
- Stack Protection
- Context Protection
- OS Protection
- Hardware Error management

Safety E2E Protection for safe communication to other ECUs

Safety OS:
- Data Protection
- Stack Protection
- Context Protection
- OS Protection
- Hardware Error management

Safety TimE Protection for alive supervision, deadline and controlflow monitoring

* example configuration; EB tresos Safety products allow a very flexible partitioning depending on the customer project needs.
EB tresos Safety TimE Protection

Alive Supervision
- Control flow monitoring
- Deadline Monitoring
The Protection Wrapper is customized for each SW-C with the smallest amount of overhead.
A Safe Operating System

Supervision Mode

- Save caller context
- Prepare protected Kernel environment
- Task activation
- Event handling
- Error handling
- ...
A Safe Operating System

**Supervision Mode**

- Task queue
- Event list
- ...

**System Call**
- Save caller context
- Prepare protected Kernel environment

**OS Service**
- Task activation
- Event handling
- Error handling
- ...

**Dispatch**
- Determine and prepare next task

**Context Switch**
- Switch Memory Partition
- Load registers including Stack Pointer and Program Counter

**EB tresos Safety OS: Certified up to ASIL D / SIL 3**
Availability

EB tresos Safety Products are available for several Microcontrollers.

![Freescale](image1.png) ![Texas Instruments](image2.png) ![ST](image3.png) ![Infineon](image4.png) ![Renesas](image5.png)
Agenda

- About EB Automotive
- The Importance of Functional Safety
- EB tresos Safety Products
- **New Challenges: Multi-Core Architectures for Safety Applications**
Multi-Core Architectures for Safety Applications

Core0

SWC

SWC

SWC

RTE

OS

BSW

MCU

ASIL SW

QM SW

OS

BSW

ASIL SW

QM SW

© Elektrobit (EB), 2014
New Challenges

Multi-Core Architectures for Safety Applications

Core0

SWC

SWC

RTE

OS

BSW

Core1

Core2

ASIL SW

QM SW

MCU

© Elektrobit (EB), 2014 21
Architecture: The CDD Approach

- Use a standard single core AUTOSAR system on one core. This acts as a master for the rest of the system.

- Use CDD(s) to connect to the other cores.
New Challenges

One AUTOSAR Core + CDD Cores

Already used in projects ("first step to multi-core")
New Challenges

Architecture: Multiple OS Instances

• Use different operating systems on different cores

• Loose coupling still possible via core-2-core CDD

• Safety OS instances need to protect themselves from other cores
**New Challenges**

**Multi-Core Examples (1)**

![Diagram](image)

- Core0
  - SWC
  - RTE
  - OS
  - BSW
  - C2C

- Core1
  - SWC
  - RTE
  - CDD
  - OS
  - C2C

- Core2
  - CDD

- MCU

**ASIL SW**

**QM SW**

- 1x Safety OS
- 1x QM OS
- 1x CDD Core
Multi-Core Examples (2)

- Core0
  - SWC
  - RTE
  - OS
  - BSW
  - C2C

- Core1
  - SWC
  - RTE
  - CDD
  - OS
  - C2C

- Core2
  - SWC
  - RTE
  - CDD
  - OS

MCU

- ASIL SW
- QM SW

- 1x Safety OS
- 2x QM OS
New Challenges

Multi-Core Examples (3)

- 1x Safety OS
- 1x Multi-core QM OS
The AUTOSAR 4.0 approach:

• Group SWCs to partitions and spread them across the available cores
• BSW on one core
• One configuration for all cores
Multi-Core Safety OS
Multi-Core Challenges

• **Lessons learned from multi-core OS (QM):**
  – Mapping of SWCs is the critical topic for success / failure
    • Communication time between cores can be quite high
  – General benefit to timing is often overestimated

• **Multi-core and safety require advanced concepts for:**
  – Cross-core API (e.g., task activation on other cores)
  – Error handling (on which core shall ErrorHook() be invoked?)
Summary

- **Functional Safety Software** is becoming more and more important for the automotive industry
- EB offers a complete **Safety Product Portfolio (OS, TimE, E2E, RTE)**
- **Multi-Core** architectures are available – but with limitations
- Under development: **EB tresos Safety OS** with seamless multi-core integration
Contact us!

Sales.automotive@elektrobit.com
automotive.elektrobit.com