COMMUNICATION CENTRIC DESIGN FOR COMPOSABILITY & DATA CONSISTENCY IN AUTOMOTIVE EMBEDDED SYSTEMS

SIMON KRAMER
CORPORATE RESEARCH, ROBERT BOSCH GMBH
Communication Centric Design

Agenda

Legacy: Automotive Embedded Systems
Future execution platforms
Challenges combining both worlds
Conclusions
Communication Centric Design

Agenda

Legacy: Automotive Embedded Systems
Future execution platforms
Challenges combining both worlds
Conclusions
In a modern car there are about 100 different ECUs interconnected over several busses and networks.
Communication Centric Design
Legacy: Automotive Embedded Systems

Example – Engine Control

- Infineon AURIX
  - Up to 3 Cores
- Periodic & Angle-Synchronous Tasks
- Scheduled with Fixed Priority Preemptive Scheduling
- on an OSEK compliant Operating System
Communication Centric Design
Legacy: Automotive Embedded Systems

Software View

AUTOSAR – Layered Architecture

Runnable Communication Graph
To avoid data inconsistencies during execution, efficient mechanisms have to be introduced.
Communication Centric Design

Agenda

Intelligent mobility systems of the future
Future execution platforms
Challenges combining both worlds
Conclusions
Communication Centric Design
Future E/E-Architectures

- **FUTURE**
  - Vehicle centralized E/E architecture
  - Vehicle Cloud Computing
  - Vehicle Computer & Zone ECUs

- **TOMORROW**
  - (Cross) Domain centralized E/E architecture
  - Domain Fusion
  - Central Cross Domain ECUs

- **TODAY**
  - Distributed E/E architecture
  - Integration
  - Functional Integration
  - Each function has his ECU

Increasing No of SW
Future execution platforms

**Hardware availability:**

- Increasing autonomy requires fail operational or at least fail degraded systems
- It can no longer be assumed that the driver can maintain controllability of the vehicle in the case of loss of function
  - E.g. In case of failure, braking system needs to be able to warn driver/occupant and ensure a safe stop
- Detecting errors and resetting/turning off system is no longer a plausible safety concept

👉 Demand for demonstrating the absolute failure rates of hardware will increase
Communication Centric Design

Future execution platforms

Software Reliability:

- Systematic faults covered by development process
- For consolidation of functions onto one common ECU, Freedom from Interference has to be shown
  - Spatial Isolation
  - Temporal Isolation
- Hypervisors are used for isolation
  - Spatial Isolation ✔️
  - Temporal Isolation ✗

⇒ Current Hypervisors are not sufficient to guarantee temporal isolation efficiently
Communication Centric Design

Agenda

Legacy: Automotive Embedded Systems
Future execution platforms
Challenges combining both worlds
Conclusions
Communication Centric Design
Consolidation of ECUs – Single Application

Task_1ms

Task_5ms

Task_10ms

5ms 6ms
Communication Centric Design

Consolidation of ECUs – Two Applications

2nd Application adds to the 10ms Task
Communication Centric Design
Timed Communication – Theory

- Tasks behave according to a Logical Execution Time (LET)
  - Inputs are (logically) read at beginning of time interval
  - Outputs are (logically) written at end of time interval
  - Independent of when task actually runs within interval
Communication Centric Design

Portability, Integratability, Interoperability

Replaceability / Portability
Same behavior at different CPU speed

Integratability / Extensibility
Integration without side effects

Interoperability
Verifiability by deterministic communication

© Tom Henzinger et al., IST Austria
Communication Centric Design

Timed Communication - Implementation

High Priority
CopyIn - Interrupt

Global Memory

Task A
local memory

Task B
local memory

Global Memory

High Priority
CopyOut - Interrupt
The following experiments were conducted with SymTA/S. Details on how the transformation of existing applications to logical execution time for analysis can be done, will be presented at ECRTS ’17.
Communication Centric Design

Transformation
Communication Centric Design

Logical Execution Time – Single Application

With LET-Semantics, the latency of the given communication path increases
Communication Centric Design

Logical Execution Time – Two Applications

10ms

11ms

15.925ms

15.990ms
Communication Centric Design
Logical Execution Time - Composability

Latency is (almost) constant during addition of new applications or functions. This also holds when moving a task to another core or when migrating to new hardware.
Communication Centric Design

Agenda

- Legacy: Automotive Embedded Systems
- Future execution platforms
- Challenges combining both worlds
- Conclusions
Future E/E-Architectures, will consist of centralized control units handling cross-domain functions. For safe integration, enhanced mechanisms are needed that ensure freedom from interference.

For spatial separation Hypervisors are used to guarantee independence, but are inefficient in the time domain.

The concept of Logical Execution Time ensures composability in the time-domain thus also giving independence.