The AUTOSAR Timing Model
---
Status and Challenges

Dr. Kai Richter
Symtavision GmbH, Germany
Symtavision GmbH – Who we are!

- Spin-off from Technical University of Braunschweig, Germany, founded May 2005
- Timing and scheduling analysis tool suite SymTA/S
- 30+ MY research and development of technology
- Expertise in system integration
- Primary market: Automotive
Symtavision Expertise: Real-Time Systems Analysis

- Real-time correctness ➔ Reliability / Dependability

- Optimization ➔ Cost
  - component selection
  - dimensioning
  - scheduling

- Flexibility ➔ Quality
Solution: Flexible, Modular SymTA/S Tool Suite
Overview

- AUTOSAR in general & target use cases
- Top-down: SW architecture vs. execution platform
- A closer look to key technical details
- Bottom-up: Integration & timing analysis practice
- Implications w.r.t AUTOSAR goals
- Conclusion
Overview

- AUTOSAR in general & target use cases
- Top-down: SW architecture vs. execution platform
- A closer look to key technical details
- Bottom-up: Integration & timing analysis practice
- Implications w.r.t AUTOSAR goals
- Conclusion
Key AUTOSAR Concepts

- portable software components
- virtual function bus (VFB)
- ports and connectors
- several communication semantics (send/recv, client/server)
- crossing module boundaries (function distribution)
- crossing company boundaries (supply chain, black box)
- configurable/customizable run-time environment

→ Needs standardized APIs to facilitate implementation!
Key AUTOSAR "Methodology and RTE"

- Flexible mapping of software components ...
- ... enabled by standardized run-time environment (RTE)
Mapping in More Detail: SW Component Structure and Execution Platform

Vehicle Function

- Sensor
- SWC
- SWC1
- Actuator SWC
- Actuator

- Sensor SWC
- SWC-1
- Actuator SWC
- Actuator

- RTE
- BSW

Signal Path / Data Flow

- Standardized RTE eases compiling & linking together several SW components from different teams/vendors/...
Typical AUTOSAR Use Cases

- Function distribution & partitioning
  - one function - several SW components one several ECUs
  - one ECU - several SW-Cs from different functions / vendors

- Adding new functions
  - product variants, face lifts, platforms

- Optimizations
  - Configuration (CAN IDs, signal-to-frame assignment, etc.)
  - Re-mapping of SW components
  - Network modifications (topology, protocols, gatewaying)

- New business models
  - Software as a product
  - Improved supply-chain "contracting" (liabilities)
Overview

- AUTOSAR in general & target use cases
- **Top-down: SW architecture vs. execution platform**
- A closer look to key technical details
- Bottom-up: Integration & timing analysis practice
- Implications w.r.t AUTOSAR goals
- Conclusion
Introduction of Timing Effects: Framework

- Function development imposes timing constraints
- High-level specification based on SW components
- AUTOSAR goal: break down the software structure into "manageable" blocks
  - timing chains and timing chain segments
  - connected at hand-over points (HOPs)
  - consider each segment / HOP individually
- Goals:
  - divide and conquer "timing analysis" top-down
  - assignment of responsibilities
  - locally verifiable, then result composition bottom-up
Timing Chains and Hand-Over Points (HOPs)

INTER-ECU communication

INTRA-ECU communication

end-to-end timing chain

timing chain segments

© Dr. Kai Richter, Symtavision GmbH
The AUTOSAR Timing Model - Status and Challenges
ARTIST2 Workshop, 23.3.06 Innsbruck
Introduction of Local Timing Effects

Reasoning about timing requires considering two views:
- static **software components**
- vs. dynamic **execution platform** behavior
  - operating systems and scheduling;
    SW components vs. runnables and tasks
  - communication semantics;
    SW-C structure vs. timing dependencies
  - middleware / driver structure;
    standardized protocols vs.
    non-standardized implementation & BSW
Overview

- AUTOSAR in general & target use cases
- Top-down: SW architecture vs. execution platform
- A closer look to key technical details
- Bottom-up: Integration & timing analysis practice
- Implications w.r.t AUTOSAR goals
- Conclusion
SW-Components vs. "Runnables" and Tasks

- SW architecture:
  2 SW components,
  6 runnables

- Implementation: 3 Tasks

- Schedule and timing dependencies
Challenge: Associating Schedules with Timing Chain Segments

- software component w/ 3 runnables
- sequential model
- actual implementation
- meaning?

what about runnable B
start of runnable A
end of runnable C
Software Component Structure vs. Timing Dependencies

- Software component view captures "logical" dependencies (data flow)

- Implementation timing dependencies can be very different!!!
  - time-driven and event-driven activation
  - send/recv and client/server communication (remote procedure call)
  - over- / undersampling
Software Component Structure

vs. Timing Dependencies

- Software component view captures "logical" dependencies (data flow)

- Timing dependencies can be very different!!!
  - time-driven and event-driven activation
  - send/recv and client/server communication (remote procedure call)
  - over- / undersampling
Sender-Receiver vs. Client-Server

- **INTRA-ECU communication:** both SW-Cs on one ECU
  - merely an issue of software structure
  - global register vs. local variable (with get Method)

- **INTER-ECU communication:** SW-Cs on different ECUs
  - has large influence on bus / ECU timing
Sender-Receiver vs. Client-Server II

- sender-receiver w/ cyclic tasks and frames
  - bus message timing
  - ECU 2 timing

- client-server solution w/ asynchronous servers and frames
Protocols vs. Non-Standardized BSW

Frame generation timing (cyclic and/or event-driven)

Buffering strategy (FIFO, priority ordered, hybrid)

Use of message objects (hardware buffers)
Priority Queue vs. FIFO in CAN Networks

- buffering strategy (inside ECU) has huge influence on network timing

**Shared priority-ordered buffer**

- high-priority frames must wait for low-priority frames
- low-priority frames benefit from FIFO
- blocking due to non-preemptiveness

**Shared FIFO Buffer**

- undermines the CAN protocol's priority scheme

- 3 messages share a FIFO
- low-priority frames benefit from FIFO
- high-priority frames must wait for low-priority frames
Challenge:
Associating Schedules with Timing Chain Segments

complex mutual dependencies

complex mutual dependencies
Summary: Local Timing Effects

Complex timing

- is not directly reflected in the **software architecture**
- is induced by the **execution platform**!
  - runnables and tasks
  - timing dependencies and communication semantics
  - non-standardized drivers and middleware (BSW)

- etc...
Overview

- AUTOSAR in general & target use cases
- Top-down: SW architecture vs. execution platform
- A closer look to key technical details
- Bottom-up: Integration & timing analysis practice
- Implications w.r.t AUTOSAR goals
- Conclusion
Bottom-up: Timing effects during integration

Key Message: Local Changes can have Global Effects !!!
**Example:**
Task Timing Changed, e.g. Function Added

constraints still valid?

new
Example: New Frame on Network

- Direct dependencies
- Indirect dependencies
- Buffer overflow
- Schedule distortion

Screenshots by SYMTA VISION
Example:
COM Layer Queuing Changed (FIFO -> priority)
Use Case: System Integration (white box)

- two individual subsystems
- integrated using shared bus

Question: How can this be analyzed & controlled?
Use Case: System Integration (black box)

Even worse:
- Only partial information available
- How to analyze this at all?
Timing Analysis in Practice Today

- Local analysis of individual components
  - Good systematic approaches available
  - But mostly simplified "environment models"
  - ➞ Later integration problems

- Testing of (sub-) systems after integration
  - Whole environment available
  - But: unknown critical interactions
    - Prohibits corner case coverage
  - ➞ Decreasing reliability
Established V-Model Design Process

- Design
  - Analysis
  - System Design
  - Module Design
  - Implementation

- Test
  - Requirements Test
  - System Test
  - Module Test / Analysis
  - Function Test

coverage?

composition?
Summary: Bottom-Up System Integration

- Many local decisions have global effect, and are mutually dependent

- Technical Issue: System-level modeling of complex timing interaction

- Business Issue: Contracting & data availability along complex supply chains

- Current practice needs improvements

- Key challenge for the AUTOSAR Timing Model!
Overview

- AUTOSAR in general & target use cases
- Top-down: SW architecture vs. execution platform
- A closer look to key technical details
- Bottom-up: Integration & timing analysis practice
- Implications w.r.t AUTOSAR goals
- Conclusion
Review: AUTOSAR Goals

AUTOSAR shall be a vehicle for:
- Integration of SW-Cs from different SW suppliers
- Integration of ECUs from different tier-1 suppliers
- Platform design
  - re-use, extensibility, platform variants
  - portability and configurability at all levels

Approach:
- Standardized software architecture
- Modular and flexible function integration
Challenge: Timing Dependencies

- SW architecture does not reflect timing dependencies

- Timing is
  - mapping dependent (execution platform)
  - not as compositional/modular as the software architecture
  - complex
  - a fundamental technical issue

- Timing currently not thoroughly addressed by AUTOSAR
- counters platform independent software & portability
What is needed?

- Controlling timing dependencies requires reasonable specification models that are supported by analysis (tools)

"There is no point in modeling something that cannot be analyzed !!!" (during some timing team meeting)

- Appropriate timing model → technology
- Appropriate design "culture" → business processes
How Could a Successful Timing Model Look Like?

- captures the complex dynamic timing dependencies, and the environment
- considers the used mechanisms (OS, protocols, BSW,...)
- enables de- / composition & local timing analysis
- allows black-box integration and IP protection
- applicable at different levels of detail

![Diagram of AUTOSAR Timing Model]

© Dr. Kai Richter, Symtavision GmbH
The AUTOSAR Timing Model - Status and Challenges
ARTIST2 Workshop, 23.3.06 Innsbruck
Software Suppliers can do: Timing Characterization

+ information about communication
  (volume & access type)
+ information about activation
  events, interrupts, timers...
ECU Suppliers can do: Timing Analysis on ECUs

OS, scheduling, drivers, BSW, interrupts, timers

black boxes

interface captures dynamic I/O behavior (jitter, min/max delays)
OEM can do: Control Timing on Bus/Network

network config, protocols, gateways, etc.

black box ECU

constraint not met

interface captures
dynamic component interactions
Design Process Tomorrow?

Design

- Performance Estimation
- Architecture Optimization
- Flexibility Analysis

Analysis

- Requirements Test
- System Test
- Module Test

OEM

System Design

Supplier

Implementation

Refinement

Integration

Verification

Function Test

Bottleneck Detection

Optimization

Analysis

- Requirement
- Performance
- Architecture
- Flexibility

Next steps?
Cultural Issues

- Many approaches to timing modeling exist
- None has been chosen yet for AUTOSAR
- Why ???

Timing challenges require re-thinking of roles !!!
Suppliers Role

- Traditional role of Suppliers
  - function implementation
  - execution platform development
  - ..

- New to suppliers
  - responsible for ECU-network interactions
  - very detailed requirements / constraints
  - traceable verification, clear responsibility / liability
  - disclosure of information relevant for timing
  - more competition due to comparability
OEMs Role

- Traditional role of OEMs in E/E design
  - function design (Matlab, etc.)
  - prototyping
  - taking suppliers liable for correct functioning

- New to OEMs
  - network timing effects out of supplier responsibility
  - timing is a technical problem requiring a technical solution (no management solution)
  - consideration of SW architecture and execution platforms
  - dealing with systematic timing and QoS contracts

- OEM needs to reason about integration much earlier
  - Quality can not be added at the end of "cooking" (like salt)!
Supplier-OEM Communication Scenario

Function Design & Simulation @ OEM

Matlab/Simulink Simulation

Prototype & Test @ OEM

Critical Supply Chain Communication

ECU design @ Supplier

Integration & Test @ OEM

SYMTAVISION
New OEM Responsibilities and Possibilities

- Facing timing as a technical challenge, OEMs can
  - understanding network timing ➔ more systematic dimensioning, configuration, optimization
  - focusing on the interaction of ECUs with the network ➔ more systematic timing constraints for suppliers (timing chains and HOPs) ➔ increasing integration reliability / reduced risk
  - better understanding of COM-layer effects ➔ systematic implementation constraints for suppliers (OEMs defines a "standard BSW core") ➔ guaranteed compliance of supplied ECUs with OEMs network
Research Bodies Role

- Traditionally
  - develop solution approaches for technical problems
  - are used to industry requesting their help
  - develop foundations for EDA tools

- AUTOSAR:
  - an entire community with an obvious problem ...
  - ... long time not asking for direct assistance

- Why is that?
Industry-Research Mismatch ???

Automotive Industry

complex systems,
manifold dependencies

revolutionary problems
Industry-Research Mismatch ???

Automotive Industry

- complex systems, manifold dependencies
- revolutionary problems

Research Community

- clear semantics, well-defined interactions
- revolutionary solutions

© Dr. Kai Richter, Syntavision GmbH
Conclusion

- Timing is "quite new" to automotive industry (esp. OEMs)
- SW architecture view not sufficient to capture timing
- Must take into account the execution platform systematically, is complex
- Needs formal models -> EDA Tools -> confident users
- Allows engineers to reason about alternatives
- Need to come:
  - SW engineering view enhancements
  - better (more systematic) platform mechanisms / basic software
  - more flexible design rules
  - revised “way of thinking” (especially for OEMs)