Mentor, A Siemens Business
Automotive Business Unit (ABU)
Cockpit Domain Consolidation
GENIVI 20th All Member Meeting
The automotive system architecture is changing: Fusion of Multi-Domains on the same ECUs

Example for a combination of IVI and Cluster on a modern ECU system
The Automotive Architecture is changing!

Key architectural directions:

- A centralized computer system
- A set of vehicle zonal IO concentrators
- A highly scalable and modular EE hardware structure
- Software centric approach with a Service Oriented Architecture
- Real-time, deterministic capable Ethernet AVB/TSN network
- Tooling supporting software platform configuration and Agile Continuous Integration & Test Software development flow
Mentor CDC - Cockpit Domain Consolidation
Next step towards Central Compute Platform

FEATURES AND BENEFITS:
A platform ready for your project specific porting
improving time & cost to market

Technical and functional safety concept

Software safety libraries (ASIL-B) & Safety Island for Safety, Entertainment and Automotive Audio features

Implement architectures for Shared Graphics, Video, Audio

IVI stack, Bluetooth*, Wi-Fi, Tuner*, CAN connectivity,
Rear View Camera, Multi-Media playback*

Mentor AUTOSAR on Safety Island, MCAL integration

QM to ASIL domain communication including certification

QM Cluster (Linux) and IVI (Linux / Android)

CDCBase - A basis feature set ready to combine
IVI & Cluster on high compute ECU systems.

* Depending on customer requirements, OSS or 3rd party solutions will be integrated by Mentor. Additional costs are subject to project specifications.
What are the challenges? How is this interesting for GENIVI?

Integrate several operating system on one ECU e.g. Linux, Android, adaptive AUTOSAR, RTOS

- Some components are needed for each/several operating systems and benefit from standardized interfaces and solutions
- Communication in between hypervised OSs not standardized (used to be vehicle bus between ECUs)
- Communication from OS to hypervisor not standardized
- Interfaces to drivers specific for each SoC
- Interaction of components of different safety standards
- OS decisions are typically OEM specific, there are lots of possible combinations
- Integration of GENIVI standardized components in Android followed by certification by Google makes usage of DRM and Play Store easier
- Standardize interfaces to security components: Trusted Execution Environment OS, Hardware Security Module

GENIVI is already working on some of these topics and willing to expand the scope based on demand!
Challenges: Standard interface to Hypervisor

Challenge
- Hypervisor interfaces vary between SoCs
- Drivers vary and are additionally dependent on operating systems
- Lots of possible combinations, big test effort

What needs to be solved?
- Standard interfaces to hypervisor
- Standard drivers
- Can be tested a lot easier on more different systems e.g. by reusing tests
- Test effort smaller for bigger test depth

GENIVI Focus
- Hypervisor Project – Virtual Device API standardization
Challenges: Internal Communication

Challenge
- CommonAPI / Some/IP only on Linux systems
- For Android no specific standard, using native Android mechanisms
- RTOS using propriety solutions

What needs to be solved?
- Integration on other operating systems to be able to directly use services offered in other components
- Transport between guest systems depends on used SoC
- Add end to end protection features like checksum & counters

GENIVI Focus
- Interface definition language based on CommonAPI (Franca IDL) available for
  - D-Bus
  - SOME/IP
  - Currently only for Linux
Challenges: Standardize basic components (1/2)

Audio

Challenge
- Mixing of sources from several domains (QM and ASIL) on one controller and to one speaker

GENIVI Focus
- AudioManager available for Linux

Graphics

Challenge
- Handling and use cases of buffer sharing (which is SoC specific)
- DRM handling on Android (Trusted Execution Environment which is SoC specific)

GENIVI Focus
- RAMSES already available for distributed rendering on several operating systems
Challenges: Standardize basic components (2/2)

Lifecycle

Challenge

- Lifecycle Master that controls guest system lifecycles
- Standardize interfaces from guest systems to master
- Standardize interfaces to hypervisor for shutdown, reset, suspend

GENIVI Focus

- NSM (Node State Manager) available for Linux

Logging

Challenge

- Same base logging mechanism on all guest systems
- Synchronization and consolidation of all logs from all systems

GENIVI Focus

- DLT (Diagnostic Log and Trace) available for Linux
What are the next steps – where do we need to focus?

The automotive system architecture is changing: Fusion of Multi-Domains on the same ECUs
Example for a combination of IVI and Cluster (CDC) on a modern ECU system

- What are the key trends in the fusion of the automotive system architecture?
- How does the CDC fusion affect other domains in the car?
- How can we ensure safety relevant aspects with the fusion?
- Which standards do we need for an ECU / SoC for the fusion?
- How can we ensure that integration does not compromise cyber security of individual domains?
- What are Verification and Validation challenges?
- Which automotive interfaces do we need for future ECU / SoCs?
- Challenges to realize a high performance inter domain and inter SoC communication technologies?
- What domain consolidations could / will be next?
Q & A
Thank you for your attention!