Hypervisor in Automotive - Critical Use cases

Background

Hypervisor is a well known hardware virtualization technique that allows multiple virtual machines called guests identified by their OS to run on a host system. The virtual guests have access (exclusive or shared) to underlying hardware resources including CPU, memory and peripherals. In the context of the virtual guest, it appears as if all of the allocated hardware is natively available exclusively to that machine. Hypervisor in most cases also provides a means of controlling the execution state of the virtual guests. Hypervisors are increasingly being used in modern automotive software systems for combining safety critical and safety agnostic platforms as well as for consolidating multiple ECU functions into one.

Purpose

The purpose of this document is to capture the critical use cases that needs to be considered for using an Hypervisor in the context of automotive in vehicle infotainment. As Instrument clusters functions fall in adjacent yet very close domain, some of the cross functional aspects between clusters and infotainment needs to be considered as well. The objective of defining these critical use cases is to create a sounding board for defining candidate system requirements and model architecture covering various types of hypervisor. Although Hypervisor is a low level technical software, the design decisions made can have far reaching impact on the overall user experience especially on the system performance.

Assumption

The following are the assumptions that the reader needs to make while going through the use cases. Eventually, it may be possible to challenge the assumption itself and arrive at contradictory architectures that may be even better and well suited for all practical purposes.

ID	Assumption
A1	The overall system involves at least one instance of Hypervisor software
A2	The overall system involves at least one instance of vehicle domain running AUTOSAR compliant software

Use cases

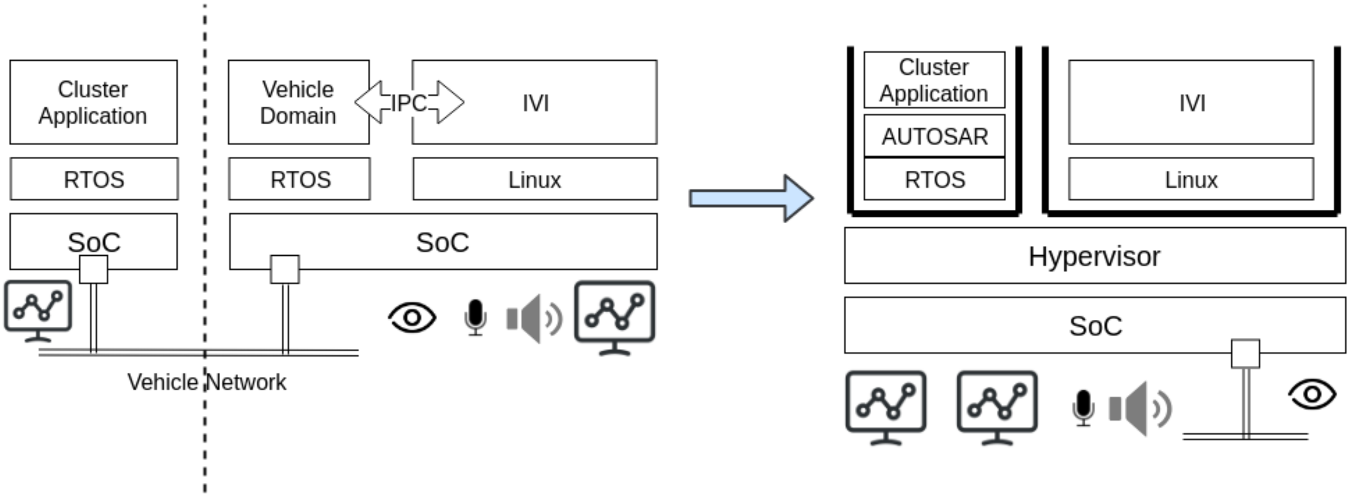
ID	Title	Description / Flow of events	Pertinent Challenges / Key points	Remarks/Comments
HVS-UC-01	Rear view Camera at Startup	1. User inserts the key in the key fob 2. (Possibly)System initialization begins in the background 3. User engages the reverse gear within 1 second 4. User turns the key to crank the vehicle within 2 seconds of key insertion 5. Full system initialization begins and system beings to show the boot up animation / splash screen / boot up audio 6. System displays the Rear View Camera output on screen along with park distance guide	1. Which domain/virtual guest controls the first camera stream? 2. Which domain/virtual guest controls the first display output? 3. If there is a need for handover between virtual guests, what is the underlying protocol for handover? 4. What happens if the user immediately turns off the engine?	1. Consider the scenario when the RVC output and park distance guide are being displayed on Instrument Cluster as well. 2. Extending #1, what if the Instrument Cluster logic is actually from a virtual guest? Would the safety critical aspects of cluster apply? What are the limitations with respect to timing?
HVS-UC-02	Entry into vehicle with an active call over Bluetooth	1. User inserts the key in the fob 2. (Possibly)System initialization begins in the background 3. User turns the key to crank the vehicle 4. Full system initialization begins and system beings to show the boot up animation / splash screen / boot up audio 5. Immediately a Bluetooth call is detected and audio path is routed to cabin speakers and microphone is used for audio input	1. Which domain/virtual guest controls the first audio such as boot up audio? 2. How is the handover of audio channel performed between the first audio source and call audio? 3. Supposing the LUC was USB and USB media is connected at startup; If the USB media playback was active in a domain /virtual guest other than the one with active call	1. Considering an internal DSP with control over UART and audio over I2S, it takes about 150ms to establish the audio path 2. If there were an external DSP with audio over MOST / EAVB, it takes about x ms to establish the audio path
HVS-UC-03	Entry into vehicle with an active call over Bluetooth + External amplifier over Ethernet AVB	1. User inserts the key in the fob 2. (Possibly)System initialization begins in the background 3. User turns the key to crank the vehicle 4. Full system initialization begins and system beings to show the boot up animation / splash screen / boot up audio 5. Immediately a Bluetooth call is detected and audio path is routed to cabin speakers and microphone is used for audio input	1. At startup which domain/virtual guest owns the EAVB control as well as data transfer? 2. Audio path should be established within 150 ms of the Bluetooth call getting detected	

HVS-UC-04	Incoming Telematics call over Ethernet AVB at startup with the vehicle in Reverse Gear	1. User enters the vehicle and cranks the vehicle 2. While the system initialization is in progress, an incoming Telematics call is detected by the TCU 3. TCU forwards the call notification to Infotainment ECU over EAVB 4. User engages the reverse gear 5. Infotainment system prompts the user to accept the call 6. User accepts the call 7. Call audio path is routed to cabin speakers and the microphone is used for audio input	1. What is the potential delay in receiving the notification? Note that this is influenced by which domain owns the EAVB at startup 2. How is the contention of resources managed? For e.g. if the I/O is performed using DMA, what considerations have to be made in the context of hypervisor	1. Telematics ECU (TCU) is interfaced with the Infotainment ECU over EAVB 2. Rear view camera is interfaced over digital serial interface / EAVB
HVS-UC-05	Ongoing Bluetooth call in one virtual guest and Skype call in another virtual guest	1. User is on an active phone call over Bluetooth in the context of a domain /virtual guest 2. User receives an incoming Skype call in the context of a second domain/virtual guest 3. System displays a prompt to *end and accept* / *hold and accept* the skype call 4. User ends the active call and accepts the skype call	1. How is audio arbitration handled in this case? 2. How is the call audio data handled? 3. Latency of switching audio path should be < 200 ms	1. The first domain in this case could possibly be Linux and the second domain could be Android for Apps with Skype being an app running on Android
HVS-UC-06	Restriction of access to underlying CAN Interface	1. Infotainment system is up and running and comprises of a Vehicle Domain and Application domain running as virtual guests on an Hypervisor 2. User installs an application that attempts to write to the CAN Interface Register (as the SoC and hence the CAN Register's memory mapped address is known) 3. Application is prevented from writing to the CAN Registers		

Hypervisor System Architecture Candidates

This section depicts the various architectures motivating the need for Hypervisor in the context of automotive infotainment

ECU Consolidation



Cluster Application
RTOS
SoC
Vehicle Domain
IPC
IVI
RTOS
Linux
SoC
Vehicle Network
Cluster Application
AUTOSAR
RTOS
IVI
Linux
Hypervisor
SoC