GENIVI®

RENESAS
BIG IDEAS FOR EVERY SPACE

TECHNOLOGY BRIEF

FEBRUARY 2019
Category: Vehicle Domain Interaction

Display Sharing

Summary

Five primary techniques for distributing graphics and HMI between cooperating systems have been identified in the GENIVI Graphics Sharing and Distributed HMI project (GSHA) [1]:

- Surface Sharing
- API Remoting
- GPU Sharing
- Display Sharing
- Shared State, independent rendering.

This brief discusses **Display Sharing** which provides support in hardware for compositing the display output from multiple operating systems into a single final display buffer. Using this, a combined HMI can be created from independent systems without any additional interaction.

Key Characteristics

- The physical display can be shared across multiple operating systems or major functional units of a single operating system.
- Hardware Compositor block provides multiple layers that can be assigned to different functional units. The hardware then combines (composites) them to create the final display buffer.
- The Compositor hardware block is typically close to or part of the Display Unit hardware block.
- The different systems do not need to know about each other.
- No need to have protocols to pass graphics between operating systems.
- Hardware may provide additional features that enhance isolation of the different functional units.

Description

Display Sharing is used to allow multiple operating systems to share a physical display while retaining control over how their output is combined. Typically, this control over the separation is an important requirement in the system design. For example, the surface area of a physical display may be mostly dedicated to display cluster graphics, but areas of the display may also show IVI information such as multimedia or navigation, or the output from a rear-view camera. In this case the IVI graphics should not be allowed to overwrite the cluster graphics.

This separation is enabled by a hardware block that provides multiple hardware display layers for each display. Hardware features vary between SoCs but core support for composition of the layers and (per plane and per pixel) alpha blending for transparency, with each layer having associated image data buffers in memory is typical. Each operating system can be assigned a dedicated layer to draw into. The hardware block combines (composites) all layers into a final display buffer which is then passed on in the Display Unit for output on the physical display.

By providing control over the Z-ordering (which layer is "in front" or "behind" in the drawing order) the hardware compositor can ensure that one layer does not overwrite the other. For example, placing the layer assigned to the Linux operating system providing IVI behind the layer assigned to the safety-critical OS providing the Cluster in the Z-ordering. Per pixel alpha blending allows a visually complex composition of the different layers both in terms of transparency between the layers and the shape of the bounding area of each layer. A layer does not need to constrain its rendering into a simple box for example. Per plane alpha blending applies the same transparency value to every pixel in the layer.

Since the hardware block combines the different display layers each operating system does not need to know about the other and there is no need to have protocols to pass graphics between operating systems. Although of course architectural decisions will be made as to how to use and control the layers in the system design.

Extended features of the hardware display layer block may include data format conversion and image processing such as colour space conversion, colour keying and video processing. For example, video processing may include the automatic processing of video data from the video input block through to the display unit block, thus freeing software from controlling the buffers and combining the output. The block may also support functional safety features that check the display output. These various functions can simplify the combination and control of display output from various car systems in software.

Sharing between multiple operating systems has been discussed, but an alternative case could involve a single operating system providing all functionality and assigning the hardware layers appropriately.

Where hardware display layers are not available, a form of Display Sharing may be provided in software by the Display Manager of certain Hypervisors. The Display Manager provides a mechanism for guest VMs to pass display buffers which it then combines.

Case Study: Canvas Demo

The Renesas R-Car Canvas Demo [2] demonstrates consolidation of cockpit functionality onto a single R-Car H3 SoC running on a Salvator-X board driving three displays and using multiple operating systems. In the version discussed here the Integrity Multivisor hypervisor is used to virtualise an Integrity RTOS instance running a cluster demo and a Linux operating system instance running an IVI stack and other functionality. The first display is shared between Integrity and Linux, whilst the second and third display is dedicated to Linux. For display one, Integrity and Linux are assigned different hardware display layers, with Linux being placed behind the Integrity cluster in the Z-ordering to ensure the Linux applications cannot overwrite the cluster graphics. In R-Car Gen 3 SoCs, the hardware display layer functionality is provided by the VSPD hardware block.

Figure 1 - Display sharing between OSs

The demo implements touch gestures in the Linux instance (see Figure 2). These can be used to size and swipe applications together and between the screens. When applications are pushed to display one they are automatically docked into a dedicated area in the cluster. As applications move across display one a small amount of transparency between the two layers allows applications placed behind the cluster to show through.

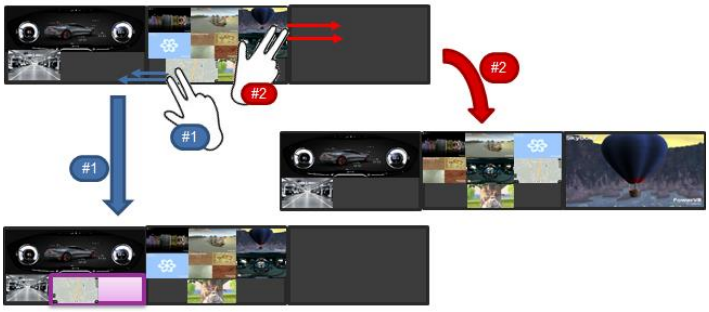

Figure 2 - Using touch to move apps between displays

Importantly the cluster does not need to understand what the Linux applications are doing and the hardware support for OS separation ensures cluster performance is maintained. Thus, simplifying the software design.

In production, Display Sharing can be usefully combined with virtualisation and functional safety technology in both software and hardware to enhance separation of display output with different safety levels. For example, the Renesas R-Car H3 SoC has multiple hardware virtualisation features to prevent one operating system stalling the processing of another operating system such as OS separation in the GPU and IPMMU (address translation and access protection for processing units and interconnects), along with hardware functional safety and safe rendering features. This protection can be further enhanced with a Hypervisor in a production system.

Authors

- Stephen Lawrence (Renesas), stephen.lawrence@renesas.com
- Participants of the Graphics Sharing working group

References & Additional Reading

1. GENIVI Domain Interaction Project on Graphics Sharing and Distributed HMI (https://at.projects.genivi.org/wiki/x/p4T0)
2. Canvas Demo case study presentation at Genivi Technical Summit Bangalore
3. Graphics Sharing Whitepaper (in development)