ACRN

Consolidate Real-Time and HMI with ACRN Hypervisor

Jack Ren, Intel ACRN Team
Table of Contents

PART 1: What is ACRN  ................................................................. page 3

PART 2: Industrial Use Case ....................................................... page 6

PART 3: Architecture ............................................................... page 7

PART 4: Roadmap ................................................................. page 14

PART 5: Call for Participation ................................................ page 15
What is ACRN™

A flexible, open-source, lightweight hypervisor for IOT workload consolidation

A Linux Foundation Project Launched in March 2018

https://projectacrn.org
Value Proposition

Core Intrinsic Values

Small Footprint
• Optimized for IOT class solutions
• Significantly smaller footprint than datacenter targeted hypervisors

Heterogeneous Workloads Consolidation
• Real time & Non-Real time
• Functionally Safe & non-safe

Open-source with Flexible Licensing
• BSD license enables proprietary Guest OS
• True Open source with a vibrant Community

Other Key Values

System Security
• Intel VT backed virtualization provides secure operating domains

Secure Containers
• Intel VT backed KATA containers as virtual machines enables added security
• Kubernetes support for KATA enables ease of deployment & management

Flexible Isolation Frameworks
• Traditional virtualization w/ Shared framework
• Safety implementation w/ Partitioned framework

Beyond-Compute sharing
• IO, Graphics, Media sharing capabilities
ACRN 1.0

Main Usage: Software Defined Cockpit

Ready for Production
• 100% Feature Test Coverage
• High Stability
• Fast Boot and Performance KPI
• 100% CTS Pass for Android Guest

Key Features
• Safety and Security Isolation (Cluster + IVI)
• Extensive Sharing Capabilities
  • Graphics, media, USB, audio, camera etc.
  • Advanced DMA/graphics buffer sharing
• Multiple OS Support
  • Clear Linux, Yocto, Ubuntu
  • Android, AliOS
• MISRA-C Compliance

Released in May 2019 @github.com/projectacrn/
Industrial: Safety + RT + HMI

Key Challenges:
- Mixed Criticality:
  - Real-Time vs non Real-Time
  - Safety vs non-Safety
  - Isolation vs Sharing

- Real-Time (Hard / Soft)
  - GBE packet IO control loop < 12us
  - MSI interrupt latency < 4us
  - Cyclic test jitter < 10us

- HMI
  - Window10

- Functional Safety
  - IEC 61508-3 (Industrial)
  - ISO 26262 (Automotive)
### Industrial Usage: Device Mapping Table

<table>
<thead>
<tr>
<th>Devices</th>
<th>Soft RTVM (RT Linux)</th>
<th>Hard RTVM (VxWorks)</th>
<th>HMI (Windows/Linux)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTC</td>
<td>Virtual</td>
<td>Virtual</td>
<td>Virtual</td>
</tr>
<tr>
<td>PCI</td>
<td>Virtual</td>
<td>Virtual</td>
<td>Virtual</td>
</tr>
<tr>
<td>UART</td>
<td>Passthru</td>
<td>Passthru</td>
<td>Virtual/Passthru</td>
</tr>
<tr>
<td>GBE Network</td>
<td>Virtual (PMD) /Passthru</td>
<td>Virtual (PMD) /Passthru</td>
<td>Virtual</td>
</tr>
<tr>
<td>TSN (i210)</td>
<td>Passthru</td>
<td>Passthru</td>
<td>N/A</td>
</tr>
<tr>
<td>Storage</td>
<td>Virtual (PMD) /Passthru</td>
<td>Virtual(PMD) /Passthru</td>
<td>Virtual</td>
</tr>
<tr>
<td>FPGA</td>
<td>Passthru</td>
<td>Passthru</td>
<td>N/A</td>
</tr>
<tr>
<td>GPU</td>
<td>N/A</td>
<td>N/A</td>
<td>Mediated Passthru</td>
</tr>
<tr>
<td>Audio</td>
<td>N/A</td>
<td>N/A</td>
<td>Passthru</td>
</tr>
<tr>
<td>USB</td>
<td>N/A</td>
<td>N/A</td>
<td>Virtual</td>
</tr>
<tr>
<td>Watchdog Timer (WDT)</td>
<td>Virtual</td>
<td>Passthru / Virtual</td>
<td>Virtual</td>
</tr>
</tbody>
</table>
ACRN + Zephyr as Safety Domain

Use Scenario

- Safety VM: Zephyr
  - Safety App
  - Safety OS

- Non-safety VM: Clearlinux
  - APP
  - Linux Kernel

Not certified

- 2 Partitions with mixed-criticality
- Static core & memory partitioning

IEC 61508 & ISO 26262 Certification Scope for ACRN

Certified by Intel and/or customers

ACRN Hypervisor

- VT-d
- EPT
- VMX
- Virtual PCI / Host bridge

BSP

BIOS

Physical Platform

LAPIC
- CPU Core

LAPIC
- CPU Core

LAPIC
- CPU Core

LAPIC
- CPU Core
Windows as HMI domain

- ACRN-GT GOP is added into OVMF to support Windows early display and Windows installation display.
- Support OVMF secure boot with vTPM for Windows secure boot chain.
- Support the Microsoft defined TLFS (Hyper-V Hypervisor Top-Level Functional Specification) minimum requirements and optional performance optimization requirements.
- Utilize Microsoft DISM tool to pre-install virtio-win drivers and gfx driver to the Windows install .iso file.
- Use GT-CLOS to prevent Windows from Cache interference.
VxWorks/RT Linux as Control domain

- **Service VM**
  - ACRN DM (Device Model)
    - virtio-net
    - virtio-blk
    - virtio-console

- **VxWorks/RT Linux UOS**
  - VIRTIO PMD FE Drivers
    - net
    - blk
    - console
  - PT Devices Drivers
    - net
    - blk
    - uart

- **Native Device Drivers**
  - ACRN Service Driver
    - net
    - blk
    - console

- **OVMF**
  - Hypercalls
  - Emulated Devices
    - vHostbridge/ vRTC / vHPET/

- **ACRN HV**
  - OSPM
  - VMX
  - VT-d
  - EPT
  - LAPIC PT
  - CAT

- **Pass-through LAPIC (except ICR/XAPICD/LDR)**
  - avoid VM-exit

- **Enable CAT**
  - isolate cache for RT VM

- **Enable virtio BE/FE as PMD**
  - avoid VM-exit

- **Configure native BIOS to disable**
  - Hyper-threading
  - Speed Step
  - Speed Shift
  - C-state
  - GT RC6
  - GFX Lower Power Mode
  - Native ASPM
  - …
Configuration for Real Time Latency Evaluation

Configuration:
• HW: Intel(R) Core(TM) i7-8650U CPU @ 1.90GHz, 8G Memory, 1M L2 cache, 8M L3 cache
• Benchmark: cyclictest (measure the scheduler jitter), running in Real-Time VM

ACRN:
• Service OS VM: Linux kernel v4.14.68-rt42
• Real-Time VM: Preempt-RT Linux: 4.14.68-rt42, with 2GB memory

Jailhouse:
• Root cell: Linux kernel v4.14.68-rt42
• Non-root cell: Preempt-RT Linux: Linux:4.14.71-rt44+, with 2GB memory
cyclic test: ACRN vs Jailhouse

Sched Jitter Distribution (1h)

Sched Jitter Histogram (1h)

Data reused from paper: <<ACRN: A Big Little Hypervisor for IOT Development>>, VEE’2019
<table>
<thead>
<tr>
<th>Area</th>
<th>V1.0@Q1'19</th>
<th>Q2'19</th>
<th>Q3'19</th>
<th>Q4'19</th>
<th>2020</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HW</strong></td>
<td>• APL NUC (UEFI)</td>
<td>• APL NUC (UEFI)</td>
<td>• APL NUC (UEFI)</td>
<td>• APL NUC (UEFI)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• KBL NUC (UEFI)</td>
<td>• KBL NUC (UEFI)</td>
<td>• KBL NUC (UEFI)</td>
<td>• KBL NUC (UEFI)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• APL UP2 (SBL)</td>
<td>• APL UP2 (SBL)</td>
<td>• APL UP2 (SBL)</td>
<td>• APL UP2 (SBL)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Denverton SoC</td>
<td>• Denverton SoC</td>
<td></td>
</tr>
<tr>
<td><strong>Hypervisor</strong></td>
<td>• Power Management (S3/S5)</td>
<td>• VxWorks as Guest</td>
<td>• Real-Time for Preempt-RT Linux</td>
<td>• Windows as guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ACRN partition mode</td>
<td>• Zephyr as Guest</td>
<td>• Real-Time for Pseudo Locking</td>
<td>• VxWorks as guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Local APIC passthrough</td>
<td>• ACRN Real-Time baseline</td>
<td>• Real-Time profiling tool</td>
<td>• Zephyr as Safety OS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Real-Time VM support</td>
<td>• ACRN Hybrid mode</td>
<td>• Real-Time Performance optimization</td>
<td>• CPU sharing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• OVMF for Clear Linux Guest</td>
<td>• Kata Container support</td>
<td>• Docker support based on Kata Containers</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>support</td>
<td>• OVMF GOP driver for GVT-g</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IOMMU interrupt remapping</td>
<td>• Device Posted Interrupt(PI)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• VM Configuration Unify</td>
<td>• Multiple IOAPIC support</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>I/O</strong></td>
<td>• GPIO virtualization</td>
<td>• SR-IOV for share mode</td>
<td>• USB hub virtualization</td>
<td>• Kubernetes support based on Kata Containers</td>
<td></td>
</tr>
<tr>
<td>virtualization</td>
<td>• QoS – Support RunC</td>
<td>• HPET Virtualization</td>
<td></td>
<td>• GVT-g Gen11 support</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• TPM2.0 Sharing (Security)</td>
<td>• Open vSwitch</td>
<td></td>
<td>• GVT-g for Windows as Guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• I2C virtualization</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Feature and dates for reference only and subject to change without notices.*
Call to Action

Join us!
If you support the ACRN project and feel that this is the right thing for the embedded ecosystem, join us in moving this project forward together as a community member. We need code contributors, users, and project direction influencers!

Contribute code!
Make a difference to the project by committing code, help us become a better project. Project code merged in the past 6 months allows you to become a voting member of the Technical Steering Committee.

All Contributions Matter
In open source projects a contribution can be anything which helps the project to accomplish its mission. Examples of Contributions beyond just code include:

- Financial Assistance
- Requirements Gathering
- Documentation
- Testing
- Bug Reporting