A Big Little Hypervisor for IoT Development

June 2019
CONTENTS

PROJECT OVERVIEW
USAGES
VALUE PROPOSITION
KEY CAPABILITIES
TECHNICAL OVERVIEW
ROADMAP 2019
GOVERNANCE
JOIN US!
Current State of Affairs – IoT Development

IoT device devices requires support for a myriad of HW resources on one platform:
Ex: Camera, Audio, Graphics, Networking, etc.

Needs to be able to support and run multiple operating systems simultaneously
Ex: Linux, Android, Windows, RTOS etc.

Needs to be able to support and run multiple SW tools and applications depending on the IoT device’s usage model

Need a way to consolidate ECUs and SW technology investment to lower total BOM costs for IoT devices productization
Virtualization is Key
Current Virtualization Landscape for IoT Development

Current Data Center Hypervisors

• Too large for embedded IoT development
• No safety-critical workload considerations
• Requires too much overhead for embedded development

Current Embedded Hypervisors

• Highly dependent on closed source proprietary solutions
  • Expensive
  • Makes product longevity difficult
• Hard partition, no ability to share resources

No Open Source Hypervisor optimized for embedded IoT development currently exists
Introduction

ACRN™ is a flexible, open-source, lightweight hypervisor - intended to enable consolidation of heterogeneous workloads, and to streamline IoT edge development.

- A Linux Foundation Project Launched in March 2018
- Version 1.0 released in May 2019
**Usages of ACRN**

### In-Vehicle Experience - Automotive (Hybrid-Mode)

- VM0: Safety VM
- VM1: Service VM
- VM2: User VMs

#### ACRN
- Partition of Functionally safe & unsafe domains
- Extensive resource sharing capabilities (graphics, media, USB, audio, etc) across VMs
- Orchestration of containerized workloads enables flexibility in design

### Industrial workloads consolidation

- VM0: Service VM
- VM1: RTOS
- VM2: HMI

#### ACRN
- Co-existence of Real-time sensitive domain with general computing domain
- Rich mediation of human interface and real-time sensitive devices/capabilities
- Flexible licensing allows deployment of proprietary licensed OS'es
## Value Proposition - ACRN

### Core Intrinsic Values

<table>
<thead>
<tr>
<th>Small Footprint</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Optimized for IOT class solutions</td>
</tr>
<tr>
<td>• Significantly smaller footprint than datacenter targeted hypervisors</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Heterogeneous Workloads consolidation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coexistence of</td>
</tr>
<tr>
<td>• Real time &amp; Non-Real time environments</td>
</tr>
<tr>
<td>• Safe &amp; Unsafe environments</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Open-source with Flexible Licensing</th>
</tr>
</thead>
<tbody>
<tr>
<td>• BSD license enables proprietary Guest OS</td>
</tr>
<tr>
<td>• True Open source with a vibrant Community</td>
</tr>
</tbody>
</table>

### Other Key Values

**System Security**
- Intel VT backed virtualization provides secure operating domains

**Functional Safety**
- Co-existence of Functionally Safe Domain with non-safe domain

**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 reduces system deployment complexity, enables heterogeneous architectures, and provide TCO advantages
### Key Capabilities

<table>
<thead>
<tr>
<th><strong>Open source w/ flexible licensing</strong></th>
<th><strong>System manageability</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Vibrant Community</td>
<td>VM Management</td>
</tr>
<tr>
<td>Linux Foundation Affiliation</td>
<td></td>
</tr>
<tr>
<td>BSD</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Diverse modes of operation</strong></th>
<th><strong>System security</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Partitioned Mode</td>
<td>Intel VT backed isolation</td>
</tr>
<tr>
<td>Shared mode</td>
<td>Framework for KATA</td>
</tr>
<tr>
<td>Hybrid (mix of partitioned &amp; shared)</td>
<td>containers</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Industrial usages &amp; standards</strong></th>
<th><strong>Rich mediation</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>MISRA-C compliant</td>
<td>Graphics, Audio, USB</td>
</tr>
<tr>
<td>Cache isolation</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Automotive usages &amp; standards</strong></th>
<th><strong>Configurability</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>FUSA certification targeted</td>
<td>CPU Assignment,</td>
</tr>
<tr>
<td>Hybrid arch. for isolated safety</td>
<td>Memory &amp; IO’s sharing &amp;</td>
</tr>
<tr>
<td>domain</td>
<td>pass-through</td>
</tr>
</tbody>
</table>
ACRN™ & OSV/ISV Vendors

Project’s Goal
Provide an embedded hypervisor reference solution to enable OSV/ISVs

A transparent enabler that provides:
• A common architecture to be used as-is
• A high quality reference stack optimized for embedded development

Productize on top of ACRN directly by adding value with:
• Proprietary Service OS or RTOS
• Commercial Licensing
• Commercial Support

Move the industry towards faster TTM
Hypervisor sits right on top of bootloader for fast booting capabilities.

Partitioning of resources to ensure safety-critical and non-safety critical domains are able to coexist on one SoC.

Rich IO mediators that allow for the Service OS to communicate directly to SoC ensuring low latency for safety critical elements.

Allows for multiple operating systems to be supported by one SoC through efficient virtualization.
Roadmap (2019-20)
Updated Q2-2019

Foundational Characteristics

- Heterogeneous Workloads consolidation
- Small Footprint (sub-50K lines)
- Open-source with a flexible licensing (BSD)
- Shared & Partitioned frameworks
- Beyond-Compute (IOs, Accelerators, Graphics/Media sharing)
- Designed for IOT market (Industrial, Automotive and beyond)

TODAY: ADVANCED SHARING

- Linux based Open source Service OS
- Linux, Android Guest OS’s
- Multiple Guest VMs in Sharing Mode
- Sharing of various IO’s

SECURE CONTAINERS

- Kata Containers as VMs
- Kubernetes support for Kata

DEEP ISOLATION

- Partition mode
- Open source Zephyr RTOS in Partition mode

GUEST OS EXPANSION

- Microsoft Windows
- Wind River VxWorks
# ACRN Open Source Roadmap in 2019

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

<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></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></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></td>
<td></td>
<td>• Denverton SoC</td>
</tr>
<tr>
<td><strong>Hypervisor</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></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>• Zephyr as Safety OS</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>• CPU sharing</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>• Docker support based on Kata Containers</td>
<td></td>
</tr>
<tr>
<td><strong>I/O virtualization</strong></td>
<td></td>
<td>• OVMF for Clear Linux Guest support</td>
<td>• Kata Container support</td>
<td>• Windows as guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• GPIO virtualization</td>
<td>• IOMMU interrupt remapping</td>
<td>• OVMF GOP driver for GVT-g</td>
<td>• Zephyr as Safety OS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• QoS – Support RunC</td>
<td>• VM Configuration Unify</td>
<td>• Device Posted Interrupt(PI)</td>
<td>• CPU sharing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• TPM2.0 Sharing (Security)</td>
<td></td>
<td>• Multiple IOAPIC support</td>
<td>• Docker support based on Kata Containers</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Windows as guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Zephyr as Safety OS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• CPU sharing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Docker support based on Kata Containers</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Windows as guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Zephyr as Safety OS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• CPU sharing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Docker support based on Kata Containers</td>
<td></td>
</tr>
</tbody>
</table>

- **SR-IOV for share mode**
- **HPET Virtualization**
- **Open vSwitch**
- **I2C virtualization**
- **USB hub virtualization**
- **Kubernetes support based on Kata Containers**
- **GVT-g Gen11 support (Elkhart Lake SoC)**
- **GVT-g for Windows as Guest**
ACRN

GOVERNANCE
Governance Ethos

**Community Involved**
- Encourage and welcome participation in the project

**Transparency**
- Committed to working transparently in the open to encourage the open source community’s involvement

**Code > Currency**
- The project is committed to a true open governance where code contributions is the main objective
Governance: Technical Steering Committee

ACRN™
A Linux Foundation Project

Technical Steering Committee
- Committee will have (5) members at project launch.
- No more than 50% of members from one company or organization
- Committee will always have an odd number of members to allow for voting decisions

Chair/Maintainer
- Most active commits, driving decisions for the project, leadership and point of contact
- Have the ability to merge code (+2)
- Ability to nominate new maintainer roles to be voted on as needed

Voting Members
- Initial appointments by founding contributors
- Election amongst contributors every 6 months (after launch). After the first election, each member is elected for one year
- Qualified voters have code merged in the past 6 months
JOIN US!
Build complex embedded systems requiring various levels of safety-criticality

Resource sharing allows for maximizing potential of resource-constrained devices while ensuring safety critical workloads are given priority to platform hardware

Open source code allows for developmental cost savings as transparent code ensures code compatibility, flexibility in vendor selection as well as helping to ensure longevity of product lifecycle while reducing support and maintenance costs of the product’s life.

Small footprint, lightweight code base to ensure inclusion of only necessary code optimized for embedded development. No superfluous code.
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
Backup

(Version of 2018)
# ACRN Open Source Roadmap in 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></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• APL NUC (UEFI)</td>
<td></td>
<td>APL NUC (UEFI)</td>
<td>APL NUC (UEFI)</td>
<td>APL NUC (UEFI)</td>
<td>APL NUC (UEFI)</td>
</tr>
<tr>
<td>• KBL NUC (UEFI)</td>
<td></td>
<td>KBL NUC (UEFI)</td>
<td>KBL NUC (UEFI)</td>
<td>KBL NUC (UEFI)</td>
<td>KBL NUC (UEFI)</td>
</tr>
<tr>
<td>• APL UP2 (SBL)</td>
<td></td>
<td>APL UP2 (SBL)</td>
<td>APL UP2 (SBL)</td>
<td>APL UP2 (SBL)</td>
<td>APL UP2 (SBL)</td>
</tr>
<tr>
<td>• Denverton SoC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Hypervisor</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Power Management (S3/S5)</td>
<td></td>
<td>VxWorks as Guest</td>
<td>Real-Time for Preempt-RT</td>
<td>Windows as guest</td>
<td></td>
</tr>
<tr>
<td>• ACRN partition mode</td>
<td></td>
<td>Zephyr as Guest</td>
<td>Linux</td>
<td>Zephyr as Safety OS</td>
<td></td>
</tr>
<tr>
<td>• Local APIC passthrough</td>
<td></td>
<td>ACRN Real-Time baseline</td>
<td>Real-Time for Pseudo Locking</td>
<td>CPU sharing</td>
<td></td>
</tr>
<tr>
<td>• Real-Time VM support</td>
<td></td>
<td>ACRN Hybrid mode</td>
<td>Real-Time profiling tool</td>
<td>Docker support based on</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>OVMM for Clear Linux Guest</td>
<td>Real-Time Performance</td>
<td>Kata Containers</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>support</td>
<td>optimization</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IOMMU interrupt remapping</td>
<td>Kata Container support</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VM Configuration Unify</td>
<td>OVMF GOP driver for GVT-g</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Device Posted Interrupt(PI)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Multiple IOAPIC support</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>I/O virtualization</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• GPIO virtualization</td>
<td></td>
<td>SR-IOV for share mode</td>
<td>USB hub virtualization</td>
<td>Kubernetes support based</td>
<td></td>
</tr>
<tr>
<td>• QoS – Support RunC</td>
<td></td>
<td>HPET Virtualization</td>
<td></td>
<td>on Kata Containers</td>
<td></td>
</tr>
<tr>
<td>• TPM2.0 Sharing (Security)</td>
<td></td>
<td>Open vSwitch</td>
<td></td>
<td>GVT-g Gen11 support</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>I2C virtualization</td>
<td></td>
<td>(Elkhart Lake SoC)</td>
<td>GVT-g for Windows as Guest</td>
</tr>
</tbody>
</table>
ACRN Look Ahead in 2019

2019 will be an exciting year for project ACRN. Several big things are planned:

- **By early Q2, we’ll welcome ACRN v1.0** and provide a stable software reference for Software-Defined-Cockpit (SDC) usage on Intel Apollo Lake platforms.

- **Real-Time OS will be supported**, opening use of ACRN in industrial scenarios needing low latency, and fast, predictable responsiveness. Initial support is for VxWorks and Zephyr OS as Real-Time guest OSes in Q2, and PREEMPT-RT Linux in Q3.

- A new **ACRN Hybrid Mode** will be completed in Q2, giving ACRN the ability to run mixed-criticality workloads. For example, running a Real-Time Guest OS with a time sensitive application and dedicated hardware resources assigned, together with a normal priority Guest OSes (UOS) running with Service OS (SOS) and sharing the remaining hardware devices.

- **Windows as Guest (WaaG)** will be officially supported in Q4, but you will see incremental features merged before that. For example, we’ll soon introduce a virtual boot loader, OVMF, that enables UEFI support for Virtual Machines required for supporting WaaG.

- **Kata Containers** will be supported in Q3. **Kata Containers** is an open source project and community working to build a standard implementation of lightweight Virtual Machines (VMs) that feel and perform like containers, but provide the workload isolation and security advantages of VMs.

- More **I/O device virtualization** will be implemented to enrich ACRN’s IoT device hypervisor capability, including GPIO virtualization in Q1, I2C virtualization in Q2 and Intel GPU Gen11 support in Q4.

- **CPU sharing** will be a big thing for ACRN. Typically used for embedded systems, a partitioned CPU will be assigned to a Guest VM to benefit the isolation and fast response from hardware. There’s also a requirement for non-critical usage for sharing CPU cores among multiple VMs to better support Kata container.

- **Functional Safety (FuSa) certification process** will be applied to ACRN core feature development, and ultimately help ACRN be deployed in industrial or automotive (SDC) uses.

*More details can be found in “ACRN Open Source Roadmap 2019”*
2019 ACRN Roadmap Feature Description

ACRN partition mode: Guest VM is created and launched from hypervisor with partitioned hardware resource (CPU, Memory, pass-through devices etc.)

Local APIC pass-through: To pass-through most of local APIC MSRs to Guest VM and deliver IRQ to Guest VM directly without needing vmexit.

Real-Time VM support: To support the Real-Time VM's power lifecycle (e.g. boot/reboot/poweroff), need to complete LAPIC passthru, ioreq completion polling, and virtio-PMD.

QoS – Support RunC: Run ACRN DM Device Model (acrndm) inside a container (runc) to guarantee the QoS for the VM's I/O.

TPM2.0 Sharing: Trusted Platform Module 2.0 virtualization on ACRN.

Real-time baseline: basic real-time support, it provides the baseline for future real-time enhancement and performance tuning.

OVMF for Clear Linux support: Open Virtual Machine Firmware (OVMF) will be used as virtual VM bootloader on ACRN. OVMF will boot ClearLinux as a User OS (UOS)

IOEMU interrupt remapping: Enable IOEMU interrupt remapping for pass-through devices to ensure they only send their interrupts to the processor they are supposed to, even if the device is misbehaving.

ACRN Hybrid mode: ACRN hypervisor can create and launch pre-defined guest(s) and the Service OS (SOS) with partitioned hardware resource. The SOS can create & launch additional UOS Guests through its Device Model (DM). Two types of Guests can co-exist, pre-defined guest(s) and those managed by the SOS (acrndm)

VM Configuration unification: Both hypervisor & DM can launch Guests with same VM configuration structure, and an offline tool will be used to create VM configuration structure info for hypervisor-launched Guests, similarly DM will create VM configuration structure info for the guests which it launches.

SR-IOV for sharing mode: To support passing through virtual function of SR-IOV device in sharing mode.

HPET Virtualization: Add HPET virtualization in ACRN-DM, prepare for removing legacy device.

Open vSwitch: Enable Open vSwitch kernel module in SOS as a bridge, to which NICs inside VMs are connected.

Real-Time for PREEMPT_RT Linux: PREEMPT_RT Linux will run as ACRN RT VM guest, with soft real-time performance.

Real-Time for Pseudo Locking: The RT VM guest can lock the real-time critical data or instructions into the cache by leveraging the Intel CAT technology.

Kata Containers support: enable Kata containers on ACRN.

OVMF GOP driver for GVT-g: OVMF GOP doesn't support GVT-g, this feature is to support GVT-g in OVMF GOP, so we can use OVMF console with GVT-g.

Device PI (posted interrupt): VT-d posted interrupt allows the PCI device interrupt to be injected to the Guest without hypervisor participation.

Real-Time profiling tool: Real-Time profiling tool will help the developers to tune the Real-Time performance.

Real-Time Perf optimization: Hypervisor and operating system level Real-Time performance will be optimized.

Multiple IOAPIC support: some platforms (such the Denverton Atom family) have multiple physical IOAPICs, ACRN will be extended to support this.

CPU sharing: Enable scheduler in hypervisor to share physical CPU resource between all SOS & UOS vCPUs

Docker support based on Kata Containers: Make Kata/ACRN work with Docker.

Zephyr as Safety OS: Zephyr will be used as safety OS which will perform the system functional safety related tasks.

Kubernetes support: Make Kata/ACRN work with Kubeenetes.

GVT-g Gen11 support: GVT-g support for Intel GPU device on Ice Lake platform

GVT-g for WaaG: GVT-g support Windows as Guest on ACRN