Jumpstart: Fast Critical Service Resumption for a Partitioning Hypervisor in Embedded Systems

Ahmad Golchin and Richard West

Boston University
DriveOS

Quest RTOS
Critical Vehicle Control Tasks

Linux
User IF
(IC, IVI, etc.)

Quest-V

Inter-sandbox

Device I/O

Vehicle Management System

ADAS Sensors

Powertrain, Chassis and Body Controllers
Why PC-class Embedded Systems?

Advantages:

- Higher Processing Capabilities
- Abundant Resources
- H/W Virtualization Technologies
- Smaller Footprint, Lower Cost, etc.

Disadvantages:

- Difficult to Provide Spatial/Temporal Isolation, etc.
- Long Boot Delays

Consolidation of 100+ ECUs into 1 Central System
Quest RTOS

● Real-time OS for multicore x86 platforms
  ○ UP2, DX1100, Intel Aero, Skull Canyon, etc.

● Dual-mode monolithic kernel

● Unified task and I/O scheduling through time-budgeted virtual CPUs (VCPUs)
  ○ Main VCPUs for task scheduling
  ○ I/O VCPUs for interrupt bottom-half scheduling

● More info: www.questos.org
The Quest-V Partitioning Hypervisor

- RTOS boot-strapped
- Support for Linux SBs
- Static partitioning:
  - CPUs, RAM, I/O
- Shared memory ISBC
- Mixed-criticality
  - Temporal & spatial separation
Boot Delay of DriveOS

Delay Components

- Firmware ~ 7 seconds
- Bootloader ~ 1.4 seconds
- Virtualization ~ 4.7 seconds
- RTOS Startup ~ 3.5 seconds
- Linux Startup ~ 11.3 seconds

Objective < 1 second for VMS
Boot Delay of DriveOS

- **Firmware and Bootloader**
  - At least 2.5 kernels between the Guest and IA-PC H/W (Minich et al)
    - UEFI Firmware
    - Intel Management Engine (IME) running Minix
    - Intel System Management Mode (SMM)
  - **Existing solutions:** NERF, Coreboot, Intel Slim
    - Reduced F/W, ROM-hosted OS images, etc.

- **Hypervisor**
  - Architectural setup, Resource Partitioning, etc.

- **Guest Kernel and Drivers**

- **User-space Services**
UEFI vs Intel Slim Bootloader

Platform: DX1100

Platform: UP2

- Linux GUI
- Linux Userspace
- Linux Kernel
- Linux Image Relocated
- RTOS Kernel & Userspace
- Sandboxes Forked
- Hypervisor
- Bootloader
- Firmware
Power Management

- ACPI(CA)
- PCI-PM
- Dynamic vs Static
- Virtual vs Real
Feasibility of a PM Solution

- 2.473 W
- 2.643 W
Jumpstart

Framework

- Quest/Linux Kernel Modules
- Quest-V Monitor Module

Function

- Turns System-wide Shutdown/Boot into ACPI S3 Suspend/Resume

Achieves

- ~600ms Quest
- ~1050ms Linux
Jumpstart Power Management

Challenges

- Unauthorized guest access to the system's Embedded Controller
  - I/O Ports & ACPI memory
- Orchestration of system-wide power transition
  - Power Master & Inter-sandbox IPC
- Resumption of critical real-time tasks
  - Idempotent vs Resumption
- Resumption of critical real-time sandbox with lower latency
  - Shared boot logic of Quest and Quest-V

Requirements

- ACPI-compliance platform
- Support of ACPI S3 natively by the guest
Control Flow

- User-space
- Kernel
- Hypervisor
- ISB IPC
- Synch. Point

```
Quest Sandbox

Ring 3
  2 Jumpstart Syscall

Ring 0
  3 Native Quest Suspend
  4 Suspend Other VMs
  7 Jumpstart Hypercall

Enter S3

VMExit
  8 - 15 Save/Rstr Guest

VMResume
  16 Native Quest Resume

Linux Sandbox

Ring 0
  1 Vehicle Power Off Request

Enter S3

VMExit
  5 Native Linux Suspend
  7 Jumpstart Hypercall

VMResume
  16 Native Linux Resume
```
Control Flow

- **Kernel**
- **Hypervisor**
- **H/W & F/W**
Resumption Delay of DriveOS with Jumpstart

Delay Components

- Firmware ~ 500 ms
- Quest-V ~ 30 ms
- Quest Guest ~ 66 ms
- Linux Guest ~ 510 ms

Quest : 600 ms
Linux : 1040 ms
Jumpstart vs Standalone Quest/Linux

Warm Boot Delays (DX1100)

- Userspace Resumed
- Drivers Resumed
- Kernel Resumed
- Linux Guest Resumed
- RTOS Guest Resumed
- Hypervisor Warm Boot
- Firmware Warm Boot
Conclusions and Future Work

- Why Jumpstart?
  - Similar power consumption of Suspend-to-RAM and Shutdown
  - Higher degree of portability
  - Faster parallel resumption of partitioned guests w.r.t. Standalone
  - Complementary to firmware optimizations

- Future Direction
  - Fast non-volatile memories and ACPI S4
Thank you!