Challenges in Supporting Virtual CPU Hotplug in SoC Based Systems (like ARM64)

Salil Mehta | **System Software Architect @Huawei, Cambridge, UK**

Thursday, 29th October 2020, 16:00 - 16:30 GMT

§ Link to KVMForum2020 Event

Agenda

- Recently, attempts have been made to add support of Virtual CPU Hotplug for ARM64 in QEMU and Linux Guest Kernel.
- It has received mixed reviews from the community.
- Some vendors have practical reasons to have such a support added.
- A few community members have expressed apprehensions about its support.
- This presentation is an attempt to highlight the key issues in supporting *Virtual* CPU Hotplug feature on ARM64 based SoCs.

Outline

1. Quick Overview
    1. Why do we need CPU Hotplug?
    2. Virtual CPU hot-(un)plug Event Sequence
2. Known Challenges with
    1. ARM64 System Architecture
    2. Host KVM
    3. QEMU Virtualizer
    4. Guest Kernel
3. Discussed Workarounds for the Known Challenges
4. Implementation Attempts Recent & Earlier
5. Problems Being Faced in Upstreaming
6. Summary What is the Way Forward?
7. Future Work
8. Acknowledgements
9. Q&A

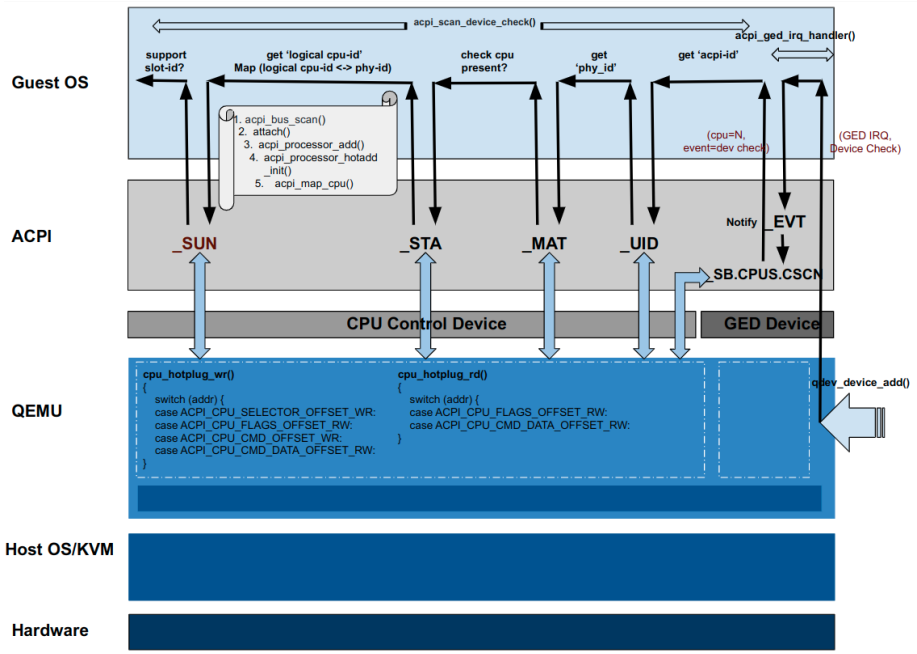
Quick Overview Why do we Need CPU Hotplug?

In general, CPU Hotplug feature [3] could be used for:

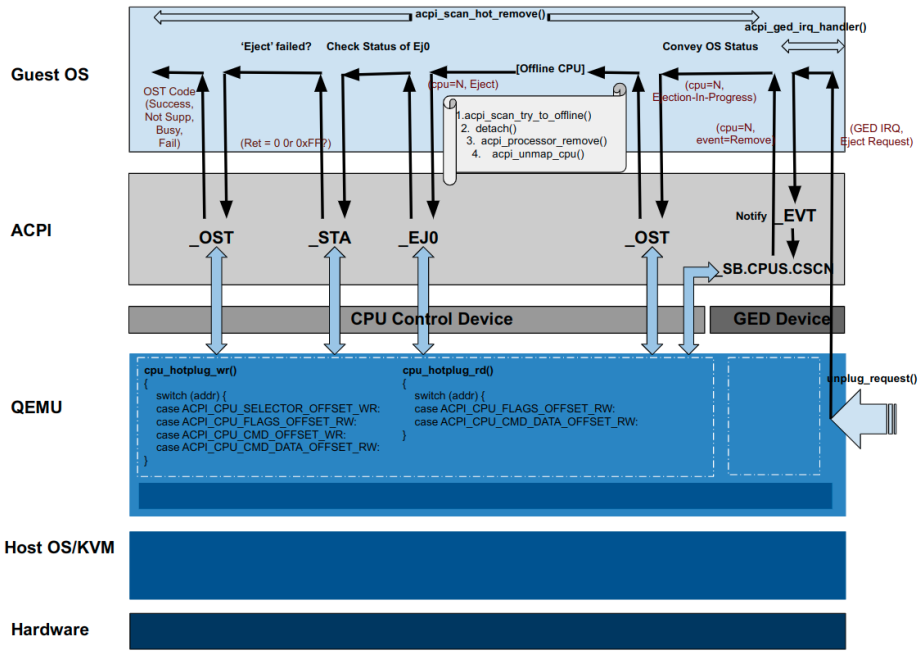
- Provisioning
  - Provisioned but Autoscaled(Vertical Pod Autoscaling) [4]
  - On-demand Provisioning (Capacity-Upgrade-on-Demand).
- Isolation of error causing CPUs/RAS (makes sense in virtual world?).
- Suspend/Resume support. {On|Off}line CPUs.
  - PSCI cpu-off based hotplug has been there in ARM64 for long [1]
  - This might not be ACPI driven. [2]

[1] arm64: add PSCI CPU_OFF-based hotplug support
[2] agent: add offline cpu interface support #478 (Justin He)
[3] CPU hotplug in the Kernel
[4] Vertical Pod Auto-scaling

Quick Overview Virtual CPU Hot-(un)plug Event Sequence



Sequence Depicting ACPI Driven vCPU Hotplug Action



Sequence Depicting ACPI Driven vCPU Hot-Unplug Action

Known Challenges with ARM64 System Architecture

- ARM64 System architecture have no concept of physical CPU hotplug.
- No specification defining a standard way to realize *virtual* CPU hotplug.
- Other ARM components like GIC have not been designed to realize physical CPU hotplug capability.
- GIC requires all the CPUs to be present at initialization.

Known Challenges with the Host KVM (1)

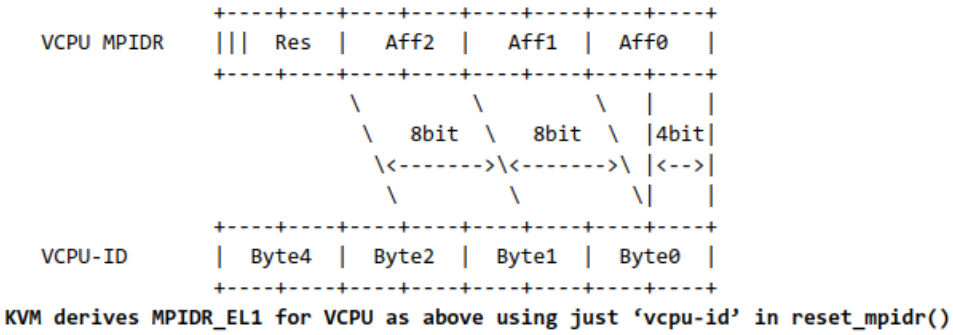
- KVM requires all vCPUs to be created during VM init time.

- Each vCPU will have VGIC related resources initialized & fixed during creation.
  - Various VGIC per-cpu static data structure.
  - Some config of related private(SGI+PPI) interrupts.
  - Resources like memory-regions(2*64K) of the related redistributors.

- Once vCPUs have been created in the Host KVM their destruction is not supported yet! (This is not specific to ARM64).

Known Challenges with the Host KVM (2)

- As of now, KVM uses vcpu-id to derive affinity value of EL1_MPIDR of vCPU
  - MPIDR uniquely identifies the vCPU in a system.
  - Should *virtual* MPIDR value be a responsibility of user-space instead of KVM?



KVM derives MPIDR_EL1 for VCPU as above using just 'vcpu-id' in reset_mpidr()

Known Challenges with QEMU Virtualizer (1)

- Due to the limitation imposed by the VGIC at Host KVM, QEMU MUST create and initialize all the vCPUs at **virtmach_init()** time.

- Realization of vCPUs(and its threads) in QEMU might not be desirable for possible vCPUs which are in disabled state.

- QEMU MUST then ensure complete realization of the GIC. This includes realization+initialization of all the redistributors and ITS (related to possible vCPUs) at KVM .

Known Challenges with QEMU Virtualizer (2)

- Un-wiring of the interrupts setup between vCPUs and the GIC requires further consideration in QOM.

- For ARM64, QEMU lacks support to correctly specify vCPU topology (soc, cluster, cores, threads). Required to uniquely identify the vCPU being plugged or unplugged. Should map to QEMU allocated (i.e. derived using some logic or even cpu topology?) MPIDR?

- QEMU lacks the support of PPTT table which shall be used to pass on the vCPU topology to the Guest Kernel.

Known Challenges with Guest Kernel

- Any ARM64 architecture related changes done inside the Kernel for the Guest should seamlessly run on Host kernel.
- vCPU hotplug might benefit from some standardization at the architecture and firmware/ACPI level.
- Any future specification allowing physical CPU hotplug must not be unduly constrained by a vCPU hotplug interface defined now.

Discussed Workarounds for the Known Challenges (1)

Various workarounds have been discussed within the community to get around the limitations,

- All possible vCPUs created within the KVM and QEMU at VM initialization.
- QEMU only realizes possible vCPUs that are not disabled. Hence, the vCPU threads are not spawned for disabled vCPUs.
- QEMU also realizes redistributors related to all the possible vCPUs and initializes their related data-structures in the KVM VGIC. This is done after all possible vCPUs have been pre-created.

[1] VCPU hotplug on KVM/ARM - Marc Zyngier
[2] VCPU hotplug on KVM/ARM - Andrew, Marc, Igor
[3] VCPU hotplug on KVM/ARM - Maran Wilson

Discussed Workarounds for the Known Challenges (2)

- QEMU and Host KVM enhanced to support user-space configuration of the MPIDR value of the vCPU. (A. Jones et al.)
- On vCPU hot-unplug, QEMU parks and powers down(PSCI off) the vCPUs.
- QEMU provides complete MADT Table including all possible vCPU interfaces and its redistributors to the Guest Kernel.
- At boot time, Guest Kernel uses info from MADT Table to size its various data-structures with all possible vCPUs (including initialization of related redistributors/ITS like Table base addresses etc.)

Implementation Attempts Recent & Earlier

Starting from most recent ones to earlier attempts,

- [PATCH RFC 0/4] Changes to Support *Virtual* CPU Hotplug for ARM64 (Salil Mehta/Zhukeqian)
  https://lkml.org/lkml/2020/6/25/434
- [PATCH RFC 00/22] Support of Virtual CPU Hotplug for ARMv8 Arch (Salil Mehta/Zhukeqian)
  https://www.mail-archive.com/qemu-devel@nongnu.org/msg712010.html
- [RFC PATCH v2 0/3] Support CPU hotplug for ARM64 (Xiongfeng Wang)
  https://lkml.org/lkml/2019/6/28/1157
- [PATCH 0/3] arm/virt: refine virt.c code and implement hot_add_cpu interface (Li Zhang)
  https://lists.gnu.org/archive/html/qemu-devel/2017-05/msg05992.html
- [RFC PATCH 0/6] hw/arm/virt: Introduce cpu topology support (Andrew Jones)
  https://lists.nongnu.org/archive/html/qemu-devel/2018-07/msg01168.html
- agent: add offline cpu interface support #478 (Justin He)
  https://github.com/kata-containers/agent/pull/478

Problems Being Faced in Upstreaming

- ARM64 System Architecture does not supports Physical CPU Hotplug. Therefore, due to absence of the suitable specification, concern has been raised to avoid divergent system architecture being invented by different vendors. This has left the Kernel patches stranded.

- QEMU patches cannot proceed till the time Guest Kernel patches are duly considered by the kernel community.

- Need more reviews and involvement by community.

Summary What is the Way Forwards?

- ACPI based support of virtual VCPU Hotplug feature is a much requested feature for the use-cases stated earlier in the slides.
- We have proposed a practical implementation, but seem to be making little progress in upstreaming.
- How do we overcome the resistance to implementing a virtual only feature, while minimizing possible clashes with a potential future physical equivalent?
- Would some sort of minimal specification to ensure some consistency of implementation(in the virtual case), alleviate such concerns? How to do this?

Future Work

Below need to be completed,

- Discussions related to specification work (ARM Arch/Firmware level)?
- Live Migration support
- Support of hotplug with NUMA
- CPU Topology
- PPTT Support to handover right vCPU topology info to guest
- Test cases
- Docs need to be updated.

Acknowledgements

I would like to take this opportunity to thank below individuals and the other people of the community who have contributed to the present and past work or the discussions and have pitched-in their ideas. I am just trying to carry forward the passed baton!

- Marc Zyngier (Google)
- James Morse (ARM)
- Sudeep Holla (ARM)
- Mark Rutland (ARM)
- Jia (Justin) He (ARM)
- Alex Bennée (Linaro)
- Ashok Kumar (Broadcom)
- Igor Mammedov (Redhat)
- Andrew Jones (Redhat)
- Michael S. Tsirkin (Redhat)
- Xiongfeng Wang (Huawei)
- Zhukeqian (Huawei)
- Jonathan Cameron (Huawei)
- Shameerali Kolothum Thodi (Huawei)
- Hanjun Guo (Huawei)
- John Garry (Huawei)
- Xuwei (Huawei)
- Zengtao (Huawei)

Comments we can Discuss?

Q&A

Thank You!

KVM FORUM

HUAWEI

THE LINUX FOUNDATION