Linaro Open Discussion Meeting: Updates

Usage of _STA.Enable AND online-capable Bits

Issues+proposal

Salil Mehta

Huawei Technologies UK R&D Ltd.

Date: 5th Oct 2022

Discussion Contents

1. Forward ported QEMU location
    I. This is the forward port of the RFC V1 floated in Jun 2020 with some minor fixes
    Link: https://github.com/salil-mehta/qemu.git virt-cpuhp-armv8/rfc-v1-port29092022

2. Can we use _STA.Enabled for Identifying whether processor can be made present or Not present?
Issues Identified:
    I. _STA.Enable = 1 always. See how the unplug protocol works.
    Reference: https://sched.co/eE4m (Slide 5 – ACPI Hot Unplug exchanges)
    II. CPU is not removed cleanly I.e. arch_unregister_cpu() is not called.
    III. GICC Online-capable Bit can resolve this issue
    Link: https://github.com/salil-mehta/linux.git virt-cpuhp-arm64/rfc-v2/jmorse-pres-eq-poss-cpu

3. Catch: online-capable not sufficient for removing cold-booted cpus.
    I. Can we treat cpus with GICC.Enabled=1 just like online-capable cpus (I.e. GICC.online-capable=1) when unplugging or plugging them back after boot?

4. Some ordering problems during init. Please see the screen shot

HUAWEI

Discussion Contents

```
[   73.641469] [acpi_processor_make_enabled] cpu4 is PRESENT, FW STA is ENABLED
[   73.642769] sysfs: cannot create duplicate filename '/devices/system/cpu/cpu4'
[   73.643575] CPU: 2 PID: 54 Comm: kworker/u12:2 Not tainted 6.0.0-rc4-188821-g321ee5476a27-dirty #57
[   73.644590] Hardware name: QEMU KVM Virtual Machine, BIOS 0.0.0 02/06/2015
[   73.645335] Workqueue: kacpi_hotplug acpi_hotplug_work_fn
[   73.646185] Call trace:
[   73.646451]  dump_backtrace+0xdc/0xe8
[   73.646851]  show_stack+0x18/0x50
[   73.647212]  dump_stack_lvl+0x68/0x84
[   73.647618]  dump_stack+0x18/0x34
[   73.647991]  sysfs_warn_dup+0x60/0x80
[   73.648414]  sysfs_create_dir_ns+0xe4/0x100
[   73.648885]  kobject_add_internal+0x98/0x220
[   73.649367]  kobject_add+0x94/0x108
[   73.649759]  device_add+0xf8/0x8a8
[   73.650145]  device_register+0x20/0x30
[   73.650569]  register_cpu+0xf0/0x1b0
[   73.650974]  arch_register_cpu+0x5c/0x70
[   73.651415]  acpi_processor_add+0x410/0x680
[   73.651886]  acpi_bus_attach+0x12c/0x228
[   73.652334]  acpi_bus_scan+0x58/0x110
[   73.652745]  acpi_device_hotplug+0x208/0x470
[   73.653227]  acpi_hotplug_work_fn+0x24/0x40
[   73.653697]  process_one_work+0x1d0/0x320
[   73.654148]  worker_thread+0x4c/0x400
[   73.654561]  kthread+0x110/0x120
[   73.654924]  ret_from_fork+0x10/0x20
[   73.655334] kobject_add_internal failed for cpu4 with -EEXIST, don't try to register things with the same name in the same directory.
[   73.656688] acpi ACPI0007:04: Enumeration failure
```

Another ordering issue?

```
[    0.124713] register_cpu_capacity_sysctl: too early to get CPU1 device!
[    0.125457] register_cpu_capacity_sysctl: too early to get CPU2 device!
[    0.126204] register_cpu_capacity_sysctl: too early to get CPU3 device!
[    0.126966] register_cpu_capacity_sysctl: too early to get CPU4 device!
[    0.127785] register_cpu_capacity_sysctl: too early to get CPU5 device!
```

Discussion Contents

5. Present == Possible has problems
    I. User interface ambiguous (?)
    II. Suggestion: selectively exposing present cpus can solve above issues
        - Experimented with above and it works with forward ported QEMU repo
        - Link: https://github.com/salil-mehta/linux.git virt-cpuhp-arm64/rfc-v2/jmorse-variant-with-cond-present-cpu
    III. Can keeping present==possible create unnecessary memory allocation/bloating problems during initialization especially when cpu number is bound to go up?

6. Need to decide pros and cons of each approach presented in below repositories properly?
    I. James Approach with online-capable and present==possible (some fixes)
        - https://github.com/salil-mehta/linux.git virt-cpuhp-arm64/rfc-v2/jmorse-pres-eq-poss-cpu
    II. Variant of James approach with online-capable and conditionally present cpus
        - https://github.com/salil-mehta/linux.git virt-cpuhp-arm64/rfc-v2/jmorse-variant-with-cond-present-cpu

Minutes of Meeting

1. Salil presented some updates on the testing of the James Kernel patches with the QEMU.
   - https://git.gitlab.arm.com/linux-arm/linux-jm.git virtual_cpu_hotplug/rfc/v0

2. Forward ported QEMU with some fixes was shared (by Salil)
   - https://github.com/salil-mehta/qemu.git virt-cpuhp-armv8/rfc-v1-port29092022

3. Some discussions on the use of _STA.Enable Bit during remove of cpu which was causing crash.
   - There is an assumption in the patches that _STA.ENA=0 while cpus are being removed.

4. Issue of present == possible with the James patches was also discussed
   - Could we get around this by conditionally making cpus present in the kernel (by Salil)
     - "ACPI says present but Linux still says not present" is an inconsistent representation and can lead to future maintenance problems (by James)
       - What kind of future problems? (needs more debate?)

5. Issue with removing cold-booted cpus was also discussed
   - Jonathan suggested keeping a variable in the kernel to identify the cpu which was earlier cold-booted or we could even use GICC Enabled/online-capable flag bits from MADT Table.

Minutes of Meeting

6. A thought to evolve the ACPI handshake protocol between firmware and kernel was also discussed
   - Jonathan floated an idea of using the _OSC method ?
   - James mentioned the limitation that if GICC.Enabled=1 during boot then none of the _STA fields could change as it effects the functionality of the 'kexec'

7. James would be using the forward ported QEMU repo for further testing and fixing. He might not be available for next few weeks as he would shift his focus on MPAM.

8. QEMU still has lots of issue to be resolved, Salil shall continue to work in refining those and help James in further resolving the issues with his approach

9. A variant of James approach with conditionally making CPU present has also been shared with the ARM folks for their humble consideration.
   - https://github.com/salil-mehta/linux.git virt-cpuhp-arm64/rfc-v2/jmorse-variant-with-cond-present-cpu
   - This has been found working in all the cases. Although, the issue about inconsistency between ACPI and kernel needs a thorough discussion!

Thanks