Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024

ISA extension management/enablement in heterogeneous systems

Andrew Jones <ajones@ventanamicro.com>

LINUX
PLUMBERS
CONFERENCE Vienna, Austria / Sept. 18-20, 2024

Overview

- What type of heterogeneous systems are we talking about?
- Challenges with considering those types of heterogeneous systems
- Discussion

Heterogeneous systems

- Types of heterogeneous systems
  - CPUs integrated with coprocessors (GPUs, NPUs, other accelerators…)
    - Run on “main” cores and schedule specific workloads on coprocessors
    - Not in the scope of this discussion
  - Single-ISA: CPUs integrated together which have identical ISA but differ in characteristics affecting power consumption
    - Potentially different clock frequencies, cache sizes, etc.
    - Mostly not in the scope of this discussion
  - Overlapping-ISA: CPUS integrated together which have overlapping, but not identical ISA
    - Example: Some Arm SoCs only have the AArch32 feature on a subset of cores
    - The main focus of this discussion

Challenges with considering overlapping-ISA heterogeneous systems

kernel

- Available vs. enabled extensions
  - Enabled may additionally require
    - Compiler support
    - Kconfig=y
    - Both compiler support and Kconfig=y
    - Other dependencies checked at detection time and/or at alternative patching time
- Must check enabled *hart-common* extensions
  - Otherwise alternatives would be broken
  - Can't hotplug a hart that is missing anything from *hart-common*

KVM

- Checks *per-hart* extensions
  - Leave it to the VMM to pin VCPUs if necessary
    - Otherwise VCPU migration would be broken
    - Arm KVM VMM has to pin VCPUs on big.LITTLE due to MIDR passthrough
- VMM determines supported extensions with KVM ioctls
  - Can't use hwprobe since KVM may not support what hwprobe advertises
  - No current KVM ioctl equivalent for hwprobe's *which-cpus*

usermode

- Hopefully libraries and applications are learning to use hwprobe
- hwprobe returns the AND of the usermode exposed *per-hart* extensions for the task's cpumask
  - hwprobe's *which-cpus* allows collecting a set of harts supporting given extensions
    - Deepak Gupta[1] made good arguments as to why this would be quite difficult to do with shared libraries and competing affinity selections

[1]https://lore.kernel.org/all/CAKC1njRqWYOsF9bQvWX99DhP8Ji_wDUc8J8N41=N6J_tncM3=A@mail.gmail.com/

Backup
(U-mode extension management)

Usermode extensions without kernel enablement

- Extensions that don’t need senvcfg bits set or other support from the kernel
  - Still need to wait until kernel at least enumerates them in hwprobe?
    - Or is there anyway to automatically pass their availability through?
    - Or will usermode probe with test instructions and SIGILL handlers?