XUANTIE

RISC-V Heterogeneous Programming Paradigm

Atomic IO Enqueue (AIOE) Extension & AIOE with Virtualization

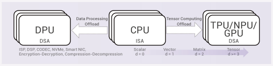
Author: Guo Ren, Alibaba Damo Academy
Email: guoren.gr@alibaba-inc.com

Abstract

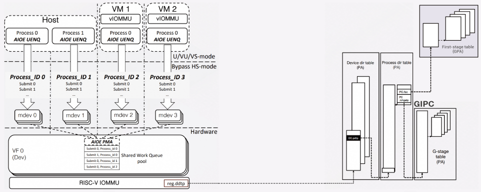
In the era of artificial intelligence, a single CPU architecture can no longer meet the demands of diverse intelligent computing workloads. Consequently, heterogeneous computing has emerged as the mainstream approach. Integrating different Domain-Specific Accelerators (DSAs) into computing systems could enhance overall computational efficiency. For instance, high-dimensional tensor computing tasks can be offloaded to TPUs/NPUs/GPGPUs, while data stream processing tasks are delegated to DPUs. The challenge of efficiently managing DSAs in heterogeneous systems has become a prominent industry focus, driving several technological advancements:

- PCI-e 5.0 and CXL 2.0 have introduced Deferrable Memory Write (DWMr) TLPs
- Armv8.7/9.2 has incorporated ST64BV0 instructions for 64-byte atomic I/O enqueue operations
- The x86 architecture has implemented ENQCMD instructions with comparable functionality

These innovations collectively reduce control latency and optimize system resource utilization.



To help RISC-V adapt to this trend, the presentation introduces the Atomic IO Enqueue Extension (AIOE) and its usage with virtualization. The AIOE extension is designed for RV64 ISA, which includes one PMA definition, two U-mode instructions, two S-mode instructions, a single S-mode CSR, and two envcfg control bits. The presentation also introduces how to use AIOE under the virtualization scenario, which involves the new proposal for RISC-V IOMMU: G-stage table In Process Context (GIPC). With the help of AIOE and GIPC, RISC-V could explore a new heterogeneous programming paradigm from HPC to embedded scenarios.



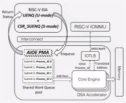
❶ Atomic IO Enqueue (AIOE) Extension

"Atomic IO Enqueue" (AIOE) extension is designed for the RV64 ISA, which contains one PMA definition, two user instructions, two supervisor instructions, one single S-mode CSR, and two envcfg control bits:

AIOE PMA	Atomic IO Enqueue Physical Memory Attribute
CSR_SUENQ	Supervisor Read Write CSR for UENQ instructions
UENQ.64B	User Enqueue Instruction for 64-byte
UENQ.32B	User Enqueue Instruction for 32-byte (Optional)
SENQ.64B	Supervisor Enqueue Instruction for 64-byte
SENQ.32B	Supervisor Enqueue Instruction for 32-byte (Optional)
CSR_MENVCFG.SUENQ	Control bit for SENQ & CSR_SUENQ in S/HS/VS-mode
CSR_HENVCFG.SUENQ	**Control bit for SENQ & CSR_SUENQ in VS-mode (AIOE under Virtualization)**

UENQ.64B (64-byte Atomic IO Store)

Unprivileged Enqueue (UENQ) instruction of the atomic IO single-write 64-Byte with/without the status result. The 64-Byte store data is formed as data [511/255:32]:<CSR_SUENQ>[31:0] from 8 consecutive registers.



CSR_SUENQ (for Process_ID)

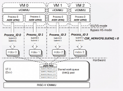
Privileged CSR register that replaces the lowest bits of the store data of UENQ.64B as Process_ID.

AIOE PMA (for Security)

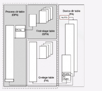
Atomic IO Enqueue (AIOE) Physical Memory Attributes (PMA) defines SENQ and UENQ target address attribute.

❷ AIOE with Virtualization

- ***CSR_HENVCFG.SUENQ = 0***
- The vIOMMU maintains First-stage table
- VMM maintains Device dir tables, Process dir tables, G-stage tables
- ***Process_ID*** distinguishes VMs

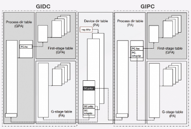


*However, the current RISC-V IOMMU distinguishes VM domains by **Device_ID**.*



G-stage table In Device Context **(GIDC)**

G-stage table In Process Context (GIPC) Extension



G-stage table In Process Context (GIPC) Extension:

- *Process_ID distinguishes VMs*
- *Process dir table is based on PA and maintained by the VMM*
- *No Nested Page Table Walk*
- *No GPA->PA TLB entries*

▶ For more information, please visit ***xrvm.com***