Multi-Level Interrupt Design
in RISC-V Linux

Atish Patra, Principal R&D Engineer

Damien Le Moal, Director, System Software Group
## Why do we need it?

- Low Interrupt Latency is Key for high performant system

<table>
<thead>
<tr>
<th>Design Choice</th>
<th>Pro</th>
<th>Con</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Only</td>
<td>Multiplexing</td>
<td>Higher Interrupt latency</td>
</tr>
<tr>
<td>Local Only</td>
<td>Lower interrupt latency</td>
<td>Hardware complexity</td>
</tr>
<tr>
<td>Multi-level</td>
<td>Lower interrupt latency and multiplexing</td>
<td>Code complexity</td>
</tr>
</tbody>
</table>
Interrupt Design in RISC-V

• Platform Level Interrupt controller (PLIC)
  – Centralized Interrupt prioritization & routing
  – Hart arbitration
  – Sends only a single external interrupt signal to hart
  – All external devices are connected to PLIC

• Local interrupts
  – Timer interrupt, Software Interrupt (Inter Processor Interrupts)
    • Core-Local Interruptor (CLINT) manages control & status registers in U54
    • Local interrupts (0-47) in U54
  – Connected to individual harts directly
  – Minimal latency
Linux view of RISC-V Hardware

U54 Core

Devices

PLIC

HLIC

CPU0

Timer interrupt
Software interrupt
Local interrupt
Local interrupt

CPU3

Timer interrupt
Software interrupt
Local interrupt
Local interrupt

©2018 Western Digital Corporation or its affiliates. All rights reserved.
Interrupt management design in Linux

[Diagram showing the design of interrupt management in Linux, including hardware components like Device, Clock, SW Interrupt, and Local, and kernel components like Arch, Generic Interrupt management core, PLIC irqchip driver, HLIC irqchip driver, SW interrupt handler, clock driver, Device driver, Device driver, and Local interrupt handler driver.]
See you at the Poster Session!!