Boot Interrupts

Author:

- Sean V Kelley <sean.v.kelley@linux.intel.com>

System Message: WARNING/2 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\PCI\[linux-master][Documentation][PCI]boot-interrupts.rst, line 7)

Cannot extract bibliographic field "Author" containing anything other than a single paragraph.

Overview

On PCI Express, interrupts are represented with either MSI or inbound interrupt messages (Assert_INTx/Deassert_INTx). The integrated IO-APIC in a given Core IO converts the legacy interrupt messages from PCI Express to MSI interrupts. If the IO-APIC is disabled (via the mask bits in the IO-APIC table entries), the messages are routed to the legacy PCH. This in-band interrupt mechanism was traditionally necessary for systems that did not support the IO-APIC and for boot. Intel in the past has used the term "boot interrupts" to describe this mechanism. Further, the PCI Express protocol describes this in-band legacy wire-interrupt INTx mechanism for I/O devices to signal PCI-style level interrupts. The subsequent paragraphs describe problems with the Core IO handling of INTx message routing to the PCH and mitigation within BIOS and the OS.

Issue

When in-band legacy INTx messages are forwarded to the PCH, they in turn trigger a new interrupt for which the OS likely lacks a handler. When an interrupt goes unhandled over time, they are tracked by the Linux kernel as Spurious Interrupts. The IRQ will be disabled by the Linux kernel after it reaches a specific count with the error "nobody cared". This disabled IRQ now prevents valid usage by an existing interrupt which may happen to share the IRQ line:

```
irq 19: nobody cared (try booting with the "irqpoll" option)
CPU: 0 PID: 2988 Comm: irq/34-nipalk Tainted: 4.14.87-rt49-02410-g4a640ec-dirty #1
Hardware name: National Instruments NI PXIe-8880/NI PXIe-8880, BIOS 2.1.5f1 01/09/2020
Call Trace:

<IRQ>
 ? dump_stack+0x46/0x5e
 ? __report_bad_irq+0x2e/0xb0
 ? note_interrupt+0x242/0x290
 ? nNIKAL100_memoryRead16+0x8/0x10 [nikal]
 ? handle_irq_event_percpu+0x55/0x70
 ? handle_irq_event+0x4f/0x80
 ? handle_fasteoi_irq+0x81/0x180
 ? handle_irq+0x1c/0x30
 ? do_IRQ+0x41/0xd0
 ? common_interrupt+0x84/0x84
</IRQ>

handlers:
irq_default_primary_handler threaded usb_hcd_irq
Disabling IRQ #19
```

Conditions

The use of threaded interrupts is the most likely condition to trigger this problem today. Threaded interrupts may not be reenabled after the IRQ handler wakes. These "one shot" conditions mean that the threaded interrupt needs to keep the interrupt line masked until the threaded handler has run. Especially when dealing with high data rate interrupts, the thread needs to run to completion; otherwise some handlers will end up in stack overflows since the interrupt of the issuing device is still active.

Affected Chipsets

The legacy interrupt forwarding mechanism exists today in a number of devices including but not limited to chipsets from AMD/ATI, Broadcom, and Intel. Changes made through the mitigations below have been applied to drivers/pci/quirks.c

Starting with ICX there are no longer any IO-APICs in the Core IO's devices. IO-APIC is only in the PCH. Devices connected to the Core IO's PCIe Root Ports will use native MSI/MSI-X mechanisms.

Mitigations

The mitigations take the form of PCI quirks. The preference has been to first identify and make use of a means to disable the routing

to the PCH. In such a case a quirk to disable boot interrupt generation can be added. [1]

IntelÂ® 6300ESB I/O Controller Hub

Alternate Base Address Register:

BIE: Boot Interrupt Enable

0	Boot interrupt is enabled.
1	Boot interrupt is disabled.

IntelÂ® Sandy Bridge through Sky Lake based Xeon servers:

Coherent Interface Protocol Interrupt Control

dis_intx_route2pch/dis_intx_route2ich/dis_intx_route2dmi2:
When this bit is set. Local INTx messages received from the IntelÂ® Quick Data DMA/PCI Express ports are not routed to legacy PCH - they are either converted into MSI via the integrated IO-APIC (if the IO-APIC mask bit is clear in the appropriate entries) or cause no further action (when mask bit is set)

In the absence of a way to directly disable the routing, another approach has been to make use of PCI Interrupt pin to INTx routing tables for purposes of redirecting the interrupt handler to the rerouted interrupt line by default. Therefore, on chipsets where this INTx routing cannot be disabled, the Linux kernel will reroute the valid interrupt to its legacy interrupt. This redirection of the handler will prevent the occurrence of the spurious interrupt detection which would ordinarily disable the IRQ line due to excessive unhandled counts. [2]

The config option X86_REROUTE_FOR_BROKEN_BOOT_IRQS exists to enable (or disable) the redirection of the interrupt handler to the PCH interrupt line. The option can be overridden by either pci=ioapicreroute or pci=noioapicreroute. [3]

More Documentation

There is an overview of the legacy interrupt handling in several datasheets (6300ESB and 6700PXH below). While largely the same, it provides insight into the evolution of its handling with chipsets.

Example of disabling of the boot interrupt

- IntelÂ® 6300ESB I/O Controller Hub (Document # 300641-004US) 5.7.3 Boot Interrupt
  https://www.intel.com/content/dam/doc/datasheet/6300esb-io-controller-hub-datasheet.pdf
- IntelÂ® XeonÂ® Processor E5-1600/2400/2600/4600 v3 Product Families Datasheet - Volume 2: Registers (Document # 330784-003) 6.6.41 cipintrc Coherent Interface Protocol Interrupt Control
  https://www.intel.com/content/dam/www/public/us/en/documents/datasheets/xeon-e5-v3-datasheet-vol-2.pdf

Example of handler rerouting

- IntelÂ® 6700PXH 64-bit PCI Hub (Document # 302628) 2.15.2 PCI Express Legacy INTx Support and Boot Interrupt https://www.intel.com/content/dam/doc/datasheet/6700pxh-64-bit-pci-hub-datasheet.pdf

If you have any legacy PCI interrupt questions that aren't answered, email me.

Cheers,

Sean V Kelley sean.v.kelley@linux.intel.com

[1] https://lore.kernel.org/r/12131949181903-git-send-email-sassmann@suse.de/

[2] https://lore.kernel.org/r/12131949182094-git-send-email-sassmann@suse.de/

[3] https://lore.kernel.org/r/487C8EA7.6020205@suse.de/