8678 Hypervisor GIC SPI 中断测试命令介绍

GRTNBL20230901

Version 0.1 , 2024/10/14

公开 或 内部使用

变更履历表

版本	日期	修订人	修订内容
0.1	2024/10/14	武阳	初始版本

目录

1 Terminology

- VM（virtual machine）虚拟机，通过软件模拟的具有完整硬件系统功能的、运行在一个完全隔离环境中的完整计算机系统。
- PE（Processing element）处理器核心。
- GIC（Generic Interrupt Controller）通用中断控制器，在 Armv8-A 平台上用于处理中断。
- SPI（Shared peripheral interrupts）共享外设中断，分发器可以将其路由到指定的 PE。

2 Overview

MT8678 Hypervisor 针对 SPI 中断，新增一组测试命令，用于模拟外设产生一个 SPI 中断，同时，根据 trace log，可以进一步分析 SPI 中断在 Hypervisor 当中，分发到 VM，所消耗的时间。命令及其使用方法如下：

```
Usage: irq_gen <irq> <repeat> <interval>

irq:      IRQ number (only support IRQ >= 32)
repeat:   How many IRQ will be generated
interval: Delay time (ms) to generate next IRQ
```

3 Hypervisor 中断量测时间点

中断测试工具目前提供四个时间量测点，记录在 ktrace data 里，说明如下：

示例：

```
IRQ-164 [005] .... 269.478592: tracing_mark_write: I|100000001|CPU5-IRQ-164-MANUAL-GEN
IRQ-164 [005] .... 269.478593: tracing_mark_write: B|100000001|CPU5-IRQ-164
IRQ-164 [005] .... 269.478596: tracing_mark_write: E|100000001|CPU5-IRQ-164
IRQ-164 [000] .... 269.478599: tracing_mark_write: I|100000001|INJECT-VIRQ164
```

1. 测试工具通过 GICD 寄存器触发中断：

   I|100000001|CPU5-IRQ-164-MANUAL-GEN

2. Hypervisor 进入中断处理函式：

   B|100000001|CPU5-IRQ-164

3. Hypervisor 结束中断处理函式：

E|100000001|CPU5-IRQ-164

4. Hypervisor 注入虚拟中断并返回 Guest OS：

I|100000001|INJECT-VIRQ164

由以上四个时间点，就可以分析出下列延迟时间数据：

- 硬件产生中断到 Hypervisor 开始处理中断
- Hypervisor 处理中断到注入虚拟中断并返回 Guest OS

以上时间延迟可以量测出 Hypervisor 处理 Guest OS 中断的效能，但返回 Guest OS 之后，需经过多少延迟才能进入 Guest OS 中断处理函式，这一段时间取决于 Guest OS 本身的行为，不包含在本工具的量测范围内。

4 irq_gen 测试命令示例

4.1 SOS 当中确认需要测试的 SPI 中断号码

```
root@FVP:/ cat /proc/interrupts
           CPU0       CPU1       CPU2       CPU3
 11:       1051       1281       1113       1317     GICv3  27 Level     arch_timer
 13:         18          0          0          0     GICv3  42 Level     uart-pl011
 14:          0          0          0          0     GICv3 254 Edge      acrn-hsm
 15:          0          0          0          0     GICv3  15 Edge      trusty
 17:         10          0          0          0     GICv3  36 Level     virtio0
 18:          1          0          0          0     GICv3  37 Level     virtio1
 19:          0          0          0          0     GICv3  38 Level     virtio2
 20:          0          0          0          0     GICv3 232 Level     9002000.irq_generator
 21:          0          0          0          0     GICv3 253 Level     simple-fake-irq
 22:          0          0          0          0     GICv3  34 Level     rtc-pl031
 23:          0          0          0          0     GICv3 380 Level     acrn,irq-notify
 24:          0          0          0          0     GICv3  32 Level     acrn,irq-notify
 25:          0          0          0          0     GICv3 178 Edge      b00000000.nebula_rproc_remote0
 36:          0          0          0          0     GICv3  23 Level     arm-pmu
 37:          0          1          0          0     GICv3 245 Edge      trusty-vqueue-notify
 38:          0          0          0          0 9030000.pl061   3 Edge      GPIO Key Poweroff
IPI0:        19         29         94         55       Rescheduling interrupts
IPI1:       998       1134       1787       1390       Function call interrupts
IPI2:         0          0          0          0       CPU stop interrupts
IPI3:         0          0          0          0       CPU stop (for crash dump) interrupts
IPI4:         0          0          0          0       Timer broadcast interrupts
IPI5:         7          4          5          8       IRQ work interrupts
IPI6:         0          0          0          0       CPU wake-up interrupts
Err:          0
root@FVP:/
```

以 simple-fake-irq 为例，通过 proc 文件系统，可以确认其 hwirq 为 253。

4.2 开启 ktrace

通过 Yocto SOS debugfs 打开 hypervisor IRQ ktrace 记录功能

```
echo irq > /sys/kernel/debug/nebula_trace/ktrace/set_event
echo 1 > /sys/kernel/debug/nebula_trace/ktrace/enable
```

4.3 启动 irq test

从 Yocto SOS 串口或 adb，通过 nblruncmd 工具发送测试命令给 hypervisor

```
nblruncmd -t k irq_gen 253 10 50
```

4.4 关闭 ktrace

通过 Yocto SOS debugfs 关闭 ktrace，并保存 ktrace 记录到/data/ktrace.raw

```
echo 0 > /sys/kernel/debug/nebula_trace/ktrace/enable
```

Ktrace 收集的 raw data 文件会记录在/data/ktrace.raw

4.5 ktrace.raw 解析

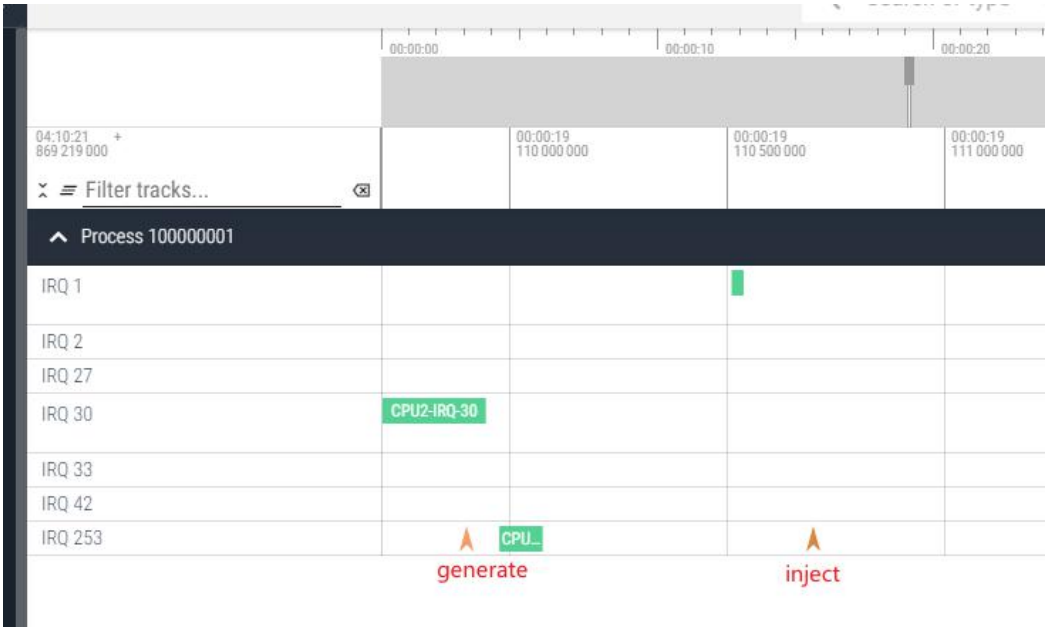
使用 adb pull 命令将/data/ktrace.raw 下载到本机，并通过 parser 工具进行进一步的解析。

```
parser/parse.py -k ktrace.raw -o ktrace.log
```


5 Ktrace.log 日志解析

5.1 Perfetto 展示

https://ui.perfetto.dev



5.2 ktrace.log 展示

D:\Workspace\GICV3\ktrace.log - Notepad++ [Administrator]

文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P) 窗口(W) ?

新建文本文档.txt | ktrace.log

```
1852  IRQ-30 [003] .... 15039.741466: tracing_mark_write: B|100000001|CPU3-IRQ-30
1853  IRQ-30 [003] .... 15039.741549: tracing_mark_write: E|100000001|CPU3-IRQ-30
1854  IRQ-30 [002] .... 15039.776599: tracing_mark_write: B|100000001|CPU2-IRQ-30
1855  IRQ-253 [002] .... 15039.776740: tracing_mark_write: I|100000001|CPU2-IRQ-253-MANUAL-GEN      generate irq
1856  IRQ-30 [002] .... 15039.776797: tracing_mark_write: E|100000001|CPU2-IRQ-30
1857  IRQ-253 [002] .... 15039.776826: tracing_mark_write: B|100000001|CPU2-IRQ-253
1858  IRQ-253 [002] .... 15039.776905: tracing_mark_write: E|100000001|CPU2-IRQ-253      hypervisor handle irq
1859  IRQ-1 [000] .... 15039.777326: tracing_mark_write: B|100000001|CPU0-IRQ-1
1860  IRQ-1 [000] .... 15039.777356: tracing_mark_write: E|100000001|CPU0-IRQ-1
1861  IRQ-253 [000] .... 15039.777524: tracing_mark_write: I|100000001|INJECT-VIRQ253      inject vm
1862  IRQ-30 [003] .... 15039.797095: tracing_mark_write: B|100000001|CPU3-IRQ-30
1863  IRQ-30 [003] .... 15039.797172: tracing_mark_write: E|100000001|CPU3-IRQ-30
1864  IRQ-27 [000] .... 15039.797346: tracing_mark_write: I|100000001|INJECT-VIRQ27
1865  IRQ-30 [003] .... 15039.855052: tracing_mark_write: B|100000001|CPU3-IRQ-30
1866  IRQ-30 [000] .... 15039.855088: tracing_mark_write: B|100000001|CPU0-IRQ-30
1867  IRQ-30 [003] .... 15039.855136: tracing_mark_write: E|100000001|CPU3-IRQ-30
1868  IRQ-30 [000] .... 15039.855281: tracing_mark_write: E|100000001|CPU0-IRQ-30
1869  IRQ-27 [000] .... 15039.855310: tracing_mark_write: I|100000001|INJECT-VIRQ27
1870  IRQ-27 [000] .... 15039.855449: tracing_mark_write: I|100000001|INJECT-VIRQ27
```

