Обработка аппаратных прерываний

На аппаратных платформах, поддерживаемых Windows, внешние прерывания ввода-вывода поступают на одну из линий контроллера прерываний. В свою очередь, контроллер прерывает работу процессора по отдельной линии.

Как только работа процессора будет прервана, этот процессор требует от контроллера запрос прерывания (Interrupt request, IRQ). Контроллер прерываний превращает IRQ в номер прерывания, использует этот номер в качестве индекса в структуре под названием таблица диспетчеризации прерываний (Interrupt dispatch table, IDT) и передает управление соответствующей процедуре обработки прерывания. Во время загрузки системы Windows заполняет IDT указателями на процедуры ядра, обрабатывающими каждое прерывание и исключение.

Windows отображает аппаратные IRQ-запросы на номера прерываний в IDT и также использует IDT для настройки обработчиков системных прерываний для исключений. Например, в системах x86 и x64 номер исключения для ошибки отсутствия страницы (исключения, которое возникает, когда поток пытается обратиться к странице виртуальной памяти, которая не определена или отсутствует) имеет значение 0xe (14).

Таким образом, запись 0xe в IDT указывает на системный обработчик ошибки обращения к странице. Хотя архитектура, поддерживаемая Windows, допускает до 256 записей в IDT-таблице, количество IRQ-запросов, поддерживаемых на конкретной машине, определяется конструкцией используемого контроллера прерываний.

Эксперимент: просмотр IDT

Содержимое IDT, включая информацию о том, какие обработчики системных прерываний назначены операционной системой Windows прерываниям (включая исключения и IRQ-запросы), можно посмотреть, используя команду отладки ядра !idt. Команда !idt без ключей показывает упрощенный вывод, который включает только зарегистрированные аппаратные прерывания (и на 64-разрядных машинах обработчики системных прерываний процессора).

В следующем примере показано, как выглядит вывод команды !idt:

```
lkd> !idt

Dumping IDT:

00: fffff80001a7ec40 nt!KiDivideErrorFault
```

01: fffff80001a7ed40 nt!KiDebugTrapOrFault

02: fffff80001a7ef00 nt!KiNmiInterrupt Stack = 0xFFFFF80001865000

03: fffff80001a7f280 nt!KiBreakpointTrap

04: fffff80001a7f380 nt!KiOverflowTrap

05: fffff80001a7f480 nt!KiBoundFault

06: fffff80001a7f580 nt!KiInvalidOpcodeFault

07: fffff80001a7f7c0 nt!KiNpxNotAvailableFault

08: fffff80001a7f880 nt!KiDoubleFaultAbort Stack = 0xFFFFF80001863000

09: fffff80001a7f940 nt!KiNpxSegmentOverrunAbort

0a: fffff80001a7fa00 nt!KiInvalidTssFault

0b: fffff80001a7fac0 nt!KiSegmentNotPresentFault

0c: fffff80001a7fc00 nt!KiStackFault

0d: fffff80001a7fd40 nt!KiGeneralProtectionFault

0e: fffff80001a7fe80 nt!KiPageFault

10: fffff80001a80240 nt!KiFloatingErrorFault

11: fffff80001a803c0 nt!KiAlignmentFault

12: fffff80001a804c0 nt!KiMcheckAbort Stack = 0xFFFFF80001867000

13: fffff80001a80840 nt!KiXmmException

1f: fffff80001a5ec10 nt!KiApcInterrupt

2c: fffff80001a80a00 nt!KiRaiseAssertion

2d: fffff80001a80b00 nt!KiDebugServiceTrap

2f: fffff80001acd590 nt!KiDpcInterrupt

37: fffff8000201c090 hal!PicSpuriousService37 (KINTERRUPT fffff8000201c000)

3f: fffff8000201c130 hal!PicSpuriousService37 (KINTERRUPT fffff8000201c0a0)

51: fffffa80045babd0 dxgkrnl!DpiFdoLineInterruptRoutine (KINTERRUPT fffffa80045bab40)

52: fffffa80029f1390 USBPORT!USBPORT_InterruptService (KINTERRUPT fffffa80029f1300)

62: fffffa80029f15d0 USBPORT!USBPORT_InterruptService (KINTERRUPT fffffa80029f1540)

USBPORT!USBPORT_InterruptService (KINTERRUPT fffffa80029f1240)

72: fffffa80029f1e10 ataport!IdePortInterrupt (KINTERRUPT fffffa80029f1d80)

81: fffffa80045bae10 i8042prt!I8042KeyboardInterruptService (KINTERRUPT

fffffa80045bad80)

82: fffffa80029f1ed0 ataport!IdePortInterrupt (KINTERRUPT fffffa80029f1e40)

90: fffffa80045bad50 Vid+0x7918 (KINTERRUPT fffffa80045bacc0)

91: fffffa80045baed0 i8042prt!I8042MouseInterruptService (KINTERRUPT fffffa80045bae40)

a0: fffffa80045bac90 vmbus!XPartPncIsr (KINTERRUPT fffffa80045bac00)

a2: fffffa80029f1210 sdbus!SdbusInterrupt (KINTERRUPT fffffa80029f1180)

rimmpx64+0x9FFC (KINTERRUPT fffffa80029f10c0)

rimspx64+0x7A14 (KINTERRUPT fffffa80029f1000)

rixdpx64+0x9C50 (KINTERRUPT fffffa80045baf00)

a3: fffffa80029f1510 USBPORT!USBPORT_InterruptService (KINTERRUPT fffffa80029f1480)

HDAudBus!HdaController::Isr (KINTERRUPT fffffa80029f1c00)

a8: fffffa80029f1bd0 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f1b40)

a9: fffffa80029f1b10 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f1a80)

aa: fffffa80029f1a50 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f19c0)

ab: fffffa80029f1990 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f1900)

ac: fffffa80029f18d0 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f1840)

ad: fffffa80029f1810 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f1780)

ae: fffffa80029f1750 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f16c0)

af: fffffa80029f1690 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f1600)

b0: fffffa80029f1d50 NDIS!ndisMiniportMessageIsr (KINTERRUPT fffffa80029f1cc0)

b1: fffffa80029f1f90 ACPI!ACPIInterruptServiceRoutine (KINTERRUPT fffffa80029f1f00)

b3: fffffa80029f1450 USBPORT!USBPORT_InterruptService (KINTERRUPT fffffa80029f13c0)

c1: fffff8000201c3b0 hal!HalpBroadcastCallService (KINTERRUPT fffff8000201c320)

d1: fffff8000201c450 hal!HalpHpetClockInterrupt (KINTERRUPT fffff8000201c3c0)

d2: fffff8000201c4f0 hal!HalpHpetRolloverInterrupt (KINTERRUPT fffff8000201c460)

df: fffff8000201c310 hal!HalpApicRebootService (KINTERRUPT fffff8000201c280)

e1: fffff80001a8e1f0 nt!KiIpiInterrupt

e2: fffff8000201c270 hal!HalpDeferredRecoveryService (KINTERRUPT fffff8000201c1e0)

e3: fffff8000201c1d0 hal!HalpLocalApicErrorService (KINTERRUPT fffff8000201c140)

fd: fffff8000201c590 hal!HalpProfileInterrupt (KINTERRUPT fffff8000201c500)

fe: fffff8000201c630 hal!HalpPerfInterrupt (KINTERRUPT fffff8000201c5a0)

На системе, предоставившей вывод для данного эксперимента, ISR-процедура клавиатуры в драйвере устройства (I8042prt.sys) назначена прерыванию с номером 0x81. Можно также увидеть, что как ранее было объяснено, прерывание 0xe соответствует системной функции KiPageFault.

Каждый процессор имеет отдельную IDT-таблицу, поэтому другие процессоры, если нужно, могут запускать другие ISR-процедуры. Например, в мультипроцессорной системе прерывание от часов получает каждый процессор, но только один процессор в ответ на это прерывание обновляет показания системных часов.

Тем не менее все процессоры используют прерывание для замера кванта времени потока и для инициации перепланирования по истечении этого кванта.

Кроме того, некоторые настройки системы могут требовать обработку определенных прерываний от устройств конкретным процессором.