**INTERRUPTS**

In Linux, interrupt signals are the distraction which diverts processor to a new activity outside from normal flow of execution. This new activity is called interrupt handler or interrupt service routine (ISR) and that distraction is Interrupts.

do\_IRQ

[\_\_visible](https://elixir.bootlin.com/linux/latest/ident/__visible) unsigned int [\_\_irq\_entry](https://elixir.bootlin.com/linux/latest/ident/__irq_entry) [do\_IRQ](https://elixir.bootlin.com/linux/latest/ident/do_IRQ)(struct [pt\_regs](https://elixir.bootlin.com/linux/latest/ident/pt_regs) \*regs)

{

struct [pt\_regs](https://elixir.bootlin.com/linux/latest/ident/pt_regs) \*old\_regs = [set\_irq\_regs](https://elixir.bootlin.com/linux/latest/ident/set_irq_regs)(regs);

struct [irq\_desc](https://elixir.bootlin.com/linux/latest/ident/irq_desc) \* desc;

/\* high bit used in ret\_from\_ code \*/

unsigned [vector](https://elixir.bootlin.com/linux/latest/ident/vector) = ~regs->orig\_ax;

[entering\_irq](https://elixir.bootlin.com/linux/latest/ident/entering_irq)();

/\* entering\_irq() tells RCU that we're not quiescent. Check it. \*/

[RCU\_LOCKDEP\_WARN](https://elixir.bootlin.com/linux/latest/ident/RCU_LOCKDEP_WARN)(![rcu\_is\_watching](https://elixir.bootlin.com/linux/latest/ident/rcu_is_watching)(), "IRQ failed to wake up RCU");

desc = [\_\_this\_cpu\_read](https://elixir.bootlin.com/linux/latest/ident/__this_cpu_read)([vector\_irq](https://elixir.bootlin.com/linux/latest/ident/vector_irq)[[vector](https://elixir.bootlin.com/linux/latest/ident/vector)]);

if (![handle\_irq](https://elixir.bootlin.com/linux/latest/ident/handle_irq)(desc, regs)) {

[ack\_APIC\_irq](https://elixir.bootlin.com/linux/latest/ident/ack_APIC_irq)();

if (desc != [VECTOR\_RETRIGGERED](https://elixir.bootlin.com/linux/latest/ident/VECTOR_RETRIGGERED)) {

[pr\_emerg\_ratelimited](https://elixir.bootlin.com/linux/latest/ident/pr_emerg_ratelimited)("%s: %d.%d No irq handler for vector\n",

\_\_func\_\_, [smp\_processor\_id](https://elixir.bootlin.com/linux/latest/ident/smp_processor_id)(),

[vector](https://elixir.bootlin.com/linux/latest/ident/vector));

} else {

[\_\_this\_cpu\_write](https://elixir.bootlin.com/linux/latest/ident/__this_cpu_write)([vector\_irq](https://elixir.bootlin.com/linux/latest/ident/vector_irq)[[vector](https://elixir.bootlin.com/linux/latest/ident/vector)], [VECTOR\_UNUSED](https://elixir.bootlin.com/linux/latest/ident/VECTOR_UNUSED));

}

}

[exiting\_irq](https://elixir.bootlin.com/linux/latest/ident/exiting_irq)();

[set\_irq\_regs](https://elixir.bootlin.com/linux/latest/ident/set_irq_regs)(old\_regs);

return 1;

}

| **Vector range** | **Use** |
| --- | --- |
| 0–19 (0x0-0x13) | Nonmaskable interrupts and exceptions |
| 20–31 (0x14-0x1f) | Intel-reserved |
| 32–127 (0x20-0x7f) | External interrupts (IRQs) |
| 128 (0x80) | Programmed exception for system calls |
| 129–238 (0x81-0xee) | External interrupts (IRQs) |
| 239 (0xef) | Local APIC timer interrupt |
| 240 (0xf0) | Local APIC thermal interrupt (introduced in the Pentium 4 models) |
| 241–250 (0xf1-0xfa) | Reserved by Linux for future use |
| 251–253 (0xfb-0xfd) | Interprocessor interrupts |
| 254 (0xfe) | Local APIC error interrupt (generated when the local APIC detects an erroneous condition) |
| 255 (0xff) | Local APIC spurious interrupt (generated if the CPU masks an interrupt while the hardware device raises it) |

| **IRQ** | **INT** | **Hardware device** |
| --- | --- | --- |
| 0 | 32 | Timer |
| 1 | 33 | Keyboard |
| 2 | 34 | PIC cascading |
| 3 | 35 | Second serial port |
| 4 | 36 | First serial port |
| 6 | 38 | Floppy disk |
| 8 | 40 | System clock |
| 10 | 42 | Network interface |
| 11 | 43 | USB port, sound card |
| 12 | 44 | PS/2 mouse |
| 13 | 45 | Mathematical coprocessor |
| 14 | 46 | EIDE disk controller's first chain |
| 15 | 47 | EIDE disk controller's second chain |

Reference:

1) <https://www.oreilly.com/library/view/understanding-the-linux/0596005652/ch04s06.html>

2) <https://notes.shichao.io/lkd/ch7/>