RISC based Interrupts

Fetch next. Inst

↓

Decode Inst.

↓

Execute Inst.

↓

Exit ← Are there ← NO ← Check if there is ← Status Register
anymore   any new interrupt?

Yes

inst?

↓ Yes

Curr. Progm   Check the priority of ← Interrupt Priority List (IPL)
priority higher interrupt against current
                 program.   00 – 40
                                                  ↓   ↘
                   interrupt's   lowest   highest
                   priority is higher.

Production of interrupts
is paused.

↓

Save the content of   **Kernel Stack:**
the microprocessor to   Part of the O/S, which
the kernel stack in   is always loaded in the
RAM.   RAM is kernel.
                                      Kernel has a stack to
                                      hold interrupted job's
                                      data to help system
                                      resume them.

↓

load ISR to serve/   **Interrupt Service Routine**
Handle interrupt.   **(ISR):** An o/s executive
                                          to deal with the
↓   interrupts.

ISR makes use of IDL
to find related program
and gets out of memory,   **Interrupt Dispatch List**
after loading the program   **(IDL):** A list that
                                      holds the associations
↓   of the interrupts with
                                      their related programs.
Interrupt is served

↓

Previous job is loaded
back from kernel stack.

prev. job
starts executing.   ↓

Interrupts are resumed.