Architektúra a správa prerušení v Operačnom Systéme

Operačné systémy

Teoretický Model: Prerušenia vs. Polling

Polling (Cyklické dopytovanie)

CPU aktívne a opakovane kontroluje stav zariadenia v slučke. Táto metóda je neefektívna ("busy-waiting"), pretože procesor plytvá miliónmi cyklov čakaním na pomalé I/O operácie, namiesto vykonávania užitočnej práce.

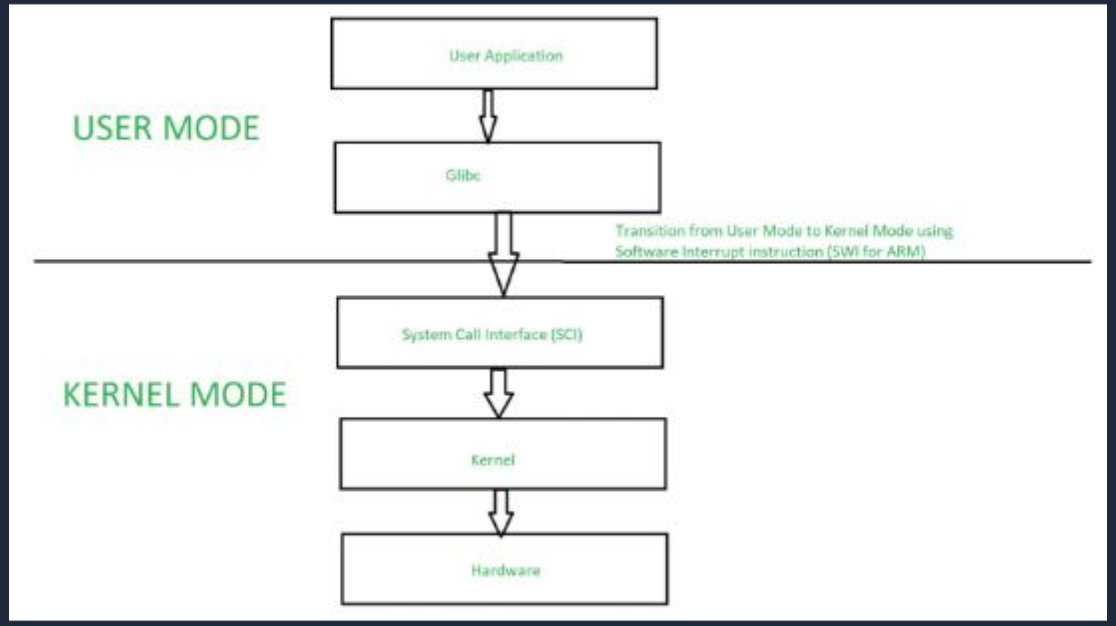
Interrupts (Prerušenia)

Udalosťami riadený model. Hardvér vyšle asynchrónny signál CPU iba vtedy, keď vyžaduje pozornosť. CPU môže medzičasom vykonávať iné procesy, čo maximalizuje priepustnosť systému a odozvu.

Duálny Režim a Ochrana Pamäte

Pre bezpečnosť systému hardvér vynucuje striktné oddelenie režimov:

- **User Mode (Používateľský režim):** Obmedzený prístup k hardvéru. Aplikácie bežia tu.
- **Kernel Mode (Jadrový režim):** Plný prístup k inštrukciám a pamäti. Tu beží obsluha prerušenia (ISR).
- **Prechod (Context Switch):** Prerušenie slúži ako "brána", ktorá atomicky prepne CPU do jadrového režimu cez preddefinovaný vektor.



Taxonómia Systémových Udalostí

Hardvérové Prerušenia

Asynchrónne signály z externých zariadení (NIC, klávesnica). Sú maskovateľné a môžu nastať kedykoľvek počas behu inštrukcií.

Výnimky (Exceptions)

Synchrónne udalosti generované CPU pri chybe (napr. Page Fault, Divide by Zero). Delia sa na Faults (opraviteľné) a Traps (ladiace).

Softvérové Prerušenia

Úmyselne vyvolané inštrukciou (napr. INT 0x80 v x86). Historicky sa používali na implementáciu systémových volaní (System Calls).

Taxonómia Systémových Udalostí

Feature	Hardware Interrupt	Exception (Fault)	Exception (Trap)	Software Interrupt
Source	External Device	CPU Execution	CPU Execution	Instruction (INT n)
Timing	Asynchronous	Synchronous	Synchronous	Synchronous
Return	Next Instruction	Restart Instruction	Next Instruction	Next Instruction
Purpose	I/O Notification	Error Handling	Debugging	System Calls

Evolúcia Hardvéru: Od PIC k APIC

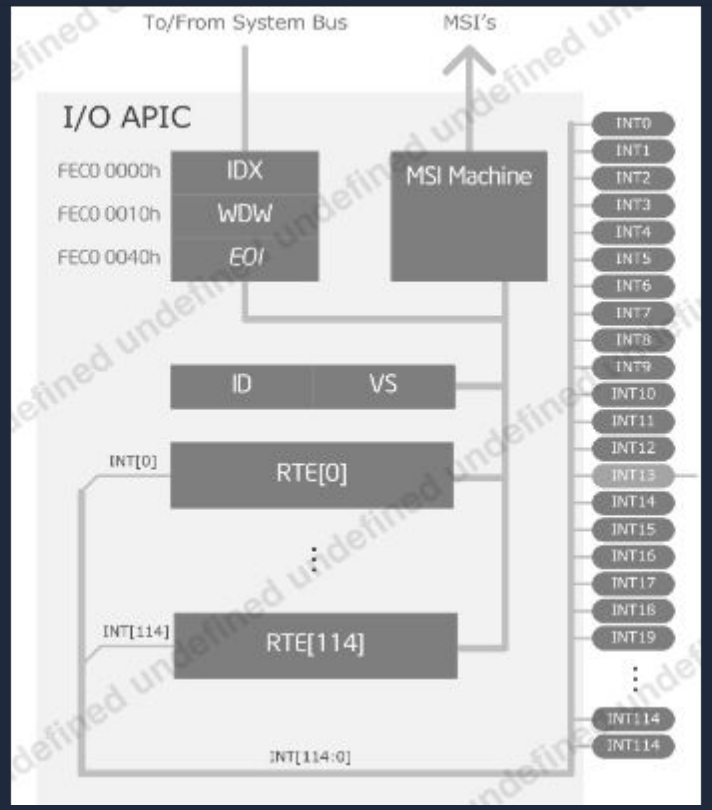
Legacy: 8259A PIC

Pôvodný radič pre jednoprocesorové systémy. Obmedzený počet liniek (15 IRQ). Nedokázal smerovať prerušenia na konkrétne jadrá v multiprocesorových systémoch.

Modern: APIC Architektúra

Pre SMP systémy. Skladá sa z dvoch častí:

- **Local APIC:** Súčasť každého CPU jadra. Prijíma lokálne prerušenia a IPI.
- **I/O APIC:** Pripája periférie a distribuuje signály konkrétnym jadrám (IRQ Affinity).



Message Signaled Interrupts (MSI/MSI-X)

V moderných PCIe zberniciach sú fyzické vodiče pre prerušenia minulosťou. Nahrádzajú ich správy v pamäti.

Výhody MSI-X:

- **In-band signalizácia:** Zápis do pamäte eliminuje potrebu dedikovaných pinov.
- **Škálovateľnosť:** Podpora až 2048 vektorov na zariadenie.
- **Multiqueue Networking:** Každá fronta sieťovej karty (RX/TX Queue) môže mať vlastné prerušenie smerované na iné CPU jadro pre masívny paralelizmus.

Dátové Štruktúry Jadra Linuxu

struct irq_desc

Hlavný deskriptor pre každú IRQ linku. Obsahuje stav, zámky a ukazovateľ na zoznam akcií. Uložené v Radix Tree pre efektívne vyhľadávanie.

struct irq_chip

Abstrakcia hardvérového radiča. Obsahuje nízkoúrovňové metódy ako irq_mask, irq_unmask a irq_ack pre konkrétny čip (napr. IOAPIC, GIC).

struct irqaction

Reprezentuje požiadavku ovládača. Obsahuje ukazovateľ na samotnú funkciu obsluhy (ISR) a flags (napr. IRQF_SHARED pre zdieľanie linky).

Priebeh Spracovania: Top Half

1	2	3	4
Hardware Trigger	**Entry Stub**	**Generic IRQ**	**Driver ISR**
Zariadenie signalizuje APIC, CPU preruší prácu a skočí na vektor v IDT.	Assembly kód uloží registre (kontext) a prepne na jadrový IRQ stack.	Funkcia do_IRQ nájde deskriptor a zavolá handle_irq.	Vykoná sa rýchla obsluha ovládača. Vráti IRQ_WAKE_THREAD ak treba viac času.

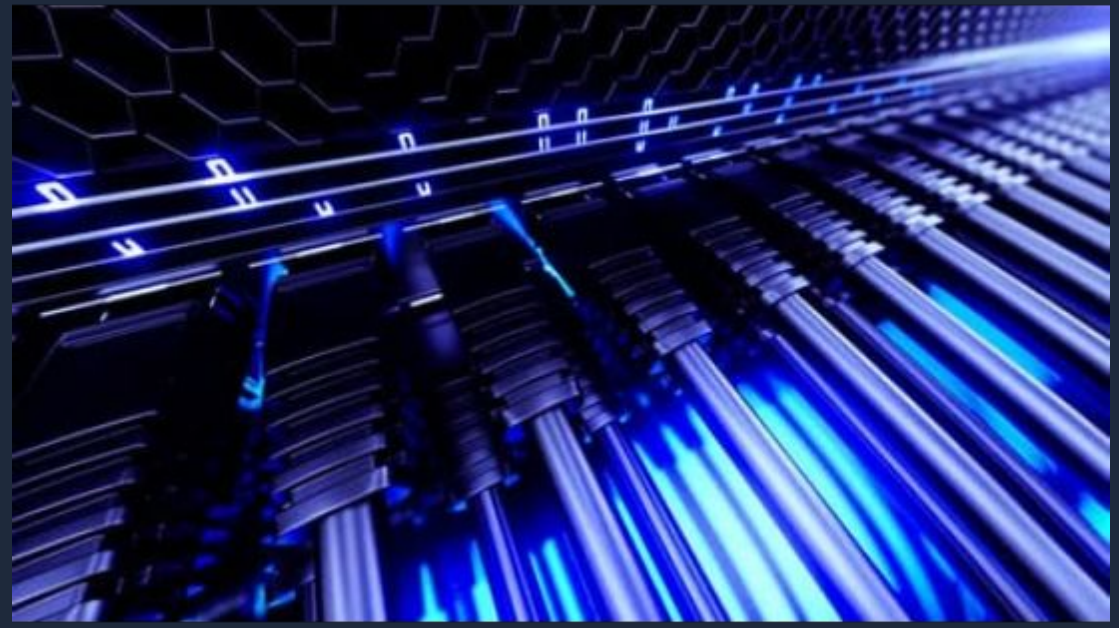
Odložené Spracovanie (Bottom Half)

Mechanizmus	Kontext	Vlastnosti	Použitie
SoftIRQ	Atomic (Interrupt)	Vysoko paralelný, re-entrantný. Beží na viacerých CPU naraz.	Kritické podsystémy (Sieť, Timer, Block IO).
Tasklet	Atomic (Interrupt)	Serializovaný (nebeží paralelne na tom istom type). *Deprecated.*	Staršie ovládače (nahrádzané Threaded IRQ).
Workqueue	Process Context	Môže spať (sleep/block). Plánovaný schedulerom.	Bežné úlohy, alokácia pamäte, I/O operácie.
Threaded IRQ	Process Context	Moderný štandard. Dedikované kernel vlákno pre IRQ.	Väčšina moderných ovládačov, podpora Real-Time.

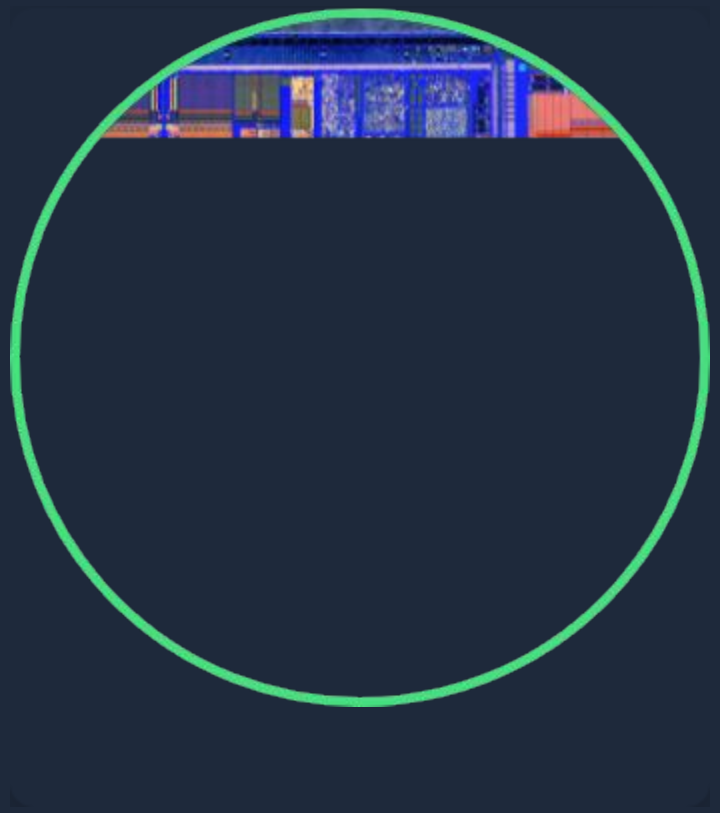
NAPI: Hybridný Model pre Siete

Pri vysokorýchlostných sieťach (10Gb+) by čisté prerušenia zahltili CPU ("Receive Livelock"). NAPI to rieši prepínaním režimov:

- **Prvé prerušenie:** Zobudí ovládač a **zakáže** ďalšie prerušenia na NIC.
- **Polling (SoftIRQ):** Jadro cyklicky vyberá pakety z buffera (batch processing).
- **Complete:** Keď je buffer prázdny alebo budget vyčerpaný, polling končí a prerušenia sa znova povolia.



Výzvy v SMP: IPI a Afinita



Inter-Processor Interrupts (IPI)

Jadra musia komunikovať. Jedno jadro môže poslať IPI inému, napr. pre TLB Shootdown (zneplatnenie cache) alebo prebudenie schedulera.

IRQ Affinity (Smerovanie)

Linux umožňuje nastaviť smp_affinity, čím určí, ktoré jadrá obsluhujú ktoré prerušenia. To je kľúčové pre výkon, aby sa predišlo "cache thrashing" (neustálemu presunu dát medzi L1/L2 cache rôznych jadier).

Záver

- Prerušenia sú základom multitaskingu, umožňujúc asynchrónnu komunikáciu s hardvérom.
- Evolúcia od PIC k MSI-X odráža potrebu škálovateľnosti v moderných serveroch.
- Linux jadro používa sofistikované delenie na Top Half (rýchla odozva) a Bottom Half (ťažká práca).
- Optimalizácie ako NAPI a Threaded IRQs sú nevyhnutné pre výkon a stabilitu systému.