計算機組織與 RISC-V CPU 設計 · 第 21 章

特權架構、CSR 與例外處理

U/S/M 特權模式 / CSR 指令 / 例外與中斷流程 / Sv32 虛擬記憶體

摘要

使用者層指令 (第 18 章) 之外, 處理器還需要一套「管理機器」的機制: 這就是 RISC-V 特權架構 (ISA Manual Volume II)。本章介紹三級特權模式 (M/S/U) 與其分工; 控制與狀態暫存器 **(CSR)** 及六條 Zicsr 指令; 例外與中斷的完整硬體流程 (mtvec/mepc/mcause/mstatus 的角色與 mret 返回); 計時器與軟體中斷 (CLINT)、外部中斷 (PLIC); 以及 **Sv32** 分頁虛擬記憶體如何落實第 8 章的概念。這是把第一部作業系統支援、第二部指令集與第三部硬體實作縫合起來的一章。

目錄

1 特權模式

編碼	模式	角色
11	M(Machine)	最高權限; 所有實作必備; 韌體/SBI 所在層
01	S(Supervisor)	作業系統核心; 管理虛擬記憶體
00	U(User)	應用程式

- 微控制器級: 只有 **M**(或 M+U);Linux 級:M+S+U。
- 各層以「陷入 (trap)」向上、以 `mret`/`sret` 向下;SBI(Supervisor Binary Interface) 是 S 對 M 的呼叫介面 (OpenSBI 韌體)。
- 較高模式可以看到並控制較低模式的一切; 反之不能——保護的根基 (第 8 章)。

2 CSR 與 Zicsr 指令

CSR(Control and Status Register) 有獨立的 12 位元位址空間 (至多 4096 個)。六條指令 (I 型,opcode=SYSTEM):

`csrrw rd,csr,rs1`	原子交換:rd ← csr;csr ← rs1
`csrrs rd,csr,rs1`	讀並置位:rd ← csr;csr ← csr | rs1
`csrrc rd,csr,rs1`	讀並清位:rd ← csr;csr ← csr & ∼rs1
`csrrwi/csrrsi/csrrci`	同上,rs1 換成 5 位元立即值

重要 M 模式 CSR:

CSR	位址	功能
mstatus	0x300	全域狀態:MIE(中斷總開關)、MPIE、MPP(陷入前模式)⋯
misa	0x301	實作的 ISA(RV32IMAC 等)
mie / mip	0x304/0x344	各中斷源的致能/懸置位
mtvec	0x305	陷入向量基底位址 (+ 模式: 直接/向量式)
mepc	0x341	陷入時的 PC(mret 返回目標)
mcause	0x342	陷入原因 (最高位 =1 中斷、=0 例外)
mtval	0x343	附加資訊 (壞位址、壞指令)
mhartid	0xF14	本 hart 編號
cycle/instret	0xC00/0xC02	週期/指令計數器 (Zicntr)

3 例外與中斷的硬體流程

3.1 名詞

例外 **(exception)**: 由指令同步引發 (非法指令、位址不對齊、頁錯失、ecall); 中斷 **(interrupt)**: 非同步外部事件 (計時器、外部裝置、軟體中斷); 陷入 **(trap)**: 兩者引發的控制轉移統稱。

3.2 進入陷阱 (硬體自動完成)

當例外發生或 (MIE=1 且某中斷 pending∧enabled) 時, 以 M 模式陷入為例, 硬體原子地:

1. `mepc` ← 出事的 PC(中斷則為下一條未執行指令);
2. `mcause` ← 原因碼;`mtval` ← 附加資訊;
3. `mstatus.MPIE` ← MIE;`mstatus.MIE` ← 0(關中斷);`mstatus.MPP` ← 陷入前模式;
4. 模式切到 M;PC ← `mtvec`(向量模式下中斷跳 base+4×cause)。

3.3 返回

處理常式以 `mret` 返回:PC ← mepc;MIE ← MPIE; 模式 ← MPP。軟體負責保存/恢復通用暫存器 (對照第 3 章「脈絡保存」與第 12 章「精確例外」)。

最小 M 模式陷阱處理常式

```
    la   t0, trap_handler
    csrrw x0, mtvec, t0        # 設定向量
    csrrsi x0, mstatus, 8      # MIE=1 開總中斷
    ...
trap_handler:
    addi sp, sp, -64           # 保存脈絡 (示意)
    sw   ra, 0(sp)
    sw   t0, 4(sp)
    ...
    csrrs t0, mcause, x0       # 讀原因
    bltz  t0, is_interrupt     # 最高位=1 -> 中斷
    # 例外: 若是 ecall (cause=11), mepc 需 +4 跳過該指令
    csrrs t1, mepc, x0
    addi  t1, t1, 4
    csrrw x0, mepc, t1
is_interrupt:
    ...                        # 分派處理
    lw   ra, 0(sp)
    ...
    addi sp, sp, 64
    mret
```

3.4 標準中斷源

- 軟體中斷 (cause 3): 寫 CLINT 的 msip——核間通知 (IPI);
- 計時器中斷 (cause 7):mtime ≥ mtimecmp 時觸發——OS 排程的心跳 (第 8 章);
- 外部中斷 (cause 11):PLIC 聚合所有裝置中斷, 依優先權裁決後通知 hart;ISR 讀 claim 暫存器領取來源 (第 7 章)。

4 Sv32 虛擬記憶體

S 模式提供 Sv32 分頁 (第 8 章理論的落實):

- `satp` CSR:MODE(0= 關/1=Sv32)+ 根頁表實體頁號;
- 32 位元虛擬位址 = VPN[1](10 位)+ VPN[0](10 位)+ 頁內偏移 (12 位)——二層頁表, 每層 1024 項;
- PTE 含 PPN 與 V/R/W/X/U/G/A/D 位;R=W=X=0 表示指向下一層;
- 4 KiB 頁;VPN[0] 全部落在同層時可形成 4 MiB 大頁 **(megapage)**;
- 頁錯失 → cause 12/13/15(指令/載入/儲存頁錯失) 陷入 S 模式;`sfence.vma` 同步 TLB。

重點觀念

到此, 教材三條線收攏: 第 8 章的「OS 需要什麼」、本章的「ISA 提供什麼」、第 23–26 章的「硬體怎麼做」。我們的教學 CPU 將實作 M 模式的最小子集 (mtvec/mepc/mcause + ecall/mret + 計時器中斷可作為進階習題), 讓你端到端看穿一次 trap 的旅程。

5 本章重點回顧

本章要點

- 三級特權:M(韌體)/S(OS)/U(應用);trap 向上、mret/sret 向下。
- CSR 獨立位址空間;csrrw/s/c(+i) 六條指令原子讀改寫。
- 陷入四步: 存 mepc/mcause → 關中斷存 MPIE/MPP → 切 M 模式 → 跳 mtvec;mret 逆轉。
- CLINT(軟體 + 計時器中斷)、PLIC(外部中斷聚合)。
- Sv32:satp 指根頁表、二層各 10 位、4 KiB 頁; 頁錯失 = 例外 12/13/15。

6 複習問題

1. 例外與中斷的差異? 各舉三例並給出 mcause 值。
2. 為什麼陷入時硬體要自動把 MIE 存到 MPIE 並清 0?
3. ecall 處理完為何要手動把 mepc+4? 頁錯失處理完為何不能 +4?
4. 寫出讀取 mhartid 並判斷「若非 hart 0 則自旋等待」的組語。
5. Sv32 下虛擬位址 0x1234ABCD 的 VPN[1]、VPN[0]、offset 各是多少?
6. 計時器中斷如何支撐分時多工?(串連第 8 章)

參考資料

[1] RISC-V International, *ISA Manual Vol. II: Privileged Architecture*, ver. 20260120.

[2] OpenSBI – RISC-V Open Source Supervisor Binary Interface. https://github.com/riscv-software-src/opensbi

[3] D. Patterson and A. Waterman, *The RISC-V Reader*, Ch. 10.