計算機組織與 RISC-V CPU 設計 · 第 14 章

指令階層平行與超純量處理器

超純量 / 指令相依 / 暫存器重新命名 / 亂序執行 / 現代處理器前瞻

摘要

管線讓多條指令的不同階段重疊; 超純量 **(superscalar)** 更進一步: 每個週期發射多條指令到多個平行的功能單元, 挖掘指令階層平行 **(ILP)**。本章討論超純量設計的核心問題: 哪些相依限制了平行 (真資料相依、反相依、輸出相依、程序相依、資源衝突); 機器平行的三要素 (發射寬度、視窗大小、重新命名); 指令發射政策 (循序/亂序)、暫存器重新命名與重排序緩衝器如何消除假相依並維持精確例外。本章對應 Stallings 第 14 章, 並補充現代視角。

目錄

1 超純量 vs. 超管線

- 超管線 **(superpipelined)**: 把每級再切細, 以更高時脈重疊更多指令——時間軸上更密。
- 超純量 **(superscalar)**: 複製功能單元 (多個 ALU、載入/儲存單元、分支單元), 每週期提取/解碼/發射 2–8 條指令——空間軸上並排。

兩者可疊加。超純量的效益上限由程式本身的 **ILP** 決定: 量測顯示一般整數程式相鄰指令間的可平行度平均僅 2–3, 必須靠大視窗與投機執行去更遠處找平行。

2 限制平行的五種相依

1. 真資料相依 **(RAW)**: 後指令讀前指令的結果——無法消除, 只能前遞或等待。
2. 程序相依 **(**控制相依**)**: 分支之後的指令能否執行取決於分支結果; 變長指令更慘 (解碼前不知邊界)。
3. 資源衝突: 功能單元、埠、匯流排不夠——複製資源可解。
4. 反相依 **(WAR)**: 後指令要寫的暫存器, 前指令還沒讀完——假相依, 源於暫存器名稱重用。
5. 輸出相依 **(WAW)**: 兩指令寫同一暫存器, 順序不能顛倒——亦是假相依。

三種資料相依

```
add x5, x6, x7   # (1)
sub x8, x5, x9   # (2) RAW: 讀 (1) 寫的 x5
or  x6, x10,x11  # (3) WAR: 寫 (1) 要讀的 x6
add x8, x12,x13  # (4) WAW: 與 (2) 都寫 x8
```

(2) 是真相依;(3)(4) 只因「名字」衝突——若 (3) 改寫到別的暫存器、(4) 的結果不覆蓋錯順序, 平行就合法。這就是重新命名的動機。

3 機器平行的三要素

Smith 等人的經典模擬結論:

1. 發射寬度: 同時發射的指令數;
2. 視窗大小: 向前看多少條指令找可發射者;
3. 暫存器重新命名: 消除 WAR/WAW。

三者必須均衡: 沒有重新命名, 加寬發射幾乎無益; 視窗太小, 再多功能單元也餵不飽。此外還需配合高品質分支預測 (第 12 章) 與足夠的記憶體頻寬。

4 指令發射政策

- 循序發射、循序完成: 最簡單; 任何停頓都阻塞後面全部。

- 循序發射、亂序完成: 長延遲指令 (除法、載入未命中) 不擋住後面的獨立指令; 需處理 WAW。
- 亂序發射、亂序完成: 解碼與執行之間設指令視窗; 只要運算元備妥、單元有空就發射。WAR/WAW 以重新命名解決。

4.1 暫存器重新命名

硬體維護一大池實體暫存器, 把 ISA 的結構暫存器動態映射過去: 每次寫入分配新的實體暫存器。上例中 (3) 的 x6 寫入新實體暫存器 P_k, 與 (1) 讀的舊 P_j 無關——WAR 消失。

4.2 重排序緩衝器與精確例外

亂序執行的結果先寫入重排序緩衝器 **(ROB)**, 再按程式順序提交 **(commit)** 到結構狀態。好處: (1) 分支誤預測時, 未提交的投機結果直接丟棄;(2) 精確例外——例外指令之前的都已提交、之後的都可撤銷,OS 看到乾淨的中斷點 (第 3、21 章的脈絡保存因此可行)。

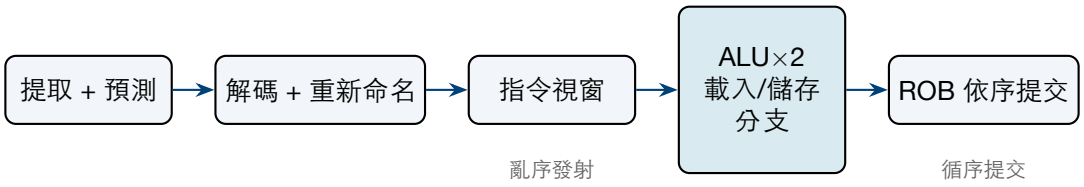

圖 1 現代亂序超純量的骨架: 亂序執行、循序提交

5 現代視角

Stallings 第 6 版以 Pentium 4 與 PowerPC G4 為例; 今日的旗艦核心 (Apple 性能核、Intel P-core、SiFive P870 等 RISC-V 高階核) 規模已達:8–10 寬解碼、300+ 項 ROB、數百實體暫存器——但骨架仍是本章所述。同時, 功耗牆使業界不再單純加寬, 而以「大小核」"每瓦 ILP"取捨;GPU 則選擇放棄 ILP 挖掘、以數千執行緒的資料平行取勝 (第 16 章)。EPIC/IA-64(書中專章) 嘗試把調度全交給編譯器, 商業上失敗, 反證了動態調度的價值。

重點觀念

理解本章的最好方法是對照第 25 章: 我們的教學用五級管線是單發射、循序機器, 它遭遇的前遞與停頓問題, 在超純量中被放大數倍——重新命名、視窗、ROB 都是同一批問題在更寬機器上的系統性解法。

6 本章重點回顧

本章要點

- 超純量 = 每週期多發射; 效益受程式 ILP、分支與記憶體行為限制。
- 五種限制:RAW(真)、控制、資源、WAR/WAW(假, 可用重新命名消除)。
- 機器平行三要素: 發射寬度、視窗、重新命名——必須均衡。
- ROB 循序提交: 支持投機執行回滾與精確例外。

7 複習問題

1. 超純量與超管線的差異? 兩者能否並存?
2. 為何 WAR/WAW 稱為「假相依」? 重新命名如何消除之?
3. 亂序完成為何需要 ROB 才能支援精確例外?
4. 指令視窗大小為何與發射寬度必須匹配?
5. 對序列 `lw x5,0(x6); add x7,x5,x8; mul x9,x10,x11; sub x12,x9,x7` 標注所有相依, 並給出雙發射亂序機器上的最短排程。

參考資料

[1] W. Stallings, *Computer Organization and Architecture*, 6th ed., Chapter 14.

[2] J. E. Smith and G. S. Sohi, “The Microarchitecture of Superscalar Processors,” *Proc. IEEE*, 1995.

[3] R. M. Tomasulo, “An Efficient Algorithm for Exploiting Multiple Arithmetic Units,” *IBM Journal*, 1967.