計算機組織與 RISC-V CPU 設計 · 第 2 章

計算機演進與效能設計

計算機簡史 / 摩爾定律 / 效能平衡 / 效能量測與 Amdahl 定律

摘要

本章回顧計算機的演進歷史——從 ENIAC 與 von Neumann 的 IAS 機器, 經電晶體、積體電路, 到微處理器與現代多核心時代; 並說明貫穿全書的核心議題: 效能設計。我們將介紹摩爾定律及其影響、處理器與記憶體之間日益擴大的速度鴻溝、以及系統設計者為了「平衡」各部件效能所採取的手段。最後介紹效能量測的基本方法: 時脈、CPI、MIPS、基準程式 (benchmark) 與 Amdahl 定律。本章對應 Stallings 第 2 章, 並補充至現代多核心與 RISC-V 的發展。

目錄

1 計算機簡史

1.1 第一代: 真空管 (1946–1957)

ENIAC(Electronic Numerical Integrator And Computer) 是世界上第一部通用電子數位計算機, 由賓州大學的 Mauchly 與 Eckert 設計建造, 用於計算彈道表。1946 年完工時, 它重達 30 噸、占地 1500 平方英尺、使用超過 18,000 支真空管、耗電 140 kW; 每秒可執行 5000 次加法。ENIAC 使用十進位而非二進位, 由 20 個累加器組成, 每個可保存一個 10 位十進位數; 它的程式設計是以手動接線與開關完成的——「寫程式」意味著重新插拔電纜。

1.1.1 von Neumann 機器與儲存程式概念

ENIAC 的程式設計方式極為不便。1945 年, 參與 ENIAC 計畫的數學家 John von Neumann 提出了 **EDVAC** 報告, 闡述儲存程式概念 **(stored-program concept)**: 程式與資料一樣, 以二進位形式儲存在記憶體中, 計算機從記憶體中讀取指令並執行, 修改程式只需改寫記憶體內容。

1946 年,von Neumann 在普林斯頓高等研究院著手設計 **IAS** 計算機 (1952 年完成), 它是所有後續通用計算機的原型。IAS 的結構包含:

- 主記憶體:1000 個字 (word), 每字 40 位元, 儲存資料與指令;
- 算術邏輯單元 **(ALU)**: 對二進位資料進行運算, 含累加器 AC、乘商暫存器 MQ;
- 程式控制單元: 含程式計數器 PC、指令暫存器 IR、記憶體位址暫存器 MAR、記憶體緩衝暫存器 MBR, 負責解譯並執行記憶體中的指令;
- 輸入/輸出設備。

重點觀念

今日幾乎所有計算機仍是 **von Neumann** 機器: 程式與資料共用同一記憶體、循序提取指令執行。你將在第 22–26 章用 Verilog 實作的 RISC-V CPU, 其提取—解碼—執行迴圈與 1952 年的 IAS 在概念上一脈相承。

1.2 第二代: 電晶體 (1958–1964)

1947 年貝爾實驗室發明電晶體, 以固態元件取代真空管, 更小、更便宜、發熱更少。第二代計算機 (如 IBM 7094、DEC PDP-1) 帶來更複雜的 ALU 與控制單元、高階程式語言, 並出現了資料通道 (I/O 專用處理器) 與多工器 (集中管理 I/O) 等系統結構特徵。

1.3 第三代: 積體電路 (1965–1971)

1958 年積體電路 (IC) 發明, 使邏輯閘與記憶體單元能大量製作在同一片矽晶圓上。這一時期的代表是 **IBM System/360**(1964): 史上第一個計畫性的計算機家族——同一結構、多種價位與效能的機型, 軟體完全相容, 奠定了「結構與組織分離」的商業典範; 以及 **DEC PDP-8**: 第一部迷你電腦, 採用匯流排結構 (Omnibus,96 條訊號線), 取代 IBM 的中央交換結構, 此後匯流排結構被幾乎所有迷你與微電腦採用。

1.4 第四代之後:LSI/VLSI 與微處理器 (1972–)

隨著晶片密度提升 (LSI、VLSI、ULSI),1971 年 Intel 推出 **4004**——第一顆把 CPU 完整做在單一晶片上的微處理器 (4 位元)。里程碑包括:8008(1972, 第一顆 8 位元)、8080(1974, 第一顆通用微處理器)、8086(1978,16 位元,x86 結構之始)、80386(1985,32 位元)、Pentium 系列 (1993–, 超純量), 以及 2000 年代的多核心轉向。

記憶體技術的躍遷

1970 年 Fairchild 推出第一顆相對大容量的半導體記憶體 (256 位元);1974 年起, 半導體記憶體的每位元價格降到磁芯記憶體之下, 從此主記憶體全面半導體化。此後晶片密度大約每年增加 **60%**(每三年約四倍),DRAM 容量從 1K 一路演進到現在的數十 Gb。

2 摩爾定律與設計的驅動力

2.1 摩爾定律

Intel 共同創辦人 Gordon Moore 於 1965 年觀察到: 單一晶片上可容納的電晶體數目約每 **18–24** 個月倍增。此趨勢 (摩爾定律) 持續了超過半個世紀, 其後果包括:

1. 晶片成本幾乎不變, 功能卻指數成長, 單位邏輯的成本指數下降;
2. 元件間距縮短, 電氣路徑變短, 操作速度提升;
3. 計算機變小, 應用場域擴大;
4. 供電與散熱需求降低;
5. 晶片內連線比跨晶片焊接可靠, 可靠度提升。

2.2 效能失衡: 處理器—記憶體鴻溝

微處理器速度的成長 (得益於管線、分支預測、超純量、亂序執行等組織技術, 詳見第 12、14 章) 遠快於 DRAM 存取時間的改善, 兩者之間形成速度鴻溝。系統設計者以多種手段緩解:

- 增加每次存取「取回」的位元數 (加寬 DRAM、寬匯流排);
- 在 DRAM 介面加入快取或緩衝 (第 5 章的 SDRAM、advanced DRAM);
- 在處理器與主記憶體之間插入多層快取記憶體 (第 4 章), 降低平均存取延遲;
- 提高互連頻寬 (更高速的匯流排、匯流排階層、交換式互連, 第 3 章)。

注意

I/O 裝置同樣構成瓶頸: 週邊愈來愈快 (千兆網路、NVMe SSD、高解析顯示), 資料搬移需求巨大。解法包括快取與緩衝、更高速的互連、多匯流排階層, 以及 DMA 與 I/O 處理器 (第 7 章)。設計的關鍵永遠是平衡 **(balance)**:CPU、記憶體、互連與 I/O 的吞吐能力必須互相匹配, 系統效能取決於最弱的一環。

2.3 現代的轉折: 功耗牆與多核心

2000 年代中期, 提高時脈頻率遭遇功耗牆: 動態功耗約與頻率成正比、與電壓平方成正比, 散熱成為無法逾越的限制。業界因而轉向:

- 多核心 **(multicore)**: 與其讓單一核心更複雜, 不如在同一晶片放多個較簡單的核心 (第 16 章);
- 專用加速器:GPU、NPU、DSP 等異質運算;
- 能效導向設計: 行動與嵌入式市場使「每瓦效能」成為第一指標。RISC-V 的興起 (2010 年於 UC Berkeley 誕生、2015 年成立基金會) 正是搭上這波浪潮: 精簡、模組化、開放的 ISA 讓任何團隊都能為特定應用打造客製核心。

3 效能量測

3.1 時脈與 CPI

處理器由時脈 **(clock)** 驅動, 頻率 f(Hz) 與週期 $\tau = 1/f$。程式的執行時間可分解為:

$$T = I_c \times \mathrm{CPI} \times \tau$$

其中 I_c 為程式執行的指令數 (instruction count),CPI(cycles per instruction) 為平均每指令週期數。更細地, 若第 i 類指令的週期數為 CPI_i、出現次數為 I_i:

$$\mathrm{CPI} = \frac{\sum_i \mathrm{CPI}_i \times I_i}{I_c}$$

CPI 計算

某程式在 400 MHz 處理器上執行, 指令組成如下: 算術邏輯 60%(CPI=1)、載入/儲存且快取命中 18%(CPI=2)、分支 12%(CPI=4)、快取未命中的記憶體參考 10%(CPI=8)。

$$\mathrm{CPI} = 0.6(1) + 0.18(2) + 0.12(4) + 0.10(8) = 2.24$$

$$\mathrm{MIPS} = \frac{f}{\mathrm{CPI} \times 10^6} = \frac{400 \times 10^6}{2.24 \times 10^6} \approx 178$$

3.2 MIPS 與 MFLOPS 的陷阱

MIPS(每秒百萬指令) 易受指令集影響: 同一高階程式, 在 CISC 機器上編譯出較少但較複雜的指令, 在 RISC 機器上編譯出較多但較簡單的指令;MIPS 值高不代表程式跑得快。MFLOPS 只對浮點密集應用有意義。因此業界改用基準程式 **(benchmark)**:

- **SPEC CPU**: 以真實應用程式組成的套件, 量測相對於參考機器的比值, 幾何平均後得到 SPECspeed / SPECrate;
- 嵌入式領域:CoreMark、Dhrystone(DMIPS/MHz 仍常見於 RISC-V 核心的規格表);
- 系統層:TPC(交易處理)、MLPerf(機器學習) 等。

3.3 Amdahl 定律

若程式中可平行化 (或可加速) 的比例為 f, 該部分獲得 N 倍加速, 整體加速比為:

$$\text{Speedup} = \frac{1}{(1-f) + \dfrac{f}{N}}$$

當 $N \to \infty$, 加速比上限為 $1/(1-f)$。

Amdahl 定律

程式有 90% 的執行時間可平行化 ($f = 0.9$)。使用 10 顆處理器:Speedup $= 1/(0.1 + 0.09) \approx 5.3$; 使用 100 顆:$1/(0.1 + 0.009) \approx 9.2$; 即使無限多顆, 也不超過 10。串行部分決定天花板——這是多核心時代最重要的一課。

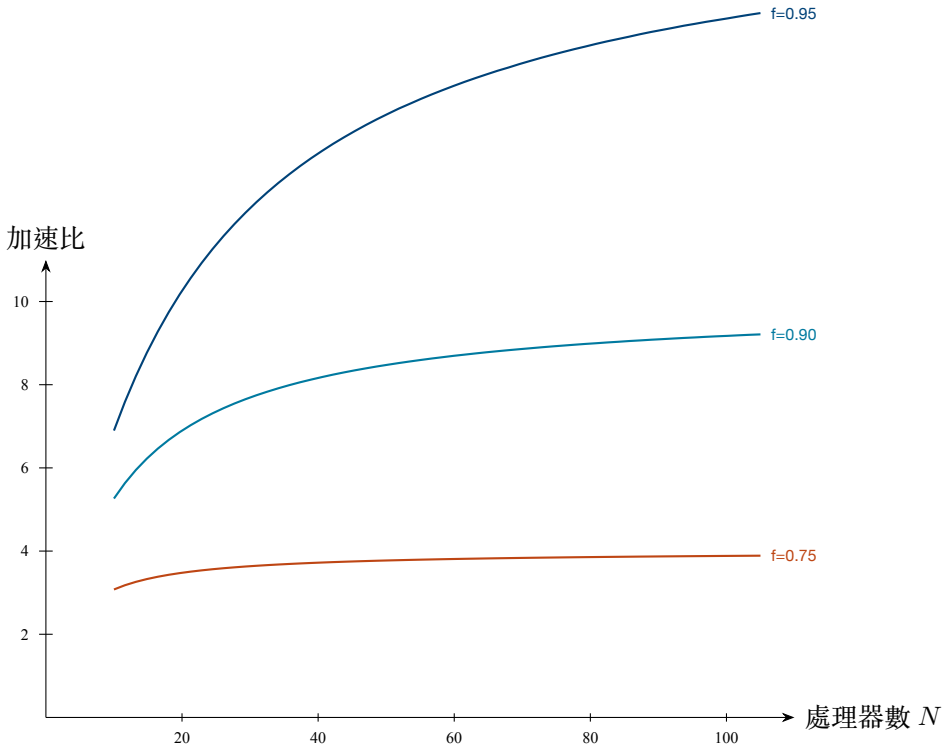

圖 1 Amdahl 定律: 串行比例限制了可達到的加速比

4 本章重點回顧

本章要點

- 儲存程式概念 (von Neumann / IAS) 是現代計算機的基礎: 程式與資料同存於記憶體。
- 世代演進: 真空管 → 電晶體 → IC → VLSI 微處理器 → 多核心/異質運算。
- 摩爾定律驅動密度指數成長; 處理器—記憶體速度鴻溝催生了快取階層與高速互連。
- 設計核心是平衡:CPU、記憶體、互連、I/O 的吞吐必須匹配。
- $T = I_c \times \text{CPI} \times \tau$; 基準程式比 MIPS 更能反映真實效能;Amdahl 定律給出加速上限

$1/(1-f)$。

5 複習問題

1. 什麼是儲存程式概念? 它解決了 ENIAC 的什麼問題?
2. 列出 IAS 計算機的四大部件, 並與第 1 章的計算機頂層結構對照。
3. 摩爾定律的內容為何? 列舉三項它帶來的後果。
4. 什麼是處理器—記憶體速度鴻溝? 列舉三種緩解手段。
5. 某 2 GHz 處理器執行一程式:50% 指令 CPI=1、30% CPI=2、20% CPI=4。求平均 CPI、MIPS, 以及執行 10^9 條指令所需時間。
6. 為什麼 MIPS 不適合跨結構比較效能?SPEC 基準如何解決此問題?
7. 程式的串行比例為 5%, 依 Amdahl 定律,16 核與 256 核的加速比各是多少? 極限是多少?

參考資料

[1] W. Stallings, *Computer Organization and Architecture*, 6th ed., Chapter 2.

[2] G. E. Moore, “Cramming more components onto integrated circuits,” *Electronics*, vol. 38, no. 8, 1965.

[3] Standard Performance Evaluation Corporation, https://www.spec.org

[4] G. M. Amdahl, “Validity of the single processor approach to achieving large scale computing capabilities,” *AFIPS 1967*.