ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
Phạm Quốc Cường

KIẾN TRÚC MÁY TÍNH

NHÀ XUẤT BẢN ĐẠI HỌC QUỐC GIA
TP HỒ CHÍ MINH - 2017

Software requires hardware to be able to perform some work, while hardware is able to do some work without software.

LỜI NÓI ĐẦU

Sự ra đời của "Máy tính" đã dẫn đến cuộc cách mạng văn minh lần thứ ba, cách mạng thông tin (information revolution), bên cạnh hai cuộc cách mạng nông nghiệp và công nghiệp trước đó. Trước khi có sự ra đời của máy tính, những ứng dụng rất bình thường ngày nay như ứng dụng tìm kiếm thông tin, ứng dụng mạng internet, ứng dụng điện thoại di động,... đều được xem là các ứng dụng viễn tưởng. Với sự ra đời và phát triển mạnh mẽ của các máy tính trong hơn bảy thập kỷ qua, các ứng dụng này không những trở thành sự thật mà còn phổ biến trong đời sống xã hội.

Sách *"Kiến trúc Máy tính"* này được viết ra với mục đích phục vụ cho các độc giả học tập và nghiên cứu các vấn đề xoay quanh kiến trúc, hoạt động và đánh giá hiệu suất các máy tính. Như vậy có thể thấy rằng, đối tượng khảo sát chính trong sách này là "máy tính". Vậy máy tính là gì? Theo định nghĩa của từ điển Cambridge[1] thì máy tính (chính xác phải gọi là "máy tính điện tử đa dụng" - computer) là "một **máy điện tử** được sử dụng để lưu trữ, tổ chức và tìm kiếm các từ, số và hình ảnh nhằm mục đích tính toán và điều khiển các máy khác". Một định nghĩa đơn giản hơn có thể được dùng để định nghĩa máy tính là "máy tính là một **thiết bị điện tử đa dụng** có thể được lập trình để thực hiện một tập hợp các tác vụ số học hoặc luận lý một cách **tự động**". Điều này có nghĩa rằng, bất kỳ thiết bị nào thỏa mãn định nghĩa trên đều có thể được gọi là máy tính. Vì vậy, dựa theo định nghĩa này, máy tính ngày nay có muôn hình vạn trạng và có nhiều loại khác nhau. Chi tiết về các loại máy tính khác nhau sẽ được trình bày trong sách này.

Nội dung chính của quyển sách này sẽ xoay quanh chủ đề "kiến trúc" của một máy tính, vậy "kiến trúc máy tính" là gì? "kiến trúc máy tính" được định nghĩa là "việc lựa chọn và kết nối các thành phần phần cứng

[1] phiên bản online tại "http://dictionary.cambridge.org/"

một cách khoa học và nghệ thuật nhằm tạo nên các máy tính đáp ứng được yêu cầu về chức năng, hiệu suất và giá thành"[2]. Do đó, nội dung chính của quyển sách sẽ xoay quanh các chủ đề *Đánh giá hiệu suất máy tính*, *Kiến trúc tập lệnh*, *Máy tính số học*, *Kiến trúc bộ xử lý* và *Kiến trúc phân cấp bộ xử lý*. Các chủ đề này sẽ được trải dài trong 5 chương. Ở mỗi chương sẽ có các bài tập cũng cố kiến thức sau mỗi chương. Với 5 chương xoay quanh các chủ đề vừa nêu, mục tiêu chính của quyển sách này là sẽ giúp người đọc hiểu được cấu trúc và tổ chức của một hệ thống máy tính cũng như những nguyên tắc hoạt động cơ bản của nó. Ngoài ra, đối với người đọc có kiến thức về các **ngôn ngữ đặc tả phần cứng**, quyển sách này có thể giúp người đọc thiết kế và hiện thực được các khối chức năng cơ bản của một hệ thống máy tính từ đó xây dựng nên một hệ thống máy tính đơn giản dùng các board mạch khả cấu hình.

Trong quá trình hoàn thiện quyển sách, tác giả đã nhận được sự giúp đỡ của rất nhiều đồng nghiệp cũng như Ban Chủ nhiệm Khoa Khoa học và Kỹ thuật Máy tính, Trường Đại học Bách Khoa - ĐHQG-HCM. Xin chân thành gửi lời cảm ơn đến các đồng nghiệp trong Khoa và Ban Chủ nhiệm khoa. Tác giả cũng xin chân thành gửi lời cảm ơn đến các tác giả của các tài liệu tham khảo đã cung cấp những thông tin quý báu giúp hoàn thành quyển sách này.

Trong quá trình biên soạn sách chắc chắn sẽ không thể tránh khỏi những thiếu sót. Tác giả rất mong nhận được sự đóng góp của người đọc. Mọi sự đóng góp xin vui lòng gửi về:

Phạm Quốc Cường
Khoa Khoa học và Kỹ thuật Máy tính, Trường Đại học Bách Khoa
268 Lý Thường Kiệt, Phường 14, Quận 10, TP.HCM
Điện thoại: (08)386487256 - NB: 5843
Email: cuongpham@hcmut.edu.vn

Phạm Quốc Cường
TP.HCM, tháng 12 năm 2016

[2]Định nghĩa tại trang web WWW Computer Architecture Page - `http://pages.cs.wisc.edu/~arch/www/`

TÓM TẮT

Sách *"Kiến trúc Máy tính"* bao gồm 5 chương mỗi chương gồm hai phần chính là nội dung của chương đó và phần bài tập nhằm củng cố kiến thức. Nội dung chính của các chương như sau:

Chương 1: **Các vấn đề cơ bản trong thiết kế máy tính**. Chương này tập trung chủ yếu vào việc giới thiệu các công nghệ và các thế hệ máy tính từ khi ra đời vào năm 1946 đến nay. Để có thể so sánh được sức mạnh tính toán của các máy tính trong cùng một thế hệ hoặc giữa các thế hệ cần phải so sánh hiệu suất tính toán của chúng. Chương này trình bày cách tính toán đo đạc hiệu suất cũng như công suất tiêu thụ của các hệ thống máy tính.

Chương **??**: **Kiến trúc tập lệnh**. Chương này sẽ giới thiệu bốn nguyên tắc thiết kế tập lệnh của một hệ thống máy tính. Tập lệnh MIPS sẽ được lấy làm ví dụ và trình bày chi tiết ba nhóm lệnh là *nhóm lệnh số học và luận lý, nhóm lệnh chuyển dữ liệu* và *nhóm lệnh hỗ trợ ra quyết định*. Tuy nhiên, máy tính chỉ có thể hiểu và thực thi được các lệnh máy (là chuỗi các ký số nhị phân). Do đó, chương này cũng trình bày phương pháp mã hóa và các định dạng lệnh máy trong kiến trúc MIPS.

Chương **??**: **Bộ tính toán số học**. Chương **??** trình bày các lệnh tính toán số học trong tập lệnh MIPS với các giá trị số nguyên. Tuy nhiên, các bài toán trong thực tế không chỉ đơn giản cần các số nguyên và các phép toán đơn giản như trên. Do đó, cần phải có những phép tính khác đối với số nguyên như phép tính nhân và chia. Ngoài ra, cần phải có những cách biểu diễn số thực và các phép toán liên quan đến số thực. Mục tiêu chính của Chương **??** này sẽ trình bày các vấn đề sau: các lệnh nhân và chia trong tập lệnh MIPS chuẩn cũng như kiến trúc phần cứng cho việc tính toán phép nhân và chia. Biểu diễn số thực trong máy tính và các lệnh xử lý số thực trong tập lệnh MIPS cũng sẽ được trình bày trong chương này.

Chương **??**: **Bộ xử lý**. Chương này sẽ trình bày hai dạng hiện thực khác nhau của máy tính theo kiến trúc MIPS. Cách hiện thực thứ nhất khá đơn giản khi mà ở đó mỗi lệnh sẽ được hoàn thành trong một chu kỳ xung nhịp. Điều này có nghĩa là chu kỳ xung nhịp phải đủ dài để thời gian thực thi của tất cả các giai đoạn khi thực thi một lệnh bất kỳ phải nhỏ hơn hoặc bằng thời gian một chu kỳ. Mặc dù cách hiện thực này đơn giản nhưng hiệu suất không cao và chỉ mang tính chất tham khảo. Cách hiện thực thứ hai là cách hiện thực được sử dụng nhiều trong thực tế hơn, đó là cách hiện thực theo cơ chế xử lý ống (pipeline). Ở cách hiện thực này, quá trình thực thi một lệnh sẽ được chia thành nhiều giai đoạn khác nhau và mỗi giai đoạn sẽ được hoàn tất trong một chu kỳ. Tuy nhiên, tại một thời điểm sẽ có nhiều giai đoạn của nhiều lệnh khác nhau cùng được thực thi song song. Do đó, hiệu suất của cách hiện thực theo cơ chế xử lý ống sẽ cao hơn so với cách hiện thực theo mô hình đơn giản.

Chương **??**: **Hệ thống bộ nhớ phân cấp**. Tổ chức bộ nhớ ảnh hưởng đến hiệu suất của toàn hệ thống bộ nhớ. Do đó, tổ chức bộ nhớ tốt sẽ góp phần tăng đáng kể hiệu suất hệ thống. Bộ nhớ trong một hệ thống máy tính sẽ được thành nhiều lớp với kích thước và tốc độ truy xuất của các lớp khác nhau. Tương ứng với tốc độ truy xuất cao thì công nghệ hiện thực có giá thành cao hơn. Chương này sẽ trình bày các công nghệ hiện thực bộ nhớ khác nhau hiện đang được sử dụng. Dựa vào những công nghệ này, bộ nhớ máy tính sẽ được tổ chức theo mô hình phân cấp với các phương pháp tiếp cận khác nhau nhằm đạt được hiệu suất xử lý cao nhất với giá thành hợp lý nhất.

Mục lục

DANH SÁCH HÌNH VẼ

DANH SÁCH BẢNG

1

CÁC VẤN ĐỀ CƠ BẢN TRONG THIẾT KẾ MÁY TÍNH

1.1. GIỚI THIỆU

Được giới thiệu lần đầu tiên vào đầu năm 1946, trải qua hơn 7 thập kỷ quá trình phát triển của các máy tính cũng như các hệ thống tính toán vẫn đang tiếp tục phát triển liên tục. Nhu cầu tính toán ngày một gia tăng đặc biệt là khi con người bước vào kỷ nguyên dữ liệu lớn (big data era). Chương 1 này sẽ giới thiệu sơ lược về lịch sử hình thành và phát triển các máy tính. Mặc dù kiến trúc và công nghệ chế tạo các máy tính từ dùng trong các thiết bị gia dụng cho đến điện thoại thông minh hay các hệ thống tính toán hiệu năng cao là tương đồng, nhưng các ứng dụng tính toán khác nhau sẽ có những yêu cầu thiết kế khác nhau. Do đó, Chương 1 cũng sẽ trình bày cách phân loại các máy tính trong các hệ thống hệ thống tính toán từ đơn giản đến phức tạp. Sự phát triển của các máy tính cũng như các hệ thống tính toán không thể tách rời sự phát triển của các công nghệ chế tạo phần cứng, do vậy tóm tắt về các công nghệ chế tạo cho bộ xử lý và bộ nhớ của các máy tính cũng sẽ được giới thiệu trong chương này.

1

Đối với người sử dụng máy tính, hai tham số quan trọng của máy tính là hiệu suất và công suất. Chương này sẽ trình bày các yếu tố ảnh hưởng đến hiệu suất máy tính cũng như các cách tính toán hiệu suất máy tính và so sánh sức mạnh tính toán giữa các máy tính và hệ thống tính toán với nhau. Các phân tích về công suất tiêu thụ cũng sẽ được trình bày trong chương này.

1.2. LỊCH SỬ PHÁT TRIỂN VÀ PHÂN LOẠI MÁY TÍNH

Được giới thiệu lần đầu tiên vào năm 1946 và trải qua hơn 7 thập kỷ phát triển, các "máy tính điện tử đa dụng" (từ đây về sau sẽ được gọi chung là "máy tính") ngày càng có các bộ xử lý hoạt động ở tốc độ cao hơn, kích thước các thành phần ngày càng giảm, dung lượng bộ nhớ tăng và khả năng xử lý cũng như tốc độ các thiết bị ngoại vi cũng tăng. Ngày nay, máy tính xuất hiện trong hầu hết tất cả các ngành công nghiệp, trong mọi mặt của đời sống và có trong rất nhiều thiết bị phục vụ cuộc sống con người cũng như các nhu cầu khác nhau của con người.

1.2.1. LỊCH SỬ PHÁT TRIỂN

Trải qua quá trình phát triển liên tục trong hơn 6 thập niên, các máy tính có thể được chia thành 4 thế hệ khác nhau.[1]

MÁY TÍNH THẾ HỆ THỨ NHẤT

Thế hệ máy tính thứ nhất, thế hệ máy tính sử dụng các **đèn chân không** (vacuum tubes) được bắt đầu năm 1946 khi máy tính ENIAC được sản xuất tại trường Đại học Pennsylvania, Hoa Kỳ. Bóng đèn chân không là các thiết bị điều khiển dòng điện giữa các cực được chứa trong các thiết bị chân không như trong Hình 1.1. Các máy tính thế hệ thứ nhất có kích thước và khối lượng rất lớn. Máy tính ENIAC dùng đến 18.000 bóng đèn chân không, có khối lượng lên đến 30 tấn và có kích thước 1.400 m^2. Công suất tiêu thụ của máy tính ENIAC cũng rất lớn khi lên đến 140 kW. Mặc dù có khối lượng và kích thước rất lớn nếu so sánh với các máy tính cá

[1] Thế hệ thứ tư hiện vẫn còn nhiều ý kiến tranh cãi. Nhiều nhà khoa học phân chia thế hệ thứ tư thành các thế hệ khác nhau. Tuy nhiên, trong sách này tác giả chọn cách phân chia các máy tính thành 4 thế hệ, dựa trên công nghệ chế tạo.

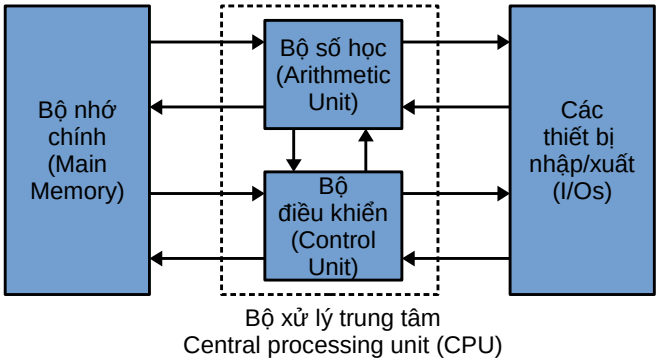
nhân hiện nay, nhưng máy tính ENIAC chỉ có thể xử lý 5.000 phép cộng trong một giây. Khác với các máy tính thế hệ sau tính toán trên hệ đếm nhị phân (binary), máy tính ENIAC tính toán dựa trên hệ đếm thập phân (decimal).

Hình 1.1: Bóng đèn chân không. Nguồn hình ảnh từ Internet.

Máy tính ENIAC được lập trình bằng cách thay đổi các công tắc, quá trình này rất phức tạp và nhàm chán. Quá trình lập trình sẽ hiệu quả hơn nếu như các chương trình có thể được biểu diễn dưới dạng phù hợp cho việc lưu trữ nó trong bộ nhớ. Một máy tính sẽ đọc các *lệnh* (instructions) từ bộ nhớ và thực thi chúng. Các chương trình sẽ được cài đặt hoặc thay đổi bằng cách thay đổi một phần bộ nhớ. Ý tưởng này được gọi là *mô hình lưu trữ chương trình* (*stored-program concept*) được phát triển bởi John von Neumann năm 1945 và được dùng cho đến ngày nay. Hiện tại, hầu hết các máy tính đều sử dụng mô hình lưu trữ chương trình hay còn được gọi là mô hình **von Neumann**.

Hình 1.2 trình bày sơ đồ tổng quát của mô hình máy tính von Neumann. Theo mô hình này, một máy tính sẽ có ba thành phần chính là:

- **Bộ nhớ chính** (main memory): là nơi lưu trữ các lệnh của các chương trình và dữ liệu;
- **Bộ xử lý trung tâm** (Central processing unit - CPU): bao gồm hai thành phần là **Bộ xử lý số học** (Arithmetic unit) có nhiệm vụ tính toán trên dữ liệu nhị phân và **Bộ điều khiển** (Control unit) có nhiệm vụ diễn giải các lệnh đọc được từ bộ nhớ và thực thi chúng; và



Hình 1.2: Mô hình máy tính von Neumann.

- **Các thiết bị nhập xuất** (Inputs/Outputs): dùng để nhận dữ liệu từ người sử dụng hoặc xuất kết quả cho người sử dụng dưới sự điều khiển của Bộ điều khiển.

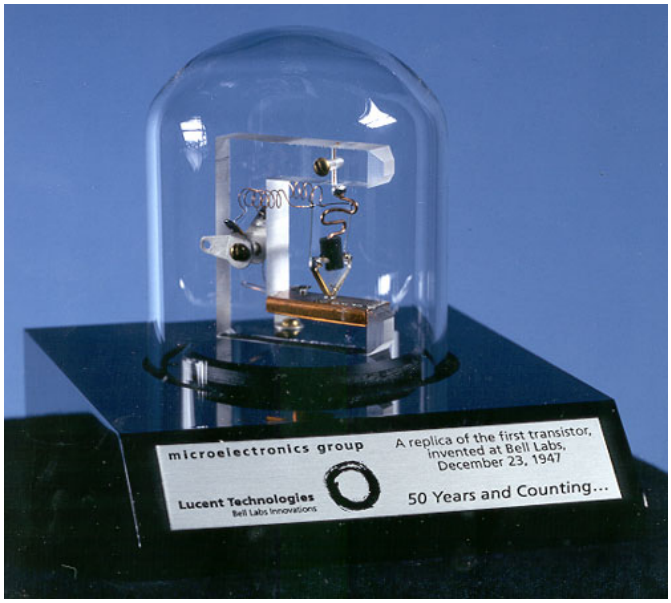
Năm 1946, John von Neumann và các cộng sự tại Viện Nghiên cứu Nâng cao tại Princeton, Hoa Kỳ bắt đầu thiết kế máy tính IAS dựa trên mô hình lưu trữ chương trình này. Mặc dù phải đến năm 1951, máy tính IAS mới được hoàn thành nhưng nó được xem như là một bản mẫu (prototype) cho tất cả các máy tính đa dụng thế hệ sau. Máy tính IAS có bộ nhớ gồm 1000 vị trí lưu trữ, còn được gọi là *từ nhớ* [2] (word) có kích thước 40 bit mỗi từ nhớ dùng để lưu trữ lệnh của các chương trình và dữ liệu. Khác với máy tính ENIAC, máy tính IAS xử lý dữ liệu kiểu nhị phân.

Trong thế hệ máy tính thứ nhất này, hai dòng máy tính được thương mại hóa là UNIVAC (Universal Automatic Computer) và IBM. UNIVAC trong giai đoạn này đã cho ra đời các dòng máy tính UNIVAC I, UNIVAC II, và loạt máy UNIVAC 1100. IBM giới thiệu hai máy tính 701 dùng cho mục đích tính toán khoa học vào năm 1953 và máy tính 702 dùng cho mục đích tính toán các ứng dụng thương mại vào năm 1955. Đây là hai máy tính đầu tiên của dòng máy IBM 700 đã làm nên tên tuổi của IBM trong lĩnh vực sản xuất máy tính.

[2] Không có một định nghĩa chính xác cho thuật ngữ *từ nhớ*. Tổng quát, một từ nhớ là tập hợp có thứ tự các bit và là đơn vị để lưu trữ, truyền đạt và xử lý thông tin trong một máy tính cụ thể. Thông thường một từ nhớ sẽ có kích thước bằng với kích thước lệnh trong các bộ xử lý có kích thước lệnh cố định.

MÁY TÍNH THẾ HỆ THỨ HAI

Sự thay đổi lớn đầu tiên trong các máy tính điện tử đa dụng là sự thay thế các bóng đèn chân không bằng các **bóng bán dẫn** (transistor). So với các bóng đèn chân không thì các bóng bán dẫn có kích thước nhỏ hơn, giá thành rẻ hơn và sản sinh ra ít nhiệt lượng khi hoạt động hơn. Mặc dù được phát triển lần đầu tiên vào năm 1947 bởi phòng thí nghiệm Bell Labs (Hình 1.3), nhưng mãi đến cuối những năm 1950 thì máy tính điện tử đa dụng chế tạo hoàn toàn bằng các bóng bán dẫn mới được thương mại hóa. Năm 1957 được xem là năm kết thúc của máy tính thế hệ thứ nhất để chuyển sang máy tính thế hệ thứ hai, máy tính được chế tạo bằng các bóng bán dẫn. Việc sử dụng các bóng bán dẫn để chế tạo máy tính đánh dấu sự bắt đầu của thế hệ máy tính thứ hai vào năm 1958. Những máy tính đầu tiên trong thế hệ này chứa khoảng 10.000 bóng bán dẫn và tăng dần đến hàng trăm ngàn bóng bán dẫn ở các máy tính ra đời sau trong cùng thế hệ.



Hình 1.3: Một phiên bản của bóng bán dẫn đầu tiên trên thế giới được phát triển bởi Bell Labs năm 1957.

Hai nhà sản xuất đi tiên phong trong việc giới thiệu và ứng dụng công

nghệ mới ở giai đoạn này là NCR và RCA. Không lâu sau đó, IBM giới thiệu dòng máy tính 7000. So với dòng máy tính 700 ở thế hệ thứ nhất, các máy tính thuộc dòng 7000 ở thế hệ này của IBM có hiệu suất và dung lượng bộ nhớ tăng đáng kể và kích thước nhỏ hơn nhiều. Kết quả so sánh này cũng đúng khi so sánh giữa các máy được sản xuất trước và sau trong cùng dòng IBM 7000. Điều này được minh họa qua Bảng 1.1, so sánh các máy tính IBM ở hai thế hệ thứ nhất và thứ hai. Ngoài ra, thế hệ máy tính thứ hai còn chứng kiến các sự thay đổi khác như sự xuất hiện của các bộ tính toán số học và luận lý (arithmetic and logic units) cũng như các bộ điều khiển (control units) phức tạp hơn, việc sử dụng các ngôn ngữ lập trình cấp cao, và sự ra đời của phần mềm hệ thống (system software).

Bảng 1.1: So sánh một số máy tính IBM trong thế hệ thứ nhất và thế hệ thứ hai

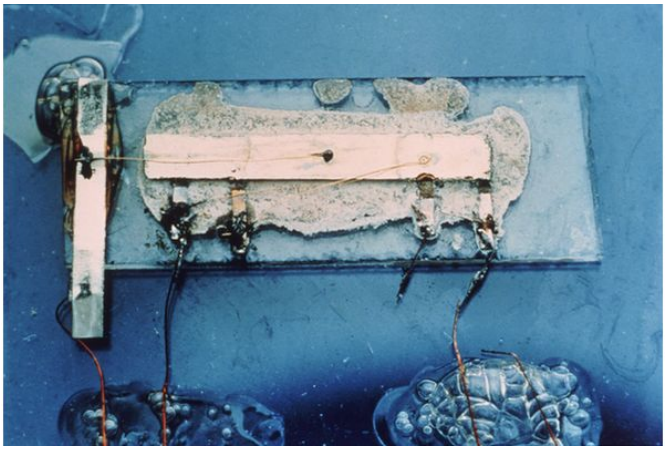
Thế hệ	Tên	Năm sản xuất	Công nghệ	Truy xuất bộ nhớ	Bộ nhớ
I	701	1952	Đèn chân không	30 μs	2-4 K
	704	1955	Đèn chân không	12 μs	4-32 K
	709	1958	Đèn chân không	12 μs	32 K
II	7090	1960	Bóng bán dẫn	2.18 μs	32 K
	7094 I	1962	Bóng bán dẫn	2.0 μs	32 K
	7094 II	1964	Bóng bán dẫn	1.4 μs	32 K

MÁY TÍNH THẾ HỆ THỨ BA

Mặc dù các bóng bán dẫn được sử dụng thay thế các bóng đèn chân không trong sản xuất máy tính và đạt được những cải tiến, tuy vậy các bóng bản dẫn này được chế tạo và đóng gói riêng biệt. Một máy tính hay đơn giản hơn là một thiết bị điện tử sẽ được chế tạo bằng cách nối dây kết nối những bóng bán dẫn này lại với nhau trên một bảng mạch. Quá trình chế tạo máy tính từ bóng bán dẫn cho tới các bảng mạch tốn kém và phức tạp đặc biệt là khi nhu cầu tính toán ngày càng cao. Khi số lượng bóng bán dẫn cần thiết cho một máy tính ngày càng lớn thì các máy tính chế tạo theo công nghệ hiện tại không đáp ứng được; do đó thế hệ máy tính thứ ba dựa trên mạch tích hợp (integrated circuits) ra đời.

Mạch tích hợp được định nghĩa là: *"một thiết bị được tạo thành từ các thành phần điện tử như là các bóng bán dẫn hay các điện trở kết nối với*

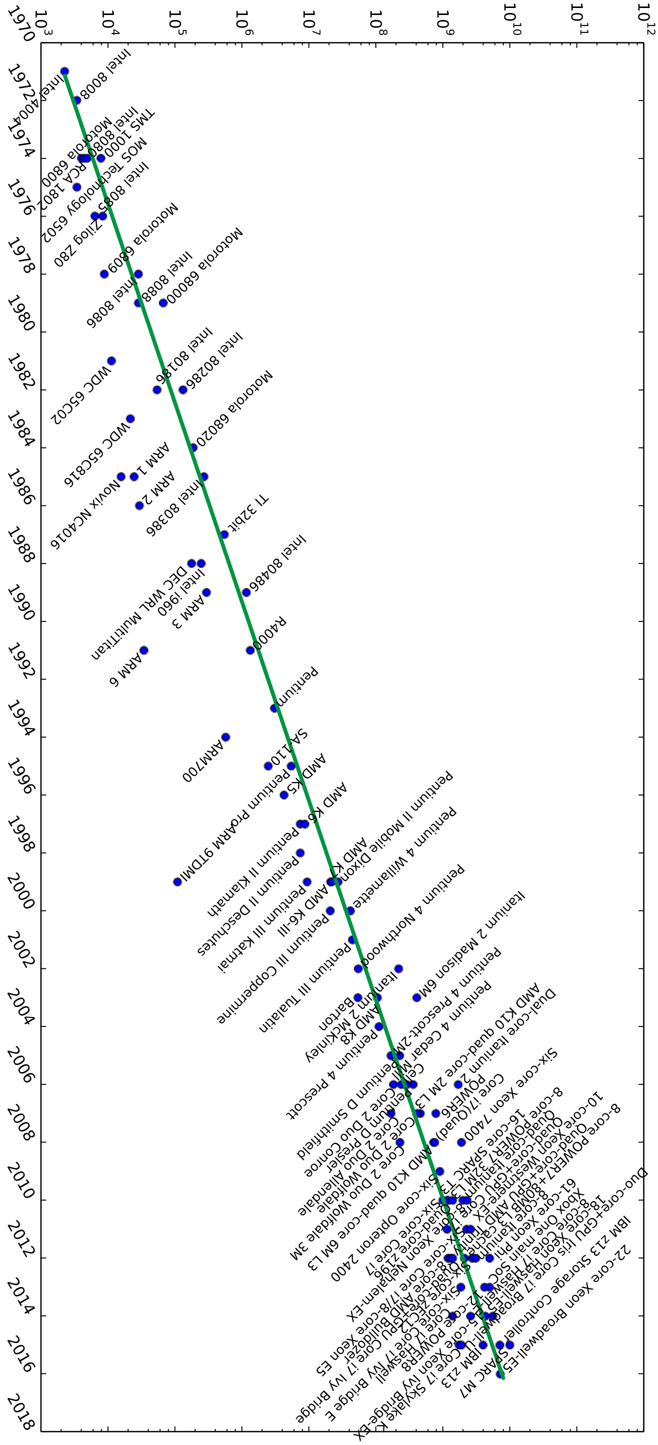
nhau được hiện thực trên một mảnh nhỏ vật liệu bán dẫn (semiconducting material)". Mạch tích hợp đầu tiên hoạt động được trên thế giới được giới thiệu vào tháng 9 năm 1958 chỉ gồm một vài bóng bán dẫn và có kích thước phần mạch chính là 11.11 mm×1.59 mm như Hình 1.4. Sự ra đời của mạch tích hợp cũng là sự bắt đầu của máy tính thế hệ thứ ba, máy tính chế tạo bằng các mạch tích hợp.



Hình 1.4: Mạch tích hợp hoạt động được đầu tiên trên thế giới.

Các mạch tích hợp ở giai đoạn đầu chỉ bao gồm một số lượng rất nhỏ bán dẫn trên một chip. Trải qua quá trình phát triển liên tục, đến năm 2014 đã có thể tích hợp trên 20 tỉ bóng bán dẫn trên một chip. Tháng 7 năm 2016, Altera (một hãng công nghệ sản xuất chip tái cấu hình của Mỹ đã được mua lại bởi tập đoàn Intel) đã công bố chip được tích hợp trên 30 tỉ bóng bán dẫn. Sự phát triển của số lượng bóng bán dẫn trên một chip trong giai đoạn vừa qua tuân theo tiên đoán của Gordon Moore (còn được gọi là định luận Moore), một nhà đồng sáng lập Intel vào những năm 1970 đó là số lượng bóng bán dẫn trên một chip sẽ tăng gấp đôi sau mỗi 18 đến 24 tháng. Hình 1.5 thống kê số lượng các bóng bán dẫn trên các bộ xử lý nổi tiếng từ năm 1971 đến năm 2011 và so sánh với định luật Moore (đường thẳng). Có thể thấy rằng, sự phát triển về số lượng bóng bán dẫn của các mạch tích hợp nói chung và các bộ xử lý nói riêng tuân theo tiên đoán của Moore phát biểu vào những năm 1970.

Hai dòng máy tính nổi tiếng trong thế hệ này là: IBM System/360 và



Hình 1.5: So sánh số lượng bóng bán dẫn trên các bộ xử lý và định luật Moore. Nguồn số liệu https://en.wikipedia.org/wiki/Transistor_count

DEC PDP-8. Dòng máy tính IBM System/360 ra đời năm 1964 đánh dấu sự thành công của IBM trong lĩnh vực sản xuất máy tính ở giai đoạn này khi chiếm đến 70% thị trường máy tính trên toàn thế giới. Tuy nhiên do chuyển hoàn toàn sang công nghệ mạch tích hợp mới nên các máy tính IBM System/360 không tương thích với dòng máy tính trước đó của IBM. Dòng máy tính IBM System/360 này bao gồm nhiều mẫu khác nhau phục vụ cho nhu cầu khác nhau của khách hàng về khả năng tính toán và giá thành. Tuy nhiên, các máy tính trong dòng máy tính IBM System/360 này sử dụng cùng một tập lệnh và cùng một hệ điều hành; do đó khi người sử dụng nâng cấp máy tính thì không cần phải thay đổi chương trình đang sử dụng. Dòng máy này không những là kim chỉ nam cho các dòng máy tính thế hệ sau của IBM mà còn có ảnh hưởng rất lớn đến cả ngành công nghiệp máy tính.

Đến thời điểm bắt đầu của thế hệ máy tính thứ ba, hầu hết các máy tính có hiệu suất trung bình trở lên đều đòi hỏi phải được đặt trong một phòng máy lạnh và có giá thành lên đến hàng trăm ngàn USD. Cùng năm với sự ra đời của dòng máy tính IBM System/360, hãng DEC (Digital Equipment Corporation) giới thiệu máy tính nhỏ PDP-8 dùng công nghệ mạch tích hợp đủ nhỏ để có thể đặt trên bàn ở các phòng thí nghiệm. Mặc dù máy tính PDP-8 không có đầy đủ các chức năng như một máy tính cỡ lớn như IBM System/360, nhưng với giá thành khá rẻ so với các dòng máy cỡ lớn chỉ khoảng 16.000 USD thì máy tính PDP-8 thu hút được lượng khá lớn người sử dụng.

MÁY TÍNH THẾ HỆ THỨ TƯ

Sau thế hệ thứ ba, các nhà khoa học vẫn chưa thống nhất với nhau về định nghĩa các thế hệ máy tính tiếp theo. Mặc dù cho đến hiện nay, công nghệ mạch tích hợp dùng các bóng bán dẫn vẫn đang được sử dụng rộng rãi, tuy nhiên số lượng bóng bán dẫn được tích hợp trên một chip đã tăng rất đáng kể và tuân theo định luật Moore. Các mạch tích hợp được chia thành các loại như trong Bảng 1.2 dựa vào số lượng bóng bán dẫn được tích hợp trên một chip. Dựa trên sự phân loại các mạch tích hợp, một số nhà nghiên cứu đề nghị chia các máy tính sau thế hệ thứ ba thành các thế hệ máy tính LSI, VLSI và ULSI. Tuy nhiên, do công nghệ chế tạo và

kiến trúc của các máy tính này là gần như tương đồng, nên thường được gọi chung là máy tính thế hệ thứ tư. Đặc điểm của máy tính thế hệ thứ tư này là sử dụng bộ nhớ bán dẫn và các vi xử lý (microprocessor).

Ban đầu khi mạch tích hợp ra đời, mạch tích hợp chủ yếu được dùng để chế tạo các bộ xử lý. Bộ nhớ của các máy tính ra đời trong những năm 1950 và 1960 dùng vật liệu sắt từ (ferromagnetic material). Hai vấn đề lớn nhất của bộ nhớ dùng vật liệu sắt từ là giá thành và dữ liệu bị xóa sau khi đọc. Năm 1970, hãng công nghệ Fairchild giới thiệu **bộ nhớ bán dẫn** (semiconductor memory) đầu tiên trên thế giới giải quyết được vấn đề xóa dữ liệu khi đọc và có tốc độ đọc cao hơn hẳn bộ nhớ dùng vật liệu sắt từ. Đến năm 1974, khi giá thành sản xuất bộ nhớ bán dẫn được giảm xuống, bộ nhớ bán dẫn được dùng trong hầu hết các máy tính.

Bảng 1.2: Các loại mạch tích hợp

Tên	Năm ra đời	Số lượng bóng bán dẫn	Số lượng cổng luận lý
SSI	1964	1 đến 10	1 đến 12
MSI	1968	10 đến 500	13 đến 99
LSI	1971	500 đến 20.000	100 đến 9.999
VLSI	1980	20.000 đến 1.000.000	10.000 đến 99.999
ULSI	1984	1.000.000 trở lên	100.000 trở lên

Đi đôi với sự ra đời bộ nhớ bán dẫn, công nghệ sản xuất bộ xử lý cũng có những bước phát triển vượt bậc. Ngày càng nhiều bộ phận thành phần được tích hợp trên cùng một chip, do đó số lượng chip cần dùng để xây dựng nên một bộ xử lý càng ít. Năm 1971 chứng kiến một sự thay đổi toàn diện trong lĩnh vực chế tạo bộ xử lý máy tính đó là sự ra đời của bộ **vi xử lý** (microprocessor) 4004 của Intel. Chip Intel 4004 là một chip đơn đầu tiên chứa tất cả các bộ phận thành phần của một bộ xử lý trung tâm của máy tính. Mặc dù chỉ có thể thực hiện được phép cộng hai số 4 bit và thực hiện phép nhân bằng cách cộng nhiều lần nhưng bộ vi xử lý 4004 đánh dấu quá trình phát triển mạnh mẽ của các bộ vi xử lý cả về khả năng tính toán lẫn công suất tiêu thụ.

Năm 1972 Intel giới thiệu bộ vi xử lý thế hệ tiếp theo của 4004 là 8008 có khả năng xử lý 8 bit thay vì 4 bit như 4004. Tuy nhiên cả hai bộ vi xử lý 4004 và 8008 đều chỉ được sử dụng cho các ứng dụng chuyên biệt.

Năm 1974 chứng kiến sự ra đời của vi xử lý đa dụng (general-purpose microprocessor) 8 bit đầu tiên Intel 8080. Đây là mốc đánh dấu cho thế hệ máy tính thứ 4, thế hệ máy tính dùng các bộ vi xử lý và bộ nhớ bán dẫn. Trong những năm tiếp theo sau, các hãng công nghệ liên tục giới thiệu các bộ xử lý các thế hệ tiếp có khả năng xử lý 16, 32, và 64 bit sử dụng các công nghệ chế tạo mạch tích hợp tiên tiến có kích thước các bóng bán dẫn ngày càng nhỏ. Bảng 1.3 so sánh các bộ xử lý tiêu biểu của Intel trong giai đoạn những năm 1970 và giai đoạn gần đây.

Bảng 1.3: Sự phát triển của các bộ xử lý Intel

Giai đoạn những năm 1970

	4004	**8008**	**8080**	**8086**	**8088**
Năm sản xuất	1971	1972	1974	1978	1979
Tần số clock	108 kHz	108 kHz	2 MHz	5 → 10 MHz	5 → 8 MHz
Độ rộng bus	4 bit	8 bit	8 bit	16 bit	8 bit
Số lượng BBD[1]	2.300	3.500	6.000	29.000	29.000
Kích thước BBD	10 μm		6 mm	3 mm	6 mm
Quản lý bộ nhớ	640 Bytes	16 KB	64 KB	1 MB	1 MB

(1) BDD: bóng bán dẫn (transistor)

Giai đoạn gần đây

	Pentium 4	**Core 2 Duo**	**Core i7-920**	**Core i7-6700K**
Năm sản xuất	2000	2006	2008	8/2015
Tần số clock	1.3-1.8 GHz	1.06-1.12 GHz	2.66 GHz	4 GHz
Độ rộng bus	64 bit	64 bit	64 bit	64 bit
Số lượng BBD[1]	42 triệu	167 triệu	731 triệu	1.35 tỉ
Kích thước BBD	180 nm	65 nm	45 nm	14 nm
Quản lý bộ nhớ	64 GB	64 GB	24 GB	64 GB
Bộ đệm	256 KB L2	2 MB L2	6 MB L2	8 MB L2

1.2.2. PHÂN LOẠI MÁY TÍNH

Mặc dù các máy tính hiện nay được sản xuất dưới cùng một công nghệ chế tạo phần cứng cho cả bộ vi xử lý và bộ nhớ cũng như đều dựa trên mô hình máy tính von Neumann, các máy tính được dùng trong những nhóm ứng dụng khác nhau sẽ có những đặc điểm và yêu cầu khác nhau.

1

Một cách tổng quát, các máy tính có thể được phân chia thành bốn loại khác nhau.

MÁY TÍNH CÁ NHÂN

Máy tính cá nhân (Personal Computer - PC) là loại máy tính có thể nói là phổ biến nhất hiện nay. Khái niệm máy tính cá nhân ra đời trong khoảng 35 năm trở lại đây. Các máy tính cá nhân có những đặc điểm riêng biệt là hiệu suất tốt cho một người sử dụng tại một thời điểm, giá thành thấp và thường thực hiện những ứng dụng của bên thứ ba (third-party software).

MÁY CHỦ

Máy chủ (Server) là khái niệm để chỉ những máy tính thường được truy xuất chỉ thông qua mạng (network). Các máy chủ thường được dùng để thực hiện một ứng dụng phức tạp - ví dụ các ứng dụng tính toán khoa học - hoặc thực thi nhiều ứng dụng nhỏ - ví dụ các ứng dụng cấu thành một trang mạng (website). Các máy chủ thường được xây dựng từ những công nghệ và thành phần cơ bản giống với máy tính cá nhân nhưng cung cấp khả năng tính toán mạnh hơn, khả năng lưu trữ và truy xuất thiết bị ngoại vi lớn hơn. Máy chủ hiện nay bao gồm rất nhiều loại có giá thành và sức mạnh khác nhau. Máy chủ dòng thấp nhất có thể chỉ là một máy tính cá nhân và có giá trong khoảng một ngàn USD. Các máy chủ dòng thấp này thông thường chỉ được dùng để lưu trữ dữ liệu hoặc thực thi những ứng dụng nhỏ hoặc những trang mạng đơn giản. Bên cạnh đó, có những máy chủ có sức mạnh tính toán rất lớn và giá thành rất cao để phục vụ cho nhiều ứng dụng phức tạp và nhiều người dùng cùng lúc. Yêu cầu quan trọng nhất của các máy chủ là độ tin cậy cao (reliability) bởi vì mọi sự hư hỏng trên các máy chủ sẽ có ảnh hưởng đến rất nhiều người.

SIÊU MÁY TÍNH

Khái niệm siêu máy tính (Supercomputer) để chỉ những máy tính có khả năng tính toán vô cùng lớn. Các siêu máy tính này thường có đến hàng chục ngàn bộ xử lý, dung lượng lưu trữ lên đến nhiều terabytes[3] và có giá thành lên đến hàng trăm triệu USD. Các siêu máy tính này dùng để tính

[3] 1 terabyte bằng 2^{40} byte.

toán những ứng dụng khoa học đòi hỏi sức mạnh tính toán rất lớn như dự báo thời tiết, tìm kiếm mỏ dầu, các bài toán so trùng chuỗi DNA,... Danh sách các siêu máy tính mạnh nhất thế giới có thể tìm thấy tại trang mạng `www.top500.org`. Mặc dù đây là loại máy tính có sức mạnh tính toán lớn nhất thế giới nhưng số lượng của nó trên thị trường là ít nhất so với ba loại còn lại.

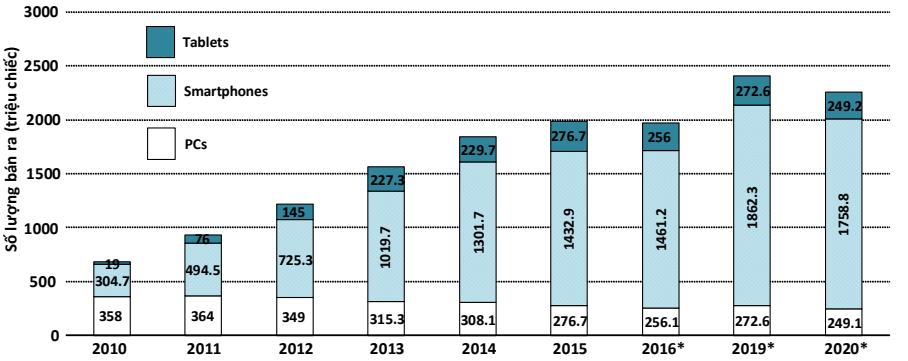
MÁY TÍNH NHÚNG

Máy tính nhúng (Embedded computer) là loại máy tính có số lượng cũng như các mức độ hiệu suất khác nhau nhiều nhất hiện nay. Máy tính nhúng bao gồm từ các bộ vi xử lý có thể tìm thấy trong xe hơi, máy tính trong các thiết bị truyền hình thông minh, cho đến mạng các bộ xử lý (network of processors) dùng điều khiển các chuyến bay. Máy tính nhúng được thiết kế để có thể thực thi hiệu quả nhất chỉ một ứng dụng hoặc một tập ứng dụng liên quan nhau. Máy tính nhúng thường được đóng gói cùng với các thiết bị phần cứng khác thành một hệ thống duy nhất. Điều đặc biệt là hầu hết người sử dụng thường không nhận thấy rằng mình đang sử dụng một máy tính. Sự bùng nổ của các máy tính nhúng đặc biệt là trong các thiết bị điện tử cầm tay làm xuất hiện *kỷ nguyên hậu máy tính cá nhân* (PostPC-era) khi mà số lượng máy tính cá nhân (PC) bán ra hàng năm ít hơn nhiều so với các loại máy tính điện tử cầm tay khác.

1.2.3. KỶ NGUYÊN HẬU MÁY TÍNH CÁ NHÂN

Hình 1.6 so sánh số lượng các điện thoại thông minh (smartphones) và các máy tính bảng (tablets) bán ra so với số lượng máy tính cá nhân truyền thống (PCs) từ năm 2010 đến nay và dự báo đến năm 2020. Có thể thấy rằng, hiện tại và trong tương lai gần, các máy tính cá nhân truyền thống đang bão hòa về số lượng bán ra trong khi số lượng máy tính bảng và số lượng điện thoại thông minh gia tăng không ngừng.

Các máy tính cá nhân đang dần dần được thay thế bằng các thiết bị cá nhân di động (personal mobile device - PMD). Các thiết bị cá nhân di động này thường là các thiết bị nhỏ có kết nối không dây có thể truy cập vào mạng internet với giá thành vài trăm USD. Các thiết bị cá nhân này hoạt động bằng pin và các phần mềm ứng dụng (thường được gọi là



Hình 1.6: Số lượng các máy tính, máy tính bảng và điện thoại thông minh bán ra. Nguồn dữ liệu `https://www.statista.com`.

"apps") có thể được tải về và cài đặt. Khác với các máy tính cá nhân, các thiết bị cá nhân di động thường không có bàn phím và chuột mà thay thế bằng các màn hình cảm ứng hoặc thậm chí là được điều khiển bằng giọng nói. Ví dụ rõ ràng nhất việc các thiết bị di động cá nhân hiện tại đang thay thế dần các máy tính cá nhân là sự bùng nổ các máy tính bảng và các điện thoại thông minh, và tương lai có thể là các kính điện tử thông minh (smart glass).

1.3. CÁC MỨC TRỪU TƯỢNG CỦA CHƯƠNG TRÌNH MÁY TÍNH

Một ứng dụng cơ bản như ứng dụng xử lý văn bản hoặc ứng dụng tính toán bảng tính có thể được phát triển bởi hàng triệu dòng lệnh và sử dụng những thư viện phần mềm phức tạp hiện thực các hàm phức tạp. Trong khi đó, phần cứng máy tính chỉ có thể xử lý những lệnh mức thấp rất đơn giản. Để chuyển đổi từ các ứng dụng phức tạp đến các lệnh phần cứng đơn giản, rất nhiều lớp phần mềm khác nhau tham gia vào việc biên dịch hoặc thông dịch ứng dụng thành các lệnh máy tính đơn giản. Các mức độ hiện thực ứng dụng sử dụng các ngôn ngữ có mức độ phức tạp khác nhau được gọi là các *mức trừu tượng* (abstraction).

1.3.1. NGÔN NGỮ MÁY

Nếu như tiếng Anh sử dụng một bảng chữ cái có 26 ký tự thì bảng chữ cái của máy tính chỉ gồm hai ký tự 0 và 1 tương ứng với hai trạng thái của các mạch điện tử là *đóng* và *ngắt* (*on* và *off*). Mỗi ký tự 0 hoặc 1 được gọi là các ký số nhị phân - *bit* (binary digit); và ngôn ngữ của máy tính (machine language) là ngôn ngữ dựa trên các số nhị phân này. Để điều khiển được máy tính và yêu cầu máy tính thực hiện những công việc cụ thể nào đó, người lập trình phải "nói" ngôn ngữ máy tính. Các tác vụ mà người lập trình có thể yêu cầu máy tính làm trực tiếp được gọi là các *lệnh máy*. Lệnh máy là một chuỗi các bit nhị phân mà máy tính có thể hiểu được và thực thi được. Ví dụ một lệnh máy chứa các số nhị phân:

```
1000110010100000
```

sẽ yêu cầu máy tính thực hiện việc cộng hai số. Chi tiết của các lệnh máy sẽ được giới thiệu trong Chương **??**. Lệnh máy là lệnh có mức trừu tượng đối với máy tính là thấp nhất vì máy tính có thể hiểu một cách trực tiếp.

1.3.2. HỢP NGỮ

Những lập trình viên đầu tiên giao tiếp và điều khiển máy tính thông qua các lệnh máy, là các chuỗi số nhị phân. Tuy nhiên, lập trình cách này rất tốn thời gian và công sức. Do đó, những ký hiệu mới gần với suy nghĩ của con người được phát minh và các ký hiệu này sau đó sẽ được chuyển đổi ra ngôn ngữ máy bằng tay. Tuy nhiên quá trình chuyển đổi này vẫn còn phức tạp và tốn nhiều công sức. Tiếp sau đó, ý tưởng *dùng máy tính để lập trình máy tính* ra đời khi các chương trình được gọi là **trình hợp dịch** (assembler) được thiết kế để chuyển đổi các lệnh dưới dạng ký hiệu thành các chuỗi nhị phân. Các lệnh được viết dưới dạng ký hiệu được gọi là *hợp ngữ* (assembly language). Ví dụ lệnh hợp ngữ:

```
add A, B
```

sẽ có lệnh máy tương đương với chuỗi nhị phân đã được giới thiệu trong phần trước. Các lệnh hợp ngữ không thể được xử lý trực tiếp bởi phần cứng máy tính mà phải thông qua trình hợp dịch để chuyển đổi thành các lệnh máy. Do đó, mức trừu tượng của hợp ngữ đối với máy tính sẽ

cao hơn lệnh máy. Một lệnh hợp ngữ tương đương với duy nhất một lệnh máy.

1.3.3. NGÔN NGỮ LẬP TRÌNH CẤP CAO

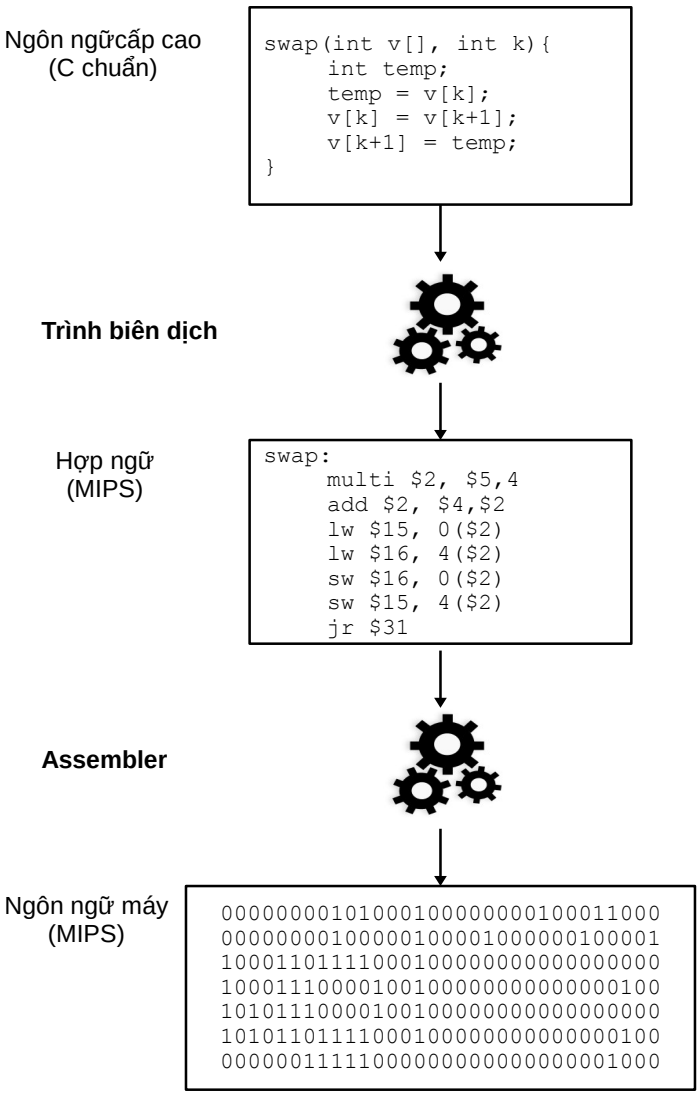
Mặc dù hợp ngữ đã cải thiện đáng kể những khó khăn gặp phải khi lập trình với ngôn ngữ máy, nhưng hợp ngữ vẫn khác xa so với những ký hiệu và cách thức một nhà khoa học ở một lĩnh vực nào đó như toán học hay hóa học mong muốn để mô phỏng một vấn đề khoa học hoặc một nhà kế toán dùng để tính toán một vấn đề tài chính. Hợp ngữ vẫn đòi hỏi người lập trình viết từng dòng lệnh mà máy tính sẽ thực hiện theo, điều này làm cho người lập trình cần phải suy nghĩ giống như máy tính.

Ngôn ngữ lập trình cấp cao (high-level programming language) ra đời nhằm giải quyết những khó khăn khi lập trình với hợp ngữ và nâng cao hiệu quả của việc lập trình. Một chương trình sẽ được dùng để chuyển đổi các chương trình viết bằng ngôn ngữ lập trình cấp cao sang hợp ngữ được gọi là *trình biên dịch* (compiler). Sự ra đời của các ngôn ngữ lập trình cấp cao và trình biên dịch là một trong những bước tiến quan trọng trong giai đoạn đầu của ngành công nghiệp điện toán. Đối với máy tính thì ngôn ngữ lập trình cấp cao có mức trừu tượng cao nhất.

Hình 1.7 trình bày mối quan hệ giữa các mức trừu tượng khác nhau. Ngày nay các ứng dụng thường được viết bằng ngôn ngữ lập trình cấp cao sau đó được biên dịch thành hợp ngữ bởi các trình biên dịch và cuối cùng được chuyển thành mã máy bởi trình hợp dịch.

1.4. CÔNG NGHỆ CHẾ TẠO BỘ XỬ LÝ VÀ BỘ NHỚ

Phần 1.2 đã giới thiệu các thế hệ máy tính khác nhau dựa trên những công nghệ sản xuất khác nhau. Ngày nay, tất cả các máy tính từ máy tính nhúng cho đến siêu máy tính cũng như các thiết bị điện tử từ đơn giản đến phức tạp đều được chế tạo bằng cách sử dụng công nghệ mạch tích hợp (integrated circuit). Các mạch tích hợp là các mạch điện tử chứa từ khoảng vài chục đến vài chục tỉ bóng bán dẫn (tại thời điểm năm 2016) kết nối với nhau theo một quy luật nào đó trên cùng một thiết bị (chip). Các mạch tích hợp đã và đang phát triển tuân theo định luật Moore. Sự



Hình 1.7: Các mức độ trừu tượng khác nhau của một chương trình. Ví dụ từ sách "Computer Organization and Design: the Hardware/Software Interface.

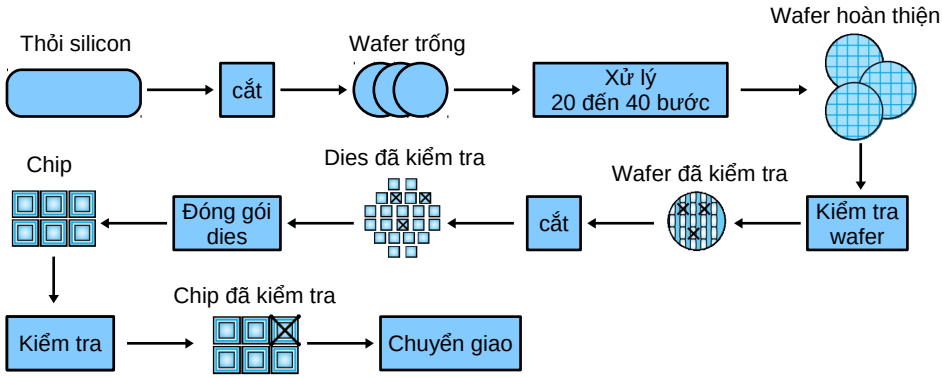
gia tăng nhanh chóng số lượng bóng bán dẫn được tích hợp trên một chip dẫn đến sự gia tăng khả năng tính toán cho các bộ xử lý và gia tăng dung lượng bộ nhớ.

Quá trình sản xuất các mạch tích hợp bắt đầu với silicon, một loại vật liệu có thể tìm thấy trong cát. Bởi vì silicon không dẫn điện tốt cũng như

không thật sự cách điện nên nó được gọi là **chất bán dẫn** (semiconductor). Ngành công nghiệp sản xuất mạch tích hợp sử dụng silicon như là thành phần chính nên còn được gọi là ngành công nghiệp bán dẫn. Trải qua quá trình hóa học đặc biệt, có thể thêm các vật liệu khác vào silicon để nó có thể có một trong ba đặc tính sau:

- Dẫn điện tốt - conductor (bằng cách thêm các dây đồng hoặc nhôm siêu nhỏ);
- Cách điện hoàn toàn - insulator (giống như nhựa hoặc kính);
- Có thể điều khiển dẫn hoặc không dẫn - transistor (giống như nhưng công tắc).

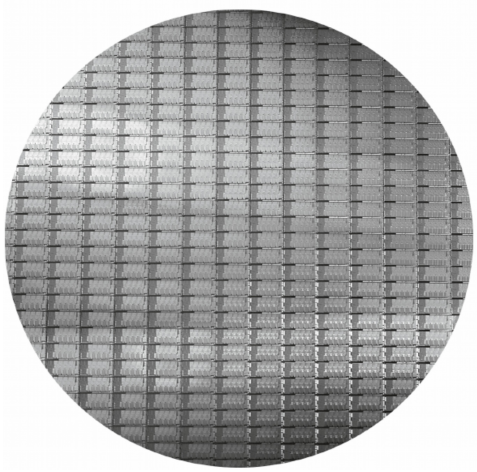
Một mạch tích hợp bao gồm sự kết hợp các phần tử thuộc ba dạng nói trên được chế tạo trên cùng một chip. Quy trình chế tạo các mạch tích hợp được tóm tắt như trong Hình 1.8.



Hình 1.8: Quy trình sản xuất các mạch tích hợp. Hình ảnh tham khảo từ sách "Computer Organization and Design: the Hardware/Software Interface.

Quá trình chế tạo các mạch tích hợp bắt đầu từ một thỏi silicon nguyên chất có kích thước chiều dài khoảng từ 30 - 60 cm (12 - 24 inches) và chiều ngang khoảng 20 - 30 cm (8 - 12 inches). Thỏi silicon này sau đó sẽ được cắt ra thành các hình tròn với chiều dày khoảng 0.25 cm (0.1 inches) được gọi là các *wafer*. Các wafer này sẽ được xử lý qua hàng loạt bước xử lý với những công nghệ và máy móc phức tạp để tạo thành các

wafer hoàn thiện chứa các chip. Mỗi chip là một hình vuông/chữ nhật nhỏ trong wafer.



Hình 1.9: Wafer khi sản xuất các chip Intel Core i7. Hình ảnh tham khảo từ sách "Computer Organization and Design: the Hardware/Software Interface.

Hình 1.9 là hình ảnh thực tế một wafer khi sản xuất các bộ xử lý Intel Core i7. Wafer này có đường kính 300 mm và mỗi chip có kích thước 20.7 × 10.5 mm. Các wafer này sau đó sẽ được kiểm tra để đánh dấu các chip hư hỏng do quá trình sản xuất. Hư hỏng có thể xảy ra do các bước xử lý gặp lỗi hoặc có thể do chỉ một hạt bụi bám vào. Wafer sau đó sẽ được cắt để tạo thành các *die*. Ví dụ trong quy trình ở Hình 1.8 một wafer được cắt ra thành 20 die, trong đó các die đánh dấu 'x' là các die hư hỏng. Các die không hư hỏng sau đó sẽ được mang đi đóng gói để tạo thành các chip. Sau khi đóng gói các chip sẽ được kiểm tra một lần nữa để loại trừ những hư hỏng trong quá trình đóng gói. Cuối cùng các chip hoàn thiện sẽ được chuyển giao cho khách hàng.

Trong quá trình sản xuất, các die hư hỏng do nhiều nguyên nhân khác nhau như đã trình bày ở trên. Tỉ lệ số die không hư hỏng trên tổng số die chứa trong một wafer được gọi là **Yield**. Thông thường Yield của một quy trình sản xuất được tính theo Công thức 1.1, trong đó α tùy thuộc vào công nghệ sản xuất (thông thường $\alpha = 4.0$), *Defect per area* là tỉ lệ hư hỏng

trên một đơn vị diện tích, và *Die area* là diện tích một die.

$$\text{Yield} = (1 + \frac{\text{Defect per area} \times \text{Die area}}{\alpha})^{-\alpha} \tag{1.1}$$

Trên một wafer sẽ có những die không nguyên vẹn do wafer hình tròn trong khi các die có hình vuông/chữ nhật. Các die ở phần rìa của wafer sẽ không nguyên vẹn và sẽ không thể sử dụng. Do đó, số lượng die trên một wafer, *Die per wafer*, được tính theo Công thức 1.2:

$$\text{Die per wafer} = \frac{\pi d^2}{4S} - \frac{\pi d}{\sqrt{2S}} \tag{1.2}$$

Trong đó, d là đường kính wafer và S là diện tích một die (*die area*). Giá trị $\pi d/\sqrt{2S}$ là phần diện tích của những die không nguyên vẹn ở rìa wafer. Những die này không phải là một die hoàn chỉnh nên phải bị bỏ đi.

Tuy nhiên, tổng số die thu được từ một wafer, *Die per wafer*, có thể được tính xấp xỉ theo Công thức 1.3.

$$\text{Die per wafer} \approx \frac{\text{Wafer area}}{\text{Die area}} = \frac{\pi d^2}{4S} \tag{1.3}$$

Giá thành sản xuất một die, *Cost per die*, được tính dựa vào giá thành sản xuất wafer, *Cost per wafer*, số lượng die thu được trên một wafer, và tỉ lệ die không hư hỏng theo Công thức 1.4:

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Die per wafer} \times \text{Yield}} \tag{1.4}$$

Kích thước các wafer là cố định và giá thành sản xuất một wafer cũng thường cố định. Giá thành sản xuất một mạch tích hợp (một die) sẽ tăng khi kích thước mạch tích hợp tăng bởi vì hai nguyên nhân chính là số lượng die trên một wafer sẽ giảm và tỉ lệ hư hỏng sẽ tăng theo sự gia tăng của kích thước die. Do đó, để giảm giá thành sản xuất các mạch tích hợp thì cần giảm kích thước các die. Kích thước các die phụ thuộc vào kiến trúc và thiết kế mạch mà nó hiện thực cũng như công nghệ được dùng để chế tạo mạch. Các công nghệ chế tạo sử dụng kích thước bóng bán dẫn

và dây dẫn nhỏ hơn liên tục được nghiên cứu đưa vào sử dụng để làm giảm giá thành. Ví dụ, các chip Intel Core i7 được sản xuất năm 2012 sử dụng công nghệ 32 nm, tức là kích thước của các bóng bán dẫn là 32 nm. Đến tháng 8 năm 2015, chip Intel Core i7 thế hệ thứ 6 được sản xuất bằng công nghệ 14 nm. Hình 1.9 minh họa một tấm wafer chứa các chip Intel Core i7 năm 2012 sản xuất theo công nghệ 32 nm.

Ví dụ 1.1.

Tính giá thành sản xuất một chip Intel Core i7 theo công nghệ 32 nm, giả sử rằng để chế tạo một wafer Intel Core i7 như trong Hình 1.9 sẽ tốn $500 và tỉ lệ hư hỏng trên một cm^2 là 0.7.

Trả lời. Áp dụng Công thức 1.2 có thể tính được số lượng chip Intel Core i7 thu được trên một wafer là:

$$\text{Die per wafer} = \frac{\pi d^2}{4S} - \frac{\pi d}{\sqrt{2S}} = \frac{\pi \times 30^2}{4 \times 2.07 \times 1.05} - \frac{\pi \times 30}{\sqrt{2 \times 2.07 \times 1.05}} \approx 280(\text{die})$$

Với tỉ lệ hư hỏng trên một cm^2 là 0.7 ta có:

$$\begin{aligned}\text{Yield} &= (1 + \frac{\text{Defect per area} \times \text{Die area}}{\alpha})^{-\alpha} \\ &= (1 + \frac{0.7 \times 2.07 \times 1.05}{4.0})^{-4.0} \approx 0.275\end{aligned}$$

Do đó, giá thành một chip Intel Core i7 chưa đóng gói (die) là:

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Die per wafer} \times \text{Yield}} \approx \frac{\$500}{280 \times 0.275} \approx \$6.494$$

1.5. HIỆU SUẤT VÀ CÔNG SUẤT

Hiệu suất của một quá trình, một hệ thống sẽ đặc trưng cho "độ lợi" hay sức mạnh của hệ thống đó. Hiệu suất của các máy tính là yếu tố duy nhất dùng để so sánh khả năng xử lý của các máy tính khác nhau. Trong khi đó, công suất tiêu thụ là một trong những yếu tố có ảnh hưởng đến phương pháp cải tiến hệ thống máy tính nhằm tăng hiệu suất.

1

1.5.1. ĐỊNH NGHĨA HIỆU SUẤT VÀ TÍNH TOÁN HIỆU SUẤT

Khi lựa chọn một máy tính, yếu tố hiệu suất là một trong những thuộc tính quan trọng để lựa chọn. Do đó, đo đạc và so sánh chính xác hiệu suất giữa các máy tính là một đòi hỏi đối với người mua máy tính và cả người thiết kế máy tính. Hiểu được cách tốt nhất để đánh giá hiệu suất máy tính và các hạn chế trong đo hiệu suất là một điểm quan trọng trong việc lựa chọn máy tính.

Khi thực thi một chương trình trên hai máy tính cá nhân khác nhau, có thể khẳng định rằng máy tính nào hoàn thành chương trình nhanh hơn thì sẽ nhanh hơn. Do đó, đối với máy tính cá nhân, có thể sử dụng đại lượng ***thời gian đáp ứng*** (response time), còn được gọi là ***thời gian thực thi*** (execution time), để đánh giá máy tính. *Thời gian đáp ứng* hay *thời gian thực thi* là tổng thời gian mà một máy tính cần thiết để hoàn thành một tác vụ cụ thể bao gồm thời gian truy xuất đĩa, thời gian truy xuất bộ nhớ, thời gian đáp ứng các thiết bị ngoại vi, thời gian xử lý của hệ điều hành, thời gian CPU thực thi,...

Đối với một server hay một trung tâm xử lý dữ liệu (data center), cùng một thời điểm sẽ có nhiều người sử dụng và nhiều ứng dụng yêu cầu được xử lý. Do đó, máy tính nhanh hơn trong trường hợp này là máy tính có thể hoàn thành xử lý được nhiều ứng dụng hơn trong một đơn vị thời gian; ví dụ một ngày hay một giờ. Số lượng tác vụ máy tính hoàn thành trong một đơn vị thời gian được gọi là ***lưu lượng*** (throughput) hay ***băng thông*** (bandwidth). Đối với server hay các máy tính dùng cho trung tâm xử lý dữ liệu thì đại lượng *lưu lượng* hay *băng thông* thường được dùng để đánh giá máy tính.

MỐI QUAN HỆ GIỮA THỜI GIAN THỰC THI VÀ LƯU LƯỢNG

Khi thay thế bộ xử lý trong một máy tính bằng một bộ xử lý nhanh hơn, tức là làm giảm thời gian thực thi một chương trình. Bởi vì thời gian thực thi chương trình giảm đi so với bộ xử lý cũ nên số lượng chương trình được xử lý trong một đơn vị thời gian sẽ tăng lên. Do đó, lưu lượng hay băng thông của máy tính tăng. Ngược lại khi thay bộ xử lý cũ bằng bộ xử lý mới cùng tốc độ nhưng có nhiều lõi (core) hơn thì không làm thay đổi thời gian thực thi một chương trình vì một chương trình chỉ được thực

thi trên một lõi. Tuy nhiên, bởi vì bộ xử lý có nhiều lõi có thể hoạt động song song nên trong một đơn vị thời gian số lượng chương trình được xử lý tăng so với bộ xử lý cũ. Điều này có nghĩa là lưu lượng hay băng thông của máy tính tăng. Nếu xét thời gian thực thi của hệ thống bằng thời gian thực thi của nhiều (n) ứng dụng thì thời gian thực thi sẽ giảm vì tại cùng một thời điểm có nhiều ứng dụng được thực thi xong xong. Do đó, tổng thời gian cần thiết thực thi n ứng dụng sẽ giảm. Từ đây có thể kết luận rằng, trong các hệ thống máy tính thực thì hai đại lượng *thời gian thực thi* và *lưu lượng* có ảnh hưởng lẫn nhau.

ĐỊNH NGHĨA VÀ SO SÁNH HIỆU SUẤT

Hiệu suất (performance) của một hệ thống máy tính là đại lượng dùng để so sánh khả năng của các máy tính. Một cách đơn giản có thể nhận thấy rằng muốn tăng hiệu suất của máy tính thì chúng ta cần giảm thời gian thực thi. Do đó, hiệu xuất của một máy tính A được định nghĩa và tính toán bằng Công thức 1.5:

$$\text{Performance}_A = \frac{1}{\text{Execution time}_A} \tag{1.5}$$

Nếu máy tính A nhanh hơn (mạnh hơn) máy tính B thì ta có mối quan hệ giữa hai hiệu suất của hai máy tính A và B là:

$$\begin{aligned} &\text{Performance}_A > \text{Performance}_B \\ \Leftrightarrow &\frac{1}{\text{Execution time}_A} > \frac{1}{\text{Execution time}_B} \\ \Leftrightarrow &\text{Execution time}_A < \text{Execution time}_B \end{aligned}$$

Điều này có nghĩa là nếu máy tính A nhanh hơn máy tính B thì thời gian thực thi của máy B sẽ lớn hơn của máy A khi hai máy cùng xử lý một ứng dụng trong điều kiện giống nhau.

Trong thiết kế máy tính, chúng ta thường so sánh một cách định lượng hiệu suất của hai máy tính. Nếu phát biểu rằng "máy tính A nhanh hơn máy tính B n lần" hay "thời gian thực thi máy tính A nhỏ hơn thời gian

thực thi máy tính B n lần" thì có nghĩa là:

$$\frac{\text{Performance}_A}{\text{Performance}_B} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = n$$

Ví dụ 1.2.

Nếu máy tính A thực thi một chương trình trong 10 s và máy tính B thực thi chương trình đó trong 20 s, hỏi máy tính A nhanh hơn máy tính B bao nhiêu lần?

Trả lời. Áp dụng công thức so sánh hiệu suất ta có:

$$\frac{\text{Performance}_A}{\text{Performance}_B} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = \frac{20}{10} = 2$$

Tức là máy tính A nhanh hơn máy tính B 2 lần; hay cũng có thể nói máy tính B chậm hơn máy tính A 2 lần.

TÍNH TOÁN HIỆU SUẤT THÔNG QUA THỜI GIAN

Thời gian xử lý là phương tiện để đo đạc hiệu suất. Máy tính nào cần ít thời gian nhất để xử lý một công việc xác định sẽ là máy tính nhanh nhất, có hiệu suất cao nhất. Cách đơn giản nhất để đo thời gian xử lý một công việc xác định là dùng đại lượng *thời gian thực thi* như đã giới thiệu ở trên hay còn được gọi là *thời gian đáp ứng* hoặc ***thời gian tổng thể*** (elapsed time). Các đại lượng này là tổng thời gian cần thiết để hoàn thành một chương trình bao gồm: thời gian truy xuất đĩa, thời gian truy xuất bộ nhớ, thời gian đáp ứng các thiết bị nhập/xuất (I/O), thời gian phí tổn hệ điều hành,...

Tuy nhiên, có một thực tế rằng máy tính cùng một lúc sẽ xử lý nhiều ứng dụng khác nhau. Máy tính thường cố gắng tối ưu đại lượng số lượng tác vụ được xử lý trong một đơn vị thời gian hơn là tối ứng thời gian thực thi cho một ứng dụng. Ngoài ra thời gian đáp ứng các thiết bị nhập xuất còn phụ thuộc vào loại thiết bị và đôi khi phụ thuộc vào người sử dụng; thời gian phí tổn hệ điều hành tùy thuộc vào trạng thái và loại hệ điều hành,... Do đó, cần phân biệt được thời gian tổng thể và thời gian bộ xử lý thực sự xử lý chương trình. Thời gian bộ xử lý trung tâm (CPU) cần thiết

để tính toán và xử lý chương trình được gọi là ***thời gian thực thi CPU*** (CPU execution time) hoặc đơn giản hơn là ***thời gian CPU*** (CPU time). Thời gian CPU không bao gồm thời gian chờ thiết bị ngoại vi hoặc thời gian xử lý ứng dụng khác. Tương tự, khi đề cập đến hiệu suất chúng ta cũng sẽ phân biệt hai loại hiệu suất đó là hiệu suất dựa trên thời gian tổng thể (còn được gọi là *hiệu suất hệ thống* - system performance)và hiệu suất dựa trên thời gian CPU (còn được gọi lạ *hiệu suất CPU* - CPU performance).

Đối với người sử dụng máy tính thì thời gian thực thi là đại lượng được quan tâm. Tuy nhiên đối với người thiết kế máy tính thì đại lượng **chu kỳ xung nhịp** (clock cycle) được quan tâm hơn vì hầu hết các máy tính đều hoạt động đồng bộ với *xung nhịp* (clock). Xung nhịp là đại lượng biểu thị cho tốc độ xử lý của các khối chức năng phần cứng. Độ dài xung nhịp được tính toán bằng thời gian chu kỳ (cycle), ví dụ 250 ps, hay bằng **tần số** (frequency, clock rate). Tần số xung nhịp được tính bằng nghịch đảo của chu kỳ xung nhịp và có đơn vị là Hz, ví dụ 4 GHz (gigahertz). Tần số xung nhịp là số chu kỳ xung nhịp trong một giây.

Giữa thời gian thực thi CPU hay thời gian CPU của một chương trình và chu kỳ (tần số) xung nhịp đồng bộ của CPU có mối quan hệ với nhau. Biểu thức 1.6 biểu diễn mối quan hệ này. Thời gian CPU cho một chương trình bằng tổng số chu kỳ xung nhịp cần thiết để thực thi chương trình (CPU clock cycles) nhân với thời gian một chu kỳ xung nhịp (Clock cycle time). Bởi vì thời gian một chu kỳ xung nhịp bằng nghịch đảo của tần số xung nhịp nên thời gian thực thi một chương trình bằng số lượng chu kỳ xung nhịp cần thiết cho chương trình chia cho tần số xung nhịp (clock rate).

$$\begin{aligned} \text{CPU time} &= \text{CPU clock cycles} \times \text{Clock cycle time} \\ &= \frac{\text{CPU clock cycles}}{\text{Clock frequency}} \end{aligned} \tag{1.6}$$

Từ Biểu thức 1.6 có thể thấy rằng, để tăng hiệu suất của một máy tính, hay giảm thời gian CPU của một chương trình trên máy tính đó thì hoặc là cần giảm số lượng chu kỳ cần thiết cho chương trình đó hoặc là giảm

thời gian một chu kỳ xung nhịp (tăng tần số xung nhịp). Tuy nhiên, việc giảm thời gian một chu kỳ xung nhịp hay tăng tần số xung nhịp thường làm gia tăng số lượng chu kỳ cần thiết để xử lý chương trình. Các chương tiếp theo sẽ phân tích kỹ hơn vấn đề này.

Ví dụ 1.3.

Máy tính A hoạt động ở tần số 2 GHz thực thi một chương trình trong 10 s. Các kỹ sư muốn thiết kế một máy tính B có khả năng thực thi chương trình đó chỉ trong thời gian 7 s bằng cách tăng tần số so với máy tính A. Tuy nhiên, khi tăng tần số thì số chu kỳ cần thiết để thực thi chương trình trên máy B cũng tăng lên 1.4 lần so với số chu kỳ cần thiết để thực thi chương trình trên máy tính A. Hỏi tần số máy tính B phải là bao nhiêu để đạt được thời gian xử lý như mong muốn.

Trả lời. Áp dụng công thức mối quan hệ giữa thời gian thực thi và tần số xung nhịp ta có:

$$\begin{aligned}
\text{CPU time}_A &= \frac{\text{CPU clock cycles}_A}{\text{clock rate}_A} \\
\text{CPU time}_B &= \frac{\text{CPU clock cycles}_B}{\text{clock rate}_B} \\
\Longrightarrow \frac{\text{CPU time}_A}{\text{CPU time}_B} &= \frac{\text{CPU clock cycles}_A}{\text{CPU clock cycles}_B} \times \frac{\text{clock rate}_B}{\text{clock rate}_A} \\
\Longrightarrow \text{clock rate}_B &= \frac{\text{CPU time}_A}{\text{CPU time}_B} \times \frac{\text{CPU clock cycles}_B}{\text{CPU clock cycles}_A} \times \text{clock rate}_A \\
\Leftrightarrow \text{clock rate}_B &= \frac{10\text{ s}}{7\text{ s}} \times 1.4 \times 2\text{ GHz} = 4\text{ GHz}
\end{aligned}$$

TÍNH TOÁN HIỆU SUẤT THÔNG QUA SỐ LỆNH

Vấn đề nảy sinh ở đây là làm sao đếm được số chu kỳ xung nhịp cần thiết cho một đoạn chương trình? Tính toán hiệu suất thông qua thời gian thực thi CPU không đề cập đến số lượng lệnh của chương trình. Tuy nhiên, một chương trình sau khi được biên dịch sẽ gồm những lệnh và những lệnh này sẽ được máy tính thực thi. Do đó, thời gian thực thi một chương trình phụ thuộc vào số lượng lệnh của chương trình. Thời gian thực thi chương trình sẽ là tích của thời gian thực thi trung bình một lệnh và tổng

số lệnh của chương trình (Instruction count). Nếu thời gian thực thi được đo dựa vào số chu kỳ thì số chu kỳ cần thiết cho một chương trình được tính theo Công thức 1.7.

$$\text{CPU clock cycles} = \text{Instructions count} \times \text{Clock cycles per instruction} \tag{1.7}$$

Trong đó đại lượng *số chu kỳ trên một lệnh* (Clock cycles per instruction) là số lượng chu kỳ trung bình cần thiết để hoàn thành một lệnh. Đại lượng số chu kỳ trên một lệnh thường được gọi là **CPI** của một chương trình hay một đoạn chương trình. Do các lệnh khác nhau sẽ thực thi những công việc khác nhau, ví dụ lệnh nhân và lệnh cộng, nên thời gian để hoàn thành các lệnh cũng sẽ khác nhau. Do đó, đại lượng CPI được tính bằng giá trị trung bình của tất cả các lệnh trong một chương trình hoặc một đoạn chương trình. CPI thường được dùng để so sánh các hiện thực phần cứng khác nhau hỗ trợ cùng một kiến trúc tập lệnh (Instruction Set Architecture - ISA) bởi vì số lượng lệnh khi sử dụng cùng một kiến trúc tập lệnh cho một chương trình là như nhau.

Ví dụ 1.4.

Một tập lệnh cùng được hiện thực cho hai máy tính khác nhau, máy A và máy B. Máy A có thể hoạt động ở tần số 4 GHz trong khi máy B chỉ hoạt động được ở tần số 2 GHz. Khi thực thi một chương trình trên máy A người ta đo được giá trị CPI là 2.0 trong trên máy B là 1.2. Hỏi máy nào thực thi chương trình này nhanh hơn và nhanh hơn bao nhiêu?

Trả lời. Bởi vì hai máy cùng hiện thực một tập lệnh nên số lượng lệnh cần thiết cho chương trình trên cả hai máy là giống nhau. Gọi giá trị này là *IC*.

Áp dụng Công thức 1.7 ta có tổng số chu kỳ cần thiết để các máy hoàn thành chương trình lần lượt là:

$$\text{CPU clock cycles}_A = \text{IC} \times \text{CPI}_A$$
$$\text{CPU clock cycles}_B = \text{IC} \times \text{CPI}_B$$

Bởi vì thời gian thực thi chương trình sẽ bằng số lượng chu kỳ cần

thiết chia cho tần số nên ta có thời gian thực thi chương trên hai máy A và B lần lượt là:

$$\text{CPU time}_A = \frac{\text{IC} \times \text{CPI}_A}{\text{frequency}_A} = \frac{\text{IC} \times 2.0}{4 \times 10^9} = \text{IC} \times 500 \text{ ps}$$
$$\text{CPU time}_B = \frac{\text{IC} \times \text{CPI}_B}{\text{frequency}_B} = \frac{\text{IC} \times 1.2}{2 \times 10^9} = \text{IC} \times 600 \text{ ps}$$

Rõ ràng, máy tính A sẽ thực thi chương trình nhanh hơn máy tính B 1.2 lần.

Từ ví dụ trên có thể thấy rằng thời gian CPU có thể được tính thông qua CPI, số lượng lệnh của chương trình và chu kỳ xung nhịp hoặc tần số xung nhịp bằng Công thức 1.8:

$$\begin{aligned}\text{CPU time} &= \text{CPU clock cycles} \times \text{Clock cycle time} \\ &= \text{Instruction count} \times \text{CPI} \times \text{Clock cycle time} \\ &= \frac{\text{Instruction count} \times \text{CPI}}{\text{clock rate}}\end{aligned} \tag{1.8}$$

Công thức 1.8 thường được sử dụng để tính toán thời gian CPU cần thiết để thực thi một chương trình bởi vì nó phản ánh ba yếu tố chính ảnh hưởng đến hiệu suất của máy tính đó là *số lượng lệnh*, *CPI*, và *tần số*. Thời gian thực thi tỉ lệ thuận với số lượng lệnh và CPI trong khi tỉ lệ nghịch với tần số xung nhịp. Công thức này thường dược dùng so sánh các hiện thực khác nhau hoặc đánh giá các thiết kế khác nhau khi biết được ba yếu tố này.

Ví dụ 1.5.

Một máy tính có ba loại lệnh A, B, và C lần lượt có CPI cho từng loại lệnh là 1, 2, và 3. Có hai chương trình dịch cùng dịch một đoạn chương trình ở ngôn ngữ cấp cao thành các lệnh thuộc ba loại lệnh trên. Số lượng lệnh cho mỗi loại ứng với từng chương trình dịch như sau:

Đoạn chương trình được dịch bởi chương trình dịch nào sẽ có thời gian thực thi nhanh hơn? Giá trị CPI trung bình cho từng đoạn chương trình được dịch bởi các chương trình dịch khác nhau là bao nhiêu?

Chương trình dịch	Số lượng lệnh các loại		
	A	B	C
1	2	4	2
2	4	1	1

Trả lời. Tổng số lượng chu kỳ cần thiết cho một đoạn chương trình sẽ bằng tổng số chu kỳ cần thiết cho các lệnh thuộc các nhóm lệnh khác nhau. Do đó, ta có thể tính số chu kỳ cần thiết cho một đoạn chương trình như sau:

$$\text{CPU clock cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{IC}_i)$$

Trong đó IC_i là số lượng lệnh loại i trong đoạn chương trình và CPI_i là CPI của loại lệnh i.

Như vậy, tổng số chu kỳ xung nhịp cần thiết để xử lý hai đoạn chương trình trên lần lượt là:

$$\text{CPU clock cycles}_1 = \sum_{i=1}^{n} (\text{CPI}_i \times \text{IC}_i) = 2 \times 1 + 1 \times 2 + 2 \times 3 = 10 \text{ (chu kỳ)}$$

$$\text{CPU clock cycles}_2 = \sum_{i=1}^{n} (\text{CPI}_i \times \text{IC}_i) = 4 \times 1 + 1 \times 2 + 1 \times 3 = 9 \text{ (chu kỳ)}$$

Đoạn chương trình được dịch bởi chương trình dịch 2 sẽ có thời gian thực thi nhanh hơn đoạn chương trình được dịch bởi trình dịch 1. Hay chương trình dịch 2 tốt hơn chương trình dịch 1 khi dịch ứng dụng cụ thể này.

CPI trung bình của các đoạn chương trình sẽ bằng tổng số chu kỳ cần thiết để thực thi đoạn chương trình chia cho số lượng lệnh trong đoạn chương trình đó. Do đó, giá trị CPI trung bình của hai đoạn chương trình trên lần lượt là:

$$\text{CPI}_1 = \frac{\text{CPU clock cycles}_1}{\text{Instruction count}_1} = \frac{10}{5} = 2.0$$

$$\text{CPI}_2 = \frac{\text{CPU clock cycles}_2}{\text{Instruction count}_2} = \frac{9}{6} = 1.5$$

Trong trường hợp này $\text{CPI}_2 < \text{CPI}_1$ và tương đồng với kết quả chương trình dịch 2 tốt hơn chương trình dịch 1. Tuy nhiên, chỉ dựa vào yếu tố

CPI không thể kết luận được điều này. Ví dụ minh họa để dành cho người đọc.

Tổng kết lại, các yếu tố ảnh hưởng đến hiệu suất của một chương trình bao gồm: *giải thuật, ngôn ngữ lập trình, trình biên dịch, kiến trúc* và *cấu trúc phần cứng.*

1.5.2. CÔNG SUẤT VÀ GIỚI HẠN CÔNG SUẤT

Hiện nay, hầu hết các mạch tích hợp đều được chế tạo với công nghệ đang chiếm ưu thế là CMOS (complementary metal oxide semiconductor). Công suất tiêu thụ (power) của các mạch tích hợp này sẽ bao gồm hai thành phần là *công suất tĩnh* (static power) và *công suất động* (dynamic power). Công suất tĩnh trong mạch tích hợp dùng công nghệ CMOS là công suất tiêu thụ của các bóng bán dẫn khi nó không hoạt động. Công suất tĩnh trong công nghệ CMOS thường nhỏ và có thể bỏ qua trong tính toán. Ngược lại, công suất động là công suất tiêu thụ khi các bóng bán dẫn thay đổi trạng thái (switching). Đây là nguồn tiêu thụ năng lượng chủ yếu trong các mạch tích hợp công nghệ CMOS. Công suất động phụ thuộc vào điện dung của các bóng bán dẫn (C), nguồn cung cấp (V_{dd}) và tần số thay đổi trạng thái (f) của các bóng bán dẫn. Công suất tiêu thụ của các mạch tích hợp theo công nghệ CMOS được xấp xỉ bằng công suất động và được tính theo Công thức 1.9.

$$\text{Power} = C \times {V_{dd}}^2 \times f \tag{1.9}$$

Tần số thay đổi trạng thái của các bóng bán dẫn chính là tần số hoạt động (clock rate) của mạch tích hợp. Điện dung của mạch tích hợp phụ thuộc vào công nghệ chế tạo và số lượng bóng bán dẫn trong mạch tích hợp. Do đó, nếu điện áp nguồn (V_{dd}) không đổi thì công suất tiêu thụ sẽ tăng tuyến tính với sự tăng tần số xung nhịp. Điều này cản trở sự phát triển của máy tính do công suất tiêu thụ càng lớn thì quá trình tản nhiệt làm mát hệ thống càng phức tạp.

Do đó, phải giảm công suất tiêu thụ bằng cách giảm điện áp nguồn cung cấp cho mạch tích hợp. Điều này có thể thực hiện được bằng cách

tạo ra các thế hệ mạch tích hợp sử dụng các công nghệ tiên tiến hơn. Thông thường công nghệ mới ra đời sẽ giảm được khoảng 15% điện áp nguồn. Trong hơn 20 năm qua, điện áp nguồn đã giảm từ 5 V xuống 1 V.

Ví dụ 1.6.

Giả sử thiết kế được một bộ xử lý đơn giản hơn bộ xử lý cũ nên điện dung giảm 15%. Do sử dụng công nghệ mới nên điện áp nguồn cũng chỉ bằng 85% điện áp nguồn trước đây. Bộ xử lý mới hoạt động ở tần số cao hơn bộ xử lý cũ 20%. Hỏi công suất tiêu thụ bộ xử lý mới tăng hay giảm so với công suất tiêu thụ bộ xử lý cũ?

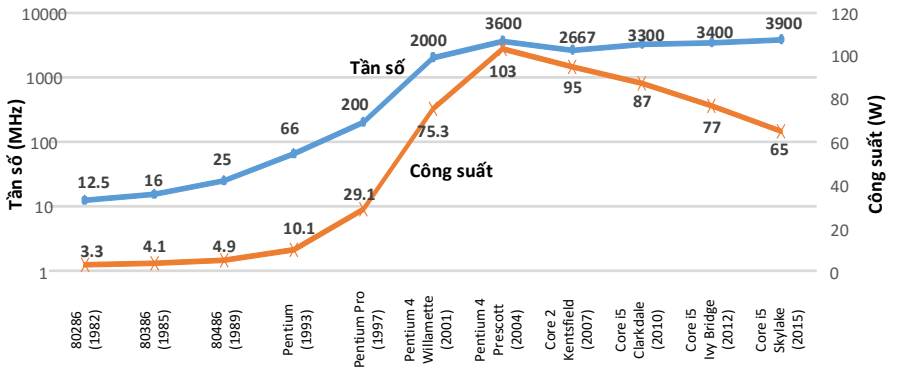
Trả lời. Áp dụng Công thức 1.9 ta có mối quan hệ giữa công suất tiêu thụ của hai bộ xử lý:

$$\frac{P_{new}}{P_{old}} = \frac{0.85C \times (0.85V_{dd})^2 \times 1.2f}{C \times V_{dd}^2 \times f} = 0.74$$

Vậy, công suất tiêu thụ bộ xử lý mới bằng 0.74 lần công suất tiêu thụ bộ xử lý cũ (giảm 26%).

Tuy nhiên, vấn đề đang gặp phải hiện nay là không thể hạ thấp điện áp nguồn hơn nữa vì khi điện áp nguồn quá thấp thì các bóng bán dẫn sẽ không hoạt động chính xác. Do đó, vấn đề nâng cao tần số xung nhịp để tăng hiệu suất hoạt động của các mạch tích hợp nói chung và của các máy tính chế tạo theo công nghệ CMOS nói riêng đang gặp phải vấn đề về giới hạn công suất (power wall). Hình 1.10 cho thấy mối quan hệ giữa công suất tiêu thụ và tần số xung nhịp của mười một thế hệ máy tính nổi tiếng của Intel trong hơn 30 năm qua. Cả tần số xung nhịp và công suất tiêu thụ đều tăng nhanh chóng trong những thập niên đầu và bắt đầu chững lại trong những năm gần đây do vấn đề giới hạn công suất.

Để giải quyết vấn đề công suất tiêu thụ, các chip được đóng gói có kích thước lớn để dễ tản nhiệt và các chip có thể được tắt những phần không được sử dụng tại từng thời điểm. Mặc dù có nhiều công nghệ tản nhiệt khác nhau có thể áp dụng cho những chip có công suất tiêu thụ lớn hơn nhiều, ví dụ 300 watts, tuy nhiên các công nghệ này rất tốn kém và



Hình 1.10: Mối quan hệ giữa công suất tiêu thụ và tần số xung nhịp của mười một thế hệ máy tính x86 nổi tiếng của Intel. Nguồn số liệu http://www.intel.com

không thể áp dụng cho các máy tính cá nhân và đặc biệt là cho các thiết bị di động.

1.6. CÁC LỖI SAI THƯỜNG GẶP

Trong thiết kế cũng như đánh giá máy tính, các lỗi sai sau đây thường xảy ra.

Lỗi 1.1. *Có thể cải tiến một thành phần nào đó của một máy tính để tăng hiệu suất toàn máy tính một lượng tỉ lệ với phần cải tiến*

Lỗi sai này có thể được chứng minh qua ví dụ sau và áp dụng định luật Amdahl. Giả sử rằng một chương trình thực thi trên một máy tính trong thời gian 50 giây, trong đó thời gian xử lý phép toán số thực chiếm 80%. Câu hỏi đặt ra là phải cải tiến việc xử lý phép toán số thực bao nhiêu lần để thời gian xử lý của chương trình giảm đi 5 lần. Theo định luật Amdahl ta có mối quan hệ giữa thời gian thực thi trước và sau khi cải tiến như sau:

$$\text{Thời gian sau cải tiến} = \frac{\text{Thời gian phần ảnh hưởng bởi cải tiến}}{\text{Tỉ lệ cải tiến}} + \text{Thời gian phần không bị ảnh hưởng bởi cải tiến}$$

Trong trường hợp này, thời gian phần ảnh hưởng bởi cải tiến là thời

gian xử lý các phép toán số thực là 40 giây (chiếm 80% thời gian thực thi chương trình), thời gian mong muốn sau cải tiến là 10 giây (cải tiến thời gian thực thi 5 lần), thời gian không ảnh hưởng bởi cải tiến tức là thời gian thực thi các phép toán khác phép toán số thực là 10 giây (50 giây - 40 giây = 10 giây). Do đó, áp dụng định luật Amdahl:

$$10 = \frac{40}{n} + 10$$

Rõ ràng là không thể tìm được n, tức là không thể chỉ cải tiến phép tính số thực để có thể cải tiến thời gian thực thi toàn chương trình 5 lần.

Lỗi 1.2. *Chỉ sử dụng một phần của công thức đánh giá hiệu suất để tính toán hiệu suất.*

Như đã trình bày trong Phần 1.5.1, hiệu suất của một máy tính được tính toán thông qua ba tham số: *tần số xung nhịp, số lượng lệnh* và *CPI*. Một trong những lỗi sai thường gặp là chỉ dùng một hoặc hai trong ba tham số trên để đánh giá hiệu suất của một máy tính. Một ví dụ thường thấy là dùng khái niệm *số triệu lệnh trên một giây* (million instructions per second - MIPS) để so sánh hiệu suất của hai máy tính. MIPS được định nghĩa là số triệu lệnh máy tính thực hiện được trong một giây và được tính theo Công thức 1.10:

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6} = \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{clock rate}} \times 10^6} = \frac{\text{clock rate}}{\text{CPI} \times 10^6} \tag{1.10}$$

Từ định nghĩa MIPS có thể thấy rằng máy tính nhanh hơn sẽ có MIPS lớn hơn. Tuy nhiên theo Công thức 1.10 thì MIPS chỉ phụ thuộc vào hai trong ba tham số đánh giá hiệu suất do đó sẽ gặp các vấn đề sau đây:

- Nếu hai máy tính có tập lệnh khác nhau thì không thể dùng đại lượng MIPS để so sánh vì số lượng lệnh sẽ khác nhau;
- MIPS phụ thuộc vào CPI và CPI của các chương trình khác nhau là khác nhau mặc dù trên cùng một máy tính, do đó một máy tính sẽ có nhiều giá trị MIPS tùy thuộc vào chương trình;

- Một chương trình có thể được biên dịch ra nhiều lệnh hơn nhưng các lệnh này lại thực thi nhanh hơn (như trong Ví dụ 1.5), do đó MIPS độc lập với hiệu suất.

1.7. KẾT CHƯƠNG

Được ra đời vào năm 1946, máy tính điện tử đa dụng đã trải qua quá trình phát triển nhanh chóng cả về công nghệ chế tạo lẫn về năng lực tính toán. Mặc dù còn nhiều tranh cải nhưng sự phát triển của các thế hệ máy tính có thể chia thành bốn thế hệ dựa vào công nghệ chế tạo ra nó: thế hệ thứ nhất sử dụng đèn chân không, thế hệ thứ hai sử dụng các bóng bán dẫn, thế hệ thứ ba hình thành cùng với sự ra đời của mạch tích hợp và thế hệ máy tính thứ tư dùng công nghệ mạch tích hợp lớn.

Mạch tích hợp là cơ sở để phát triển các loại máy tính hiện tại từ siêu máy tính đến các máy tính nhúng. Công nghệ mạch tích hợp đã và đang phát triển nhanh chóng cả về kích thước các bóng bán dẫn lẫn số lượng các bóng bán dẫn được tích hợp trên cùng một chip. Tuy nhiên, hiệu suất của các máy tính phụ thuộc không chỉ vào tần số xung nhịp, được quyết định bởi kiến trúc của máy tính và công nghệ chế tạo, mà còn phụ thuộc vào hai đại lượng quan trọng khác là số chu kỳ trung bình mỗi lệnh (CPI) và số lượng lệnh của chương trình.

Trong thời gian gần đây, tần số xung nhịp của các máy tính hầu như không thể tăng do giới hạn công suất tiêu thụ của các chip. Do đó, xu hướng phát triển của các máy tính ngày nay là xu hướng đa xử lý và các bộ xử lý nhiều lõi. Các vấn đề này sẽ được giới thiệu chi tiết ở các chương tiếp theo.

1.8. CÂU HỎI ÔN TẬP VÀ BÀI TẬP

Bài tập 1.1. Cho bảng số liệu sau:

a. Tính Yield của các loại chip nếu giả sử $\alpha = 4$?

b. Tại sao Yield của chip Sun Niagara lại nhỏ hơn Yield của AMD Operton nhiều mặc dù cả hai có cùng tỉ lệ hư hỏng?

Chip	Kích thước chip (mm^2)	Tỉ lệ hư hỏng (cm^2)	Công nghệ sản xuất (nm)	Số triệu transistors/chip
IBM Power5	389	0.3	130	276
Sun Niagara	380	0.75	90	279
AMD Operton	199	0.75	90	233

Bài tập 1.2. Giả sử rằng, một công ty đang chuẩn bị đầu từ dây chuyền sản xuất chip IBM Power5 (chip IBM Power5 cũ như trong Bài tập 1.1). Phòng chiến lược của công ty dự báo rằng nếu đầu tư dây chuyền sản xuất mới thì cần đầu tư 1 tỉ USD. Sau khi đầu tư dây chuyền sản xuất mới xong thì số lượng chip bán ra trong mỗi tháng sẽ nhiều hơn hiện tại 3 lần và mỗi chip mới có thể bán với giá cao hơn chip cũ 2 lần. Biết rằng chip IBM Power5 sản xuất theo dây chuyền mới sẽ có kích thước là 185 mm^2, tỉ lệ hư hỏng là 0.7 trên mỗi cm^2. Giá thành sản xuất mỗi wafer đường kính 300 mm theo cả dây chuyền cũ lẫn mới đều là 500 USD. Hiện tại giá bán mỗi chip IBM Power5 cao hơn giá thành sản xuất là 40%.

a. Tính giá thành sản xuất chip IBM Power5 cũ và mới?

b. Tính lợi nhuận của mỗi chip IBM Power5 cũ và mới?

c. Nếu hiện tại mỗi tháng bán được 500.000 chip IBM Power5, cần bao nhiêu tháng để có thể thu hồi vốn đầu tư cho dây chuyền sản xuất mới?

Bài tập 1.3. Giả sử rằng khi phân tích một đoạn chương trình người ta thấy rằng có 25% số lượng lệnh là các lệnh xử lý số thực và 75% số lượng lệnh là các lệnh không xử lý số thực. CPI trung bình của các lệnh xử lý số thực là 4.0 trong khi CPI trung bình của các lệnh không xử lý số thực là 1.33. Phân tích theo khía cạnh khác người ta thấy rằng có 2% số lượng lệnh là lệnh xử khai căn số thực (floating point square root) và CPI của lệnh này là 20. Giả sử có hai cách tiếp cận để cải tiến hiệu suất của đoạn chương trình trên. Cách 1 là tiến hành cải tiến CPI của lệnh xử lý khai căn số thực thành 2. Cách thứ 2 là tiến hành cải tiến CPI của nhóm lệnh xử lý số thực thành 2.5. Hãy cho biết cách cải tiến nào tốt hơn?

1

Bài tập 1.4. Đoạn chương trình gồm 1000 lệnh trong đó lệnh load/store chiếm 30%, lệnh jump chiếm 10%, 20% lệnh rẽ nhánh, còn lại là các lệnh về đại số. Biết CPI của lệnh load/store là 2.5, lệnh jump là 1, lệnh rẽ nhánh là 1.5, và lệnh đại số là 2. Biết máy tính có tần số là 2 GHz.

a. Tính thời gian thực thi của đoạn chương trình trên?

b. Tính CPI trung bình của đoạn chương trình trên?

c. Người ta tiến hành cải tiến lệnh load/store sao cho thời gian thực thi của nó giảm đi một nửa. Tính speedup của hệ thống.

Bài tập 1.5. Giả sử một chương trình có 50×10^6 lệnh số thực, 110×10^6 lệnh số nguyên, 80×10^6 lệnh chuyển dữ liệu, và 16×10^6 lệnh rẽ nhánh. CPI cho các loại lệnh lần lượt là 1, 1, 4, và 2.

a. Phải cải tiến CPI của lệnh số thực thành bao nhiêu để chương trình chạy nhanh gấp 2 lần?

b. Cần phải cải tiến CPI của lệnh chuyển dữ liệu thành bao nhiêu để chương trình chạy nhanh hơn 2 lần?

c. Tính thời gian thực thi của chương trình nếu CPI của lệnh số nguyên và số thực giảm 40% trong khi CPI của lệnh chuyển dữ liệu và lệnh rẽ nhánh giảm 30%, biết rằng bộ xử lý hoạt động ở tần số 2 GHz?

Bài tập 1.6. Giả sử thiết kế được một bộ xử lý mới cần điện thế giảm 20% và có tần số tăng 20% so với bộ xử lý cũ. Tính tỉ lệ $\frac{P_{new}}{P_{old}}$ với P là công suất tiêu thụ?