VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Trường Đại học Bách khoa Hà Nội

-----oOo-----

Hồ sơ công nghệ bộ vi xử lý

Giáo viên hướng dẫn: PGS. TS. Phạm Văn Hải

Sinh viên thực hiện: Tạ Quang Tùng

MSSV: 20154280

Lớp: KSTN-CNTT-K60

Mục lục

Thuật ngữ **3**

Danh mục hình ảnh **4**

Chương I: Tổng quan về bộ vi xử lý **5**

1. Khái niệm bộ vi xử lý 5
2. Cấu trúc bộ vi xử lý 6
3. Lịch sử phát triển các bộ vi xử lý 8

Chương II: Các kiến trúc tập lệnh **11**

1. Khái niệm và phân loại kiến trúc tập lệnh 11
2. Kiến trúc tập lệnh CISC 12
3. Kiến trúc tập lệnh RISC 13
4. So sánh kiến trúc RISC với kiến trúc CISC 14

Chương III: Các công nghệ tăng hiệu năng bộ vi xử lý **17**

1. Xử lý đường ống (Instruction Pipelining) 17
2. Vi xử lý đa nhân (Multi-core processor) 19
3. Siêu phân luồng (Hyper-threading) 20

Chương IV: Vi điều khiển **23**

1. Giới thiệu về vi điều khiển 23
2. Thiết kế của vi điều khiển 24

Kết luận **27**

Tài liệu tham khảo **28**

Thuật ngữ

Ký hiệu viết tắt	Ý nghĩa
Microprocessor	Bộ vi xử lý
Microcontroller	Vi điều khiển
CPU: Central Processing Unit	Bộ xử lý trung tâm
ALU: Arithmetic Logic Unit	Khối tính toán logic
CU: Control Unit	Khối điều khiển
Register	Thanh ghi
Instruction	Lệnh máy
RISC	Reduced Instruction Set Computer
CISC	Complex Instruction Set Computer
Instruction Pipeline	Xử lý đường ống
Clock rate	Tần số xung nhịp
Logical Processor	Bộ xử lý ở mức logic
ISA: Instruction Set Architecture	Kiến trúc tập lệnh
MCU: Microcontroller unit	Vi điều khiển

Danh mục hình ảnh

Hình 1: Bộ vi xử lý Core i7 của Intel 5

Hình 2: Cấu trúc của CPU 6

Hình 3: Mô hình hóa ALU 7

Hình 4: Sun UltraSPARC, một bộ vi xử lý RISC 14

Hình 5: Xử lý đường ống trong RISC cổ điển 18

Hình 6: Một vi xử lý 2 nhân với 2 mức cache 19

Hình 7: Vi điều khiển ATmega 23

Chương I: Tổng quan về bộ vi xử lý

1. Khái niệm bộ vi xử lý

Hình 1: Bộ vi xử lý Core i7 của Intel

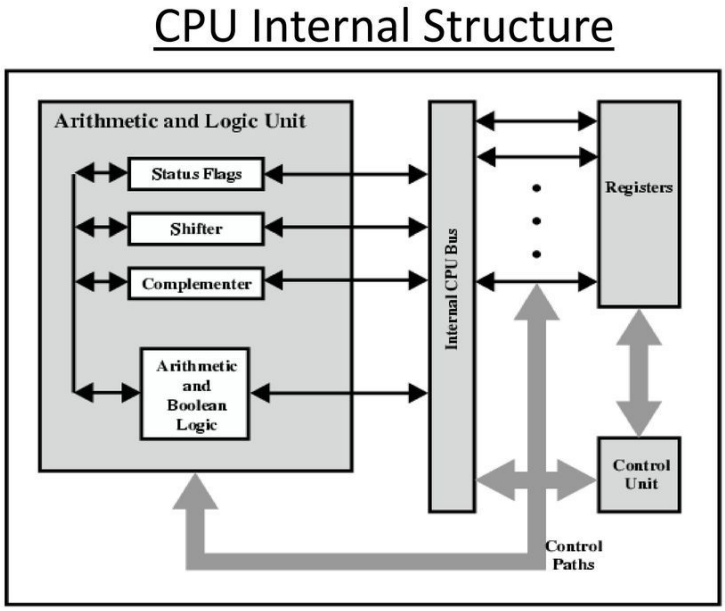
Bộ vi xử lý (microprocessor) là một máy tính nhỏ hoặc CPU (đơn vị xử lý trung tâm) được sử dụng để tính toán, thực hiện phép toán logic, kiểm soát hệ thống và lưu trữ dữ liệu,... Vi xử lý sẽ xử lý các dữ liệu đầu vào / đầu ra (input/output) thiết bị ngoại vi và đưa ra kết quả trở lại để chúng hoạt động.

Trước khi xuất hiện các bộ vi xử lý, các CPU được tạo thành từ các mạch tích hợp (IC) cỡ nhỏ riêng biệt. Mỗi mạch tích hợp chỉ chứa khoảng vài chục transitor. Ngày này, mỗi CPU có thể chứa đến hàng triệu MOS Transitor siêu nhỏ (Kích thước chỉ vài chục đến vài trăm nm). Quá trình tạo ra các vi xử lý được diễn ra gần như hoàn toàn tự động làm cho giá thành của các bộ vi xử lý giảm đi đáng kể.

Bộ vi xử lý là một thiết bị xử lý tính toán tổng quát. Tuy nhiên ngoài ra còn có các thiết bị được tính toán với mục đích đặc trưng như:

- Bộ xử lý tín hiệu số (DSP) chuyên biệt cho tính toán trên tín hiệu số.
- Bộ xử lý đồ họa (GPU) được thiết kế chủ yếu cho render ảnh thời gian thực.
- Vi điều khiển (microcontroller) tích hợp sẵn bộ vi xử lý và các thiết bị ngoại vi được sử dụng trong các hệ thống nhúng.
- Hệ thống trên một vi mạch (Systems on chip) là một thiết bị tích hợp sẵn một hoặc nhiều các vi xử lý hoặc các vi điều khiển.

2. Cấu trúc bộ vi xử lý



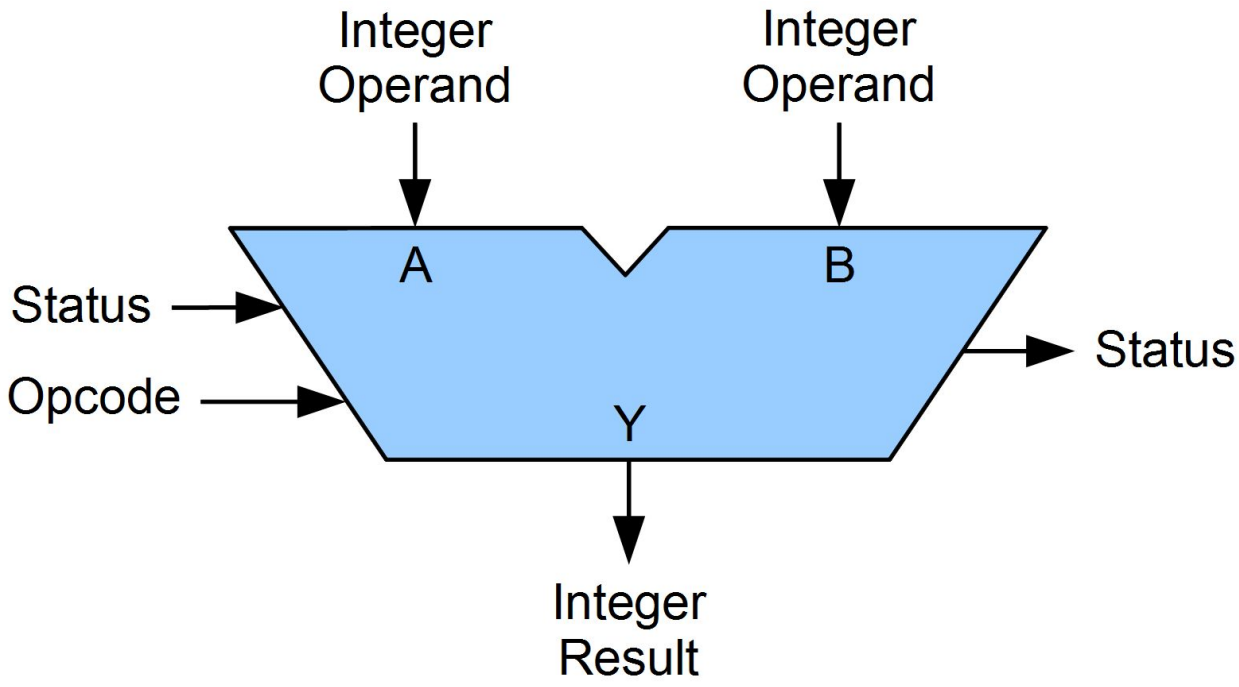
Hình 2: Cấu trúc của CPU

Một bộ vi xử lý bao gồm các thành phần chính là:

- Khối tính toán logic (ALU).

- Khối điều khiển (CU).
- Tập các thanh ghi (Register).

Khối tính toán logic (ALU):



Hình 3: Mô hình hóa ALU

ALU là mạch số tổ hợp thực hiện tính toán số học và tính toán trên bit cho các số nguyên, ngược lại với khối tính toán số phẩy động (FPU). Đầu vào khối ALU bao gồm các toán hạng số nguyên, trạng thái của các cờ trước phép tính và mã số của phép toán cần thực hiện (opcode). Đầu ra là giá trị kết quả và trạng thái cờ sau phép tính.

Các phép toán thực hiện trên ALU bao gồm:

- Phép toán số học: cộng, cộng có dư, trừ, trừ có mượn, bù hai, tăng, giảm.
- Phép toán bitwise: AND, OR, XOR, bù một.
- Phép dịch bit: dịch trái, dịch phải, xoay trái, xoay phải.

Khối điều khiển (CU):

CU là khối điều hướng hoạt động của bộ vi xử lý. CU kiểm soát trình tự di chuyển dữ liệu giữa các đơn vị con của CPU đồng thời cũng sinh ra các tín hiệu điều khiển cho phần còn lại của máy tính.

Thanh ghi (Register):

Là nơi lưu trữ dữ liệu tạm thời trong quá trình tính toán, địa chỉ của thanh ghi được mã hóa trực tiếp trong instruction và cũng được giải mã trực tiếp từ CU.

Ngoài các thành phần chính kể trên, các CPU hiện đại còn bao gồm nhiều đơn vị khác như:

- FPU: Dùng để tính toán số phẩy động.
- MMU - Memory Management Unit: Dùng cho phân trang, phân đoạn, tạo địa chỉ ảo.

3. Lịch sử phát triển các bộ vi xử lý

Việc phát minh ra các bộ vi xử lý giá rẻ đã làm thanh đổi xã hội hiện đại. Các bộ vi xử lý trong các máy tính cá nhân được sử dụng cho tính toán, chỉnh sửa văn bản và giao tiếp qua internet. Rất nhiều bộ vi xử lý là một phần của các hệ thống nhúng, cung cấp điều khiển số cho vô số các thiết bị hàng ngày như điện thoại, TV,...

Thế hệ đầu tiên:

Là khoảng thời gian những năm từ 1971 đến 1973. Intel chế tạo ra vi xử lý đầu tiên là 4004 vào năm 1971 với tốc độ xung nhịp là 108KHz. Kích thước một từ nhớ là 4 bit nên chỉ có thể biểu diễn giá trị từ -8 đến 7. Vì vậy 4004 không được

sử dụng vào trong tính toán số học mà được áp dụng vào điều khiển thiết bị khác.

Thế hệ thứ hai:

Intel 8008 là bước tiến triển tiếp theo, cũng là vi xử lý đầu tiên sử dụng 8 bit. Được tạo ra vào năm 1972, theo ngay sau đó là Intel 8080, cũng là một vi xử lý 8 bit. Tuy nhiên vì chỉ sử dụng 8 bit từ nhớ nên cũng không được áp dụng nhiều vào tính toán số học. Thay vào đó, 8080 chỉ được sử dụng trong các ứng dụng điều khiển.

Một số các vi xử lý khác cũng xuất hiện vào thời điểm này như 6800 từ Motorola, Z-80 từ Zilog.

Những vi xử lý sản xuất thời điểm này có giá cao bởi vì chỉ dựa trên công nghệ chế tạo vi xử lý NMOS.

Thế hệ thứ ba:

Vào khoảng năm 1978, Intel tạo ra vi xử lý 8086, vi xử lý đầu tiên 16 bit. Với kích thước từ nhớ 16 bit có thể biểu diễn một khoảng khá lớn giá trị số nguyên từ -32768 đến +32767 nên đã được áp dụng vào trong tính toán số học. Vi xử lý này cũng trở nên rất phổ biến không chỉ ở ứng dụng trong điều khiển mà còn cho tính toán số. Tốc độ của những vi xử lý này nhanh hơn thế hệ thứ hai 4 lần. Cũng cùng lúc đó, Motorola chế tạo vi xử lý 16 bit là 68000. Zilog cũng phát hành vi xử lý Z-8000.

Thế hệ thứ tư:

Vào đầu những năm 80, Intel phát hành vi xử lý 32 bit có mã hiệu Intel 80386, sử dụng công nghệ chế tạo HCMOS. Nhờ kích thước từ nhớ 32 bit, Intel 80386 đã trở nên rất phổ biến cho việc tính toán. Vào cùng thời điểm, Motorola cũng tạo ra vi xử lý 32 bit là 68020. Intel sau đó phát hành 80486, gần giống như vi xử lý

80386 nhưng tích hợp bộ vi xử lý tính toán số học. Vào đầu những năm 90, Intel tiếp tục phát hành vi xử lý 80586 với tên là vi xử lý Pentium. Những vi xử lý này có tộc độ tính toán số học cực và thực thi instruction kì nhanh. Vi xử lý Pentium 4 phát hành năm 2000 có tổng cộng 42 triệu transitor hoạt động với tần số 1.5 GHz và có tốc độ tính toán 1500 MIPS (Million Instructions Per Second - triệu instruction trên giây).

Thế hệ thứ năm:

Từ năm 1995 cho đến hiện nay, các thế hệ vi xử lý hiệu năng cao sử dụng từ nhớ 64 bit xuất hiện. Đồng thời là càng ngày càng phổ biến của các vi xử lý đa nhân và sự phổ biến rộng rãi của các máy tính nhúng và các thiết bị cầm tay khiến cho nhu cầu về các vi xử lý nhỏ gọn, hiệu năng cao, tiêu thụ ít điện năng ngày càng được xem trọng.

Chương II: Các kiến trúc tập lệnh

1. Khái niệm và phân loại kiến trúc tập lệnh

Kiến trúc tập lệnh, hay Instruction Set Architecture (ISA), là một mô hình trừu tượng của máy tính. Quá trình hiện thực hóa một ISA được gọi là Implementation. Một ISA cho phép các implementation có thể có khác nhau lớn về hiệu năng, kích thước vật lý, giá thành,... Bởi vì ISA chỉ phục vụ cho mục đích làm giao diện giữa phần mềm và phần cứng. Một phần mềm được viết cho một ISA có thể chạy trên nhiều phần cứng có cùng implementation. Từ đó tăng tính tương thích nhị phân giữa các thế hệ máy tính. Sự phát triển của các ISA đã làm giảm đáng kể giá thành của máy tính và tăng cường tính ứng dụng của máy tính trong thực tế.

Một ISA định nghĩ tất cả mọi thứ mà một lập trình viên mã máy cần phải biết để viết chương trình cho máy tính. Khác ISA khác nhau một cách tổng quan ở kiểu dữ liệu chúng hỗ trợ, ở trạng thái của máy tính (trạng thái lưu ở bộ nhớ chính hoặc ở trong thanh ghi) và ngữ nghĩa của các trạng thái đó (như là tính nhất quán bộ nhớ hoặc chế độ địa chỉ), ở tập lệnh (tập các instruction mà cấu thành nên ngôn ngữ máy của máy tính đó) và ở mô hình input/output.

Một vài loại máy ảo có các bytecode cũng được coi là một ISA như là Smalltalk, JVM, Common Language Runtime của Microsoft, hay như WebAssembly cho ứng dụng Web.

Một ISA có thể được phân loại bằng nhiều cách khác nhau. Một trong những phân loại phổ biến là theo độ phức tạp của ISA đó:

- Complex Instruction Set Computer - CISC: Có nhiều các instruction đặc trưng mà hiếm khi được sử dụng trong ứng dụng thực tế.

- Reduced Instruction Set Computer - RISC: Đơn giản hóa bộ vi xử lý bằng việc chỉ thực thi một số instruction mà hay được sử dụng trong các chương trình.

Một vài kiểu ISA khác như là các kiến trúc Very Long Instruction Word (VLIW), kiến trúc Long Instruction Word (LIW) hay kiến trúc Explicitly Parallel Instruction Computing (EPIC). Những kiến trúc thuộc các loại này được tạo ra nhằm mục đích khai thác khả năng song song ở mức instruction của máy tính.

Ngoài ra còn một số kiểu ISA độ phức tạp ít hơn cũng được nghiên cứ như: Minimal Instruction Set Computer (MISC) và One Instruction Set Computer (OISC).

2. Kiến trúc tập lệnh CISC

CISC là các máy tính mà một instruction có thể thực thi nhiều các lệnh nhỏ hơn (như là load dữ liệu từ bộ nhớ, tính toán số học, lưu dữ liệu trở lại bộ nhớ) hoặc có khả năng thực hiện các phép toán nhiều bước hoặc có nhiều chế độ địa chỉ trong cùng một instruction. Thuật ngữ này được sử dụng chỉ sau khi thuật ngữ RISC xuất hiện, cũng thường được sử dụng để ám chỉ những kiến trúc không phải là RISC. Một bộ vi xử lý RISC hiện đại có thể phức tạp hơn một vi điều khiển sử dụng tập lệnh của kiến trúc CISC cả về mức độ phức tạp của mạch tích hợp lẫn số lượng instruction.

Ví dụ một số các kiến trúc tập lệnh mà được gán nhãn CISC như: System/360, z/Architecture, PDP-11, VAX và Data General Nova. Các bộ vi xử lý và các vi điều khiển phổ biến mà cũng được gán nhãn CISC như: Motorola 6800, 6809 và 68000; Intel 8080, iAPX432 và họ vi xử lý x86; Zilog Z80, Z8 và họ Z8000; Họ Intel 8051 và nhiều vi xử lý khác.

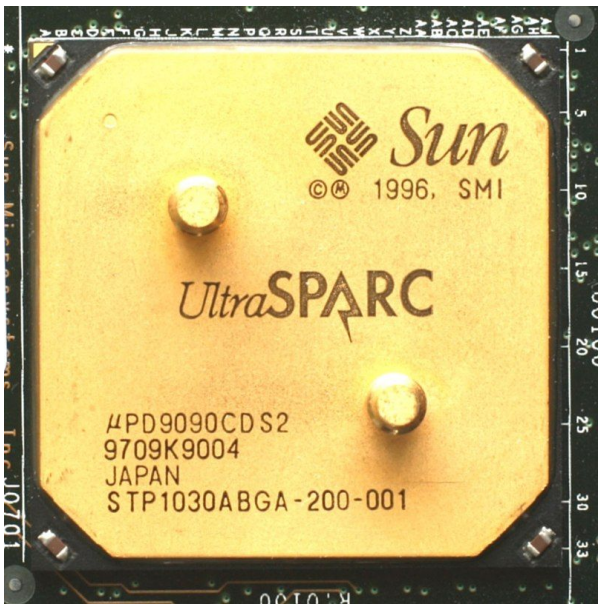
Trước khi kiến trúc RISC xuất hiện và phổ biến, rất nhiều kiến trúc máy tính cố gắng giảm thiểu "khoảng cách ngữ nghĩa" giữa tập lệnh mã máy và các cấu trúc của ngôn ngữ lập trình bậc cao như gọi thủ tục, vòng lặp hay chế độ địa chỉ phức tạp. Từ đó cho phép việc truy cập các cấu trúc dữ liệu và mảng chỉ cần dùng một instruction. Các instruction cũng được mã hóa một cách nhỏ gọn từ đó giúp các chương trình trong kiến trúc CISC có kích thước nhỏ hơn đặc biệt là tại thời điểm những năm 1960 là một ưu điểm rất lớn vì tiết kiệm chi phí lưu trữ, cũng như tốc độ xử lý nhanh hơn. Ngoài ra nó cũng tăng năng suất khi lập trình với tập lệnh CISC ngay cả ở mức lập trình hợp ngữ. Kích thước chương trình nhỏ cũng có ích ngay cả trong các máy tính hiện nay do tốc độ truy cập bộ nhớ nhỏ hơn rất nhiều tốc độ xử lý của CPU, các CPU phải sử dụng các bộ nhớ giới hạn kích thước cache để tăng tốc truy cập dữ liệu và mã máy.

Tuy nhiên, cũng chính vì tập lệnh phức tạp của kiến trúc CISC nên dẫn tới nhiều những vấn đề như: thiết kế những instruction mà không bao giờ dùng đến, những instruction mà khó có thể thiết kế hiệu quả ở mức phần cứng, yêu cầu một lượng lớn công sức để có thể song song hóa hay pipeline.

3. Kiến trúc tập lệnh RISC

RISC là một kiểu kiến trúc tập lệnh mà cho phép ít chu kỳ thực thi cho mỗi instruction hơn là CISC. Đã có nhiều định nghĩa về RISC được đề xuất, nhưng nhìn chung RISC là những máy tính có tập lệnh nhỏ và tổng quát hơn là có nhiều những câu lệnh chuyên biệt hóa. Một đặc tính thường có khác của RISC là phân chia rõ ràng các lệnh load/store từ bộ nhớ với các lệnh xử lý khác. Mặc dù RISC đã có tiền thân từ các máy tính sản xuất vào những năm 1960 và 1970. Tuy nhiên những khái niệm rõ ràng và hiện đại về RISC mới được đặt ra vào những năm 1980. Đặc biệt là nhờ vào hai dự án đến từ trường đại học Stanford

(kiến trúc MIPS) và từ trường đại học Berkeley (kiến trúc SPARC) mà khái niệm RISC được trở nên phổ biến. Một vài dự án thành công khác như là kiến trúc PowerPC và Power ISA của IBM.



Hình 4: Sun UltraSPARC, một bộ vi xử lý RISC

Một số kiến trúc RISC phổ biến như: ARC, Alpha, Am29000, ARM, Atmel AVR, Blackfin, i860, i960, M88000, MIPS, PA-RISC, Power ISA (bao gồm cả PowerPC), RISC-V, SuperH và SPARC. Trong đó kiến trúc ARM được sử dụng phổ biến ở thị trường sản xuất smartphone và máy tính bảng như iPhone, Android, iPad,... Các bộ vi xử lý RISC cũng đã được sử dụng trong các siêu máy tính như máy tính Summit, siêu máy tính mạnh nhất vào thời điểm 11/2018.

4. So sánh kiến trúc RISC với kiến trúc CISC

Một trong những ưu điểm chính của kiến trúc CISC là trình biên dịch không phải chuyển các câu lệnh bậc cao thành nhiều các instruction cho máy tính. Kiến trúc RISC chỉ sử dụng những instruction đơn giản mà có thể thực thi trong một chu

trình clock. Ưu điểm của kiến trúc RISC là phần cứng yêu cầu sử dụng ít các transtitor hơn, dẫn đến có nhiều khoảng trống để cho các thanh ghi hơn. Hơn nữa, do các instruction thực hiện đồng bộ thời gian hơn nên kiến trúc RISC dễ thực thi pipeline hơn. Tuy nhiên các trình biên dịch phải thực hiện nhiều hơn việc dịch các câu lệnh bậc cao sang mã máy.

Bảng so sánh giữa CISC và RISC:

RISC	CISC
Nhấn mạnh vào phần cứng	Nhấn mạnh vào phần mềm
Có chứa các instruction phức tạp, yêu cầu đa xung clock	Chỉ bao gồm các instruction đơn giản, đơn xung clock
Các thao tác load và store bộ nhớ có thể kết hợp vào các instruction	Thao tác load và store yêu cầu các instruction riêng biệt
Kích thước code nhỏ, tần số xung nhịp cao	Kích thước code lơn, tần số xung nhịp thấp
Các transitor chủ yếu sử dụng xử lý các lệnh phức tạp	Các transitor được sử dụng cho việc tạo nhiều thanh ghi

Mặc dù có nhiều ưu điểm, kiến trúc RISC mất đến hơn một thập kỷ để chiếm chỗ đứng trong thị trường vi xử lý. Phần lớn là do hỗ trợ kém về phần mềm hơn CISC. Một phần nữa là do sự hiện diện của Intel. Tuy các vi xử lý CISC càng ngày càng khó phát triển, Intel vẫn có đủ tiềm lực tài chính để phát triển và sản xuất ra các vi xử lý mạnh mẽ. Mặc dù vi xử lý RISC có thể vượt qua vi xử lý CISC của Intel ở một vài khía cạnh, tuy nhiên sự khác biệt là không đáng kể dẫn đến người dùng không có mong muốn thay đổi công nghệ đang sử dụng.

Ngày này, các vi xử lý của Intel gần như là các vi xử lý còn lại duy nhất sử dụng kiến trúc CISC. Nguyên nhân là do sự phát triển trong công nghệ sản xuất RAM dẫn đến giá thành giảm đi đáng kể. Đồng thời các trình dịch ngày càng trở nên tinh vi hơn nên việc kiến trúc RISC nhấn mạnh vào phần mềm thay vì phần cứng cho tối ưu hóa hiệu năng nên lý tưởng hơn.

Chương III: Các công nghệ tăng hiệu năng bộ vi xử lý

1. Xử lý đường ống (Instruction Pipelining)

Instruction Pipeline là kỹ thuật xử lý song song ở mức instruction trong một bộ vi xử lý nhằm mục đích giữ các phần luôn luôn hoạt động. Bằng việc chia các instruction thành các bước tuần tự nhỏ hơn, mỗi bước được thực hiện riêng biệt bởi các đơn vị nhỏ hơn trong vi xử lý, từ đó làm cho các phần của các instruction thực hiện song song. Từ đó tăng tốc độ xử lý của CPU lên dù cho clock rate không thay đổi nhưng có thể làm tăng độ trễ xử lý do bản thân quá trình xử lý pipeline tạo nên.

Số lượng các bước con trong xử lý đường ống:

- Pipeline trong kiến trúc RISC cổ điển có 5 bước:
    - Instruction Fetch - Nạp instruction.
    - Instruction Decode - Giải mã instruction.
    - Execute - Thực thi.
    - Memory Access - Truy cập bộ nhớ.
    - Register Write Back - Ghi ngược trở lại.
- Các vi điều khiển Atmel AVR và PIC có 2 bước.
- Rất nhiều thiết kế bao gồm đường ống có tới 7, 10 và thập chí 20 bước (như vi xử lý Intel Pentium 4).

Clock cycle / Instr. No.	1	2	3	4	5	6	7
1	IF	ID	EX	MEM	WB		
2		IF	ID	EX	MEM	WB	
3			IF	ID	EX	MEM	WB
4				IF	ID	EX	MEM
5					IF	ID	EX

Hình 5: Xử lý đường ống trong RISC cổ điển

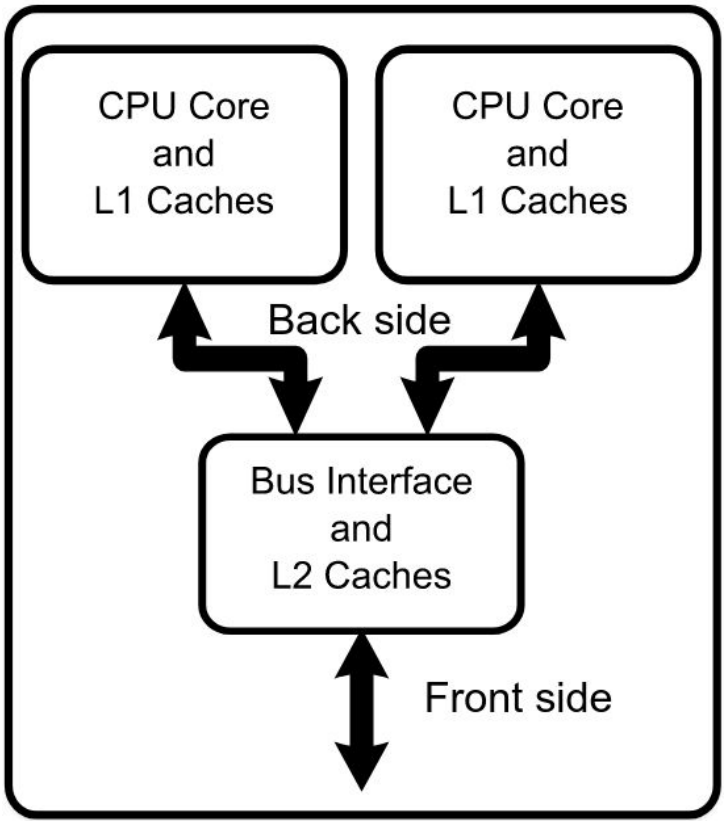
Hazard trong xử lý đường ống

Trong xử lý instruction tuần tự thông thường, các instruction phải kết thúc trước khi instruction khác bắt đầu. Tuy nhiên điều này không còn đúng trong xử lý pipeline. Hazard xảy ra khi có sự phụ thuộc giữa các instruction mà chúng không thể được thực hiện song song.

Có 3 loại hazard:

- Hazard dữ liệu: Xảy ra khi có sự phụ thuộc dữ liệu từ instruction này sang instruction khác.
- Hazard cấu trúc: Xảy ra khi một phần của bộ vi xử lý được sử dụng đồng thời bởi hai hay nhiều instruction tại cùng một thời điểm.
- Hazard rẽ nhánh: Xảy ra khi có instruction rẽ nhánh trong chuỗi các instruction được thực thi.

2. Vi xử lý đa nhân (Multi-core processor)



Hình 6: Một vi xử lý 2 nhân với 2 mức cache

Vi xử lý đa nhân là vi xử lý máy tính mà có nhiều hơn hai khối xử lý, gọi là core, mỗi khối đọc và thực thi instruction độc lập như thể máy tính có nhiều các vi xử lý. Các nhà sản xuất CPU thường tích hợp các core lên cùng một mạch tích hợp. Tuy công nghệ chế tạo vi xử lý càng ngày càng phát triển nhưng hiện nay đã gần như đạt tới giới hạn vật lý về kích thước của transitor. Đồng thời tốc độ xung nhịp của CPU đã không còn có thể tăng lên đáng kể sau vài năm vì nhiều lý do như nhiệt độ, tiêu thụ điện năng. Vì lý do đó mà các bộ vi xử lý đa nhân càng ngày càng phổ biến. Ngoài ứng dụng trong sản xuất CPU, vi xử lý đa nhân còn được áp dụng rộng rãi trong nhiều lính vựng tính toán khác như xử lý tín hiệu số (DSP), đồ họa (GPU).

Ưu điểm của vi xử lý đa nhân:

Do khoảng cách giữa các CPU core là nhỏ và nằm trên cùng một IC cho phép các mạch đồng bộ hóa cache hoạt động hiệu quả với tần số xung nhịp cao hơn so với ở cách xa nhau. Tín hiệu truyền giữa các CPU trong khoảng cách ngắn, do đó cũng hạn chế suy giảm cường độ tín hiệu, cho phép nhiều dữ liệu có thể truyền qua trong một khoảng thời gian.

CPU đa nhân yêu cầu ít không gian hơn là thiết đa vi xử lý đối xứng (Symmetric Multiprocessor - SMP), đồng thời cũng yêu cầu sử dụng ít năng lượng hơn.

Nhược điểm của vi xử lý đa nhân:

Để đạt được hiệu quả trong sử dụng tài nguyên tính toán của vi xử lý đa nhân thì cả hệ điều hành lẫn ứng dụng phải được chỉnh sửa cho tính toán song song. Ứng dụng có tăng được hiệu năng hay không phụ thuộc rất nhiều vào việc sử dụng đa luồng trong ứng dụng.

Việc tích hợp nhiều core vào cùng một IC cũng khiến vấn đề tản nhiệt cho vi xử lý trở nên khó khăn hơn. Hơn nữa, hiệu năng của hệ thống tính toán không chỉ phụ thuộc vào tốc độ các core mà còn phụ thuộc vào tốc độ bus hệ thống, băng thông của bộ nhớ chính. Trong một báo cáo năm 2009, Dr Jun Ni chỉ ra rằng nếu một mà sử dụng gần hết băng thông bộ nhớ thì hai core chỉ có thể tăng hiệu năng lên được 30% đến 70%; tuy nhiên nếu băng thông không bị giới hạn thì có thể đạt tới 90% hiệu năng cải thiện.

3. Siêu phân luồng (Hyper-threading)

Hyper-threading là công nghệ đa luồng đồng thời được sử dụng bởi Intel. Về mặt kiến trúc, một bộ xử lý áp dụng công nghệ Hyper-threading sẽ có hai logical processor cho mỗi core, mỗi một logical processor sẽ bao gồm một tập các trạng

thái riêng biệt. Logical processor có thể được tạm dừng, được ngắt hoặc được thực thi luồng một cách độc lập với các logical processor khác trong cùng một core. Các logical processor trong cùng một core sẽ chia sẻ các tài nguyên tính toán như: khối thực thi, bộ nhớ cache và system bus.

Hyper-thread hoạt động bằng việc nhân đôi một số phần của bộ vi xử lý nhưng không nhân đôi tài nguyên tính toán. Những phần được nhân đôi là những phần lưu trữ trạng thái của vi xử lý khi tính toán như:

- Các thanh ghi điều khiển:
    - Các thanh ghi cờ.
    - Các thanh ghi mặt nạ ngắt.
    - Các thanh ghi khối quản lý bộ nhớ.
    - Các thanh ghi trạng thái.
- Các thanh ghi tổng quát:
    - Các thanh ghi tính toán số học, logic.
    - Các thanh ghi địa chỉ.
    - Các thanh ghi đếm.
    - Các thanh ghi chỉ số.
    - Các thanh ghi stack.
    - Các thanh ghi xử lý xâu.
- Các thanh ghi pipeline, lưu trữ các bước trong xử lý đường ống.

Nếu nhìn từ khía cạnh hệ điều hành và người dùng thì các logical processor giống như các bộ xử lý vật lý thông thường.

Intel sử dụng hyper-threading trong các vi xử lý kiến trúc x86 vào năm 2002 cho vi xử lý Foster MP-based Xeon. Đồng thời Intel cũng sử dụng hyper-threading trong các vi xử lý Northwood-based Pentium 4 tần số xung nhịp 3.06 GHz vào cùng năm đó. Các vi xử lý đời sau đó đều có hyper-threading như Pentium 4 HT,

Pentium 4 Extreme Edition, Pentium Extreme Edition và các thế hệ Core i sau đó.

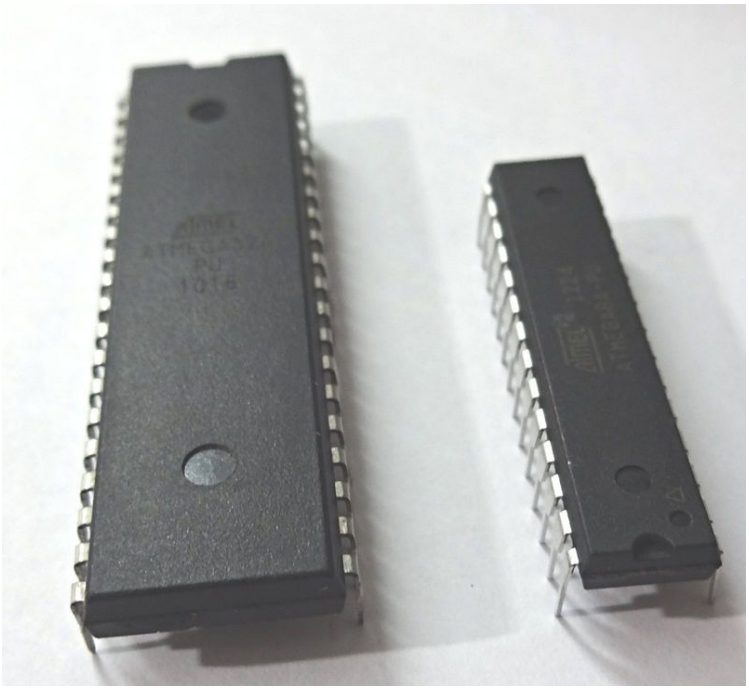
Về hiệu năng thì theo Intel, vi xử lý đầu tiên áp dụng hyper-threading chỉ sử dụng nhiều hơn 5% diện tích so với các vi xử lý không có hyper-threading, tuy nhiên hiệu năng lại tăng đến 15-30%. Pentium 4 cũng có hiệu năng cao hơn 30% so với cùng vi xử lý không áp dụng hyper-threading. Nhưng trong tính toán hiệu năng cao, hyper-threading có thể cản thiện hiệu năng trong một vài ứng dụng, nhưng không phải toàn bộ, thậm chí là giảm đi so với khi không sử dụng.

Khi mà hyper-threading bắt đầu xuất hiện vào năm 2002, rất nhiều những hệ điều hành phổ biến khi đó không được tối ưu cho hyper-threading. Vào năm 2006, hyper-threading bị chỉ trích vì sử dụng năng lượng không hiệu quả.

Chương IV: Vi điều khiển

1. Giới thiệu về vi điều khiển

Vi điều khiển (microcontroller unit - MCU) là một máy tính nhỏ trên cùng một vi mạch tích hợp. Trong thuật ngữ hiện đại thì vi điều khiển tương tự nhưng đơn giản hơn System on a chip (SoC). Một SoC có thể bao gồm các vi điều khiển như là các thành phần của nó. Một vi điều khiển có thể có một hay nhiều các CPU (các core xử lý) đồng thời với bộ nhớ mà input/output ngoại vi có thể lập trình được. Bộ nhớ của vi điều khiển thường bao gồm bộ nhớ chính RAM, bộ nhớ chương trình ROM và bộ nhớ lưu trữ lâu dài flash.



Hình 7: Vi điều khiển ATmega

Các vi điều khiển thường được thiết kế cho các ứng dụng nhúng, ngược lại so với các bộ vi xử lý thường được sử dụng trong các máy tính cá nhân. Vi điều khiển được sử dụng phần lớn trong các thiết điều khiển tự động như thiết bị điều

khiển từ xa, thiết bị y tế, đồ chơi,... Bởi việc giảm kích thước và giá thành so với sử dụng vi điều khiển, bộ nhớ, thiết bị vào ra đơn giản, vi điều khiển rất phù hợp cho các thiết bị tự động giá thành thấp. Trong ngữ cảnh của Internet of Things, vi điều khiển là giải pháp phổ biến và kinh tế cho việc thu thập dữ liệu, cảm biến và điều khiển thế giới thực.

Vào năm 2002, khoảng 55% lượng CPU bán ra trên toàn thế giới là các vi điều khiển và vi xử lý 8 bit. Khoảng trên 2 tỉ vi điều khiển 8 bit được bán ra vào năm 1997 và theo Semico, có khoảng hơn 4 tỉ vi điều khiển 8 bit được bán ra năm 2006, thị trường vi điều khiển cũng tăng 36.5% vào năm 2010 và 12% vào năm 2011.

Một ngôi nhà thông thường ở các nước phát triển có trung bình khoảng 4 vi xử lý đa mục đích nhưng lại có tới khoảng trên 30 vi điều khiển. Một xe hơi thông thường có trung bình khoảng 30 vi điều khiển. MCU cũng có thể được tìm thấy trong các thiết bị điện tử như máy giặt, lò vi sóng và điện thoại bàn. Giá thành sản xuất một vi điều khiển cũng chỉ khoảng $0.10. Giá thành cũng ngày càng giảm theo thời gian, một vi điều khiển rẻ nhất là $0.03 vào năm 2018 và có một số vi điều khiển 32 bit cũng chỉ có giá 1 USD.

2. Thiết kế của vi điều khiển

Phần lớn vi điều khiển sử dụng hiện nay là trong các thiết bị nhúng. Tuy rằng có một số thiết bị nhúng là rất tinh vi, phần lớn chỉ yêu cầu sử dụng bộ nhớ và kích thước chương trình nhỏ, không yêu cầu có hệ điều hành, độ phức tạp thấp. Yêu cầu xử lý của vi điều khiển từ đó cũng khác so với vi xử lý thông thường.

Yêu cầu về ngắt:

Vi điều phải cung cấp đáp ứng thời gian thực với các sự kiện bên trong hệ nhúng. Khi một sự kiện nào đó xảy ra, một tín hiệu ngắt báo cho bộ xử lý dừng việc tính toán hiện tại và bắt đầu dịch vụ ngắt (Interrupt Service Routine hay ISR) sau đó trả về vị trí xử lý instruction trước đó. Những nguồn gây ngắt có thể phụ thuộc vào thiết bị, và thông thường bao gồm những sự kiện như là tràn bộ đếm thời gian, hoàn thành xong chuyển đổi từ tín hiệu số sang liên tục hoặc ngược lại, mức logic bị thay đổi khi một nút được ấn hay dữ liệu được nhận qua kênh truyền. Khi mà tiêu thụ năng lượng là một vấn đề quan trọng đối với thiết bị nhúng, ngắt cũng có thể làm chức năng đánh thức vi điều khiển từ trạng thái ngủ năng lượng thấp.

Yêu cầu về chương trình:

Một trương trình cho vi điều khiển phải chiếm đủ bộ nhớ vi điều khiển, thông thường nó là một giá trị nhỏ. Đồng thời việc thêm thêm bộ nhớ ngoại vi cũng có chi phí cao. Phụ thuộc vào từng loại thiết bị, bộ nhớ chương trình có thể vĩnh viễn, chỉ đọc mà chỉ có thể được lập trình ở nhà máy; hoặc là bộ nhớ có thể xóa hoặc thay đổi được.

Yêu cầu về tính năng đặc trưng:

Các vi điều khiển thường có một vài cho đến một vài chục các chân input/output tổng quát (GPIO). Các chân GPIO có thể được cấu hình bằng phần mềm để là chân input hoặc output. GPIO thường được sử dụng để đọc tín hiệu cảm biến hoặc tín hiệu bên ngoài. Cũng có thể được sử dụng để điều khiển các thiết bị khác như LED, động cơ.

Nhiều các hệ thống nhúng cần có khả năng đọc tín hiệu từ cảm biến mà sinh ra tín hiệu liên tục. Vì vậy các vi xử lý thông thường kèm theo bộ chuyển đổi từ tín hiệu tương tự sang tín hiệu số. Bộ chuyển đổi từ tín hiệu số sang tương tự thì ít được tích hợp hơn.

Một khối điều khiển tín hiệu PWM cũng thông thường được tích hợp sẵn vào vi điều khiển cho các chức năng như điều khiển điện thế, điều khiển động cơ mà không cần sử dụng nhiều tài nguyên tính toán của CPU. Ngoài ra khối truyền tín hiệu UART cũng được dùng phổ biến.

Kết luận

Nhờ sự phát triển mạnh mẽ của công nghệ vi xử lý mà ngành công nghệ thông tin đã trở thành ngành gây ảnh hưởng nhất tới sự thay đổi khoa học xã hội. Tuy nhiên do giới hạn vật lý mà sự phát triển đó đang dần chậm lại. Thay vào đó là sự phát triển của các vi xử lý đa nhân giúp làm tăng hiệu năng nhưng theo một cách khác. Để có thể sử dụng hiệu quả các vi xử lý đa nhân này cũng cần có sự thay đổi đáng kể trong việc xây dựng các phần mềm khai thác được tính toán đồng bộ và song song.

Với sự phát triển của các thiết bị cầm tay như điện thoại, máy tính bảng cũng dẫn tới nhu cầu về các bộ vi xử lý kiến trúc ARM tăng cao. Đồng thời với xu hướng phát triển của công nghệ IoT cũng yêu cầu cao về các hệ thống nhúng và các vi điều khiển. Hi vọng hồ sơ công nghệ này sẽ giúp người đọc có cái nhìn tổng quát hơn về công nghệ vi xử lý cũng như vi điều khiển hiện nay.

Tài liệu tham khảo

[1] https://en.wikipedia.org/wiki/Microprocessor

[2] https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/

[3] https://en.wikipedia.org/wiki/Instruction_pipelining

[4] https://en.wikipedia.org/wiki/Hazard_(computer_architecture)

[5] https://en.wikipedia.org/wiki/Hyper-threading

[6] https://en.wikipedia.org/wiki/Instruction_set_architecture

[7] https://en.wikipedia.org/wiki/Reduced_instruction_set_computer#Use_of_RISC_architectures

[8] https://en.wikipedia.org/wiki/Multi-core_processor