

TRƯỜNG ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

KIẾN TRÚC MÁY TÍNH

Computer Architecture

Course ID: IT3030

Version: CA2021.2

Nguyễn Kim Khánh

Thông tin liên hệ

- Nguyễn Kim Khánh, PhD.
 - Khoa Kỹ thuật máy tính
 - Department of Computer Engineering
- Office: 802-B1
- e-mail: khanhnk@soict.hust.edu.vn
khanh.nguyenkim@hust.edu.vn
- Mobile: 0913.585533

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

2

Mục tiêu học phần

- Sinh viên được trang bị các kiến thức cơ sở về Kiến trúc tập lệnh và Tổ chức của máy tính, cũng như những nguyên tắc cơ bản trong thiết kế máy tính.
- Sau khi học xong học phần này, sinh viên có khả năng:
 - Tìm hiểu kiến trúc tập lệnh của các máy tính cụ thể
 - Lập trình hợp ngữ
 - Đánh giá hiệu năng máy tính và cải thiện hiệu năng của chương trình
 - Khai thác và quản trị hiệu quả các hệ thống máy tính
 - Phân tích và thiết kế máy tính

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

3

Tài liệu học tập

- Bài giảng Kiến trúc máy tính (.pdf)
- Textbooks:
 - [1] Stallings W., *Computer Organization and Architecture*, 10th ed., Pearson 2016
 - [2] Patterson D., Hennessy J., *Computer Organization and Design*, 5th ed., Morgan Kaufmann, 2014
- Phần mềm hỗ trợ học tập:
 - MARS (MIPS Assembler and Runtime Simulator)
(<http://courses.missouristate.edu/kenvollmar/mars/>)

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

4

Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

5

Kiến trúc máy tính

Chương 1 GIỚI THIỆU CHUNG

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

6

Nội dung

- 1.1. Máy tính và phân loại máy tính
- 1.2. Khái niệm kiến trúc máy tính
- 1.3. Sự tiến hóa của công nghệ máy tính
- 1.4. Hiệu năng máy tính

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

7

1.1. Máy tính và phân loại máy tính

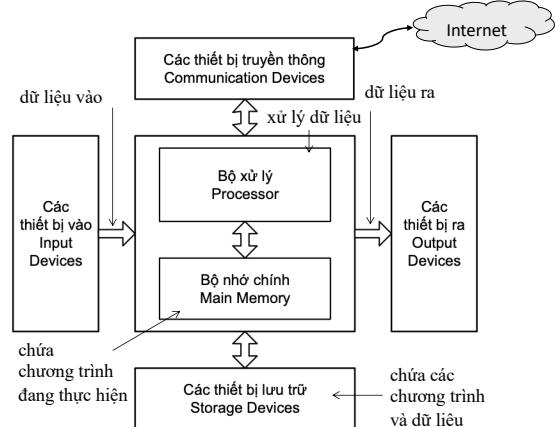
- Máy tính (Computer) là thiết bị điện tử thực hiện các công việc sau:
 - Nhận dữ liệu vào,
 - Xử lý dữ liệu theo dãy các lệnh được nhớ sẵn bên trong,
 - Đưa dữ liệu (thông tin) ra.
 - Dãy các lệnh nằm trong bộ nhớ để yêu cầu máy tính thực hiện công việc cụ thể gọi là chương trình (program).
- Máy tính hoạt động theo chương trình

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

8

Sơ đồ đơn giản của máy tính



NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

9

Phân loại máy tính

- Máy tính cá nhân (PC - Personal Computers)
 - Desktop computers, Laptop computers
 - Máy tính đa dụng
- Thiết bị di động (PMD - Personal Mobile Devices)
 - Smartphones, Tablet Computers
 - Kết nối Internet
- Máy tính nhúng/ IoT (Embedded Computers/ Internet of Things)
 - Máy tính được đặt ẩn trong thiết bị khác
 - Được thiết kế chuyên dụng
 - IoT – là máy tính nhúng có kết nối với Internet

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

10

Phân loại máy tính (tiếp)

- Máy chủ (Servers) – máy phục vụ
 - Dùng trong mạng để quản lý và cung cấp các dịch vụ
 - Hiệu năng và độ tin cậy cao
 - Hàng nghìn đến hàng triệu USD
- Máy tính lớn/ Siêu máy tính (Warehouse-Scale Computers/ Supercomputers)
 - WSC còn gọi là Clusters: kết nối nhiều PCs hoặc Servers với nhau dùng cho các mục đích khác nhau
 - Supercomputers: Một dạng WSC dùng cho siêu tính toán trong khoa học và kỹ thuật
 - Hàng triệu đến hàng trăm triệu USD

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

11

1.2. Khái niệm kiến trúc máy tính

- Kiến trúc máy tính bao gồm 3 khía cạnh:
 - Kiến trúc tập lệnh (Instruction Set Architecture): nghiên cứu máy tính theo cách nhìn của người lập trình
 - Tổ chức máy tính (Computer Organization) hay Vi kiến trúc (Microarchitecture): nghiên cứu thiết kế máy tính ở mức cao (thiết kế CPU, hệ thống nhớ, cấu trúc bus, ...)
 - Phần cứng (Hardware): nghiên cứu thiết kế ở mức chi tiết và công nghệ đóng gói của máy tính.
- Cùng một kiến trúc tập lệnh có thể có nhiều sản phẩm (tổ chức, phần cứng) khác nhau
- Học phần này tập trung vào Kiến trúc tập lệnh và Tổ chức máy tính

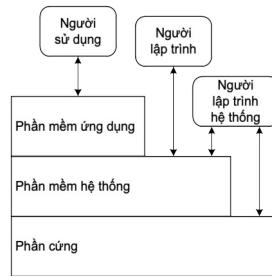
NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

12

Phân lớp máy tính

- Phần mềm ứng dụng
 - Được viết theo ngôn ngữ bậc cao
- Phần mềm hệ thống
 - Chương trình dịch (Compilers): dịch mã ngôn ngữ bậc cao thành ngôn ngữ máy
 - Hệ điều hành (Operating Systems)
 - Lập lịch cho các nhiệm vụ và chia sẻ tài nguyên
 - Quản lý bộ nhớ và lưu trữ
 - Điều khiển vào-ra
- Phần cứng
 - Bộ xử lý, bộ nhớ, mô-đun vào-ra



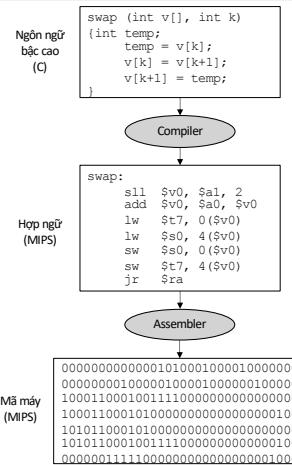
NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

13

Các mức của mã chương trình

- Ngôn ngữ bậc cao
 - High-level language - HLL
 - Mức trừu tượng gần với vấn đề cần giải quyết
 - Hiệu quả và linh động
- Hợp ngữ
 - Assembly language
 - Mô tả lệnh dưới dạng text
- Ngôn ngữ máy
 - Machine language
 - Các lệnh được mã hóa bằng số nhị phân



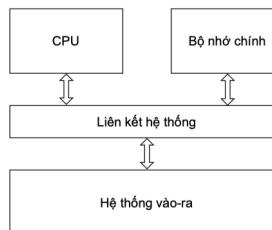
NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

14

Các thành phần cơ bản của máy tính

- Giống nhau với tất cả các loại máy tính
- Bộ xử lý trung tâm
 - Central Processing Unit - CPU
 - Điều khiển hoạt động của máy tính và xử lý dữ liệu
- Bộ nhớ chính
 - Main Memory
 - Chứa các chương trình đang thực hiện
- Hệ thống vào-ra
 - Input/Output
 - Trao đổi thông tin giữa máy tính với bên ngoài
- Liên kết hệ thống
 - System interconnection
 - Kết nối và vận chuyển thông tin



NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

15

1.3. Sự tiến hóa của công nghệ máy tính

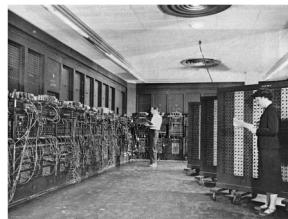
- Máy tính dùng đèn điện tử chân không (1950s)
 - Máy tính ENIAC: máy tính đầu tiên (1946)
 - Máy tính IAS: máy tính von Neumann (1952)
- Máy tính dùng transistors (1960s)
- Máy tính dùng vi mạch SSI, MSI và LSI (1970s)
 - SSI - Small Scale Integration
 - MSI - Medium Scale Integration
 - LSI - Large Scale Integration
- Máy tính dùng vi mạch VLSI (1980s)
 - VLSI - Very Large Scale Integration
- Máy tính dùng vi mạch ULSI (1990s-nay)
 - ULSI - Ultra Large Scale Integration

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

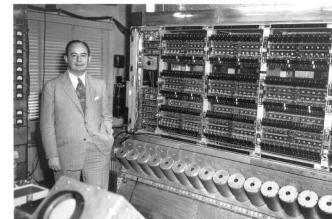
16

Máy tính đầu tiên: ENIAC và IAS



- Electronic Numerical Integrator and Computer
- Dự án của Bộ Quốc phòng Mỹ
- Do GS John Mauchly ở đại học Pennsylvania thiết kế
- 30 tấn
- Xử lý theo số thập phân

NKK-IT3030-CA2021.2

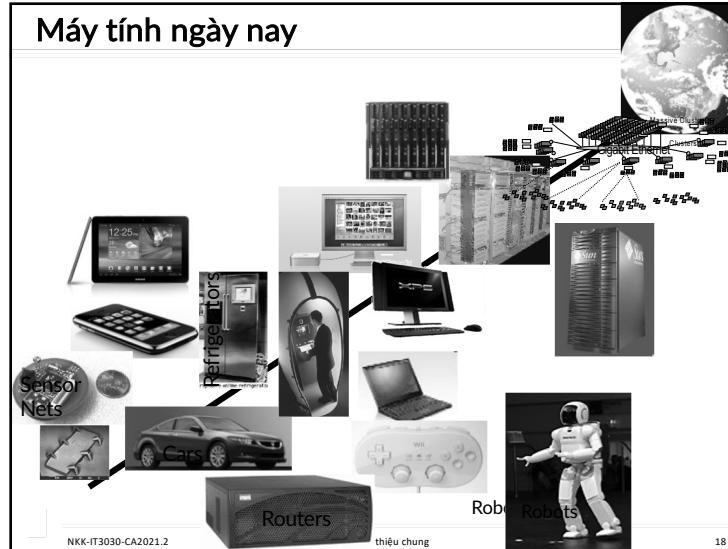


- Thực hiện tại Princeton Institute for Advanced Studies
- Do John von Neumann thiết kế theo ý tưởng "stored program"
- Xử lý theo số nhị phân
- Trở thành mô hình cơ bản của máy tính

CH1-Giới thiệu chung

17

Máy tính ngày nay



NKK-IT3030-CA2021.2

thiệu chung

18

Một số loại vi mạch số điển hình (IC- Integrated Circuits)

- Bộ vi xử lý (Microprocessors)
 - Một hoặc một vài CPU được chế tạo trên một chip
- Vi mạch điều khiển tổng hợp (Chipset)
 - Vi mạch thực hiện các chức năng nối ghép các thành phần của máy tính với nhau
- Bộ nhớ bán dẫn (Semiconductor Memory)
 - ROM, RAM, Flash memory
- Hệ thống trên chip (SoC – System on Chip) hay Bộ vi điều khiển (Microcontrollers)
 - Tích hợp các thành phần chính của máy tính trên một chip vi mạch
 - Được dùng trong smartphone, tablet và máy tính nhúng

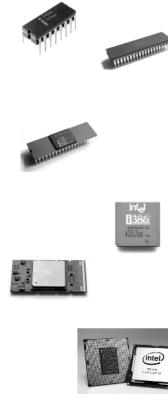
NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

19

Sự phát triển của bộ vi xử lý

- 1971: bộ vi xử lý 4-bit Intel 4004
- 1972: các bộ xử lý 8-bit
- 1978: các bộ xử lý 16-bit
 - Máy tính cá nhân IBM-PC ra đời năm 1981
- 1985: các bộ xử lý 32-bit
- 2001: các bộ xử lý 64-bit
- 2006: các bộ xử lý đa lõi (multicores)
 - Nhiều CPU trên 1 chip



NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

20

1.4. Hiệu năng máy tính

- Định nghĩa hiệu năng P (Performance):

$$\text{Hiệu năng} = \frac{1}{\text{Thời gian thực hiện}}$$

hay là: $P = \frac{1}{t}$

"Máy tính A nhanh hơn máy B k lần"

$$\frac{P_A}{P_B} = \frac{t_B}{t_A} = k$$

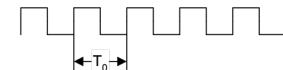
- Ví dụ: Thời gian chạy chương trình:

- 10s trên máy A, 15s trên máy B
- $t_B / t_A = 15s / 10s = 1.5$
- Vậy máy A nhanh hơn máy B 1.5 lần

NKK-IT3030-CA2021.2

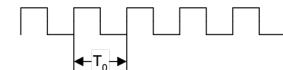
CH1-Giới thiệu chung

21



Tốc độ xung nhịp của CPU

- Về mặt thời gian, CPU hoạt động theo một xung nhịp (clock) có tốc độ xác định



- Chu kỳ xung nhịp T_0 (Clock period): thời gian của một chu kỳ
- Tốc độ (tần số) xung nhịp f_0 (Clock speed hoặc Clock rate): số chu kỳ trong 1s, đo bằng đơn vị Hz
- $f_0 = 1/T_0$
- Ví dụ: Bộ xử lý có $f_0 = 4\text{GHz} = 4 \times 10^9\text{Hz}$
- $T_0 = 1/(4 \times 10^9) = 0.25 \times 10^{-9}\text{s} = 0.25\text{ns}$

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

22

Thời gian thực hiện của CPU

- Để đơn giản, ta xét thời gian CPU thực hiện chương trình (CPU time):

Thời gian thực hiện của CPU =

Số chu kỳ xung nhịp x Thời gian một chu kỳ

$$t_{CPU} = n \times T_0 = \frac{n}{f_0}$$

trong đó: n là số chu kỳ xung nhịp

- Hiệu năng được tăng lên bằng cách:
 - Giảm số chu kỳ xung nhịp n
 - Tăng tốc độ xung nhịp f_0

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

23

Ví dụ

- Hai máy tính A và B cùng chạy một chương trình

Máy tính A:

- Tốc độ xung nhịp của CPU: $f_A = 2\text{GHz}$
- Thời gian CPU thực hiện chương trình: $t_A = 10\text{s}$

Máy tính B:

- Thời gian CPU thực hiện chương trình: $t_B = 6\text{s}$
- Số chu kỳ xung nhịp khi chạy chương trình trên máy B (n_B) bằng 1.2 lần số chu kỳ xung nhịp khi chạy chương trình trên máy A (n_A)

- Hãy xác định tốc độ xung nhịp cần thiết cho máy B (f_B)?

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

24

Ví dụ

- Ta có: $t = \frac{n}{f_0}$
- Số chu kỳ xung nhịp khi chạy chương trình trên máy A:
$$n_A = t_A \times f_A = 10s \times 2GHz = 20 \times 10^9$$
- Số chu kỳ xung nhịp khi chạy chương trình trên máy B:
$$n_B = t_B \times f_B = 1.2 \times n_A = 24 \times 10^9$$
- Vậy tốc độ xung nhịp cần thiết cho máy B:

$$f_B = \frac{n_B}{t_B} = \frac{24 \times 10^9}{6} = 4 \times 10^9 \text{Hz} = 4 \text{ GHz}$$

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

25

Số lệnh và số chu kỳ trên một lệnh

- Số chu kỳ xung nhịp của chương trình:

Số chu kỳ = Số lệnh của chương trình x Số chu kỳ trên một lệnh

$$n = IC \times CPI$$

 - n - số chu kỳ xung nhịp
 - IC - số lệnh của chương trình (Instruction Count)
 - CPI - số chu kỳ trên một lệnh (Cycles per Instruction)

- Vậy thời gian thực hiện của CPU:
$$t_{CPU} = IC \times CPI \times T_0 = \frac{IC \times CPI}{f_0}$$
- Trong trường hợp các lệnh khác nhau có CPI khác nhau, cần tính CPI trung bình

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

26

Ví dụ

- Hai máy tính A và B có cùng kiến trúc tập lệnh
- Máy tính A có:
 - Chu kỳ xung nhịp: $T_A = 250\text{ps}$
 - Số chu kỳ/ lệnh trung bình: $CPI_A = 2.0$
- Máy tính B:
 - Chu kỳ xung nhịp: $T_B = 500\text{ps}$
 - Số chu kỳ/ lệnh trung bình: $CPI_B = 1.2$
- Hãy xác định máy nào nhanh hơn và nhanh hơn bao nhiêu?

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

27

Ví dụ (tiếp)

- Ta có: $t_{CPU} = IC \times CPI_{TB} \times T_0$
- Hai máy cùng kiến trúc tập lệnh, vì vậy số lệnh của cùng một chương trình trên hai máy là bằng nhau:
$$IC_A = IC_B = IC$$
- Thời gian thực hiện chương trình đó trên máy A và máy B:

$$t_A = IC_A \times CPI_A \times T_0 = IC \times 2.0 \times 250\text{ps} = IC \times 500\text{ ps}$$

$$t_B = IC_B \times CPI_B \times T_B = IC \times 1.2 \times 500\text{ps} = IC \times 600\text{ ps}$$
- Từ đó ta có:
$$\frac{t_B}{t_A} = \frac{IC \times 600\text{ ps}}{IC \times 500\text{ ps}} = 1.2$$
- Kết luận: máy A nhanh hơn máy B 1.2 lần

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

28

CPI trung bình

- Nếu các loại lệnh khác nhau có số chu kỳ khác nhau, ta có tổng số chu kỳ:

$$n = \sum_{i=1}^K (CPI_i \times IC_i)$$

- Vậy CPI trung bình:

$$CPI_{TB} = \frac{n}{IC} = \frac{1}{IC} \sum_{i=1}^K (CPI_i \times IC_i)$$

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

29

Ví dụ

Cho bảng chỉ ra các dãy lệnh sử dụng các lệnh thuộc các loại A, B, C. Tính CPI trung bình?

Loại lệnh	A	B	C
CPI theo loại lệnh	1	2	3
IC trong dãy lệnh 1	20	10	20
IC trong dãy lệnh 2	40	10	10

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

30

Ví dụ

Cho bảng chỉ ra các dãy lệnh sử dụng các lệnh thuộc các loại A, B, C. Tính CPI trung bình?

Loại lệnh	A	B	C
CPI theo loại lệnh	1	2	3
IC trong dãy lệnh 1	20	10	20
IC trong dãy lệnh 2	40	10	10

- Dãy lệnh 1: Số lệnh = 50
 - Số chu kỳ = $= 1 \times 20 + 2 \times 10 + 3 \times 20 = 100$
 - $CPI_{TB} = 100/50 = 2.0$
- Dãy lệnh 2: Số lệnh = 60
 - Số chu kỳ = $= 1 \times 40 + 2 \times 10 + 3 \times 10 = 90$
 - $CPI_{TB} = 90/60 = 1.5$

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

31

Tóm tắt về Hiệu năng

$$CPUtime = \frac{Instructions}{Program} \times \frac{Clock cycles}{Instruction} \times \frac{Seconds}{Clock Cycle}$$

Thời gian thực hiện chương trình của CPU =
 = Số lệnh của chương trình x Số chu kỳ/lệnh x Số giây của một chu kỳ

$$t_{CPU} = IC \times CPI \times T_0 = \frac{IC \times CPI}{f_0}$$

- Hiệu năng phụ thuộc vào:
 - Thuật giải
 - Ngôn ngữ lập trình
 - Chương trình dịch
 - Kiến trúc tập lệnh
 - Phần cứng

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

32

MIPS như là thước đo hiệu năng

- MIPS: Millions of Instructions Per Second
Số triệu lệnh trên 1 giây

$$\text{MIPS} = \frac{\text{Instruction Count}}{\text{Execution Time} \times 10^6} = \frac{\text{Instruction Count}}{\frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

$$\text{MIPS} = \frac{f_0}{\text{CPI} \times 10^6}$$

$$\text{CPI} = \frac{f_0}{\text{MIPS} \times 10^6}$$

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

33

Ví dụ

- Tính MIPS của bộ xử lý với:
clock rate = 2GHz và CPI = 4

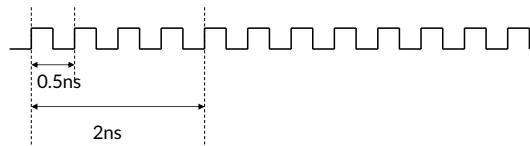
NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

34

Ví dụ

- Tính MIPS của bộ xử lý với:
clock rate = 2GHz và CPI = 4



- Chu kỳ $T_0 = 1/(2 \times 10^9) = 0.5\text{ns}$
- CPI = 4 → thời gian thực hiện 1 lệnh = $4 \times 0.5\text{ns} = 2\text{ns}$
- Số lệnh thực hiện trong 1s = $(10^9\text{ns})/(2\text{ns}) = 5 \times 10^8$ lệnh
- Vậy bộ xử lý thực hiện được 500 MIPS

NKK-IT3030-CA2021.2

CH1-Giới thiệu chung

35

Ví dụ

- Tính CPI của bộ xử lý với:
clock rate = 1GHz và 400 MIPS

NKK-IT3030-CA2021.2

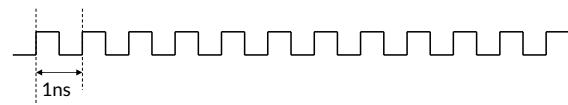
CH1-Giới thiệu chung

36

Ví dụ

Tính CPI của bộ xử lý với:

clock rate = 1GHz và 400 MIPS



- Chu kỳ $T_0 = 1/10^9 = 1\text{ns}$
- Số lệnh thực hiện trong 1 s là $400\text{MIPS} = 4 \times 10^8$ lệnh
- Thời gian thực hiện 1 lệnh = $1/(4 \times 10^8)\text{s} = 2.5\text{ns}$
- Vậy ta có: CPI = $2.5/1 = 2.5$

MFLOPS

- Sử dụng cho các hệ thống tính toán lớn
- Millions of Floating Point Operations per Second
- Số triệu phép toán số dấu phẩy động trên một giây

$$\text{MFLOPS} = \frac{\text{Executed floating point operations}}{\text{Execution time} \times 10^6}$$

- GFLOPS (10^9)
- TFLOPS (10^{12})
- PFLOPS (10^{15})

Kiến trúc máy tính

Hết chương 1



Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

2

Kiến trúc máy tính

Chương 2 **HỆ THỐNG MÁY TÍNH**

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

3

Nội dung

- 2.1. Các thành phần cơ bản của máy tính
- 2.2. Hoạt động cơ bản của máy tính
- 2.3. Liên kết trong máy tính

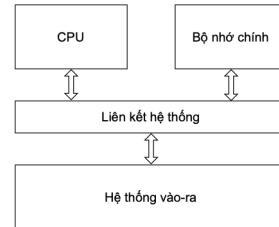
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

4

2.1. Các thành phần cơ bản của máy tính

- Bộ xử lý trung tâm
(Central Processing Unit – CPU)
 - Điều khiển hoạt động của máy tính và xử lý dữ liệu
- Bộ nhớ chính (Main Memory)
 - Chứa các chương trình đang thực hiện
- Hệ thống vào- ra (Input-Output)
 - Trao đổi thông tin giữa máy tính với bên ngoài
- Liên kết hệ thống (System interconnection)
 - Kết nối và vận chuyển thông tin
 - Bus: dạng kết nối cơ bản



NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

5

1. Bộ xử lý trung tâm (CPU)

- Chức năng:
 - điều khiển hoạt động của máy tính
 - xử lý dữ liệu
- Nguyên tắc hoạt động cơ bản:
 - CPU hoạt động theo chương trình nằm trong bộ nhớ chính.
- Là thành phần nhanh nhất trong hệ thống

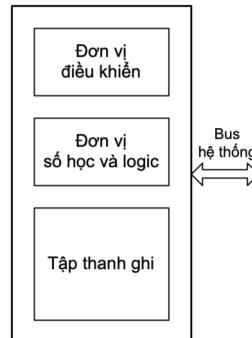
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

6

Các thành phần cơ bản của CPU

- Đơn vị điều khiển
 - Control Unit - CU
 - Điều khiển hoạt động của máy tính theo chương trình đã định sẵn
- Đơn vị số học và logic
 - Arithmetic and Logic Unit - ALU
 - Thực hiện các phép toán số học và phép toán logic
- Tập thanh ghi
 - Register File - RF
 - Gồm các thanh ghi chứa các thông tin phục vụ cho hoạt động của CPU



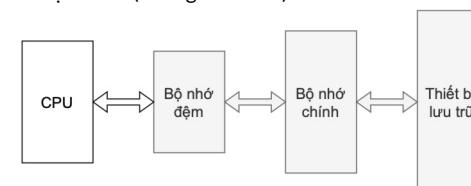
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

7

2. Bộ nhớ máy tính

- Chức năng: nhớ chương trình và dữ liệu (dưới dạng nhị phân)
- Các thao tác cơ bản với bộ nhớ:
 - Thao tác ghi (Write)
 - Thao tác đọc (Read)
- Các thành phần chính:
 - Bộ nhớ chính (Main memory)
 - Bộ nhớ đệm (Cache memory)
 - Thiết bị lưu trữ (Storage Devices)



NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

8

Bộ nhớ chính (Main memory)

- Tồn tại trên mọi máy tính
- Chứa các lệnh và dữ liệu của chương trình đang được thực hiện
- Sử dụng bộ nhớ bán dẫn
- Tổ chức thành các ngăn nhớ được đánh địa chỉ (thường đánh địa chỉ cho từng byte nhớ)
- Nội dung của ngăn nhớ có thể thay đổi, song địa chỉ vật lý của ngăn nhớ luôn cố định
- CPU muốn đọc/ghi ngăn nhớ cần phải biết địa chỉ ngăn nhớ đó

Nội dung	Địa chỉ
0100 1101	00...0000
0101 0101	00...0001
1010 1111	00...0010
0000 1110	00...0011
0111 0100	00...0100
1011 0010	00...0101
0010 1000	00...0110
1110 1111	00...0111
.	.
.	.
.	.
0110 0010	11...1110
0010 0001	11...1111

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

9

Bộ nhớ đệm (Cache memory)

- Bộ nhớ có tốc độ nhanh được đặt đệm giữa CPU và bộ nhớ chính nhằm tăng tốc độ CPU truy cập bộ nhớ
- Dung lượng nhỏ hơn bộ nhớ chính
- Sử dụng bộ nhớ bán dẫn tốc độ nhanh
- Cache thường được chia thành một số mức (L1, L2, L3)
- Cache thường được tích hợp trên cùng chip bộ xử lý

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

10

Thiết bị lưu trữ (Storage Devices)

- Còn được gọi là bộ nhớ ngoài
- Chức năng và đặc điểm
 - Lưu giữ tài nguyên phần mềm của máy tính
 - Được kết nối với hệ thống dưới dạng các thiết bị vào-ra
 - Dung lượng lớn
 - Tốc độ chậm
- Các loại thiết bị lưu trữ
 - Bộ nhớ từ: ổ đĩa cứng HDD
 - Bộ nhớ bán dẫn: ổ thể rắn SSD, ổ nhớ flash, thẻ nhớ
 - Bộ nhớ quang: CD, DVD (ít dùng)

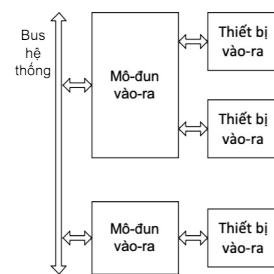
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

11

3. Hệ thống vào-ra

- Chức năng: Trao đổi thông tin giữa máy tính với thế giới bên ngoài
- Các thao tác cơ bản:
 - Vào dữ liệu (Input)
 - Ra dữ liệu (Output)
- Các thành phần chính:
 - Các thiết bị vào-ra (IO devices)
 - Các mô-đun vào-ra (IO modules)



NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

12

Các thiết bị vào-ra

- Còn được gọi là thiết bị ngoại vi (Peripherals)
- Chức năng: chuyển đổi dữ liệu giữa bên trong và bên ngoài máy tính
- Các loại thiết bị vào-ra:
 - Thiết bị vào (Input Devices)
 - Thiết bị ra (Output Devices)
 - Thiết bị lưu trữ (Storage Devices)
 - Thiết bị truyền thông (Communication Devives)

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

13

Mô-đun vào-ra

- Chức năng: nối ghép các thiết bị vào-ra với máy tính
- Mỗi mô-đun vào-ra có một hoặc một vài cổng vào-ra (I/O Port)
- Mỗi cổng vào-ra được đánh một địa chỉ xác định
- Các thiết bị vào-ra được kết nối và trao đổi dữ liệu với máy tính thông qua các cổng vào-ra
- CPU muốn trao đổi dữ liệu với thiết bị vào-ra, cần phải biết địa chỉ của cổng vào-ra tương ứng

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

14

2.2. Hoạt động cơ bản của máy tính

- Thực hiện chương trình
- Hoạt động ngắt
- Hoạt động vào-ra

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

15

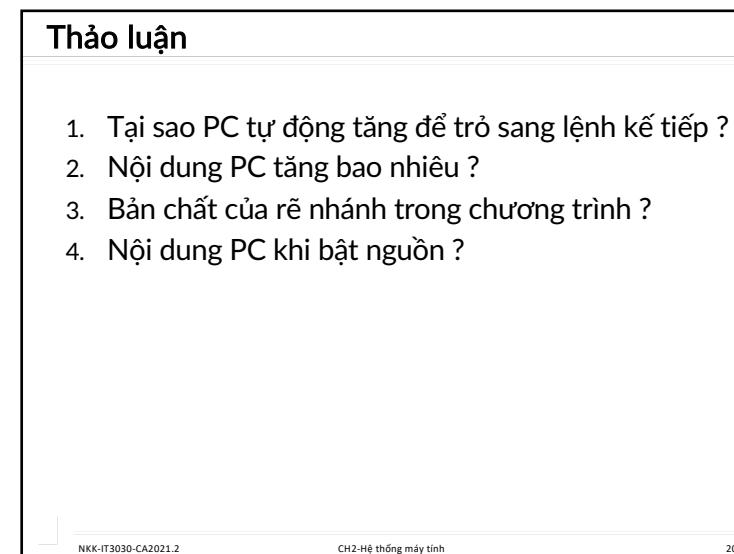
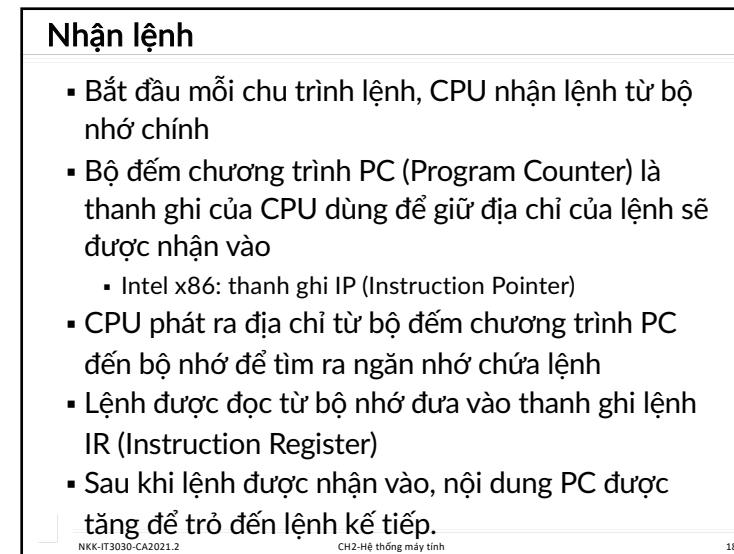
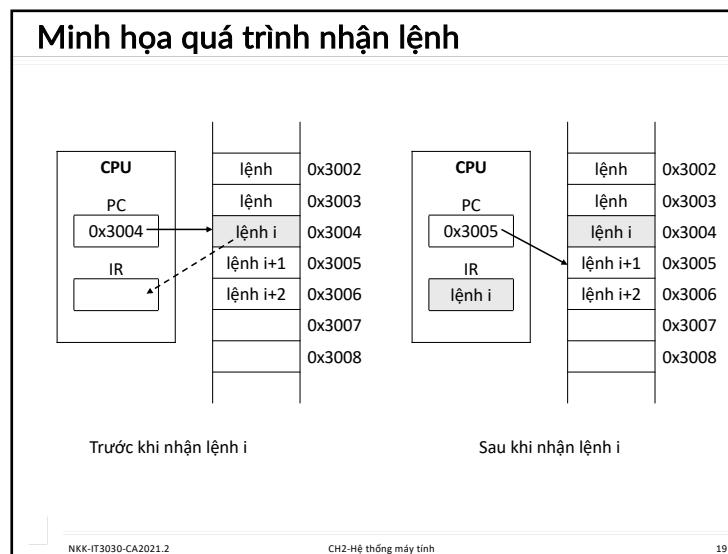
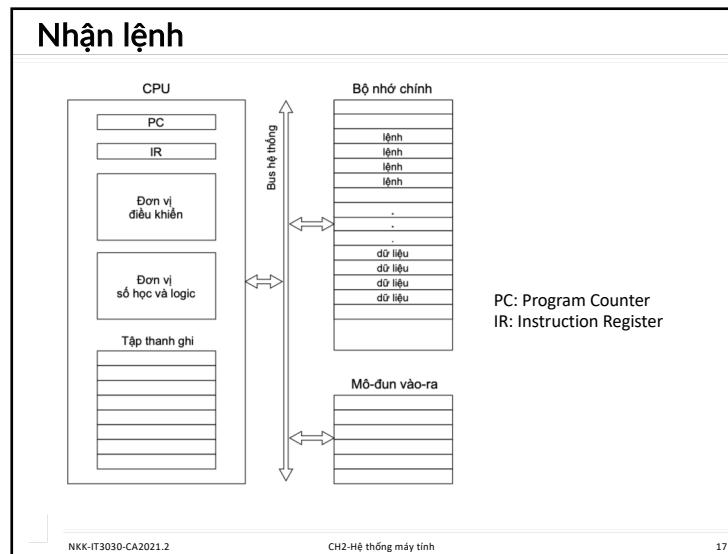
1. Thực hiện chương trình

- Là hoạt động cơ bản của máy tính
- Máy tính lặp đi lặp lại chu trình lệnh gồm hai bước:
 - Nhận lệnh
 - Thực hiện lệnh
- Hoạt động thực hiện chương trình bị dừng nếu:
 - Thực hiện lệnh bị lỗi
 - Gặp lệnh dừng
 - Tắt máy

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

16



Thực hiện lệnh

- Bộ xử lý giải mã lệnh đã được nhận và phát tín hiệu điều khiển thực hiện thao tác mà lệnh yêu cầu
- Các kiểu thao tác cơ bản của lệnh:
 - Trao đổi dữ liệu giữa CPU với bộ nhớ chính hoặc CPU với mô-đun vào-ra
 - Thực hiện các phép toán số học hoặc phép toán logic với các dữ liệu
 - Chuyển điều khiển trong chương trình: rẽ nhánh hoặc nhảy đến vị trí khác

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

21

2. Ngắt (Interrupt)

- Khái niệm chung về ngắt: Ngắt là cơ chế cho phép CPU tạm dừng chương trình đang thực hiện để chuyển sang thực hiện một chương trình con có sẵn trong bộ nhớ.
 - Chương trình con xử lý ngắt (Interrupt handlers)
- Các loại ngắt:
 - Các biệt lệ (Exceptions): gây ra do lỗi khi thực hiện chương trình (VD: tràn số, mã lệnh sai, ...)
 - Các ngắt từ bên ngoài (External Interrupts): do thiết bị vào-ra (thông qua mô-đun vào-ra) gửi tín hiệu ngắt đến CPU để yêu cầu trao đổi dữ liệu

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

22

Hoạt động với ngắt từ bên ngoài

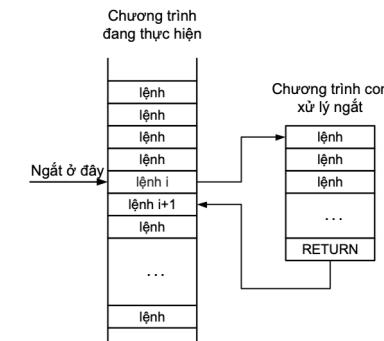
- Sau khi hoàn thành mỗi lệnh, bộ xử lý kiểm tra tín hiệu ngắt
- Nếu không có ngắt, bộ xử lý nhận lệnh tiếp theo của chương trình hiện tại
- Nếu có tín hiệu ngắt:
 - Tạm dừng (suspend) chương trình đang thực hiện
 - Cắt ngắt cảnh (các thông tin liên quan đến chương trình bị ngắt)
 - Thiết lập bộ đếm chương trình PC trả đến chương trình con xử lý ngắt tương ứng
 - Chuyển sang thực hiện chương trình con xử lý ngắt
 - Khôi phục ngữ cảnh và trả về tiếp tục thực hiện chương trình đang bị tạm dừng

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

23

Hoạt động ngắt (tiếp)

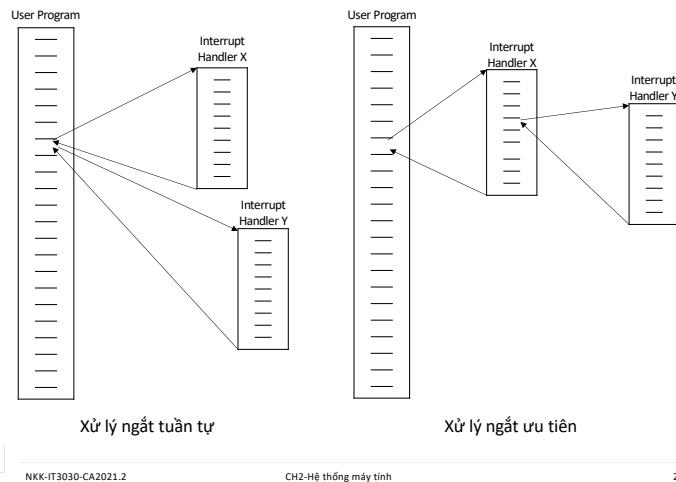


NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

24

Xử lý với nhiều tín hiệu yêu cầu ngắt



NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

25

3. Hoạt động vào-ra

- **Hoạt động vào-ra:** là hoạt động trao đổi dữ liệu giữa mô-đun vào-ra với bên trong máy tính.
- **Các kiểu hoạt động vào-ra:**
 - CPU trao đổi dữ liệu với mô-đun vào-ra bởi lệnh vào-ra trong chương trình
 - CPU trao quyền điều khiển cho phép mô-đun vào-ra trao đổi dữ liệu trực tiếp với bộ nhớ chính (DMA - Direct Memory Access).

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

26

2.3. Liên kết trong máy tính

1. Luồng thông tin trong máy tính

- Các mô-đun trong máy tính:
 - CPU
 - Mô-đun nhớ
 - Mô-đun vào-ra
- cần được kết nối với nhau

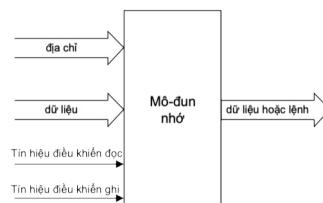
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

27

Kết nối mô-đun nhớ

- Địa chỉ đưa đến để xác định ngăn nhớ
- Dữ liệu được đưa đến khi ghi
- Dữ liệu hoặc lệnh được đưa ra khi đọc
 - Bộ nhớ không phân biệt lệnh và dữ liệu
- Nhận các tín hiệu điều khiển:
 - Điều khiển đọc (Read)
 - Điều khiển ghi (Write)



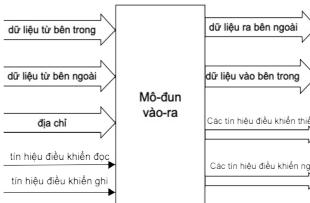
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

28

Kết nối mô-đun vào-ra

- Địa chỉ đưa đến để xác định cổng vào-ra
- Ra dữ liệu (Output)
 - Nhận dữ liệu từ bên trong (CPU hoặc bộ nhớ chính)
 - Đưa dữ liệu ra thiết bị vào-ra
- Vào dữ liệu (Input)
 - Nhận dữ liệu từ thiết bị vào-ra
 - Đưa dữ liệu vào bên trong (CPU hoặc bộ nhớ chính)
- Nhận các tín hiệu điều khiển từ CPU
- Phát các tín hiệu điều khiển đến thiết bị vào-ra
- Phát các tín hiệu ngắt đến CPU



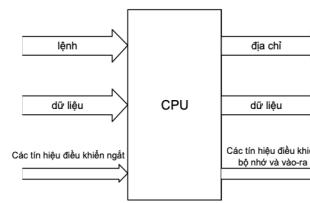
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

29

Kết nối CPU

- Phát địa chỉ đến các mô-đun nhớ hay các mô-đun vào-ra
- Đọc lệnh từ bộ nhớ
- Đọc dữ liệu từ bộ nhớ hoặc mô-đun vào-ra
- Đưa dữ liệu ra (sau khi xử lý) đến bộ nhớ hoặc mô-đun vào-ra
- Phát tín hiệu điều khiển đến các mô-đun nhớ và các mô-đun vào-ra
- Nhận các tín hiệu ngắt



NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

30

2. Liên kết bus trong máy tính

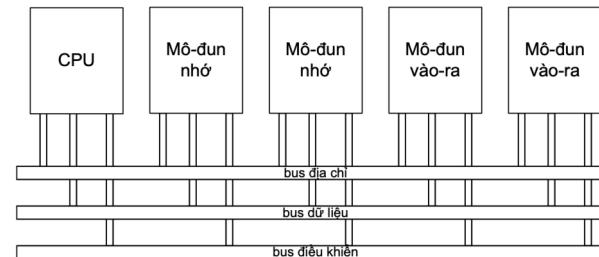
- Bus: tập hợp các đường kết nối để vận chuyển thông tin giữa các mô-đun của máy tính với nhau
- Các bus chức năng:
 - Bus địa chỉ (Address bus)
 - Bus dữ liệu (Data bus)
 - Bus điều khiển (Control bus)
- Độ rộng bus: là số đường dây của bus có thể truyền các bit thông tin đồng thời (chỉ dùng cho bus địa chỉ và bus dữ liệu)

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

31

Sơ đồ cấu trúc bus cơ bản



NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

32

Bus địa chỉ

- Chức năng: vận chuyển địa chỉ để xác định vị trí ngăn nhớ hay cổng vào-ra
- Độ rộng bus địa chỉ:
 - N bit: $A_{N-1}, A_{N-2}, \dots, A_2, A_1, A_0$
 - Số lượng địa chỉ tối đa được sử dụng là: 2^N địa chỉ (không gian địa chỉ)
 - Địa chỉ nhỏ nhất: 00 ... 000₍₂₎
 - Địa chỉ lớn nhất: 11 ... 111₍₂₎
- Ví dụ:
 - Máy tính sử dụng bus địa chỉ 32-bit ($A_{31}-A_0$), bộ nhớ chính được đánh địa chỉ cho từng byte
 - Có khả năng đánh địa chỉ cho 2^{32} bytes nhớ = 4GiB

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

33

Bus dữ liệu

- Chức năng:
 - vận chuyển lệnh từ bộ nhớ đến CPU
 - vận chuyển dữ liệu giữa các thành phần của máy tính với nhau
- Độ rộng bus dữ liệu: số bit được truyền đồng thời
 - M bit: $D_{M-1}, D_{M-2}, \dots, D_2, D_1, D_0$
 - M thường là 8, 16, 32, 64 bit
- Ví dụ:
 - Máy tính có bus dữ liệu kết nối CPU với bộ nhớ là 64-bit
 - Có thể trao đổi 8 byte nhớ ở một thời điểm

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

34

Bus điều khiển

- Chức năng: vận chuyển các tín hiệu điều khiển
- Các loại tín hiệu điều khiển:
 - Các tín hiệu điều khiển đọc/ghi
 - Các tín hiệu điều khiển ngắn
 - Các tín hiệu điều khiển bus

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

35

Một số tín hiệu điều khiển điển hình

- Các tín hiệu (phát ra từ CPU) điều khiển đọc/ghi:
 - Memory Read (MEMR): Tín hiệu điều khiển đọc dữ liệu từ một ngăn nhớ có địa chỉ xác định đưa lên bus dữ liệu (chuyển vào CPU).
 - Memory Write (MEMW): Tín hiệu điều khiển ghi dữ liệu có sẵn trên bus dữ liệu (từ CPU) đến một ngăn nhớ có địa chỉ xác định.
 - I/O Read (IOR): Tín hiệu điều khiển đọc dữ liệu từ một cổng vào-ra có địa chỉ xác định đưa lên bus dữ liệu (chuyển vào CPU).
 - I/O Write (IOW): Tín hiệu điều khiển ghi dữ liệu có sẵn trên bus dữ liệu (từ CPU) ra một cổng có địa chỉ xác định.

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

36

Một số tín hiệu điều khiển hiển thị (tiếp)

- Các tín hiệu điều khiển ngắn:

- Interrupt Request (INTR): Tín hiệu từ bộ điều khiển vào-ra gửi đến yêu cầu ngắt CPU để trao đổi vào-ra. Tín hiệu INTR có thể bị che.
- Interrupt Acknowledge (INTA): Tín hiệu phát ra từ CPU báo cho bộ điều khiển vào-ra biết CPU chấp nhận ngắt để trao đổi vào-ra.
- Non Maskable Interrupt (NMI): tín hiệu ngắt không che được gửi đến ngắt CPU.
- Reset: Tín hiệu từ bên ngoài gửi đến CPU và các thành phần khác để khởi động lại máy tính.

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

37

Một số tín hiệu điều khiển hiển thị (tiếp)

- Các tín hiệu điều khiển bus:

- Bus Request (BRQ) : Tín hiệu từ mô-đun vào-ra gửi đến yêu cầu CPU chuyển nhượng quyền sử dụng bus.
- Bus Grant (BGT): Tín hiệu phát ra từ CPU chấp nhận chuyển nhượng quyền sử dụng bus cho mô-đun vào-ra.
- Lock/ Unlock: Tín hiệu cấm/cho-phép xin chuyển nhượng bus.

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

38

Đặc điểm của bus dùng chung (shared bus)

- Nhiều mô-đun kết nối vào bus chung → cần có bộ phân xử bus
- Bus chỉ phục vụ được một yêu cầu trao đổi dữ liệu tại một thời điểm → độ trễ lớn
- Bus phải có tốc độ bằng tốc độ bus của mô-đun nhanh nhất trong hệ thống
- Khắc phục:
 - Đa bus (Multiple bus): chia thành nhiều bus:
 - Bus cho bộ xử lý
 - Bus cho bộ nhớ
 - Bus vào-ra
 - Liên kết điểm-điểm (Point to point interconnection):
 - Sử dụng phổ biến trên các máy tính hiện nay

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

39

3. Liên kết điểm-điểm

- Với bus dùng chung, khi tốc độ dữ liệu ngày càng cao, khó thực hiện các chức năng đồng bộ hóa và phân xử bus kịp thời
- Kết nối điểm-điểm có độ trễ nhỏ hơn, tốc độ dữ liệu cao hơn và khả năng mở rộng tốt hơn
- Các loại kết nối điểm-điểm phổ biến:
 - QPI – Quick Path Interconnect
 - PCIe – PCI express

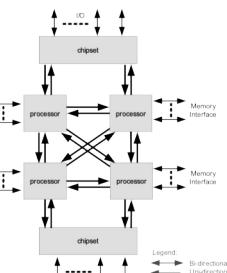
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

40

QPI - Quick Path Interconnect

- Được giới thiệu vào năm 2008
- Đa kết nối trực tiếp:
 - Kết nối trực tiếp từng cặp, loại bỏ việc phân xử bus
- Kiến trúc giao thức phân lớp
 - Kết nối các bộ xử lý với nhau và với chipset sử dụng kiến trúc giao thức phân lớp
- Truyền dữ liệu dạng gói



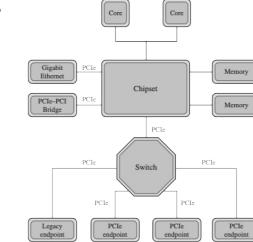
NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

41

PCI express

- PCI bus: (Peripheral Component Interconnect):**
 - Bus vào-ra với băng thông rộng độc lập với bộ xử lý để kết nối với các thiết bị vào-ra
 - Bus dùng chung
- PCI Express (PCIe)**
 - Liên kết điểm-điểm thay thế cho PCI bus
 - Kiến trúc giao thức phân lớp
 - Tốc độ cao
 - PCIe 1.0 – 6.0

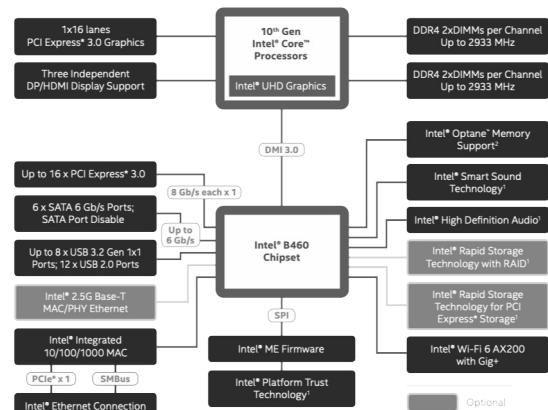


NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

42

Ví dụ liên kết trong máy tính Intel



NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

43

Kiến trúc máy tính

Hết chương 2

NKK-IT3030-CA2021.2

CH2-Hệ thống máy tính

44



Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

2

Kiến trúc máy tính

Chương 3 SỐ HỌC MÁY TÍNH

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

3

Nội dung

- 3.1. Hệ đếm (ôn tập)
- 3.2. Cơ bản về logic số (*)
- 3.3. Biểu diễn số nguyên
- 3.4. Phép cộng và phép trừ số nguyên
- 3.5. Phép nhân và chia số nguyên
- 3.6. Số dấu phẩy động

(*) dành cho sinh viên KHMT

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

4

3.1. Hệ đếm

Các hệ đếm cơ bản:

- Hệ thập phân (Decimal System)
 - con người sử dụng
- Hệ nhị phân (Binary System)
 - máy tính sử dụng
- Hệ mươi sáu (Hexadecimal System)
 - dùng để viết gọn cho số nhị phân

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

5

1. Hệ thập phân

- Cơ số 10
- 10 chữ số: 0,1,2,3,4,5,6,7,8,9
- Dùng n chữ số thập phân có thể biểu diễn được 10^n giá trị khác nhau:
 - 00...000 = 0
 - 99...999 = $10^n - 1$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

6

Dạng tổng quát của số thập phân

$$A = a_n a_{n-1} \dots a_1 a_0, a_{-1} a_{-2} \dots a_{-m}$$

Giá trị của A được hiểu như sau:

$$A = a_n 10^n + a_{n-1} 10^{n-1} + \dots + a_1 10^1 + a_0 10^0 + a_{-1} 10^{-1} + \dots + a_{-m} 10^{-m}$$

$$A = \sum_{i=-m}^n a_i 10^i$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

7

Ví dụ số thập phân

$$472.38 = 4 \times 10^2 + 7 \times 10^1 + 2 \times 10^0 + 3 \times 10^{-1} + 8 \times 10^{-2}$$

- Các chữ số của phần nguyên:

$$\begin{array}{rcl} 472 : 10 = 47 & \text{dư} & 2 \\ 47 : 10 = 4 & \text{dư} & 7 \\ 4 : 10 = 0 & \text{dư} & 4 \end{array}$$

- Các chữ số của phần lẻ:

$$\begin{array}{rcl} 0.38 \times 10 = 3.8 & \text{phần nguyên} & 3 \\ 0.8 \times 10 = 8.0 & \text{phần nguyên} & 8 \end{array}$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

8

2. Hệ nhị phân

- Cơ số 2
- 2 chữ số nhị phân: 0 và 1
- Chữ số nhị phân được gọi là **bit** (binary digit)
 - bit là đơn vị thông tin nhỏ nhất
- Dùng n bit có thể biểu diễn được 2^n giá trị khác nhau:
 - 00...000 = 0
 - 11...111 = $2^n - 1$
- Các lệnh của chương trình và dữ liệu trong máy tính đều được mã hóa bằng số nhị phân

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

9

Biểu diễn số nhị phân

Số nhị phân				Số thập phân
1-bit	2-bit	3-bit	4-bit	
0	00	000	0000	0
1	01	001	0001	1
	10	010	0010	2
	11	011	0011	3
		100	0100	4
		101	0101	5
		110	0110	6
		111	0111	7
			1000	8
			1001	9
			1010	10
			1011	11
			1100	12
			1101	13
			1110	14
			1111	15

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

10

Đơn vị dữ liệu và thông tin trong máy tính

- bit – chữ số nhị phân (binary digit): là đơn vị thông tin nhỏ nhất, cho phép nhận một trong hai giá trị: 0 hoặc 1.
- byte là một tổ hợp 8 bit: có thể biểu diễn được 256 giá trị (2^8)
- Qui ước đơn vị dữ liệu trong Khoa học máy tính:
 - KB (Kilobyte) = 2^{10} bytes = 1024 bytes
 - MB (Megabyte) = 2^{10} KB = 2^{20} bytes ($\sim 10^6$)
 - GB (Gigabyte) = 2^{10} MB = 2^{30} bytes ($\sim 10^9$)
 - TB (Terabyte) = 2^{10} GB = 2^{40} bytes ($\sim 10^{12}$)
 - PB (Petabyte) = 2^{10} TB = 2^{50} bytes
 - EB (Exabyte) = 2^{10} PB = 2^{60} bytes

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

11

Qui ước mới về ký hiệu đơn vị dữ liệu

Theo thập phân			Theo nhị phân		
Đơn vị	Viết tắt	Giá trị	Đơn vị	Viết tắt	Giá trị
kilobyte	KB	10^3	kibibyte	KiB	$2^{10} = 1024$
megabyte	MB	10^6	mebibyte	MiB	2^{20}
gigabyte	GB	10^9	gibibyte	GiB	2^{30}
terabyte	TB	10^{12}	tebibyte	TiB	2^{40}
petabyte	PB	10^{15}	pebibyte	PiB	2^{50}
exabyte	EB	10^{18}	exbibyte	EiB	2^{60}

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

12

Dạng tổng quát của số nhị phân

$$A = a_n a_{n-1} \dots a_1 a_0, a_{-1} a_{-2} \dots a_{-m}$$

Giá trị của A được tính như sau:

$$A = a_n 2^n + a_{n-1} 2^{n-1} + \dots + a_1 2^1 + a_0 2^0 + a_{-1} 2^{-1} + \dots + a_{-m} 2^{-m}$$

$$A = \sum_{i=-m}^n a_i 2^i$$

Ví dụ số nhị phân

$$1101001.1011_{(2)} =$$

$$\begin{array}{cccccccccc} 6 & 5 & 4 & 3 & 2 & 1 & 0 & -1 & -2 & -3 & -4 \\ & & & & & & & & & & \end{array} = 2^6 + 2^5 + 2^3 + 2^0 + 2^{-1} + 2^{-3} + 2^{-4}$$

$$= 64 + 32 + 8 + 1 + 0.5 + 0.125 + 0.0625$$

$$= 105.6875_{(10)}$$

Chuyển đổi số nguyên thập phân sang nhị phân

- Phương pháp 1: chia dần cho 2 rồi lấy phần dư
- Phương pháp 2: Phân tích thành tổng của các số 2^i
→ nhanh hơn

Phương pháp chia dần cho 2

- Ví dụ: chuyển đổi $105_{(10)}$

▪ $105 : 2 =$	52	dư	1	\uparrow biểu diễn số dư theo chiều mũi tên
▪ $52 : 2 =$	26	dư	0	
▪ $26 : 2 =$	13	dư	0	
▪ $13 : 2 =$	6	dư	1	
▪ $6 : 2 =$	3	dư	0	
▪ $3 : 2 =$	1	dư	1	
▪ $1 : 2 =$	0	dư	1	

$$▪ Kết quả: $105_{(10)} = 1101001_{(2)}$$$

Phương pháp phân tích thành tổng của các 2^i

- Ví dụ 1: chuyển đổi $105_{(10)}$

$$105 = 64 + 32 + 8 + 1 = 2^6 + 2^5 + 2^3 + 2^0$$

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
0	1	1	0	1	0	0	1

Kết quả: $105_{(10)} = 0110\ 1001_{(2)}$

- Ví dụ 2:

$$17000_{(10)} = 16384 + 512 + 64 + 32 + 8$$

$$= 2^{14} + 2^9 + 2^6 + 2^5 + 2^3$$

$$17000_{(10)} = 0100\ 0010\ 0110\ 1000_{(2)}$$

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

17

Chuyển đổi số lẻ thập phân sang nhị phân

- Ví dụ 1: chuyển đổi $0.6875_{(10)}$

$0.6875 \times 2 = 1.375$	phần nguyên = 1	biểu diễn theo chiều mũi tên
$0.375 \times 2 = 0.75$	phần nguyên = 0	
$0.75 \times 2 = 1.5$	phần nguyên = 1	
$0.5 \times 2 = 1.0$	phần nguyên = 1	

Kết quả: $0.6875_{(10)} = 0.1011_{(2)}$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

18

Chuyển đổi số lẻ thập phân sang nhị phân (tiếp)

- Ví dụ 2: chuyển đổi $0.81_{(10)}$

$0.81 \times 2 = 1.62$	phần nguyên = 1	↓
$0.62 \times 2 = 1.24$	phần nguyên = 1	
$0.24 \times 2 = 0.48$	phần nguyên = 0	
$0.48 \times 2 = 0.96$	phần nguyên = 0	
$0.96 \times 2 = 1.92$	phần nguyên = 1	
$0.92 \times 2 = 1.84$	phần nguyên = 1	
$0.84 \times 2 = 1.68$	phần nguyên = 1	
$0.81_{(10)} \approx 0.1100111_{(2)}$		

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

19

3. Hệ mười sáu (Hexa)

- Cơ số 16

- 16 chữ số: 0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F

- Dùng để viết gọn cho số nhị phân: cứ một nhóm 4-bit sẽ được thay bằng một chữ số Hexa

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

20

Quan hệ giữa số nhị phân và số Hexa

Ví dụ:

- $1011\ 0011_{(2)} = B3_{(16)}$
- $0000\ 0000_{(2)} = 00_{(16)}$

- $0010\ 1101\ 1001\ 1010_{(2)} = 2D9A_{(16)}$
- $1111\ 1111\ 1111\ 1111_{(2)} = FFFF_{(16)}$

4-bit	Số Hexa	Thập phân
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

21

1. Đại số Boolean

- Đại số Boolean sử dụng các biến logic và phép toán logic
- Biến logic có thể nhận giá trị 1 (TRUE) hoặc 0 (FALSE)
- Các phép toán logic cơ bản: AND, OR và NOT
 - $A \text{ AND } B = A \cdot B$ hay AB
 - $A \text{ OR } B = A + B$
 - $\text{NOT } A = A$
 - Thứ tự ưu tiên: NOT > AND > OR
- Thêm các phép toán logic: NAND, NOR, XOR
 - $A \text{ NAND } B = \text{NOT}(A \text{ AND } B) = \overline{A \cdot B}$
 - $A \text{ NOR } B = \text{NOT}(A \text{ OR } B) = \overline{A + B}$
 - $A \text{ XOR } B = A \oplus B = A \cdot B + A \cdot B$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

23

3.2. Cơ bản về logic số

- Đại số Boolean
- Các cổng logic
- Mạch tổ hợp
- Mạch dãy

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

22

Phép toán đại số Boolean

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

A	NOT A
0	1
1	0

NOT là phép toán 1 biến

A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

24

Các đồng nhất thức của đại số Boolean

$$\begin{aligned} A \cdot B &= B \cdot A \\ A \cdot (B + C) &= (A \cdot B) + (A \cdot C) \\ 1 \cdot A &= A \\ A \cdot A &= 0 \\ 0 \cdot A &= 0 \\ A \cdot A &= A \\ A \cdot (B \cdot C) &= (A \cdot B) \cdot C \\ \overline{A \cdot B} &= A + B \text{ (Định lý De Morgan)} \end{aligned}$$

$$\begin{aligned} A + B &= B + A \\ A + (B \cdot C) &= (A + B) \cdot (A + C) \\ 0 + A &= A \\ A + A &= 1 \\ 1 + A &= 1 \\ A + A &= A \\ A + (B + C) &= (A + B) + C \\ \overline{A + B} &= A \cdot B \text{ (Định lý De Morgan)} \end{aligned}$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

25

2. Các cỗng logic (Logic Gates)

- Thực hiện các phép toán logic:
 - NOT, AND, OR, NAND, NOR, XOR
- Cỗng logic một đầu vào:
 - Cỗng NOT
- Cỗng hai đầu vào:
 - AND, OR, XOR, NAND, NOR
- Cỗng nhiều đầu vào:
 - AND, OR, XOR, NAND, NOR

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

26

Ký hiệu các cỗng logic

Name	Graphical Symbol	Algebraic Function	Truth Table
AND		$F = A \cdot B$ or $F = AB$	$\begin{array}{ c c c } \hline \bar{A} & B & F \\ \hline 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ \hline \end{array}$
OR		$F = A + B$	$\begin{array}{ c c c } \hline \bar{A} & B & F \\ \hline 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ \hline \end{array}$
NOT		$F = \bar{A}$ or $F = A'$	$\begin{array}{ c c } \hline \bar{A} & F \\ \hline 0 & 1 \\ 1 & 0 \\ \hline \end{array}$
NAND		$F = \bar{A} \cdot \bar{B}$	$\begin{array}{ c c c } \hline \bar{A} & B & F \\ \hline 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ \hline \end{array}$
NOR		$F = \bar{A} + \bar{B}$	$\begin{array}{ c c c } \hline \bar{A} & B & F \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ \hline \end{array}$
XOR		$F = A \oplus B$	$\begin{array}{ c c c } \hline \bar{A} & B & F \\ \hline 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ \hline \end{array}$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

27

3. Mạch tổ hợp

- Mạch logic là mạch bao gồm:
 - Các đầu vào (Inputs)
 - Các đầu ra (Outputs)
 - Đặc tả chức năng (Functional specification)
 - Đặc tả thời gian (Timing specification)
- Các kiểu mạch logic:
 - Mạch tổ hợp (Combinational Circuits)
 - Mạch không nhớ
 - Đầu ra được xác định bởi các giá trị hiện tại của đầu vào
 - Mạch dãy (Sequential Circuits)
 - Mạch có nhớ
 - Đầu ra được xác định bởi các giá trị trước đó và giá trị hiện tại của đầu vào

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

28

Mạch tổ hợp

- Mạch tổ hợp là mạch logic trong đó đầu ra chỉ phụ thuộc đầu vào ở thời điểm hiện tại
- Là mạch không nhớ và được thực hiện bằng các cổng logic
- Mạch tổ hợp có thể được định nghĩa theo ba cách:
 - Bảng chân lý (Truth Table)
 - Dạng sơ đồ
 - Phương trình Boolean

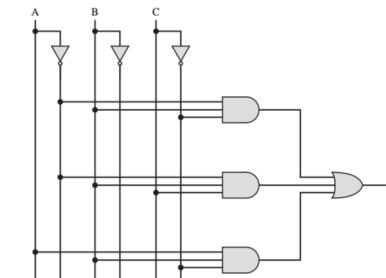
NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

29

Ví dụ

Đầu vào			Đầu ra
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0



$$F = \bar{A}\bar{B}\bar{C} + \bar{A}BC + AB\bar{C}$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

30

Bộ chọn kênh (Multiplexer - MUX)

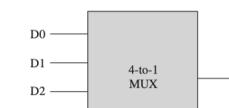
- 2^n đầu vào dữ liệu (D)
- n đầu vào chọn (S)
- 1 đầu ra dữ liệu (F)
- Mỗi tổ hợp đầu vào chọn (S) xác định đầu vào dữ liệu nào (D) sẽ được nối với đầu ra (F)

NKK-IT3030-CA2021.2

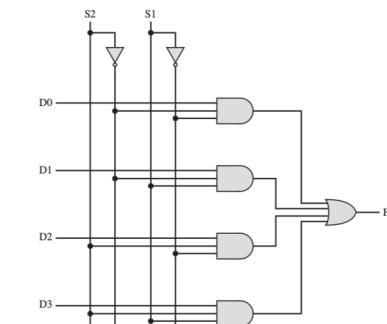
CH3-Số học và logic máy tính

31

Bộ chọn kênh 4 đầu vào



Đầu vào chọn		Đầu ra
S2	S1	F
0	0	D0
0	1	D1
1	0	D2
1	1	D3



$$F = D0 \cdot \bar{S2} \cdot \bar{S1} + D1 \cdot \bar{S2} \cdot S1 + D2 \cdot S2 \cdot \bar{S1} + D3 \cdot S2 \cdot S1$$

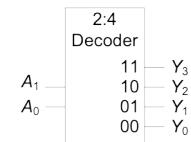
NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

32

Bộ giải mã (Decoder)

- N đầu vào, 2^N đầu ra
- Với một tổ hợp của N đầu vào, chỉ có một đầu ra tích cực (khác với các đầu ra còn lại)
- Ví dụ: Bộ giải mã 2 ra 4

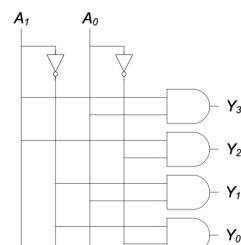


A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

33



Bộ cộng

- Bộ cộng bán phần 1-bit (Half-adder)
 - Cộng hai bit tạo ra bit tổng và bit nhớ ra
- Bộ cộng toàn phần 1-bit (Full-adder)
 - Cộng 3 bit
 - Cho phép xây dựng bộ cộng N-bit

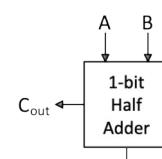
NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

34

Bộ cộng bán phần 1-bit

$$\begin{array}{r} 0 \quad 0 \\ + 0 \quad + 1 \\ \hline 0 \quad 1 \end{array} \quad \begin{array}{r} 1 \quad 1 \\ + 0 \quad + 1 \\ \hline 1 \quad 0 \end{array}$$



$$S = A \oplus B$$

$$C_{out} = AB$$

Đầu vào		Đầu ra	
A	B	S	C _{out}
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

35

Bộ cộng toàn phần 1-bit

$$S = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + ABC + A\bar{B}\bar{C}$$

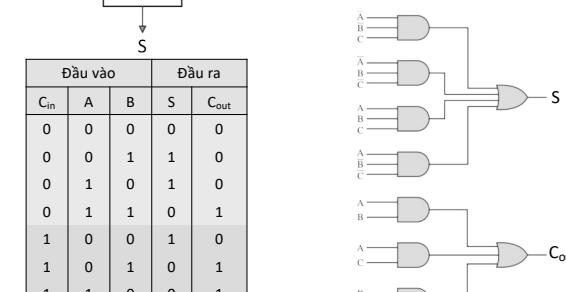
$$C_{out} = AB + AC + BC$$

Đầu vào		Đầu ra		
C _{in}	A	B	S	C _{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

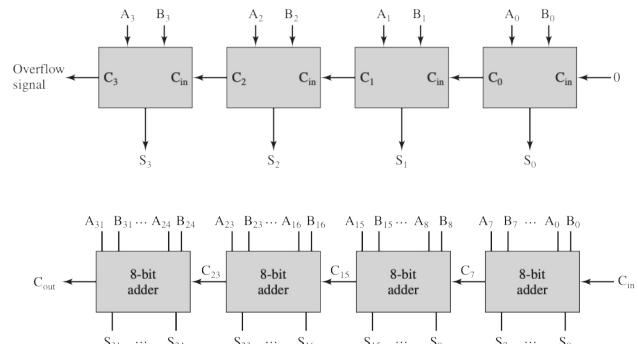
NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

36



Bộ cộng 4-bit và bộ cộng 32-bit



NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

37

4. Mạch dãy

- Mạch dãy là mạch logic trong đó đầu ra phụ thuộc giá trị đầu vào ở thời điểm hiện tại và đầu vào ở thời điểm quá khứ
- Là mạch có nhớ, được thực hiện bằng phần tử nhớ (Flip-Flop) và có thể kết hợp với các cổng logic

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

38

Các Flip-Flop cơ bản

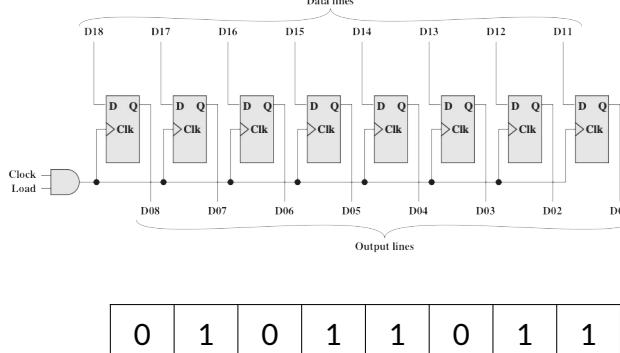
Name	Graphical Symbol	Truth Table
S-R		
J-K		
D		

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

39

Thanh ghi 8-bit song song



NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

40

3.3. Biểu diễn số nguyên

- Số nguyên không dấu (Unsigned Integer)
- Số nguyên có dấu (Signed Integer)

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

41

1. Biểu diễn số nguyên không dấu

Nguyên tắc tổng quát: Dùng n bit biểu diễn số nguyên không dấu A:

$$A = a_{n-1}a_{n-2}\dots a_2a_1a_0$$

Giá trị của A được tính như sau:

$$A = \sum_{i=0}^{n-1} a_i 2^i$$

Dải biểu diễn của A: [0, $2^n - 1$]

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

42

Ví dụ 1

- Biểu diễn các số nguyên không dấu sau đây bằng 8-bit:

$$A = 41 ; B = 150$$

Giải:

$$A = 41 = 32 + 8 + 1 = 2^5 + 2^3 + 2^0$$

$$41 = 0010\ 1001$$

$$B = 150 = 128 + 16 + 4 + 2 = 2^7 + 2^4 + 2^2 + 2^1$$

$$150 = 1001\ 0110$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

43

Ví dụ 2

- Cho các số nguyên không dấu M, N được biểu diễn bằng 8-bit như sau:

$$M = 0001\ 0010$$

$$N = 1011\ 1001$$

Xác định giá trị của chúng ?

Giải:

$$M = 0001\ 0010 = 2^4 + 2^1 = 16 + 2 = 18$$

$$N = 1011\ 1001 = 2^7 + 2^5 + 2^4 + 2^3 + 2^0$$

$$= 128 + 32 + 16 + 8 + 1 = 185$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

44

Với n = 8 bit

- Biểu diễn được các giá trị từ 0 đến 255 ($2^8 - 1$)

Chú ý:

$$\begin{array}{r} 1111\ 1111 \\ + \underline{0000\ 0001} \\ \hline 1\ 0000\ 0000 \end{array}$$

có tràn nhớ ra ngoài (Carry out)

$$255 + 1 = 0 ???$$

do vượt ra khỏi dải biểu diễn

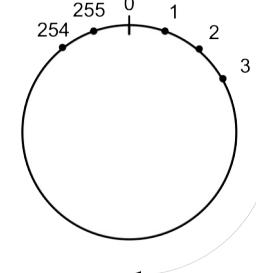
Biểu diễn nhị phân	Giá trị thập phân
0000 0000	0
0000 0001	1
0000 0010	2
0000 0011	3
0000 0100	4
...	
1111 1110	254
1111 1111	255

Trục số học với n = 8 bit

Trục số học:



Trục số học máy tính:

**Với n = 16 bit, 32 bit, 64 bit**

- n= 16 bit: dải biểu diễn từ 0 đến 65535 ($2^{16} - 1$)

- 0000 0000 0000 0000 = 0
- ...
- 0000 0000 1111 1111 = 255
- 0000 0001 0000 0000 = 256
- ...
- 1111 1111 1111 1111 = 65535

- n= 32 bit: dải biểu diễn từ 0 đến $2^{32} - 1$

- n= 64 bit: dải biểu diễn từ 0 đến $2^{64} - 1$

2. Biểu diễn số nguyên có dấu

Số bù một và Số bù hai

- Định nghĩa:** Cho một số nhị phân A được biểu diễn bằng n bit, ta có:

- Số bù một của A = $(2^n - 1) - A$
- Số bù hai của A = $2^n - A$
- Số bù hai của A = (Số bù một của A) + 1

Ví dụ

Với $n = 8$ bit, cho $A = 0010\ 0101$

- Số bù một của A được tính như sau:

$$1111\ 1111 \quad (2^8 - 1)$$

$$- \underline{0010\ 0101} \quad (A)$$

$$1101\ 1010$$

→ đảo giá trị các bit của A

- Số bù hai của A được tính như sau:

$$1\ 0000\ 0000 \quad (2^8)$$

$$- \underline{0010\ 0101} \quad (A)$$

$$1101\ 1011$$

→ thực hiện khó khăn

Quy tắc tìm Số bù một và Số bù hai

- Số bù một của A = đảo giá trị các bit của A
- (Số bù hai của A) = (Số bù một của A) + 1

Ví dụ:

$$\begin{array}{rcl} \text{Cho} & A & = 0010\ 0101 \\ \text{Số bù một của } A & = & 1101\ 1010 \\ & & + \underline{1} \end{array}$$

$$\begin{array}{rcl} \text{Số bù hai của } A & = & 1101\ 1011 \end{array}$$

Nhận xét:

$$\begin{array}{rcl} A & = & 0010\ 0101 \\ \text{Số bù hai của } A & = & + \underline{1101\ 1011} \\ 1\ 0000\ 0000 & = & 0 \end{array}$$

(bỏ qua bit nhớ ra ngoài)

→ Số bù hai của $A = -A$

Biểu diễn số nguyên có dấu theo mã bù hai

Nguyên tắc tổng quát: Dùng n bit biểu diễn số nguyên có dấu A :

$$A = a_{n-1}a_{n-2}\dots a_2a_1a_0$$

- Với A là số dương: bit $a_{n-1} = 0$, các bit còn lại biểu diễn độ lớn như số không dấu
- Với A là số âm: được biểu diễn bởi số bù hai của số dương tương ứng, vì vậy bit $a_{n-1} = 1$

Ví dụ

- Biểu diễn các số nguyên có dấu sau đây bằng 8-bit:
 $A = +58 ; B = -80$

Giai:

$$A = +58 = 0011\ 1010$$

$$B = -80$$

$$\begin{array}{rcl} \text{Ta có: } +80 & = & 0101\ 0000 \end{array}$$

$$\begin{array}{rcl} \text{Số bù một} & = & 1010\ 1111 \end{array}$$

$$\begin{array}{rcl} & & + \underline{1} \\ \text{Số bù hai} & = & 1011\ 0000 \end{array}$$

$$\begin{array}{rcl} \text{Vậy: } B = -80 & = & 1011\ 0000 \end{array}$$

Xác định giá trị của số dương

Dạng tổng quát của số dương:

$$A = 0a_{n-2} \dots a_2 a_1 a_0$$

Giá trị của số dương:

$$A = \sum_{i=0}^{n-2} a_i 2^i$$

Dải biểu diễn của số dương: $[0, (2^{n-1} - 1)]$

Xác định giá trị của số âm

Dạng tổng quát của số âm:

$$A = 1a_{n-2} \dots a_2 a_1 a_0$$

Giá trị của số âm:

$$A = -2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$$

Dải biểu diễn của số âm: $[-2^{n-1}, -1]$

Công thức xác định giá trị số âm

$$\begin{aligned} A &= 1a_{n-2} a_{n-3} \dots a_2 a_1 a_0 \\ -A &= 0\overline{a_{n-2}} \overline{a_{n-3}} \dots \overline{a_2} \overline{a_1} \overline{a_0} + 1 \\ &= \underbrace{11 \dots 111}_{n-1} - a_{n-2} a_{n-3} \dots a_2 a_1 a_0 + 1 \end{aligned}$$

$$= (2^{n-1} - 1) - \left(\sum_{i=0}^{n-2} a_i 2^i \right) + 1$$

$$\text{Vậy } A = -2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$$

Công thức tổng quát cho số nguyên có dấu

Dạng tổng quát của số nguyên có dấu A:

$$A = a_{n-1} a_{n-2} \dots a_2 a_1 a_0$$

Giá trị của A được tính như sau:

$$A = -a_{n-1} 2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i$$

Dải biểu diễn: $[-2^{n-1}, +(2^{n-1} - 1)]$

Ví dụ

- Hãy xác định giá trị của các số nguyên có dấu được biểu diễn theo mã bù hai với 8-bit như dưới đây:

- P = 0110 0010
- Q = 1101 1011

Giải:

- $P = 0110\ 0010 = 2^6 + 2^5 + 2^1 = 64 + 32 + 2 = +98$
- $Q = 1101\ 1011 = -2^7 + 2^6 + 2^4 + 2^3 + 2^1 + 2^0 = -128 + 64 + 16 + 8 + 2 + 1 = -37$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

57

Với n = 8 bit

- Biểu diễn được các giá trị từ -2^7 đến $+2^7-1$
- 128 đến +127
- Chỉ có một giá trị 0
- Không biểu diễn cho giá trị +128

Chú ý:

$+127 + 1 = -128$

$(-128) + (-1) = +127$

có tràn xảy ra (Overflow)

(do vượt ra khỏi dài biểu diễn)

Giá trị thập phân	Biểu diễn bù hai
0	0000 0000
+1	0000 0001
+2	0000 0010
	...
+126	0111 1110
+127	0111 1111
-128	1000 0000
-127	1000 0001
	...
-2	1111 1110
-1	1111 1111

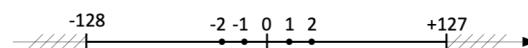
NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

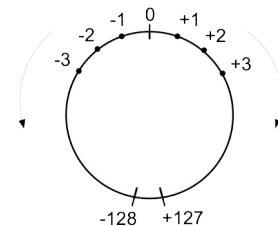
58

Trục số học số nguyên có dấu với n = 8 bit

- Trục số học:



- Trục số học máy tính:



NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

59

Với n = 16 bit, 32 bit, 64 bit

- Với n = 16bit: biểu diễn từ -2^{15} đến $2^{15}-1$

- $0000\ 0000\ 0000\ 0000 = 0$
- $0000\ 0000\ 0000\ 0001 = +1$
- ...
- $0111\ 1111\ 1111\ 1111 = +32767 \quad (2^{15} - 1)$
- $1000\ 0000\ 0000\ 0000 = -32768 \quad (-2^{15})$
- $1000\ 0000\ 0000\ 0001 = -32767$
- ...
- $1111\ 1111\ 1111\ 1111 = -1$

- Với n = 32bit: biểu diễn từ -2^{31} đến $2^{31}-1$

- Với n = 64bit: biểu diễn từ -2^{63} đến $2^{63}-1$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

60

Mở rộng bit cho số nguyên

- Mở rộng số không dấu (Zero-extended): thêm các bit 0 vào bên trái
- Mở rộng số có dấu (Sign-extended):

Số dương:

$$+19 = \begin{array}{r} 0001\ 0011 \\ \text{(8bit)} \end{array}$$

$$+19 = \begin{array}{r} 0000\ 0000\ 0001\ 0011 \\ \text{(16bit)} \end{array}$$

→ thêm các bit 0 vào bên trái

Số âm:

$$-19 = \begin{array}{r} 1110\ 1101 \\ \text{(8bit)} \end{array}$$

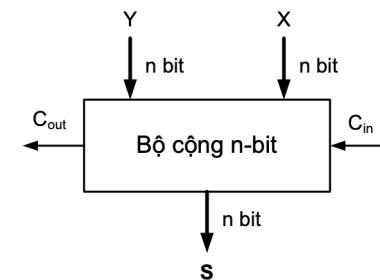
$$-19 = \begin{array}{r} 1111\ 1111\ 1110\ 1101 \\ \text{(16bit)} \end{array}$$

→ thêm các bit 1 vào bên trái

3.4. Phép cộng/trừ với số nguyên

1. Phép cộng số nguyên không dấu

Bộ cộng n-bit



Nguyên tắc cộng số nguyên không dấu

- Khi cộng hai số nguyên không dấu n-bit, kết quả nhận được là n-bit:
 - Nếu $C_{out} = 0$ → nhận được kết quả đúng
 - Nếu $C_{out} = 1$ → nhận được kết quả sai, do có nhớ ra ngoài (Carry Out)
- Hiện tượng nhớ ra ngoài xảy ra khi: tổng $> (2^n - 1)$

Ví dụ cộng số nguyên không dấu

$$\begin{array}{r} 57 = 0011\ 1001 \\ + 34 = + \underline{0010\ 0010} \\ \hline 91 \quad 0101\ 1011 = 64+16+8+2+1=91 \rightarrow \text{đúng} \end{array}$$

$$\begin{array}{r} 209 = 1101\ 0001 \\ + 73 = + \underline{0100\ 1001} \\ \hline 282 \quad 1\ 0001\ 1010 \\ \text{kết quả} = 0001\ 1010 = 16+8+2=26 \rightarrow \text{sai} \\ \text{do có nhớ ra ngoài} (C_{out}=1) \end{array}$$

Để có kết quả đúng, ta thực hiện cộng theo 16-bit:

$$\begin{array}{r} 209 = 0000\ 0000\ 1101\ 0001 \\ + 73 = + \underline{0000\ 0000\ 0100\ 1001} \\ \hline 0000\ 0001\ 0001\ 1010 \\ = 256+16+8+2 = 282 \end{array}$$

2. Phép đảo dấu

▪ Ta có:

$$\begin{array}{rcl} + 37 & = & 0010\ 0101 \\ \text{bù một} & = & 1101\ 1010 \\ \text{bù hai} & = & \begin{array}{c} + \\ \hline 1 \end{array} \\ & & 1101\ 1011 = -37 \end{array}$$

▪ Lấy bù hai của số âm:

$$\begin{array}{rcl} - 37 & = & 1101\ 1011 \\ \text{bù một} & = & 0010\ 0100 \\ \text{bù hai} & = & \begin{array}{c} + \\ \hline 1 \end{array} \\ & & 0010\ 0101 = +37 \end{array}$$

▪ Kết luận: Phép đảo dấu số nguyên trong máy tính thực chất là lấy bù hai

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

65

3. Cộng số nguyên có dấu

▪ Khi cộng hai số nguyên có dấu n-bit, kết quả nhận được là n-bit và không cần quan tâm đến bit C_{out}

- Khi cộng hai số khác dấu thì kết quả luôn luôn đúng
- Khi cộng hai số cùng dấu, nếu dấu kết quả cùng dấu với các số hạng thì kết quả là đúng
- Khi cộng hai số cùng dấu, nếu kết quả có dấu ngược lại, khi đó có tràn (Overflow) xảy ra và kết quả bị sai

▪ Hiện tượng tràn xảy ra khi tổng nằm ngoài dải biểu diễn: $[-(2^{n-1}), +(2^{n-1}-1)]$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

66

Ví dụ cộng số nguyên có dấu không tràn

- $(+70) = 0100\ 0110$
 $+ (+42) = \underline{0010\ 1010}$
 $+ 112 \quad 0111\ 0000 = +112$
- $(+97) = 0110\ 0001$
 $+ (-52) = \underline{1100\ 1100} \quad (+52=0011\ 0100)$
 $+ 45 \quad 1\ 0010\ 1101 = +45$
- $(-90) = 1010\ 0110 \quad (+90=0101\ 1010)$
 $+ (+36) = \underline{0010\ 0100}$
 $- 54 \quad 1100\ 1010 = -54$
- $(-74) = 1011\ 0110 \quad (+74=0100\ 1010)$
 $+ (-30) = \underline{1110\ 0010} \quad (+30=0001\ 1110)$
 $- 104 \quad 1\ 1001\ 1000 = -104$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

67

Ví dụ cộng số nguyên có dấu bị tràn

- $(+75) = 0100\ 1011$
 $+ (+82) = \underline{0101\ 0010}$
 $+ 157 \quad 1001\ 1101$
 $= -128+16+8+4+1 = -99 \rightarrow sai$
- $(-104) = 1001\ 1000 \quad (+104=0110\ 1000)$
 $+ (-43) = \underline{1101\ 0101} \quad (+43=0010\ 1011)$
 $- 147 \quad 1\ 0110\ 1101$
 $= 64+32+8+4+1 = +109 \rightarrow sai$
- Cả hai ví dụ đều tràn vì tổng nằm ngoài dải biểu diễn:
 $[-128, +127]$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

68

Ví dụ

- Giải thích kết quả của chương trình sau:

```
#include <stdio.h>
int main() {
    short x = 32767;
    short y = 6;
    short sum = x + y;
    printf("%d", sum);
    return 0;
}
```

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

69

3.5. Phép nhân và phép chia số nguyên

1. Nhân số nguyên không dấu

$$\begin{array}{r}
 1011 \quad \text{Số bị nhân (11)} \\
 \times \underline{1101} \quad \text{Số nhân (13)} \\
 \hline
 1011 \\
 0000 \\
 1011 \\
 \hline
 1011 \\
 \hline
 10001111 \quad \text{Tích (143)}
 \end{array}$$

Các tích riêng phần

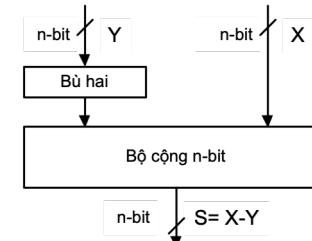
NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

71

4. Nguyên tắc thực hiện phép trừ

- Phép trừ hai số nguyên: $X - Y = X + (-Y)$
- Nguyên tắc: Lấy bù hai của Y để được $-Y$, rồi cộng với X



NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

70

Nhân số nguyên không dấu (tiếp)

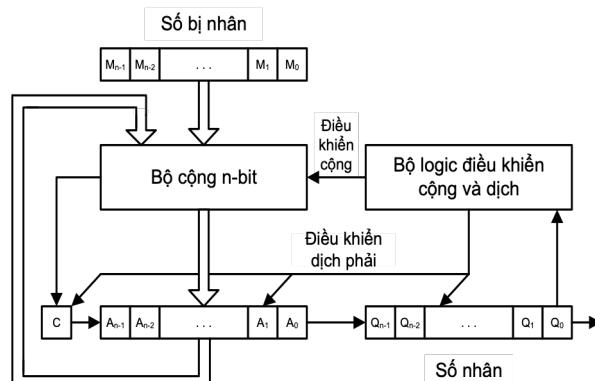
- Các tích riêng phần được xác định như sau:
 - Nếu bit của số nhân bằng 0 \rightarrow tích riêng phần bằng 0
 - Nếu bit của số nhân bằng 1 \rightarrow tích riêng phần bằng số bị nhân
 - Tích riêng phần tiếp theo được dịch trái một bit so với tích riêng phần trước đó
- Tích bằng tổng các tích riêng phần
- Nhân hai số nguyên n-bit, tích có độ dài 2n bit (không bao giờ tràn)

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

72

Bộ nhân số nguyên không dấu

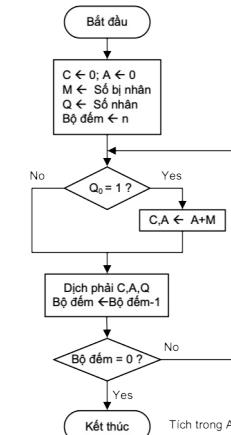


NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

73

Lưu đồ nhân số nguyên không dấu



NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

74

Ví dụ nhân số nguyên không dấu

- Số bị nhân $M = 1011 \quad (11)$
- Số nhân $Q = 1101 \quad (13)$
- Tích $= 1000\ 1111 \quad (143)$

C	A	Q
0	0000	1101
	+ 1011	
0	1011	1101
0	0101	1110
0	0010	1111
	+ 1011	
0	1101	1111
0	0110	1111
	+ 1011	
1	0001	1111
0	1000	1111

Các giá trị khởi đầu

A $\leftarrow A + M$

Dịch phải

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

75

Ví dụ nhân số nguyên không dấu (tiếp)

- Số bị nhân $M = 0110 \quad (6)$
- Số nhân $Q = 0101 \quad (5)$
- Tích $= \quad (30)$

C	A	Q
0	0000	0101
	+ 0110	
0	0110	0101
0	0011	0010
0	0001	1001
	+ 0110	
0	0111	1001
0	0011	1100
	+ 0001	
0	0001	1110

Các giá trị khởi đầu

A $\leftarrow A + M$

Dịch phải

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

76

2. Nhân số nguyên có dấu

- Sử dụng thuật giải nhân không dấu
- Sử dụng thuật giải Booth (tham khảo sách COA)

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

77

Sử dụng thuật giải nhân không dấu

- **Bước 1.** Chuyển đổi số bị nhân và số nhân thành số dương tương ứng
- **Bước 2.** Nhân hai số dương bằng thuật giải nhân số nguyên không dấu, được tích của hai số dương.
- **Bước 3.** Hiệu chỉnh dấu của tích:
 - Nếu hai thừa số ban đầu cùng dấu thì giữ nguyên kết quả ở bước 2
 - Nếu hai thừa số ban đầu là khác dấu thì đảo dấu kết quả của bước 2 (lấy bù hai)

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

78

3. Chia số nguyên không dấu

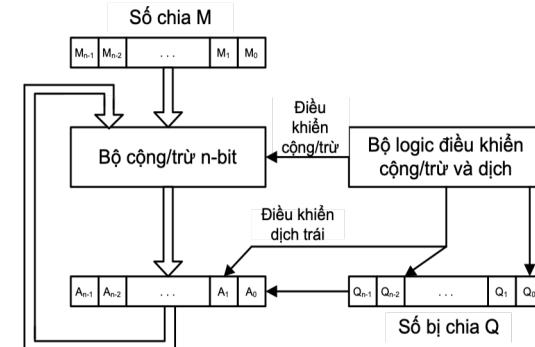
$$\begin{array}{r}
 \text{Số bị chia} \quad 10010011 \quad | \quad 1011 \quad \text{Số chia} \\
 - \underline{1011} \quad 00001101 \quad \text{Thương} \\
 001110 \\
 - \underline{1011} \\
 001111 \\
 - \underline{1011} \\
 100 \quad \text{Phần dư}
 \end{array}$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

79

Bộ chia số nguyên không dấu

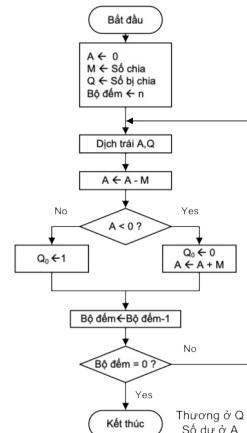


NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

80

Lưu đồ chia số nguyên không dấu



NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

81

Ví dụ: $Q = 1011 (11)$ $M = 0011 (3) \rightarrow -M = 1101$

A	Q	
0000	1011	$BD = 4$
0001	0110	dịch trái
<u>1101</u>		$A = A - M < 0$
1110		
<u>0011</u>	0110	$A = A + M$ $BD = 3$
0010	1100	dịch trái
<u>1101</u>		$A = A - M < 0$
1111		
<u>0011</u>	1100	$A = A + M$ $BD = 2$
0101	1000	dịch trái
<u>1101</u>		$A = A - M > 0$
0010	1001	$BD = 1$
0101	0010	dịch trái
<u>1101</u>		$A = A - M > 0$
0010	0011	$BD = 0$
2	3	

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

82

4. Chia số nguyên có dấu

- Bước 1. Chuyển đổi số bị chia và số chia về thành số dương tương ứng.
- Bước 2. Sử dụng thuật giải chia số nguyên không dấu để chia hai số dương, kết quả nhận được là thương Q và phần dư R đều là dương
- Bước 3. Hiệu chỉnh dấu của kết quả như sau:
▪ (Lưu ý: phép đảo dấu thực chất là thực hiện phép lấy bù hai)

Số bị chia	Số chia	Thương	Số dư
đương	đương	giữ nguyên	giữ nguyên
đương	âm	đảo dấu	giữ nguyên
âm	đương	đảo dấu	đảo dấu
âm	âm	giữ nguyên	đảo dấu

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

83

3.6. Số dấu phẩy động

- Nguyên tắc chung
 - Floating Point Number → biểu diễn cho số thực
 - Tổng quát: một số thực X được biểu diễn theo kiểu số dấu phẩy động như sau:

$$X = \pm M * R^E$$

- M là phần định trị (Mantissa),
- R là cơ số (Radix),
- E là phần mũ (Exponent).

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

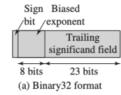
84

2. Chuẩn IEEE754-2008

- Cơ số R = 2

- Các dạng:

- Dạng 32-bit



- Dạng 64-bit



- Dạng 128-bit

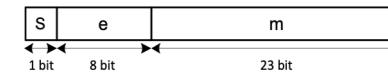


NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

85

Dạng 32-bit



- S là bit dấu:

- S = 0 → số dương
- S = 1 → số âm

- e (8 bit) là giá trị dịch chuyển của phần mũ E:

$$\text{e} = E + 127 \rightarrow \text{phần mũ } E = e - 127$$

- m (23 bit) là phần lẻ của phần định trị M:

$$M = 1.m$$

- Công thức xác định giá trị của số thực:

$$X = (-1)^S \cdot 1.m \cdot 2^{e-127}$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

86

Ví dụ 1

Xác định giá trị của các số thực được biểu diễn bằng 32-bit sau đây:

1100 0001 0101 0110 0000 0000 0000 0000

- S = 1 → số âm

- e = 1000 0010₍₂₎ = 130₍₁₀₎ → E = 130 - 127 = 3

Vậy

$$X = -1.10101100_{(2)} \cdot 2^3 = -1101.011_{(2)} = -13.375_{(10)}$$

0011 1111 1000 0000 0000 0000 0000 = ?

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

87

Ví dụ 2

Biểu diễn số thực X= 83.75₍₁₀₎ về dạng số dấu phẩy động IEEE754 32-bit

Giải:

- X = 83.75₍₁₀₎ = 1010011.11₍₂₎ = 1.0100111 × 2⁶

- Ta có:

- S = 0 vì đây là số dương

- e = E + 127 = 6 + 127 = 133₍₁₀₎ = 1000 0101₍₂₎

- Vậy:

$$X = 0100 0010 1010 0111 1000 0000 0000 0000$$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

88

Các qui ước đặc biệt

- Các bit của e bằng 0, các bit của m bằng 0, thì $X = \pm 0$
 $x000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \rightarrow X = \pm 0$
- Các bit của e bằng 1, các bit của m bằng 0, thì $X = \pm \infty$
 $x111\ 1111\ 1000\ 0000\ 0000\ 0000\ 0000\ 0000 \rightarrow X = \pm \infty$
- Các bit của e bằng 1, còn m có ít nhất một bit bằng 1, thì nó không biểu diễn cho số nào cả (NaN - not a number)

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

89

Dải giá trị biểu diễn

 2^{-127} đến 2^{+127} 10^{-38} đến 10^{+38} 

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

90

Dạng 64-bit

- S là bit dấu
- e (11 bit) là giá trị dịch chuyển của phần mũ E:
 $e = E + 1023 \rightarrow$ phần mũ E = e - 1023
- m (52 bit): phần lẻ của phần định trị M
- Giá trị số thực:

$$X = (-1)^S \cdot 1.m \cdot 2^{e-1023}$$
- Dải giá trị biểu diễn: 10^{-308} đến 10^{+308}

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

91

Dạng 128-bit

- S là bit dấu
- e (15 bit) là giá trị dịch chuyển của phần mũ E:
 $e = E + 16383 \rightarrow$ phần mũ E = e - 16383
- m (112 bit): phần lẻ của phần định trị M
- Giá trị số thực:

$$X = (-1)^S \cdot 1.m \cdot 2^{e-16383}$$
- Dải giá trị biểu diễn: 10^{-4932} đến 10^{+4932}

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

92

3. Thực hiện phép toán số dấu phẩy động

- $X_1 = M_1 \cdot R^{E_1}$
- $X_2 = M_2 \cdot R^{E_2}$
- Ta có
 - $X_1 \cdot X_2 = (M_1 \cdot M_2) \cdot R^{E_1+E_2}$
 - $X_1 / X_2 = (M_1 / M_2) \cdot R^{E_1-E_2}$
 - $X_1 \pm X_2 = (M_1 \cdot R^{E_1-E_2} \pm M_2) \cdot R^{E_2}$, với $E_2 \geq E_1$

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

93

Các khả năng tràn số

- Tràn trên số mũ (Exponent Overflow): mũ dương vượt ra khỏi giá trị cực đại của số mũ dương có thể ($\rightarrow \infty$)
- Tràn dưới số mũ (Exponent Underflow): mũ âm vượt ra khỏi giá trị cực đại của số mũ âm có thể ($\rightarrow 0$)
- Tràn trên phần định trị (Mantissa Overflow): cộng hai phần định trị có cùng dấu, kết quả bị nhớ ra ngoài bit cao nhất
- Tràn dưới phần định trị (Mantissa Underflow): Khi hiệu chỉnh phần định trị, các số bị mất ở bên phải phần định trị

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

94

Phép cộng và phép trừ

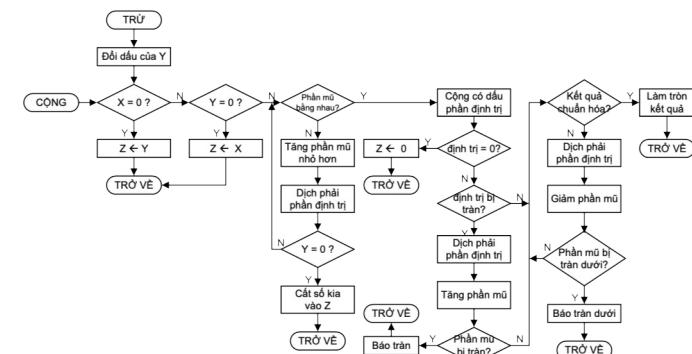
- Kiểm tra các số hạng có bằng 0 hay không
- Hiệu chỉnh phần định trị
- Cộng hoặc trừ phần định trị
- Chuẩn hoá kết quả

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

95

Thuật toán cộng/trừ số dấu phẩy động

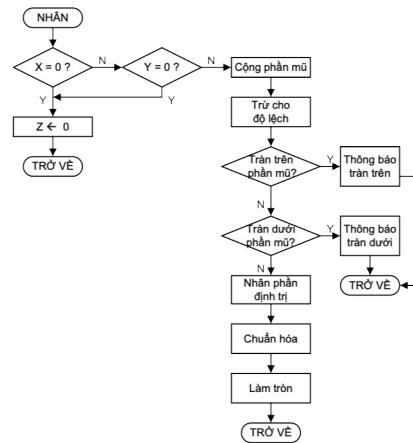


NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

96

Thuật toán nhân số dấu phẩy động

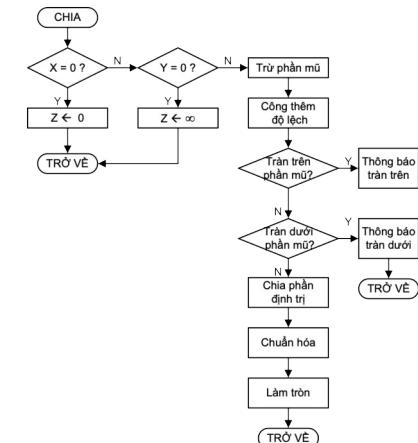


NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

97

Thuật toán chia số dấu phẩy động



NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

98

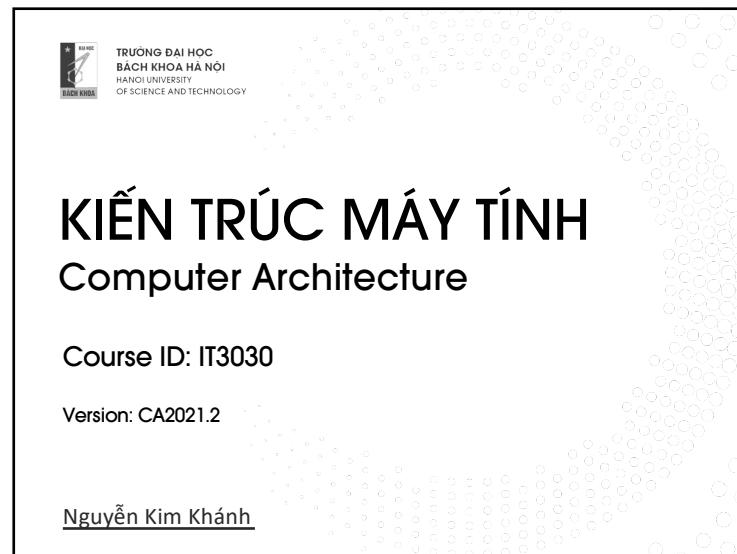
Kiến trúc máy tính

Hết chương 3

NKK-IT3030-CA2021.2

CH3-Số học và logic máy tính

99



Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

2

Kiến trúc máy tính

Chương 4 KIẾN TRÚC TẬP LỆNH

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

3

Nội dung

- 4.1. Giới thiệu chung về kiến trúc tập lệnh
- 4.2. Lệnh hợp ngữ và toán hạng
- 4.3. Mã máy
- 4.4. Các lệnh số học / logic
- 4.5. Tạo các cấu trúc điều khiển
- 4.6. Lập trình mảng dữ liệu
- 4.7. Chương trình con
- 4.8. Các lệnh số dấu phẩy động
- 4.9. Các phương pháp định địa chỉ
- 4.10. Dịch và chạy chương trình hợp ngữ
- 4.11. Các kiến trúc tập lệnh phổ biến

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

4

4.1. Giới thiệu chung về kiến trúc tập lệnh

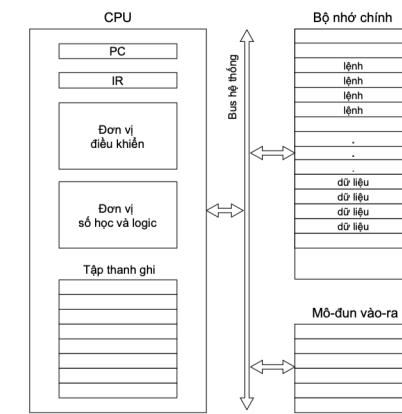
- Kiến trúc tập lệnh (Instruction Set Architecture): cách nhìn máy tính bởi người lập trình
- Vi kiến trúc (Microarchitecture): cách thực hiện kiến trúc tập lệnh bằng phần cứng
- Ngôn ngữ trong máy tính:
 - Hợp ngữ (assembly language):
 - dạng lệnh có thể đọc được bởi con người
 - biểu diễn dạng text
 - Ngôn ngữ máy (machine language):
 - còn gọi là mã máy (machine code)
 - dạng lệnh có thể đọc được bởi máy tính
 - biểu diễn bằng các bit 0 và 1

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

5

Mô hình lập trình của máy tính



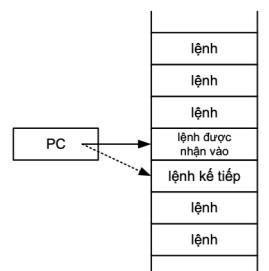
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

6

CPU nhận lệnh từ bộ nhớ

- Bộ đếm chương trình PC (Program Counter) là thanh ghi của CPU giữ địa chỉ của lệnh cần nhận vào để thực hiện
- CPU phát địa chỉ từ PC đến bộ nhớ, lệnh được nhận vào
- Sau khi lệnh được nhận vào, nội dung PC tự động tăng để trỏ sang lệnh kế tiếp
- PC tăng bao nhiêu?
 - Tùy thuộc vào độ dài của lệnh vừa được nhận
 - MIPS: lệnh có độ dài 32-bit, PC tăng 4



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

7

Giải mã và thực hiện lệnh

- Bộ xử lý giải mã lệnh đã được nhận và phát các tín hiệu điều khiển thực hiện thao tác mà lệnh yêu cầu
- Các kiểu thao tác chính của lệnh:
 - Trao đổi dữ liệu giữa CPU với bộ nhớ chính hoặc với cổng vào-ra
 - Thực hiện các phép toán số học hoặc phép toán logic với các dữ liệu (được thực hiện bởi ALU)
 - Chuyển điều khiển trong chương trình (rẽ nhánh, nhảy)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

8

CPU đọc/ghi dữ liệu bộ nhớ

- Với các lệnh trao đổi dữ liệu với bộ nhớ, CPU cần biết và phát ra địa chỉ của ngăn nhớ cần đọc/ghi
- Địa chỉ đó có thể là:**
 - Hằng số địa chỉ được cho trực tiếp trong lệnh
 - Giá trị địa chỉ nằm trong thanh ghi con trỏ
 - Địa chỉ = Địa chỉ cơ sở + giá trị dịch chuyển

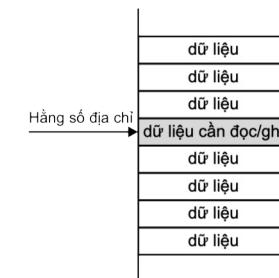
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

9

Hằng số địa chỉ

- Trong lệnh cho hằng số địa chỉ cụ thể
- CPU phát giá trị địa chỉ này đến bộ nhớ để tìm ra ngăn nhớ dữ liệu cần đọc/ghi



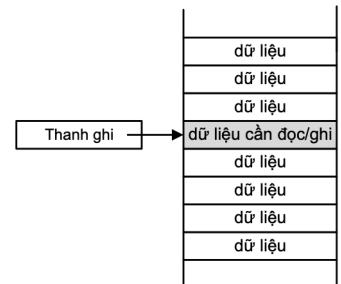
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

10

Sử dụng thanh ghi con trỏ

- Trong lệnh cho biết tên thanh ghi con trỏ
- Thanh ghi con trỏ chứa giá trị địa chỉ
- CPU phát địa chỉ này ra để tìm ra ngăn nhớ dữ liệu cần đọc/ghi



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

11

Sử dụng địa chỉ cơ sở và dịch chuyển

- Địa chỉ cơ sở (base address): địa chỉ của ngăn nhớ cơ sở
- Giá trị dịch chuyển địa chỉ (offset): giá số địa chỉ giữa ngăn nhớ cần đọc/ghi so với ngăn nhớ cơ sở
- Địa chỉ của ngăn nhớ cần đọc/ghi = (địa chỉ cơ sở) + (offset)
- Có thể sử dụng các thanh ghi để quản lý các tham số này
- Trường hợp riêng:
 - Địa chỉ cơ sở = 0
 - Offset = 0



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

12

Ngăn xếp (Stack)

- Ngăn xếp là vùng nhớ dữ liệu có cấu trúc LIFO (Last In - First Out vào sau - ra trước)
- Ngăn xếp thường dùng để phục vụ cho chương trình con
- Đầu ngăn xếp là một ngăn nhớ xác định
- Đỉnh ngăn xếp là thông tin nằm ở vị trí trên cùng trong ngăn xếp
- Đỉnh ngăn xếp có thể bị thay đổi

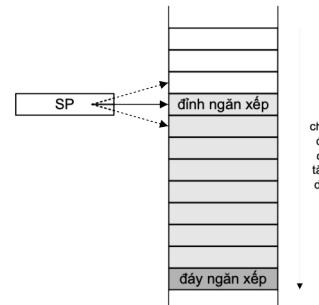
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

13

Con trỏ ngăn xếp SP (Stack Pointer)

- SP là thanh ghi chứa địa chỉ của ngăn nhớ đỉnh ngăn xếp
- Khi cất thêm một thông tin vào ngăn xếp:
 - Giảm nội dung của SP
 - Thông tin được cất vào ngăn nhớ được trả bởi SP
- Khi lấy một thông tin ra khỏi ngăn xếp:
 - Thông tin được đọc từ ngăn nhớ được trả bởi SP
 - Tăng nội dung của SP
- Khi ngăn xếp rỗng, SP trả vào đáy



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

14

Thứ tự lưu trữ các byte trong bộ nhớ chính

- Bộ nhớ chính được đánh địa chỉ cho từng byte
- Hai cách lưu trữ thông tin nhiều byte:
 - Đầu nhỏ (Little-endian): Byte có ý nghĩa thấp được lưu trữ ở ngăn nhớ có địa chỉ nhỏ, byte có ý nghĩa cao được lưu trữ ở ngăn nhớ có địa chỉ lớn.
 - Đầu to (Big-endian): Byte có ý nghĩa cao được lưu trữ ở ngăn nhớ có địa chỉ nhỏ, byte có ý nghĩa thấp được lưu trữ ở ngăn nhớ có địa chỉ lớn.
- Các sản phẩm thực tế:
 - Intel x86: little-endian
 - Motorola 680x0, SunSPARC: big-endian
 - MIPS: little-endian hoặc big-endian

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

15

Ví dụ lưu trữ dữ liệu 32-bit

Số nhị phân	0001	1010	0010	1011	0011	1100	0100	1101
Số Hexa	0x1A		0x2B		0x3C		0x4D	

little-endian	0x4D 0x3C 0x2B 0x1A	big-endian	0x4000 0x4001 0x4002 0x4003
---------------	------------------------------	------------	--------------------------------------

(0x là kí hiệu bắt đầu cho số Hexa)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

16

Tập lệnh

- Mỗi bộ xử lý được thiết kế theo một tập lệnh xác định
- Tập lệnh thường có hàng chục đến hàng trăm lệnh
- Mỗi lệnh máy (mã máy) là một chuỗi các bit (0,1) mà bộ xử lý hiểu được để thực hiện một thao tác xác định.
- Các lệnh được mô tả bằng các ký hiệu gợi nhớ dạng text, đó chính là các lệnh của hợp ngữ (assembly language)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

17

Dạng lệnh hợp ngữ

- Mã C:**
 $a = b + c;$
- Ví dụ lệnh hợp ngữ:**
`add a, b, c # a = b + c`
 trong đó:
 - add: ký hiệu gợi nhớ chỉ ra thao tác (phép toán) cần thực hiện.
 - Chú ý: mỗi lệnh chỉ thực hiện một thao tác
 - b, c: các toán hạng nguồn cho thao tác
 - a: toán hạng đích (nơi ghi kết quả)
 - phần sau dấu # là lời giải thích (chỉ có tác dụng đến hết dòng)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

18

Các thành phần của lệnh máy

Mã thao tác	Địa chỉ toán hạng
-------------	-------------------

- Mã thao tác (operation code hay opcode):** mã hóa cho thao tác mà bộ xử lý phải thực hiện
 - Các thao tác chuyển dữ liệu
 - Các phép toán số học
 - Các phép toán logic
 - Các thao tác chuyển điều khiển (rẽ nhánh, nhảy)
- Địa chỉ toán hạng:** chỉ ra nơi chứa các toán hạng mà thao tác sẽ tác động
 - Toán hạng có thể là:
 - Hằng số nằm ngay trong lệnh
 - Nội dung của thanh ghi
 - Nội dung của ngăn nhớ (hoặc cổng vào-ra)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

19

Số lượng địa chỉ toán hạng trong lệnh

- Ba địa chỉ toán hạng:**
 - `add r1, r2, r3 # r1 = r2 + r3`
 - Sử dụng phổ biến trên các kiến trúc hiện nay
- Hai địa chỉ toán hạng:**
 - `add r1, r2 # r1 = r1 + r2`
 - Sử dụng trên Intel x86, Motorola 680x0
- Một địa chỉ toán hạng:**
 - `add r1 # Acc = Acc + r1`
 - Được sử dụng trên kiến trúc thế hệ trước
- 0 địa chỉ toán hạng:**
 - Các toán hạng đều được ngầm định ở ngăn xếp
 - Không thông dụng

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

20

Các kiến trúc tập lệnh CISC và RISC

- **CISC: Complex Instruction Set Computer**
 - Máy tính với tập lệnh phức tạp
 - Các bộ xử lý: Intel x86, Motorola 680x0
- **RISC: Reduced Instruction Set Computer**
 - Máy tính với tập lệnh thu gọn
 - SunSPARC, Power PC, MIPS, ARM ...
 - RISC đối nghịch với CISC
 - Kiến trúc tập lệnh tiên tiến

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

21

Các đặc trưng của kiến trúc RISC

- Số lượng lệnh ít
- Hầu hết các lệnh truy nhập toán hạng ở các thanh ghi
- Truy nhập bộ nhớ bằng các lệnh LOAD/STORE (nạp/lưu)
- Thời gian thực hiện các lệnh là như nhau
- Các lệnh có độ dài cố định (thường là 32 bit)
- Số lượng dạng lệnh ít
- Có ít phương pháp định địa chỉ toán hạng
- Có nhiều thanh ghi
- Hỗ trợ các thao tác của ngôn ngữ bậc cao

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

22

Kiến trúc tập lệnh MIPS

- MIPS viết tắt cho:
Microprocessor without Interlocked Pipeline Stages
- Được phát triển bởi John Hennessy và các đồng nghiệp ở đại học Stanford (1984)
- Được thương mại hóa bởi MIPS Computer Systems Inc. (mips.com)
- Là kiến trúc RISC điển hình, dễ học
- Được sử dụng trong nhiều sản phẩm thực tế
- Các phần tiếp theo trong chương này sẽ nghiên cứu kiến trúc tập lệnh MIPS 32-bit
- Tài liệu:
 - MIPS Reference Data Sheet và Chapter 2 – COD
 - Introduction To MIPS Assembly Language Programming.pdf

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

23

4.2. Lệnh hợp ngữ và các toán hạng

- Thực hiện phép cộng: 3 toán hạng
 - Là phép toán phổ biến nhất
 - Hai toán hạng nguồn và một toán hạng đích
$$\text{add } a, b, c \quad \# \quad a = b + c$$
- Hầu hết các lệnh số học/logic có dạng trên
- Các lệnh số học sử dụng toán hạng thanh ghi hoặc hằng số

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

24

Tập thanh ghi của MIPS

- MIPS có tập 32 thanh ghi 32-bit
 - Được sử dụng thường xuyên
 - Được đánh số từ 0 đến 31 (mã hóa bằng 5-bit)
- Chương trình hợp dịch Assembler đặt tên:
 - Bắt đầu bằng dấu \$
 - \$t0, \$t1, ..., \$t9 chứa các giá trị tạm thời
 - \$s0, \$s1, ..., \$s7 cất các biến
- Qui ước gọi dữ liệu trong MIPS:
 - Dữ liệu 32-bit được gọi là “word”
 - Dữ liệu 16-bit được gọi là “halfword”

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

25

Tập thanh ghi của MIPS

Tên thanh ghi	Số hiệu thanh ghi	Công dụng
\$zero	0	the constant value 0, chứa hằng số = 0
\$at	1	assembler temporary, giá trị tạm thời cho hợp ngữ
\$v0-\$v1	2-3	procedure return values, các giá trị trả về của thủ tục
\$a0-\$a3	4-7	procedure arguments, các tham số vào của thủ tục
\$t0-\$t7	8-15	temporaries, chứa các giá trị tạm thời
\$s0-\$s7	16-23	saved variables, lưu các biến
\$t8-\$t9	24-25	more temporaries, chứa các giá trị tạm thời
\$k0-\$k1	26-27	OS temporaries, các giá trị tạm thời của OS
\$gp	28	global pointer, con trỏ toàn cục
\$sp	29	stack pointer, con trỏ ngăn xếp
\$fp	30	frame pointer, con trỏ khung
\$ra	31	procedure return address, địa chỉ trả về của thủ tục

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

26

Toán hạng thanh ghi

- Lệnh add, lệnh sub (subtract) chỉ thao tác với toán hạng thanh ghi
 - add rd, rs, rt # (rd) = (rs)+(rt)
 - sub rd, rs, rt # (rd) = (rs)-(rt)
- Ví dụ mã C:

$$f = (g + h) - (i + j);$$
 - giả thiết: f, g, h, i, j nằm ở \$s0, \$s1, \$s2, \$s3, \$s4
- Được dịch thành mã hợp ngữ MIPS:


```
add $t0, $s1, $s2 # $t0 = g + h
add $t1, $s3, $s4 # $t1 = i + j
sub $s0, $t0, $t1 # f = (g+h) - (i+j)
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

27

Toán hạng ở bộ nhớ

- Muốn thực hiện phép toán số học với toán hạng ở bộ nhớ, cần phải:
 - Nạp (load) giá trị từ bộ nhớ vào thanh ghi
 - Thực hiện phép toán trên thanh ghi
 - Lưu (store) kết quả từ thanh ghi ra bộ nhớ
- Bộ nhớ được đánh địa chỉ theo byte
 - MIPS sử dụng 32-bit để đánh địa chỉ cho các byte nhớ và các cổng vào-ra
 - Không gian địa chỉ: 0x00000000 – 0xFFFFFFFF
 - Mỗi word có độ dài 32-bit chiếm 4-byte trong bộ nhớ, địa chỉ của các word là bội của 4 (địa chỉ của byte đầu tiên)
- MIPS cho phép lưu trữ trong bộ nhớ theo kiểu đầu to (big-endian) hoặc kiểu đầu nhỏ (little-endian)
 - Phần mềm MARS: little-endian

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

28

Địa chỉ byte nhớ và word nhớ

Dữ liệu hoặc lệnh	Địa chỉ byte (theo Hexa)	Dữ liệu hoặc lệnh	Địa chỉ word (theo Hexa)
byte (8-bit)	0x0000 0000	word (32-bit)	0x0000 0000
byte	0x0000 0001	word	0x0000 0004
byte	0x0000 0002	word	0x0000 0008
byte	0x0000 0003	word	0x0000 000C
byte	0x0000 0004	word	0x0000 0010
byte	0x0000 0005	word	0x0000 0014
byte	0x0000 0006	word	0x0000 0018
byte	0x0000 0007	.	0xFFFF FFF4
.		.	0xFFFF FFF8
byte	0xFFFF FFFB	word	0xFFFF FFFC
byte	0xFFFF FFFC		
byte	0xFFFF FFBD		
byte	0xFFFF FFBE		
byte	0xFFFF FFCC		

2^{30} words

Địa chỉ của word nhớ là địa chỉ của byte đầu tiên trong word nhớ đó

2^{32} bytes

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

29

Lệnh lw (load word)

- Nạp (đọc) word dữ liệu 32-bit từ bộ nhớ đưa vào thanh ghi

`lw rt, imm(rs) # (rt) = mem[(rs) + imm]`

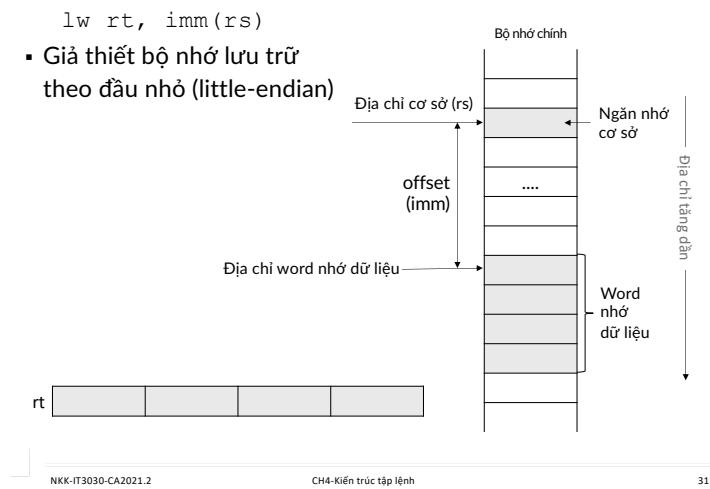
- rs: thanh ghi chứa địa chỉ cơ sở (base address)
- imm (immediate): hằng số dịch chuyển địa chỉ (offset)
→ địa chỉ của word dữ liệu cần đọc = địa chỉ cơ sở + imm
- rt: thanh ghi đích, nơi chứa word dữ liệu được nạp vào
- Giá trị địa chỉ cơ sở và imm đều chia hết cho 4

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

30

Minh họa lệnh lw

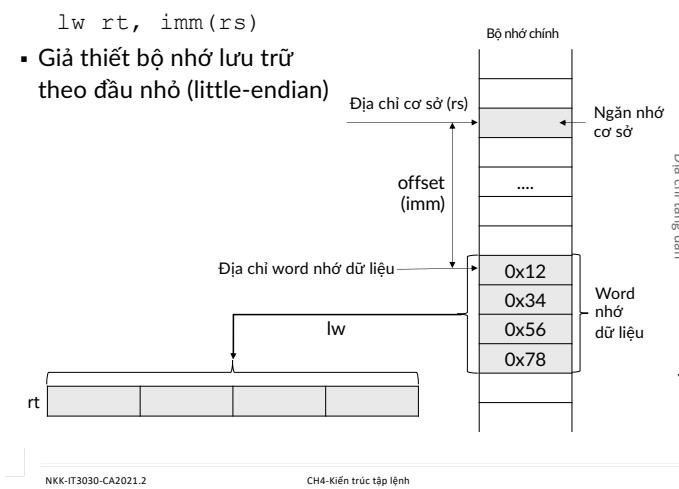


NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

31

Minh họa lệnh lw



NKK-IT3030-CA2021.2

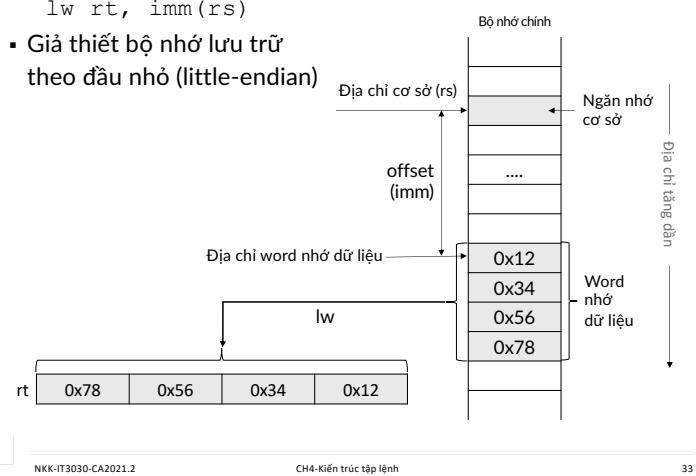
CH4-Kiến trúc tập lệnh

32

Minh họa lệnh lw

`lw rt, imm(rs)`

- Giả thiết bộ nhớ lưu trữ theo đầu nhỏ (little-endian)



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

33

Lệnh sw (store word)

`sw rt, imm(rs) # mem[(rs)+imm]=(rt)`

- `rt`: thanh ghi nguồn, chứa word dữ liệu cần ghi ra bộ nhớ
- `rs`: thanh ghi chứa địa chỉ cơ sở (base address)
- `imm` (immediate): hằng số dịch chuyển địa chỉ (`offset`)
→ địa chỉ của nơi ghi word dữ liệu = địa chỉ cơ sở + `imm`
- Giá trị địa chỉ cơ sở và `imm` đều chia hết cho 4

NKK-IT3030-CA2021.2

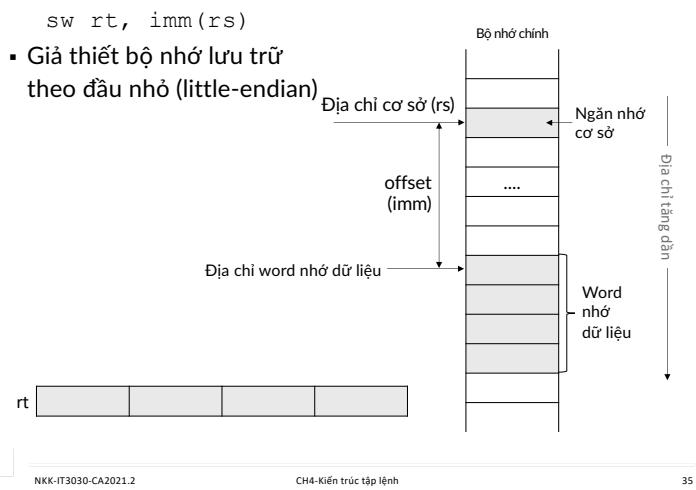
CH4-Kiến trúc tập lệnh

34

Minh họa lệnh sw

`sw rt, imm(rs)`

- Giả thiết bộ nhớ lưu trữ theo đầu nhỏ (little-endian)



NKK-IT3030-CA2021.2

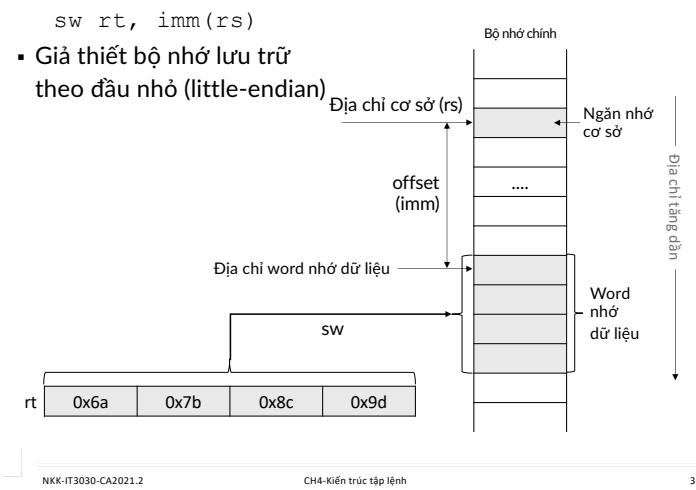
CH4-Kiến trúc tập lệnh

35

Minh họa lệnh sw

`sw rt, imm(rs)`

- Giả thiết bộ nhớ lưu trữ theo đầu nhỏ (little-endian)



NKK-IT3030-CA2021.2

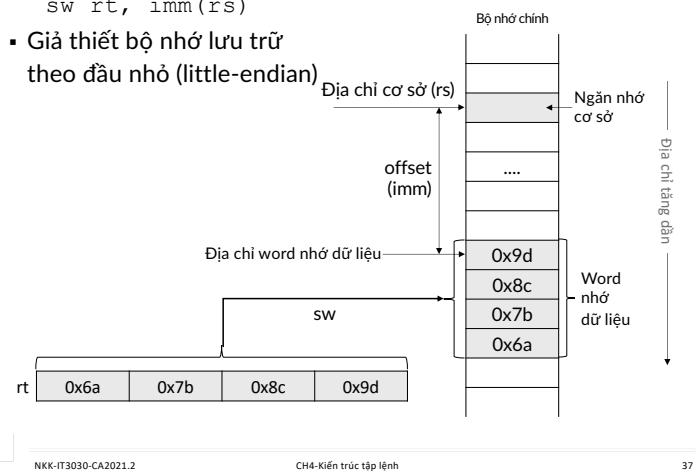
CH4-Kiến trúc tập lệnh

36

Minh họa lệnh sw

`sw rt, imm(rs)`

- Giả thiết bộ nhớ lưu trữ theo đầu nhỏ (little-endian)



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

37

Ví dụ toán hạng bộ nhớ

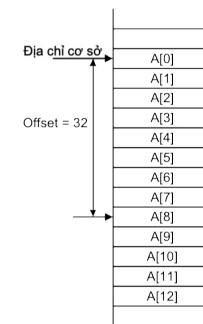
- Mã C:

// A là mảng các phần tử 32-bit

$$g = h + A[8];$$

- Cho g ở \$s1, h ở \$s2

- \$s3 chứa địa chỉ cơ sở của mảng A



Offset = 32
Địa chỉ cơ sở

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

38

Ví dụ toán hạng bộ nhớ

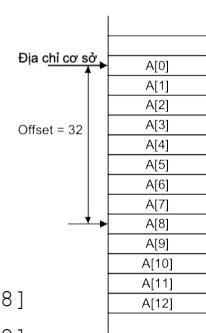
- Mã C:

// A là mảng các phần tử 32-bit

$$g = h + A[8];$$

- Cho g ở \$s1, h ở \$s2

- \$s3 chứa địa chỉ cơ sở của mảng A

Offset = 32
Địa chỉ cơ sở

- Mã hợp ngữ MIPS:

Chỉ số 8, do đó offset = 32

```
lw    $t0, 32($s3)      # $t0 = A[8]
add  $s1, $s2, $t0      # g = h+A[8]
```

offset

base register

(Chú ý: offset phải là hằng số, có thể dương, âm hoặc bằng 0)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

39

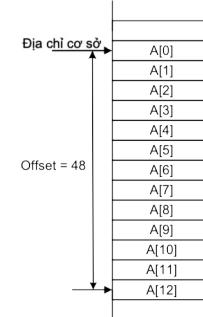
Ví dụ toán hạng bộ nhớ (tiếp)

- Mã C:

$$A[12] = h + A[8];$$

- h ở \$s2

- \$s3 chứa địa chỉ cơ sở của mảng A



Offset = 48
Địa chỉ cơ sở

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

40

Ví dụ toán hạng bộ nhớ (tiếp)

- Mã C:

$A[12] = h + A[8];$

- h ở \$s2

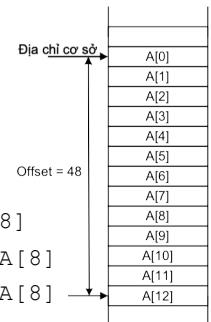
- \$s3 chứa địa chỉ cơ sở của mảng A

- Mã hợp ngữ MIPS:

```
lw $t0, 32($s3) # $t0 = A[8]
```

```
add $t0, $s2, $t0 # $t0 = h+A[8]
```

```
sw $t0, 48($s3) # A[12]=h+A[8]
```



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

41

Thanh ghi với Bộ nhớ

- Truy nhập thanh ghi nhanh hơn bộ nhớ
- Thao tác dữ liệu trên bộ nhớ yêu cầu nạp (load) và lưu (store)
 - Cần thực hiện nhiều lệnh hơn
- Chương trình dịch sử dụng các thanh ghi cho các biến nhiều nhất có thể
 - Chỉ sử dụng bộ nhớ cho các biến ít được sử dụng
 - Cần tối ưu hóa sử dụng thanh ghi

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

42

Toán hạng tức thì (immediate)

- Dữ liệu hằng số được xác định ngay trong lệnh

```
addi $s3, $s3, 4 # $s3 = $s3+4
```

- Không có lệnh trừ (subi) với giá trị hằng số

- Sử dụng hằng số âm trong lệnh addi để thực hiện phép trừ

```
addi $s2, $s1, -1 # $s2 = $s1-1
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

43

Xử lý với số nguyên

- Số nguyên có dấu (biểu diễn bằng bù hai):

- Với n bit, dài biểu diễn: $[-2^{n-1}, +2^{n-1}-1]$

- Overflow: tràn với số nguyên có dấu

- Các lệnh add, sub, addi: nếu có overflow xảy ra thì báo lỗi tràn và dừng lại

- Số nguyên không dấu:

- Với n bit, dài biểu diễn: $[0, 2^n - 1]$

- Các lệnh addu, subu, addiu: bỏ qua overflow

- Qui ước biểu diễn hằng số nguyên trong hợp ngữ MIPS:

- số thập phân: 12; 3456; -18

- số Hexa (bắt đầu bằng 0x): 0x12 ; 0x3456; 0x1AB6

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

44

Hằng số Zero

- Thanh ghi 0 của MIPS (\$zero hay \$0) luôn chứa hằng số 0
 - Không thể thay đổi giá trị
- Hữu ích cho một số thao tác thông dụng
 - Chẳng hạn, chuyển dữ liệu giữa các thanh ghi

$$\text{add \$t2, \$s1, \$zero} \quad \# \quad \$t2 = \$s1$$
 - Nạp hằng số vào thanh ghi

$$\text{addi \$s1, \$zero, 8} \quad \# \quad \$s1 = 8$$

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

45

4.3. Mã máy (Machine code)

- Các lệnh được mã hóa dưới dạng nhị phân được gọi là mã máy
- Các lệnh của MIPS:
 - Được mã hóa bằng các từ lệnh 32-bit
 - Mỗi lệnh chiếm 4-byte trong bộ nhớ, do vậy địa chỉ của lệnh trong bộ nhớ là bội của 4
 - Có ít dạng lệnh
- Số hiệu thanh ghi được mã hóa bằng 5-bit
 - \$t0 - \$t7 có số hiệu từ 8 - 15 (01000 - 01111)
 - \$t8 - \$t9 có số hiệu từ 24 - 25 (11000 - 11001)
 - \$s0 - \$s7 có số hiệu từ 16 - 23 (10000 - 10111)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

46

Các kiểu lệnh máy của MIPS

Lệnh kiểu R	op	rs	rt	rd	shamt	funct
	6 bit	5 bit	5 bit	5 bit	5 bit	6 bit

Lệnh kiểu I	op	rs	rt	imm
	6 bit	5 bit	5 bit	16 bit

Lệnh kiểu J	op	address
	6 bit	26 bit

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

47

Lệnh kiểu R (Registers)

op	rs	rt	rd	shamt	funct
6 bit	5 bit	5 bit	5 bit	5 bit	6 bit

Các trường của lệnh

- op (operation code - opcode): mã thao tác
 - với các lệnh kiểu R, op = 000000
- rs: số hiệu thanh ghi nguồn thứ nhất
- rt: số hiệu thanh ghi nguồn thứ hai
- rd: số hiệu thanh ghi đích
- shamt (shift amount): số bit được dịch, chỉ dùng cho lệnh dịch bit, với các lệnh khác shamt = 00000
- funct (function code): mã hàm → mã hóa cho thao tác cụ thể

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

48

Ví dụ mã máy của lệnh add, sub

op	rs	rt	rd	shamt	funct
6 bit	5 bit	5 bit	5 bit	5 bit	6 bit
add \$t0, \$s1, \$s2					
0	\$s1	\$s2	\$t0	0	add
0	17	18	8	0	32 (0x20)
000000	10001	10010	01000	00000	100000 (0x02324020)
sub \$s0, \$t3, \$t5					
0	\$t3	\$t5	\$s0	0	sub
0	11	13	16	0	34 (0x22)
000000	01011	01101	10000	00000	100010 (0x016D8022)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

49

Mở rộng bit cho hằng số theo số có dấu

- Với các lệnh addi, lw, sw cần cộng nội dung thanh ghi với hằng số:
 - Thanh ghi có độ dài 32-bit
 - Hằng số imm 16-bit, cần mở rộng có dấu thành 32-bit (Sign-extended)
- Ví dụ mở rộng có dấu số 16-bit thành 32-bit:

+5 =	0000 0000 0000 0101	16-bit
+5 =	0000 0000 0000 0000 0000 0101	32-bit
-12 =	1111 1111 1111 0100	16-bit
-12 =	1111 1111 1111 1111 1111 0100	32-bit

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

51

Lệnh kiểu I (Immediate)

op	rs	rt	imm
6 bit	5 bit	5 bit	16 bit

- Dùng cho các lệnh số học/logic với toán hạng tức thì và các lệnh load/store (nạp/lưu)

- Với lệnh addi:
 - rs số hiệu thanh ghi nguồn
 - rt số hiệu thanh ghi đích
 - Với lệnh lw, sw:
 - rs là số hiệu thanh ghi cơ sở
 - rt: số hiệu thanh ghi đích (lw), hoặc số hiệu thanh ghi nguồn (sw)
 - imm (immediate): hằng số nguyên 16-bit
- addi rt, rs, imm # (rt) = (rs)+SignExtImm
 lw rt, imm(rs) # (rt) = mem[(rs)+SignExtImm]
 sw rt, imm(rs) # mem[(rs)+SignExtImm] = (rt)
 (SignExtImm: hằng số imm 16-bit được mở rộng có dấu thành 32-bit)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

50

Ví dụ mã máy của lệnh addi

op	rs	rt	imm
6 bit	5 bit	5 bit	16 bit
addi \$s0, \$s1, 5			
8	\$s1	\$s0	5
8	17	16	5
001000	10001	10000	0000 0000 0000 0101 (0x22300005)
addi \$t1, \$s2, -12			
8	\$s2	\$t1	-12
8	18	9	-12
001000	10010	01001	1111 1111 1111 0100 (0x2249FFF4)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

52

Ví dụ mã máy của lệnh load và lệnh store

op	rs	rt	imm
6 bit	5 bit	5 bit	16 bit
1w	\$t0	32(\$s3)	
35	\$s3	\$t0	32
35	19	8	32
100011	10011	01000	0000 0000 0010 0000 (0x8E680020)
sw	\$s1	4(\$t1)	
43	\$t1	\$s1	4
43	9	17	4
101011	01001	10001	0000 0000 0000 0100 (0xAD310004)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

53

Lệnh kiểu J (Jump)

op	address
6 bit	26 bit

- Toán hạng 26-bit địa chỉ
- Được sử dụng cho các lệnh nhảy
 - j (jump) \rightarrow op = 000010
 - jal (jump and link) \rightarrow op = 000011
- (sẽ được giới thiệu chi tiết ở mục 4.5)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

54

4.4. Các lệnh số học/logic

Các lệnh logic: Thao tác trên các bit của dữ liệu

Phép toán logic	Toán tử trong C	Lệnh của MIPS
Shift left	<<	sll
Shift right	>>	srl
Bitwise AND	&	and, andi
Bitwise OR		or, ori
Bitwise XOR	^	xor, xori
Bitwise NOT	~	nor

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

55

Ví dụ lệnh logic kiểu R

Nội dung các thanh ghi nguồn

\$s1 0100 0110 1010 0001 1100 0000 1011 0111

\$s2 1111 1111 1111 1111 0000 0000 0000 0000

Mã hợp ngữ

and \$s3, \$s1, \$s2 \$s3

or \$s4, \$s1, \$s2 \$s4

xor \$s5, \$s1, \$s2 \$s5

nor \$s6, \$s1, \$s2 \$s6

Kết quả thanh ghi đích

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

56

Ví dụ lệnh logic kiểu R

Nội dung các thanh ghi nguồn

\$s1	0100	0110	1010	0001	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

\$s2	1111	1111	1111	1111	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

Mã hợp ngữ

Kết quả thanh ghi đích

and \$s3, \$s1, \$s2

\$s3	0100	0110	1010	0001	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

or \$s4, \$s1, \$s2

\$s4								
------	--	--	--	--	--	--	--	--

xor \$s5, \$s1, \$s2

\$s5								
------	--	--	--	--	--	--	--	--

nor \$s6, \$s1, \$s2

\$s6								
------	--	--	--	--	--	--	--	--

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

57

Ví dụ lệnh logic kiểu R

Nội dung các thanh ghi nguồn

\$s1	0100	0110	1010	0001	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

\$s2	1111	1111	1111	1111	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

Mã hợp ngữ

Kết quả thanh ghi đích

and \$s3, \$s1, \$s2

\$s3	0100	0110	1010	0001	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

or \$s4, \$s1, \$s2

\$s4	1111	1111	1111	1111	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

xor \$s5, \$s1, \$s2

\$s5								
------	--	--	--	--	--	--	--	--

nor \$s6, \$s1, \$s2

\$s6								
------	--	--	--	--	--	--	--	--

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

58

Ví dụ lệnh logic kiểu R

Nội dung các thanh ghi nguồn

\$s1	0100	0110	1010	0001	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

\$s2	1111	1111	1111	1111	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

Mã hợp ngữ

Kết quả thanh ghi đích

and \$s3, \$s1, \$s2

\$s3	0100	0110	1010	0001	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

or \$s4, \$s1, \$s2

\$s4	1111	1111	1111	1111	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

xor \$s5, \$s1, \$s2

\$s5	1011	1001	0101	1110	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

nor \$s6, \$s1, \$s2

\$s6								
------	--	--	--	--	--	--	--	--

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

59

Ví dụ lệnh logic kiểu R

Nội dung các thanh ghi nguồn

\$s1	0100	0110	1010	0001	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

\$s2	1111	1111	1111	1111	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

Mã hợp ngữ

Kết quả thanh ghi đích

and \$s3, \$s1, \$s2

\$s3	0100	0110	1010	0001	0000	0000	0000	0000
------	------	------	------	------	------	------	------	------

or \$s4, \$s1, \$s2

\$s4	1111	1111	1111	1111	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

xor \$s5, \$s1, \$s2

\$s5	1011	1001	0101	1110	1100	0000	1011	0111
------	------	------	------	------	------	------	------	------

nor \$s6, \$s1, \$s2

\$s6	0000	0000	0000	0000	0011	1111	0100	1000
------	------	------	------	------	------	------	------	------

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

60

Ví dụ lệnh logic kiểu I

Giá trị các toán hạng nguồn

\$s1	0000	0000	0000	0000	0000	0000	1111	1111
imm	0000	0000	0000	0000	1111	1010	0011	0100

← Zero-extended →

Mã hợp ngữ

andi \$s2,\$s1,0xFA34

\$s2

ori \$s3,\$s1,0xFA34

\$s3

xori \$s4,\$s1,0xFA34

\$s4

Kết quả thanh ghi đích

Chú ý: Với các lệnh logic kiểu I, hằng số imm 16-bit được mở rộng không dấu thành 32-bit (zero-extended)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

61

Ví dụ lệnh logic kiểu I

Giá trị các toán hạng nguồn

\$s1	0000	0000	0000	0000	0000	0000	1111	1111
imm	0000	0000	0000	0000	1111	1010	0011	0100

← Zero-extended →

Mã hợp ngữ

andi \$s2,\$s1,0xFA34

\$s2

ori \$s3,\$s1,0xFA34

\$s3

xori \$s4,\$s1,0xFA34

\$s4

Kết quả thanh ghi đích

Chú ý: Với các lệnh logic kiểu I, hằng số imm 16-bit được mở rộng không dấu thành 32-bit (Zero-extended)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

62

Ý nghĩa của các phép toán logic

- Phép AND: giữ nguyên một số bit trong word, xóa các bit còn lại về 0
 - Phép OR: giữ nguyên một số bit trong word, thiết lập các bit còn lại lên 1
 - Phép XOR: giữ nguyên một số bit trong word, đảo giá trị các bit còn lại
 - Phép NOT: đảo các bit trong word
 - Đổi 0 thành 1, và đổi 1 thành 0
 - MIPS không có lệnh NOT, nhưng có lệnh NOR với 3 toán hạng
 - a NOR b == NOT (a OR b)
- ```
nor $t0, $t1, $zero # $t0 = not($t1)
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

63

### Các lệnh dịch bit

| op    | rs    | rt    | rd    | shamt | funct |
|-------|-------|-------|-------|-------|-------|
| 6 bit | 5 bit | 5 bit | 5 bit | 5 bit | 6 bit |

- shamt: chỉ ra dịch bao nhiêu vị trí (shift amount)
- rs: không sử dụng, thiết lập = 00000
- Lấy nội dung thanh ghi nguồn rt dịch trái hoặc dịch phải rồi cắt sang thanh ghi đích rd; rt không thay đổi nội dung
- sll - shift left logical (dịch trái logic)
  - Dịch trái các bit và diền các bit 0 vào bên phải
  - Dịch trái  $i$  bits là nhân với  $2^i$  (nếu kết quả trong phạm vi biểu diễn 32-bit)
- srl - shift right logical (dịch phải logic)
  - Dịch phải các bit và diền các bit 0 vào bên trái
  - Dịch phải  $i$  bits là chia cho  $2^i$  (chỉ với số nguyên không dấu)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

64

### Ví dụ lệnh dịch trái sll

Lệnh hợp ngữ:

`sll $t2, $s0, 4 # $t2 = $s0 << 4`

Mã máy:

| op     | rs    | rt    | rd    | shamt | funct  |
|--------|-------|-------|-------|-------|--------|
| 0      | 0     | 16    | 10    | 4     | 0      |
| 000000 | 00000 | 10000 | 01010 | 00100 | 000000 |

(0x00105100)

Ví dụ kết quả thực hiện lệnh:

\$s0 [0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 1101] = 13

\$t2 [0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 1101 | 0000] = 13x16

Chú ý: Nội dung thanh ghi \$s0 không bị thay đổi

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

65

### Ví dụ lệnh dịch phải srl

Lệnh hợp ngữ:

`srl $s2, $s1, 2 # $s2 = $s1 >> 2`

Mã máy:

| op     | rs    | rt    | rd    | shamt | funct  |
|--------|-------|-------|-------|-------|--------|
| 0      | 0     | 17    | 18    | 2     | 2      |
| 000000 | 00000 | 10001 | 10010 | 00010 | 000010 |

(0x00119082)

Ví dụ kết quả thực hiện lệnh:

\$s1 [0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0101 | 0110] = 86

\$s2 [0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0001 | 0101] = 21 [86/4]

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

66

### Nạp hằng số vào thanh ghi

- Trường hợp hằng số 16-bit → sử dụng lệnh addi:
  - Ví dụ: nạp hằng số 0x4F3C vào thanh ghi \$s0:
  $\text{addi } \$s0, \$0, 0x4F3C \ #\$s0 = 0x4F3C$
- Trong trường hợp hằng số 32-bit → sử dụng lệnh lui và lệnh ori:
  - `lui rt, 16_bit_cao`
    - Copy 16 bit cao của hằng số 32-bit vào nửa bên trái rt
    - Xóa 16 bit bên phải của rt về 0
  - `ori rt, rt, 16_bit_thap`
    - Đưa 16 bit thấp của hằng số 32-bit vào nửa bên phải rt

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

67

### Lệnh lui (load upper immediate)

| op    | rs    | rt    | imm    |
|-------|-------|-------|--------|
| 6 bit | 5 bit | 5 bit | 16 bit |

`lui $s0, 0x21A0`

15 0 \$s0 0x21A0

15 0 16 0x21A0

001111 00000 10000 0010 0001 1010 0000

(0x3C1021A0)

Nội dung \$s0 sau khi lệnh được thực hiện:

|      |      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|------|
| \$s0 | 0010 | 0001 | 1010 | 0000 | 0000 | 0000 | 0000 | 0000 |
|------|------|------|------|------|------|------|------|------|

(0x21A00000)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

68

### Ví dụ khởi tạo thanh ghi 32-bit

- Nạp vào thanh ghi \$s0 giá trị 32-bit sau:

0010 0001 1010 0000 0100 0000 0011 1011 =0x21A0 403B

```
lui $s0, 0x21A0 # nạp 0x21A0 vào nửa cao
 # của thanh ghi $s0
ori $s0,$s0,0x403B # nạp 0x403B vào nửa thấp
 # của thanh ghi $s0
```

Nội dung \$s0 sau khi thực hiện lệnh lui

|      |      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|------|
| \$s0 | 0010 | 0001 | 1010 | 0000 | 0000 | 0000 | 0000 | 0000 |
|      | 0000 | 0000 | 0000 | 0000 | 0100 | 0000 | 0011 | 1011 |

or

Nội dung \$s0 sau khi thực hiện lệnh ori

|      |      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|------|
| \$s0 | 0010 | 0001 | 1010 | 0000 | 0100 | 0000 | 0011 | 1011 |
|------|------|------|------|------|------|------|------|------|

### 4.5. Tạo các cấu trúc điều khiển

- Các cấu trúc rẽ nhánh

- if
- if/else
- switch/case

- Các cấu trúc lặp

- while
- do while
- for

### Các lệnh nhân và chia

mul rd, rs, rt # tích chi trong 32-bit

MIPS có hai thanh ghi 32-bit: HI (high) và LO (low)

mult rs, rt # nhân số nguyên có dấu

multu rs, rt # nhân số nguyên không dấu

- Tích 64-bit nằm trong cặp thanh ghi HI/LO

div rs, rt # chia số nguyên có dấu

divu rs, rt # chia số nguyên không dấu

- HI: chứa phần dư, LO: chứa thương

mfhi rd # Move from Hi to rd

mflo rd # Move from LO to rd

### Các lệnh rẽ nhánh và lệnh nhảy

- Các lệnh rẽ nhánh: beq, bne

Rẽ nhánh đến lệnh được đánh nhãn nếu điều kiện là đúng, ngược lại, thực hiện tuần tự

beq rs, rt, L1

- branch on equal

nếu (rs == rt) rẽ nhánh đến lệnh ở nhãn L1

bne rs, rt, L1

- branch on not equal

nếu (rs != rt) rẽ nhánh đến lệnh ở nhãn L1

- Lệnh nhảy j

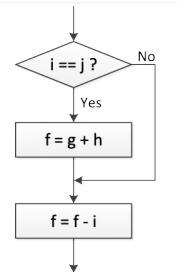
j L1

- nhảy (jump) không điều kiện đến lệnh ở nhãn L1

### Dịch câu lệnh if

- Mã C:

```
if (i==j)
 f = g+h;
 f = f-i;
 • f, g, h, i, j ở $s0, $s1, $s2, $s3, $s4
```



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

73

### Dịch câu lệnh if

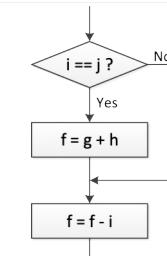
- Mã C:

```
if (i==j)
 f = g+h;
 f = f-i;
 • f, g, h, i, j ở $s0, $s1, $s2, $s3, $s4
```

- Mã hợp ngữ MIPS:

```
$s0 = f, $s1 = g, $s2 = h
$s3 = i, $s4 = j
bne $s3, $s4, L1 # Nếu i=j
add $s0, $s1, $s2 # thì f=g+h
L1: sub $s0, $s3, $s4 # f=f-i
```

Điều kiện hợp ngữ ngược với điều kiện của ngôn ngữ bậc cao



NKK-IT3030-CA2021.2

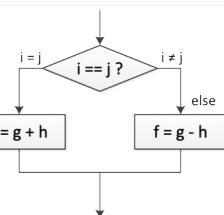
CH4-Kiến trúc tập lệnh

74

### Dịch câu lệnh if/else

- Mã C:

```
if (i==j) f = g+h;
else f = g-h;
 • f, g, h, i, j ở $s0, $s1, $s2, $s3, $s4
```



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

75

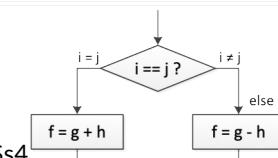
### Dịch câu lệnh if/else

- Mã C:

```
if (i==j) f = g+h;
else f = g-h;
 • f, g, h, i, j ở $s0, $s1, $s2, $s3, $s4
```

- Mã hợp ngữ MIPS:

```
bne $s3,$s4,Else # Nếu i=j
add $s0,$s1,$s2 # thì f=g+h
j Exit # thoát
Else: sub $s0,$s1,$s2 # nếu i!=j thì f=g-h
Exit: ...
```



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

76

## Dịch câu lệnh switch/case

Mã C:

```
switch (amount) {
 case 20: fee = 2; break;
 case 50: fee = 3; break;
 case 100: fee = 5; break;
 default: fee = 0;
}
// tương đương với sử dụng các câu lệnh if/else
if (amount == 20) fee = 2;
else if (amount == 50) fee = 3;
else if (amount == 100) fee = 5;
else fee = 0;
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

77

## Dịch câu lệnh switch/case

Mã hợp ngữ MIPS

```
$s0 = amount, $s1 = fee
case20:
 addi $t0,$0,20 # $t0 = 20
 bne $s0,$t0,case50 # amount == 20? if not, skip to case50
 addi $s1,$0,2 # if so, fee = 2
 j done # and break out of case
case50:
 addi $t0,$0,50 # $t0 = 50
 bne $s0,$t0,case100 # amount == 50? if not, skip to case100
 addi $s1,$0,3 # if so, fee = 3
 j done # and break out of case
case100:
 addi $t0,$0,100 # $t0 = 100
 bne $s0,$t0,default # amount == 100? if not, skip to default
 addi $s1,$0,5 # if so, fee = 5
 j done # and break out of case
default:
 add $s1,$0,$0 # fee = 0
done:
```

NKK-IT3030-CA2021.2

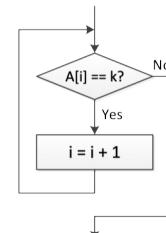
CH4-Kiến trúc tập lệnh

78

## Dịch câu lệnh vòng lặp while

Mã C:

```
while (A[i] == k) i += 1;
 ■ i ở $s3, k ở $s5
 ■ địa chỉ cơ sở của mảng A ở $s6
```



NKK-IT3030-CA2021.2

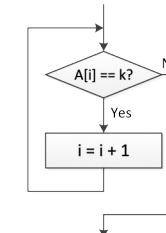
CH4-Kiến trúc tập lệnh

79

## Dịch câu lệnh vòng lặp while

Mã C:

```
while (A[i] == k) i += 1;
 ■ i ở $s3, k ở $s5
 ■ địa chỉ cơ sở của mảng A ở $s6
```



Mã hợp ngữ MIPS:

```
Loop: sll $t1, $s3, 2 # $t1 = 4*i
 add $t1, $t1, $s6 # $t1 = địa chỉ A[i]
 lw $t0, 0($t1) # $t0 = A[i]
 bne $t0, $s5, Exit # nếu A[i] != k thì Exit
 addi $s3, $s3, 1 # nếu A[i] = k thì i = i + 1
 j Loop # quay lại Loop
Exit: ...
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

80

## Dịch câu lệnh vòng lặp for

- Mã C:

```
// add the numbers from 0 to 9
int sum = 0;
int i;
for (i=0; i!=10; i++) {
 sum = sum + i;
}
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

81

## Dịch câu lệnh vòng lặp for

- Mã C:

```
// add the numbers from 0 to 9
int sum = 0;
int i;
for (i=0; i!=10; i++) {
 sum = sum + i;
}
```

- Mã hợp ngữ MIPS:

```
$s0 = i, $s1 = sum
addi $s1, $0, 0 # sum = 0
add $s0, $0, $0 # i = 0
addi $t0, $0, 10 # $t0 = 10
for: beq $s0, $t0, done # Nếu i=10, thoát
add $s1, $s1, $s0 # Nếu i<10 thì sum = sum+i
addi $s0, $s0, 1 # tăng i thêm 1
j for # quay lại for
done: ...
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

82

## Thêm các lệnh thao tác điều kiện

- Lệnh slt (set on less than)

```
slt rd, rs, rt
```

- Nếu (rs < rt) thì rd = 1; ngược lại rd = 0;

- Lệnh slti

```
slti rt, rs, constant
```

- Nếu (rs < constant) thì rt = 1; ngược lại rt = 0;

- Sử dụng kết hợp với các lệnh beq, bne

```
slt $t0, $s1, $s2 # nếu ($s1 < $s2)
```

```
bne $t0, $zero, L1 # rẽ nhánh đến L1
```

```
...
```

```
L1:
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

83

## So sánh số có dấu và không dấu

- So sánh số có dấu: slt, slti

- So sánh số không dấu: sltu, sltiu

- Ví dụ

- \$s0 = 1111 1111 1111 1111 1111 1111 1111 1111

- \$s1 = 0000 0000 0000 0000 0000 0000 0000 0001

```
slt $t0, $s0, $s1 # signed
```

- 1 < +1 → \$t0 = 1

```
sltu $t0, $s0, $s1 # unsigned
```

- 4,294,967,295 > 1 → \$t0 = 0

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

84

### Ví dụ sử dụng lệnh slt

- Mã C

```
int sum = 0;
int i;

for (i=1; i < 101; i = i*2) {
 sum = sum + i;
}
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

85

### Ví dụ sử dụng lệnh slt

- Mã hợp ngữ MIPS

```
$s0 = i, $s1 = sum
 addi $s1, $0, 0 # sum = 0
 addi $s0, $0, 1 # i = 1
 addi $t0, $0, 101 # t0 = 101
loop: slt $t1, $s0, $t0 # Nếu i>= 101
 beq $t1, $0, done # thì thoát
 add $s1, $s1, $s0 # nếu i<101 thì sum=sum+i
 sll $s0, $s0, 1 # i= 2*i
 j loop # lặp lại
done:
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

86

### 4.6. Lập trình với mảng dữ liệu

- Truy cập số lượng lớn các dữ liệu cùng loại
- Chỉ số (Index): truy cập từng phần tử của mảng
- Kích cỡ (Size): số phần tử của mảng

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

87

### Ví dụ về mảng

- Mảng 5-phần tử, mỗi phần tử có độ dài 32-bit, chiếm 4 byte trong bộ nhớ
- Địa chỉ cơ sở = 0x12348000 (địa chỉ của phần tử đầu tiên của mảng array[0])
- Bước đầu tiên để truy cập mảng: nạp địa chỉ cơ sở vào thanh ghi

|            |          |
|------------|----------|
| 0x12348000 | array[0] |
| 0x12348004 | array[1] |
| 0x12348008 | array[2] |
| 0x1234800C | array[3] |
| 0x12348010 | array[4] |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

88

### Ví dụ truy cập các phần tử

#### ■ Mã C

```
int array[5];
array[0] = array[0] * 2;
array[1] = array[1] * 2;
```

#### ■ Mã hợp ngữ MIPS

```
nạp địa chỉ cơ sở của mảng vào $s0
lui $s0, 0x1234 # 0x1234 vào nửa cao của $s0
ori $s0, $s0, 0x8000 # 0x8000 vào nửa thấp của $s0

lw $t1, 0($s0) # $t1 = array[0]
sll $t1, $t1, 1 # $t1 = $t1 * 2
sw $t1, 0($s0) # array[0] = $t1

lw $t1, 4($s0) # $t1 = array[1]
sll $t1, $t1, 1 # $t1 = $t1 * 2
sw $t1, 4($s0) # array[1] = $t1
```

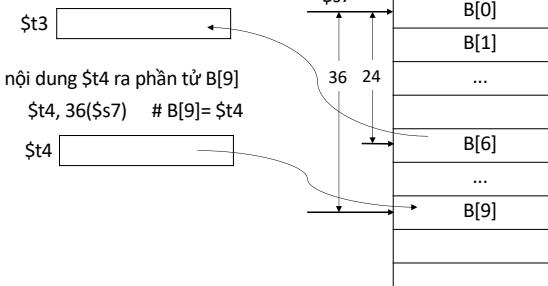
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

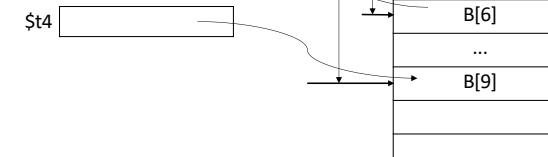
89

### Truy cập các phần tử 32-bit của mảng với chỉ số cố định

Nạp phần tử B[6] vào thanh ghi \$t3  
 lw \$t3, 24(\$s7) # \$t3 = B[6]



Cắt nội dung \$t4 ra phần tử B[9]  
 sw \$t4, 36(\$s7) # B[9] = \$t4



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

90

### Truy cập các phần tử 32-bit của mảng với chỉ số dạng biến

Nạp A[i] vào thanh ghi \$t5: (\$s3 = i)

```
sll $t0, $s3, 2 # $t0 = 4*i
add $t0, $s6, $t0 # $t0 = địa chỉ của A[i]
lw $t5, 0($t0) # $t5 = A[i]
```

lw \$t5, 0(\$t0) # \$t5 = A[i]

Cắt nội dung thanh ghi \$t6 ra phần tử A[j] (\$s4 = j)

```
sll $t1, $s4, 2 # $t1 = 4*j
add $t1, $s6, $t1 # $t1 = địa chỉ của A[j]
sw $t6, 0($t1) # A[j] = $t6
```

sw \$t6, 0(\$t1) # A[j] = \$t6

sw \$t6, 0(\$t1) # A[j] = \$t6

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

91

### Ví dụ vòng lặp truy cập mảng dữ liệu

#### ■ Mã C

```
int array[1000];
int i;

for (i=0; i < 1000; i = i + 1)
 array[i] = array[i] * 8;

// giả sử địa chỉ cơ sở của mảng = 0x23b8f000
```

#### ■ Mã hợp ngữ MIPS

```
$s0 = array base address (0x23b8f000), $s1 = i
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

92

### Ví dụ vòng lặp truy cập mảng dữ liệu (tiếp)

```
Mã hợp ngữ MIPS
$s0 = array base address (0x23b8f000), $s1 = i
khởi tạo các thanh ghi
 lui $s0, 0x23b8 # $s0 = 0x23b80000
 ori $s0, $s0, 0xf000 # $s0 = 0x23b8f000
 addi $s1, $0, 0 # i = 0
 addi $t2, $0, 1000 # $t2 = 1000
vòng lặp
loop: slt $t0, $s1, $t2 # i < 1000?
 beq $t0, $0, done # if not then done
 sll $t0, $s1, 2 # $t0 = i*4
 add $t0, $t0, $s0 # address of array[i]
 lw $t1, 0($t0) # $t1 = array[i]
 sll $t1, $t1, 3 # $t1 = array[i]*8
 sw $t1, 0($t0) # array[i] = array[i]*8
 addi $s1, $s1, 1 # i = i + 1
 j loop # repeat
done:
```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

93

### Truy cập byte và ký tự

- Các tập ký tự được mã hóa theo byte

  - ASCII: 128 ký tự

    - 95 ký thị hiển thị, 33 mã điều khiển

  - Latin-1: 256 ký tự

    - ASCII và các ký tự mở rộng

- Unicode: Tập ký tự 32-bit

  - Được sử dụng trong Java, C++, ...

  - Hầu hết các ký tự của các ngôn ngữ trên thế giới và các ký hiệu

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

94

### Các thao tác với byte/halfword

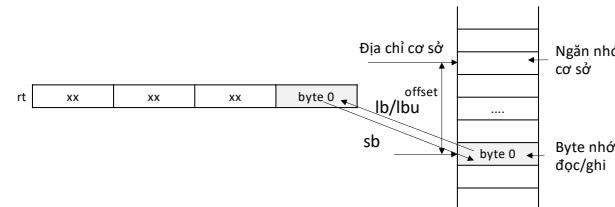
- Có thể sử dụng các phép toán logic
- MIPS có các lệnh để Nạp/Lưu byte/halfword:
  - lb rt, imm(rs) và lh rt, imm(rs)
    - Nạp 1 byte hoặc 2 byte (halfword) từ bộ nhớ vào bên phải thanh ghi đích rt
    - Phần còn lại của thanh ghi rt được mở rộng có dấu thành 32-bit (Sign-extended)
  - lbu rt, imm(rs) và lhu rt, imm(rs)
    - Phần còn lại của thanh ghi rt được mở rộng không dấu thành 32-bit (Zero-extended)
  - sb rt, imm(rs) và sh rt, imm(rs)
    - Chỉ lưu byte/halfword bên phải thanh ghi rt ra bộ nhớ

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

95

### Lệnh lb (load byte), lbu và lệnh sb (store byte)



lb rt, imm(rs) # nạp 1 byte từ bộ nhớ vào thanh ghi  
# 3 byte bên trái được mở rộng theo số có dấu

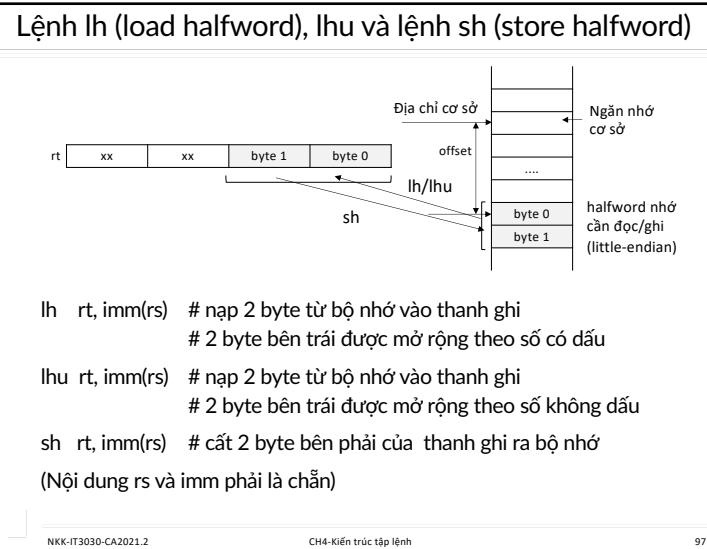
lbu rt, imm(rs) # nạp 1 byte từ bộ nhớ vào thanh ghi  
# 3 byte bên trái được mở rộng theo số không dấu

sb rt, imm(rs) # cất 1 byte bên phải của thanh ghi ra bộ nhớ  
(Nội dung rs và imm có thể chẵn hay lẻ)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

96

**Ví dụ copy String**

## ▪ Mã C:

```

void strcpy (char x[], char y[])
{ int i;
 i = 0;
 while ((x[i]=y[i]) != '\0')
 i += 1;
}

```

- Các địa chỉ của x, y ở \$a0, \$a1
- i ở \$s0

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

98

**Ví dụ Copy String**

## ▪ Mã hợp ngữ MIPS

```

add $s0,$zero,$zero # i = 0
L1: add $t1,$s0,$a1 # addr of y[i] in $t1
 lbu $t2,0($t1) # $t2 = y[i]
 add $t3,$s0,$a0 # addr of x[i] in $t3
 sb $t2,0($t3) # x[i] = y[i]
 beq $t2,$zero,L2 # exit loop if y[i]==0
 addi $s0,$s0,1 # i = i + 1
 j L1 # next iteration of loop
L2:

```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

99

**4.7. Chương trình con - thủ tục**

## ▪ Các bước yêu cầu:

1. Đặt các tham số vào các thanh ghi
2. Chuyển điều khiển đến thủ tục
3. Thực hiện các thao tác của thủ tục
4. Đặt kết quả vào thanh ghi cho chương trình đã gọi thủ tục
5. Trở về vị trí đã gọi

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

100

### Sử dụng các thanh ghi

- \$a0 - \$a3: các tham số vào (các thanh ghi 4 - 7)
- \$v0, \$v1: các kết quả ra (các thanh ghi 2 và 3)
- \$t0 - \$t9: các giá trị tạm thời
  - Có thể được ghi lại bởi thủ tục được gọi
- \$s0 - \$s7: cất giữ các biến
  - Cần phải cất/khôi phục bởi thủ tục được gọi
- \$gp: global pointer - con trỏ toàn cục cho dữ liệu tĩnh (thanh ghi 28)
- \$sp: stack pointer - con trỏ ngăn xếp (thanh ghi 29)
- \$fp: frame pointer - con trỏ khung (thanh ghi 30)
- \$ra: return address - địa chỉ trả về (thanh ghi 31)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

101

### Các lệnh liên quan với thủ tục

#### ▪ Gọi thủ tục: jump and link

```
jal ProcedureAddress
```

- Địa chỉ của lệnh kế tiếp (địa chỉ trả về) được cất ở thanh ghi \$ra
- Nhảy đến địa chỉ của thủ tục: nạp vào PC địa chỉ của lệnh đầu tiên của Thủ tục được gọi

#### ▪ Trở về từ thủ tục: jump register

```
jr $ra
```

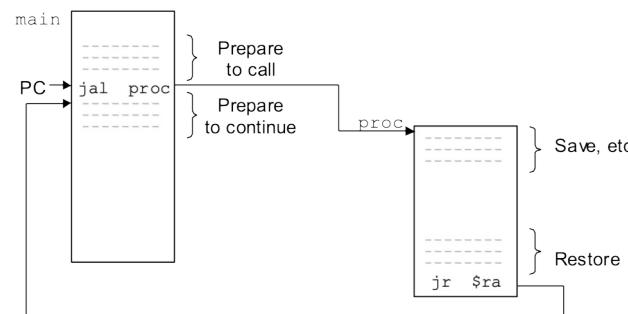
- Copy nội dung thanh ghi \$ra (đang chứa địa chỉ trả về) trả lại cho bộ đếm chương trình PC

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

102

### Minh họa gọi Thủ tục

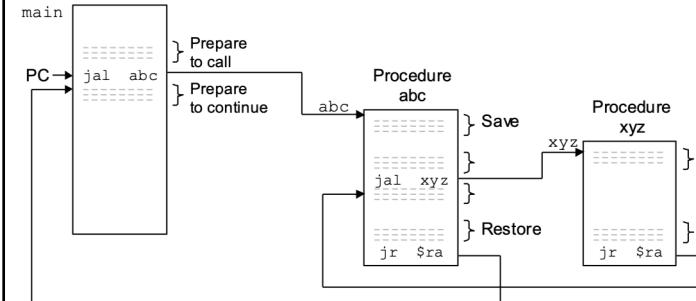


NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

103

### Gọi thủ tục lồng nhau



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

104

### Ví dụ Thủ tục lá

- Thủ tục lá là thủ tục không có lời gọi thủ tục khác
  - Mã C:
- ```
int leaf_example (int g, h, i, j)
{
    int f;
    f = (g + h) - (i + j);
    return f;
}
    
```
- Các tham số g, h, i, j ở \$a0, \$a1, \$a2, \$a3
 - f ở \$s0 (do đó, cần cất \$s0 ra ngăn xếp)
 - \$t0 và \$t1 được thủ tục dùng để chứa các giá trị tạm thời, cũng cần cất trước khi sử dụng
 - Kết quả ở \$v0

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

105

Mã hợp ngữ MIPS

leaf_example:

addi	\$sp, \$sp, -12	# tạo 3 vị trí ở stack
sw	\$t1, 8(\$sp)	# cất nội dung \$t1
sw	\$t0, 4(\$sp)	# cất nội dung \$t0
sw	\$s0, 0(\$sp)	# cất nội dung \$s0
add	\$t0, \$a0, \$a1	# \$t0 = g+h
add	\$t1, \$a2, \$a3	# \$t1 = i+j
sub	\$s0, \$t0, \$t1	# \$s0 = (g+h)-(i+j)
add	\$v0, \$s0, \$zero	# trả kết quả sang \$v0
lw	\$s0, 0(\$sp)	# khôi phục \$s0
lw	\$t0, 4(\$sp)	# khôi phục \$t0
lw	\$t1, 8(\$sp)	# khôi phục \$t1
addi	\$sp, \$sp, 12	# xóa 3 mục ở stack
jr	\$ra	# trở về nơi đã gọi

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

106

Ví dụ Thủ tục cành

- Là thủ tục có gọi thủ tục khác
 - Mã C:
- ```
int fact (int n)
{
 if (n < 1) return (1);
 else return n * fact(n - 1);
}

```
- Tham số n ở \$a0
  - Kết quả ở \$v0

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

107

### Mã hợp ngữ MIPS

**fact:**

|      |                     |                          |
|------|---------------------|--------------------------|
| addi | \$sp, \$sp, -8      | # dành stack cho 2 mục   |
| sw   | \$ra, 4(\$sp)       | # cất địa chỉ trả về     |
| sw   | \$a0, 0(\$sp)       | # cất tham số n          |
| slti | \$t0, \$a0, 1       | # kiểm tra n < 1 ?       |
| beq  | \$t0, \$zero, L1    | # nếu sai, rẽ xuống L1   |
| addi | \$v0, \$zero, 1     | # nếu đúng, kết quả là 1 |
| addi | \$sp, \$sp, 8       | # xóa 2 mục khỏi stack   |
| jr   | \$ra                | # và trả về              |
| L1:  | addi \$a0, \$a0, -1 | # giảm n                 |
| jal  | <b>fact</b>         | # gọi đệ quy             |
| lw   | \$a0, 0(\$sp)       | # khôi phục n ban đầu    |
| lw   | \$ra, 4(\$sp)       | # và địa chỉ trả về      |
| addi | \$sp, \$sp, 8       | # xóa 2 mục khỏi stack   |
| mul  | \$v0, \$a0, \$v0    | # nhân để nhận kết quả   |
| jr   | \$ra                | # và trả về              |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

108

#### 4.8. Các lệnh với số dấu phẩy động (FP)

- Các thanh ghi số dấu phẩy động
  - 32 thanh ghi 32-bit (single-precision): \$f0, \$f1, ... \$f31
  - Cặp đôi để chứa dữ liệu dạng 64-bit (double-precision): \$f0/\$f1, \$f2/\$f3, ...
- Các lệnh số dấu phẩy động chỉ thực hiện trên các thanh ghi số dấu phẩy động
- Lệnh load và store với FP
  - lwcl, ldc1, swcl, sdcl
    - Ví dụ: ldc1 \$f8, 32(\$s2)

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

109

#### Các lệnh với số dấu phẩy động

- Các lệnh số học với số FP 32-bit (single-precision)
  - add.s, sub.s, mul.s, div.s
  - VD: add.s \$f0, \$f1, \$f6
- Các lệnh số học với số FP 64-bit (double-precision)
  - add.d, sub.d, mul.d, div.d
  - VD: mul.d \$f4, \$f4, \$f6
- Các lệnh so sánh
  - c.xx.s, c.xx.d (trong đó xx là eq, lt, le, ...)
  - Thiết lập hoặc xóa các bit mã điều kiện
  - VD: c.lt.s \$f3, \$f4
- Các lệnh rẽ nhánh dựa trên mã điều kiện
  - bc1t, bc1f
  - VD: bc1t TargetLabel

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

110

#### 4.9. Các phương pháp định địa chỉ

- Phương pháp định địa chỉ (Addressing modes) là cách mã hóa trong lệnh để:
  - xác định nơi đọc/ghi toán hạng, hoặc
  - xác định địa chỉ của lệnh tiếp theo.
- MIPS có 5 phương pháp định địa chỉ
  - Định địa chỉ thanh ghi (Register Addressing)
  - Định địa chỉ tức thì (Immediate Addressing)
  - Định địa chỉ cơ sở (Base Addressing)
  - Định địa chỉ tương đối với PC (PC-relative Addressing)
  - Định địa chỉ giả trực tiếp (Pseudodirect Addressing)

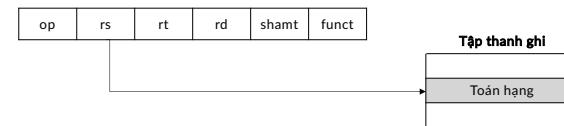
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

111

#### Định địa chỉ thanh ghi

- Các toán hạng nằm ở thanh ghi
- Tất cả các lệnh kiểu R sử dụng mode này
- Ví dụ:
  - add \$s0, \$t2, \$t3
  - sub \$t8, \$s1, \$0



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

112

### Định địa chỉ tức thì

- Toán hạng là hằng số 16-bit trong lệnh
- Ví dụ:
  - addi \$s3, \$t5, -20
  - ori \$s4, \$t7, 0xFF



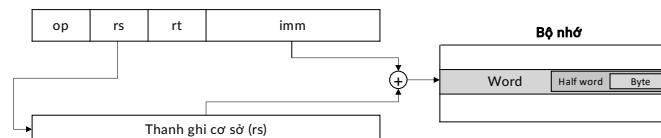
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

113

### Định địa chỉ cơ sở

- Toán hạng nằm ở bộ nhớ
- Địa chỉ toán hạng = Nội dung thanh ghi cơ sở (rs) + imm
- Ví dụ:
  - lw \$s4, 12(\$s6)  
Địa chỉ = (\$s6) + 12
  - sw \$t2, -20(\$s7)  
Địa chỉ = (\$s7) - 20



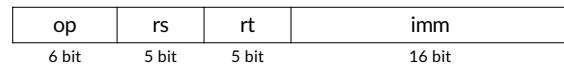
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

114

### Định địa chỉ tương đối với PC

- Lệnh máy của lệnh branch (beq, bne)
  - Mã thao tác, hai thanh ghi, hằng số
  - Rẽ xuôi hoặc rẽ ngược



#### So sánh nội dung 2 thanh ghi:

- Điều kiện đúng:  $PC \leftarrow \text{Địa chỉ đích}$ 
  - Địa chỉ của lệnh cần rẽ tới để thực hiện
  - Địa chỉ đích =  $(PC + 4) + \text{hằng số imm} \times 4$
  - Hằng số imm 16-bit có giá trị trong dải  $[-2^{15}, +2^{15} - 1]$
- Điều kiện sai:  $PC \leftarrow PC + 4$

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

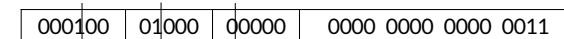
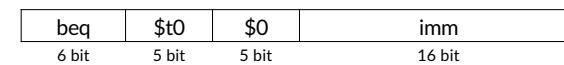
115

### Ví dụ tìm lệnh máy của lệnh beq/bne (1)

```

0x00400010 beq $t0, $0, L1
0x00400014 <lệnh kế tiếp>
0x00400018 ...
0x0040001C ...
0x00400020 L1: <lệnh tiếp theo>
0x00400024 ...

```



0x11000003

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

116

### Ví dụ tìm lệnh máy của lệnh beq/bne (2)

```

0x00400034 ...
0x00400038 L2: <lệnh tiếp theo>
0x0040003C ...
0x00400040 ...
0x00400044 ...
0x00400048 bne $s3, $s0, L2
0x0040004C <lệnh kế tiếp>

```

| bne    | \$s3  | \$s0  | imm                 |
|--------|-------|-------|---------------------|
| 6 bit  | 5 bit | 5 bit | 16 bit              |
| 5      | 19    | 16    | -5                  |
| 000101 | 10011 | 10000 | 1111 1111 1111 1011 |

0x1670FFFF

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

117

### Ví dụ xác định địa chỉ đích

Cho lệnh máy ở địa chỉ 0x00400050 là: 0x12110006

|        |       |       |                     |
|--------|-------|-------|---------------------|
| 000100 | 10000 | 10001 | 0000 0000 0000 0110 |
| 4      | 16    | 17    | 6                   |
| beq    | \$s0  | \$s1  | imm                 |

$$\begin{aligned} \text{Địa chỉ đích} &= (\text{PC}+4) + \text{imm} \times 4 = (0x00400050+4) + 6 \times 4 \\ &= 0x00400054 + 0x18 = 0x0040006C \end{aligned}$$

```

0x00400050 beq $s0, $s1, L
0x00400054 <lệnh kế tiếp>
...
...
0x0040006C L: <lệnh tiếp theo>

```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

118

### Định địa chỉ giả trực tiếp

- Đích của lệnh Jump (j và jal) có thể là bất kỳ chỗ nào trong chương trình
  - Cần mã hóa đầy đủ địa chỉ trong lệnh

| op    | address |
|-------|---------|
| 6 bit | 26 bit  |

Địa chỉ đích = PC<sub>31...28</sub> : (address): 00

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

119

### Ví dụ

```

0x0040005C j L1
0x00400060 <lệnh kế tiếp>
...
0x004000A0 L1: <lệnh tiếp theo>
0x004000A4 ...

```

Địa chỉ đích = 0x004000A0 =

|          |      |      |      |      |      |      |      |    |
|----------|------|------|------|------|------|------|------|----|
| 0000     | 0000 | 0100 | 0000 | 0000 | 0000 | 1010 | 00   | 00 |
| Lệnh máy | 0000 | 10   | 0000 | 0100 | 0000 | 0000 | 1010 | 00 |

j = 2 (0x08100028)

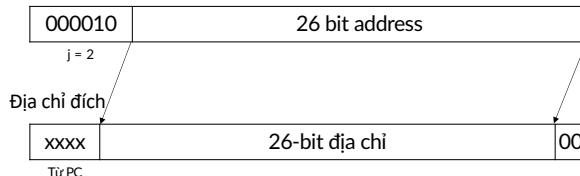
NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

120

### Cách xác định địa chỉ đích

j L1



Địa chỉ đích

Từ PC

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

121

### Ví dụ về mã hóa lệnh

|       |                      |            |    |    |    |   |   |          |
|-------|----------------------|------------|----|----|----|---|---|----------|
| Loop: | sll \$t1, \$s3, 2    | 0x00480000 | 0  | 0  | 19 | 9 | 2 | 0        |
|       | add \$t1, \$t1, \$s6 | 0x00480004 | 0  | 9  | 22 | 9 | 0 | 32       |
|       | lw \$t0, 0(\$t1)     | 0x00480008 | 35 | 9  | 8  |   |   | 0        |
|       | bne \$t0, \$s5, Exit | 0x0048000C | 5  | 8  | 21 |   |   | 2        |
|       | addi \$s3, \$s3, 1   | 0x00480010 | 8  | 19 | 19 |   |   | 1        |
|       | j Loop               | 0x00480014 | 2  |    |    |   |   | 0x120000 |
| Exit: |                      | 0x00480018 |    |    |    |   |   |          |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

122

### Ví dụ về mã hóa lệnh

|       |                      |            |    |    |    |   |   |          |
|-------|----------------------|------------|----|----|----|---|---|----------|
| Loop: | sll \$t1, \$s3, 2    | 0x00480000 | 0  | 0  | 19 | 9 | 2 | 0        |
|       | add \$t1, \$t1, \$s6 | 0x00480004 | 0  | 9  | 22 | 9 | 0 | 32       |
|       | lw \$t0, 0(\$t1)     | 0x00480008 | 35 | 9  | 8  |   |   | 0        |
|       | bne \$t0, \$s5, Exit | 0x0048000C | 5  | 8  | 21 |   |   | 2        |
|       | addi \$s3, \$s3, 1   | 0x00480010 | 8  | 19 | 19 |   |   | 1        |
|       | j Loop               | 0x00480014 | 2  |    |    |   |   | 0x120000 |
| Exit: |                      | 0x00480018 |    |    |    |   |   |          |

|                      |                                         |            |
|----------------------|-----------------------------------------|------------|
| sll \$t1, \$s3, 2    | 000000 00000 10011 01001 00010 000000   | 0x00134880 |
| add \$t1, \$t1, \$s6 | 000000 01001 10110 01001 00000 100000   | 0x01364820 |
| lw \$t0, 0(\$t1)     | 100011 01001 01000 0000 0000 0000 0000  | 0x8d280000 |
| bne \$t0, \$s5, Exit | 000101 01000 10101 0000 0000 0000 0010  | 0x15150002 |
| addi \$s3, \$s3, 1   | 001000 10011 10011 0000 0000 0000 0001  | 0x22730001 |
| j Loop               | 000010 00 0001 0010 0000 0000 0000 0000 | 0x08120000 |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

123

### 4.10. Dịch và chạy chương trình hợp ngữ

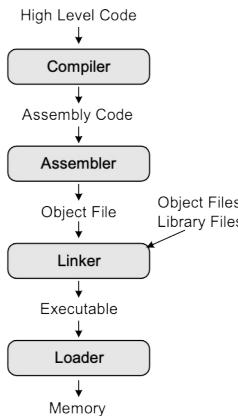
- Các phần mềm lập trình hợp ngữ MIPS:
  - MARS
  - MipsIt
  - QtSpim
- MIPS Reference Data

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

124

### Dịch và chạy ứng dụng



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

125

### Phần mềm mô phỏng MARS

- Bộ nhớ:

- $2^{32}$  bytes = 4 GiB
- Địa chỉ từ 0x00000000 đến 0xFFFFFFFF

- Vùng nhớ lệnh:

- Bắt đầu từ địa chỉ: 0x00400000

- Dữ liệu

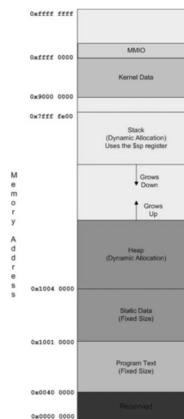
- Toàn cục/tĩnh: được cấp phát trước khi chương trình bắt đầu thực hiện
  - Bắt đầu ở địa chỉ: 0x10010000
- Động: được cấp phát trong khi chương trình thực hiện

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

126

### Bản đồ bộ nhớ



NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

127

### Ví dụ: Mã C

```

int f, g, y; // global variables

int main(void)
{
 f = 2;
 g = 3;
 y = sum(f, g);
 return y;
}

int sum(int a, int b) {
 return (a + b);
}

```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

128

### Ví dụ chương trình hợp ngữ MIPS

```

.data
f: .word 0
g: .word 0
y: .word 0
.text
main:
 addi $sp, $sp, -4 # stack frame
 sw $ra, 0($sp) # store $ra
 addi $a0, $0, 2 # $a0 = 2
 sw $a0, f # f = 2
 addi $a1, $0, 3 # $a1 = 3
 sw $a1, g # g = 3
 jal sum # call sum
 sw $v0, y # y = sum()
 lw $ra, 0($sp) # restore $ra
 addi $sp, $sp, 4 # restore $sp
 li $v0, 10
 syscall
sum:
 add $v0, $a0, $a1 # $v0 = a + b
 jr $ra # return

```

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

129

### Viết chương trình trên phần mềm MARS

```

1 .data
2 f: .word 0
3 g: .word 0
4 y: .word 0
5 .text
6 main:
7 addi $sp, $sp, -4 # stack frame
8 sw $ra, 0($sp) # store $ra
9 addi $a0, $0, 2 # $a0 = 2
10 sw $a0, f # f = 2
11 addi $a1, $0, 3 # $a1 = 3
12 sw $a1, g # g = 3
13 jal sum # call sum
14 sw $v0, y # y = sum()
15 lw $ra, 0($sp) # restore $ra
16 addi $sp, $sp, 4 # restore $sp
17 li $v0, 10
18 syscall
19 sum:
20 add $v0, $a0, $a1 # $v0 = a + b
21 jr $ra # return

```

The screenshot shows the MARS assembly editor interface with the assembly code for the program. The code is identical to the one shown in the previous window.

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

130

### Thực thi chương trình trên MARS

The screenshot shows the MARS debugger interface during execution. The assembly code is displayed in the main window, and the registers pane shows the state of the CPU registers. The registers pane includes columns for Name, Number, and Value. The value column shows the memory addresses where the register values are stored.

| Name   | Number | Value      |
|--------|--------|------------|
| \$zero | 0      | 0x00000000 |
| \$at   | 1      | 0x00000000 |
| \$t0   | 2      | 0x00000000 |
| \$v1   | 3      | 0x00000000 |
| \$s0   | 4      | 0x00000000 |
| \$a1   | 5      | 0x00000000 |
| \$a2   | 6      | 0x00000000 |
| \$t1   | 7      | 0x00000000 |
| \$t0   | 8      | 0x00000000 |
| \$t2   | 9      | 0x00000000 |
| \$t2   | 10     | 0x00000000 |
| \$t3   | 11     | 0x00000000 |
| \$s1   | 12     | 0x00000000 |
| \$s2   | 13     | 0x00000000 |
| \$t4   | 14     | 0x00000000 |
| \$s0   | 15     | 0x00000000 |
| \$s0   | 16     | 0x00000000 |
| \$a1   | 17     | 0x00000000 |
| \$s1   | 18     | 0x00000000 |
| \$s2   | 19     | 0x00000000 |
| \$s4   | 20     | 0x00000000 |
| \$s5   | 21     | 0x00000000 |
| \$t5   | 22     | 0x00000000 |
| \$t7   | 23     | 0x00000000 |
| \$t8   | 24     | 0x00000000 |
| \$t9   | 25     | 0x00000000 |
| \$sk1  | 26     | 0x00000000 |
| \$sp   | 27     | 0x00000000 |
| \$gp   | 28     | 0x00000000 |
| \$tp   | 29     | 0x00000000 |
| \$fp   | 30     | 0x00000000 |
| \$t9   | 31     | 0x00000000 |
| pc     |        | 0x00000000 |
| h1     |        | 0x00000000 |
| lo     |        | 0x00000000 |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

131

### Vùng nhớ lệnh

The screenshot shows the MARS debugger interface during execution, focusing on the memory dump and registers. The memory dump shows the state of memory at various addresses, and the registers pane shows the state of the CPU registers. The registers pane includes columns for Name, Number, and Value.

| Name   | Number | Value      |
|--------|--------|------------|
| \$zero | 0      | 0x00000000 |
| \$at   | 1      | 0x00000000 |
| \$t0   | 2      | 0x00000000 |
| \$v1   | 3      | 0x00000000 |
| \$s0   | 4      | 0x00000000 |
| \$a1   | 5      | 0x00000000 |
| \$a2   | 6      | 0x00000000 |
| \$t1   | 7      | 0x00000000 |
| \$t0   | 8      | 0x00000000 |
| \$t2   | 9      | 0x00000000 |
| \$t2   | 10     | 0x00000000 |
| \$t3   | 11     | 0x00000000 |
| \$s1   | 12     | 0x00000000 |
| \$s2   | 13     | 0x00000000 |
| \$t4   | 14     | 0x00000000 |
| \$s0   | 15     | 0x00000000 |
| \$s0   | 16     | 0x00000000 |
| \$a1   | 17     | 0x00000000 |
| \$s1   | 18     | 0x00000000 |
| \$s2   | 19     | 0x00000000 |
| \$t5   | 20     | 0x00000000 |
| \$t5   | 21     | 0x00000000 |
| \$t5   | 22     | 0x00000000 |
| \$t7   | 23     | 0x00000000 |
| \$t5   | 24     | 0x00000000 |
| \$t9   | 25     | 0x00000000 |
| \$t5   | 26     | 0x00000000 |
| \$t1   | 27     | 0x00000000 |
| \$sp   | 28     | 0x00000000 |
| \$gp   | 29     | 0x00000000 |
| \$tp   | 30     | 0x00000000 |
| \$t9   | 31     | 0x00000000 |
| pc     |        | 0x00000000 |
| h1     |        | 0x00000000 |
| lo     |        | 0x00000000 |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

132

**Tập thanh ghi**

| Name   | Number | Value        |
|--------|--------|--------------|
| \$zero | 0      | 0x00000000   |
| \$at   | 1      | 0x00000000   |
| \$v0   | 2      | 0x00000000   |
| \$v1   | 3      | 0x00000000   |
| \$a0   | 4      | 0x00000000   |
| \$a1   | 5      | 0x00000000   |
| \$a2   | 6      | 0x00000000   |
| \$a3   | 7      | 0x00000000   |
| \$t0   | 8      | 0x00000000   |
| \$t1   | 9      | 0x00000000   |
| \$t2   | 10     | 0x00000000   |
| \$t3   | 11     | 0x00000000   |
| \$t4   | 12     | 0x00000000   |
| \$t5   | 13     | 0x00000000   |
| \$t6   | 14     | 0x00000000   |
| \$t7   | 15     | 0x00000000   |
| \$s0   | 16     | 0x00000000   |
| \$s1   | 17     | 0x00000000   |
| \$s2   | 18     | 0x00000000   |
| \$s3   | 19     | 0x00000000   |
| \$s4   | 20     | 0x00000000   |
| \$s5   | 21     | 0x00000000   |
| \$s6   | 22     | 0x00000000   |
| \$s7   | 23     | 0x00000000   |
| \$t8   | 24     | 0x00000000   |
| \$t9   | 25     | 0x00000000   |
| \$sp   | 26     | 0x00000000   |
| \$k1   | 27     | 0x00000000   |
| \$gp   | 28     | 0x10000000   |
| \$gp   | 29     | 0x7ffffefffc |
| \$fp   | 30     | 0x00000000   |
| \$ra   | 31     | 0x00000000   |
| pc     |        | 0x00400000   |
| hi     |        | 0x00000000   |
| lo     |        | 0x00000000   |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

133

**Vùng nhớ dữ liệu**

| Address    | Data Segment |            |            |            |             |             |             |             |
|------------|--------------|------------|------------|------------|-------------|-------------|-------------|-------------|
|            | Value (+0)   | Value (+4) | Value (+8) | Value (+c) | Value (+10) | Value (+14) | Value (+18) | Value (+1c) |
| 0x10010000 | 0x00000000   | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000  | 0x00000000  | 0x00000000  | 0x00000000  |
| 0x10010020 | 0x00000000   | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000  | 0x00000000  | 0x00000000  | 0x00000000  |
| 0x10010040 | 0x00000000   | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000  | 0x00000000  | 0x00000000  | 0x00000000  |
| 0x10010060 | 0x00000000   | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000  | 0x00000000  | 0x00000000  | 0x00000000  |
| 0x10010080 | 0x00000000   | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000  | 0x00000000  | 0x00000000  | 0x00000000  |
| 0x100100a0 | 0x00000000   | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000  | 0x00000000  | 0x00000000  | 0x00000000  |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

134

**Nhãn và biến**

| Labels                 |            |
|------------------------|------------|
| Label                  | Address    |
| <b>fullprogram.asm</b> |            |
| main                   | 0x00400000 |
| sum                    | 0x0040003c |
| f                      | 0x10010000 |
| g                      | 0x10010004 |
| y                      | 0x10010008 |

Data     Text

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

135

**Ví dụ lệnh giả (Pseudoinstruction)**

| Pseudoinstruction  | MIPS Instructions    |
|--------------------|----------------------|
| li \$s0,0x1234AA77 | lui \$at,0x1234      |
|                    | ori \$s0,\$at,0xAA77 |
| not \$t1,\$t2      | nor \$t1,\$t2,\$0    |
| move \$s1,\$s2     | addu \$s1, \$0, \$s2 |
| nop                | sll \$0, \$0, 0      |

NKK-IT3030-CA2021.2

CH4-Kiến trúc tập lệnh

136

#### 4.11. Các kiến trúc tập lệnh phổ biến

- Kiến trúc Intel x86

- Từ 1978
- Tập lệnh phức tạp (CISC)
- Độ dài lệnh: 1-15 byte
- 16/32/64-bit

- Kiến trúc ARM

- Từ 1985
- Tập lệnh thu gọn (RISC)
- Độ dài lệnh: 4 byte (2 byte)
- 32/64-bit

#### Kiến trúc máy tính

Hết chương 4



## Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

2

## Kiến trúc máy tính

### Chương 5 BỘ XỬ LÝ

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

3

## Nội dung

- 5.1. Tổ chức của CPU
- 5.2. Thiết kế bộ xử lý theo kiến trúc MIPS
- 5.3. Kỹ thuật đường ống lệnh và song song mức lệnh

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

4

## 5.1. Tổ chức của CPU

### 1. Cấu trúc cơ bản của CPU

#### ▪ Nhiệm vụ của CPU:

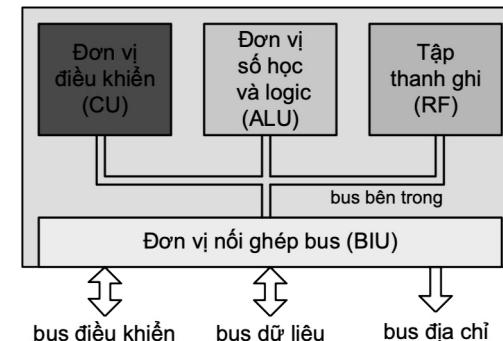
- Nhận lệnh (Fetch Instruction): CPU đọc lệnh từ bộ nhớ
- Giải mã lệnh (Decode Instruction): xác định thao tác mà lệnh yêu cầu
- Nhận dữ liệu (Fetch Data): nhận dữ liệu từ bộ nhớ hoặc các cổng vào-ra
- Xử lý dữ liệu (Process Data): thực hiện phép toán số học hay phép toán logic với các dữ liệu
- Ghi dữ liệu (Write Data): ghi dữ liệu ra bộ nhớ hay cổng vào-ra

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

5

## Sơ đồ cấu trúc cơ bản của CPU



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

6

## 2. Đơn vị số học và logic

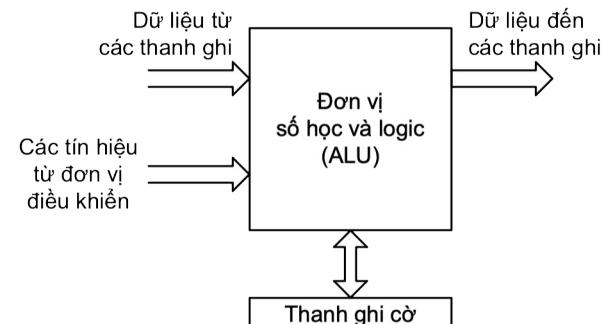
- **Chức năng:** Thực hiện các phép toán số học và phép toán logic:
  - Số học: cộng, trừ, nhân, chia, đảo dấu
  - Logic: AND, OR, XOR, NOT, phép dịch bit
- **Các cờ phép toán:** các bit ở thanh ghi cờ hiển thị trạng thái của kết quả phép toán.
  - Ví dụ cờ của các bộ xử lý Intel x86:
    - Cờ Zero (ZF): nếu kết quả phép toán = 0 thì ZF=1
    - Cờ Sign (SF): nếu kết quả phép toán < 0 thì SF=1
    - Cờ Carry (CF): nếu phép toán có nhớ ra khỏi bit cao nhất thì CF=1
    - Cờ Overflow (OF): nếu phép toán có tràn với số nguyên có dấu thì cờ OF=1

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

7

## Mô hình kết nối ALU



Thanh ghi cờ: hiển thị trạng thái của kết quả phép toán

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

8

### 3. Đơn vị điều khiển

#### ▪ Chức năng

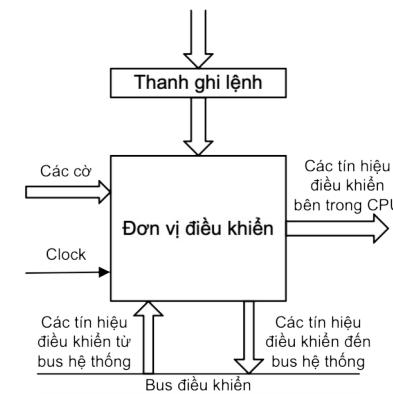
- Điều khiển nhận lệnh từ bộ nhớ đưa vào CPU
- Tăng nội dung của PC để trả sang lệnh kế tiếp
- Giải mã lệnh đã được nhận để xác định thao tác mà lệnh yêu cầu
- Phát ra các tín hiệu điều khiển thực hiện lệnh
- Nhận các tín hiệu yêu cầu từ bus hệ thống và đáp ứng với các yêu cầu đó.

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

9

### Mô hình kết nối đơn vị điều khiển



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

10

### Các tín hiệu đưa đến đơn vị điều khiển

- Clock: tín hiệu nhịp từ mạch tạo dao động bên ngoài
- Lệnh máy từ thanh ghi lệnh đưa đến để giải mã
- Các cờ từ thanh ghi cờ cho biết trạng thái của CPU
- Các tín hiệu yêu cầu từ bus điều khiển

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

11

### Các tín hiệu phát ra từ đơn vị điều khiển

- Các tín hiệu điều khiển bên trong CPU:
  - Điều khiển các thanh ghi
  - Điều khiển ALU
- Các tín hiệu điều khiển bên ngoài CPU:
  - Điều khiển bộ nhớ
  - Điều khiển các mô-đun vào-ra

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

12

## Các phương pháp thiết kế đơn vị điều khiển

- Đơn vị điều khiển vi chương trình  
(Microprogrammed Control Unit)
  - Đơn vị điều khiển nối kết cứng  
(Hardwired Control Unit)

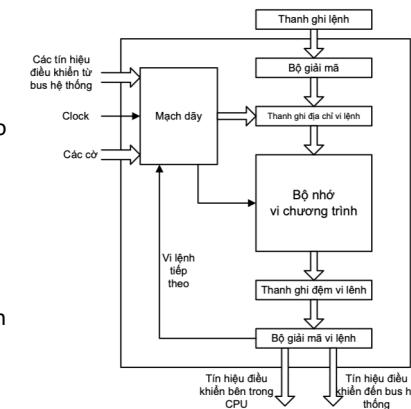
NKK-IT3030-CA2021.2

CH5-Bộ xử lý

13

## Đơn vị điều khiển vi chương trình

- Bộ nhớ vi chương trình (ROM) lưu trữ các vi chương trình (microprogram)
  - Một vi chương trình bao gồm các vi lệnh (microinstruction)
  - Mỗi vi lệnh mã hóa cho một vi thao tác (microoperation)
  - Để hoàn thành một lệnh cần thực hiện một hoặc một vài vi chương trình
  - Tốc độ chậm



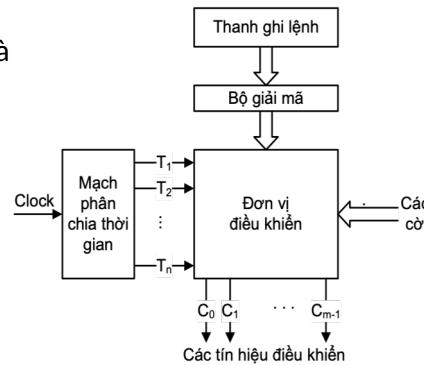
NKK-IT3030-CA2021.

CH5-Bộ xử lý

1

## Đơn vị điều khiển nối kết cứng

- Sử dụng mạch cứng để giải mã và tạo các tín hiệu điều khiển thực hiện lệnh
  - Tốc độ nhanh
  - Đơn vị điều khiển phức tạp



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

15

#### 4. Hoạt động của chu trình lệnh

## Chu trình lệnh

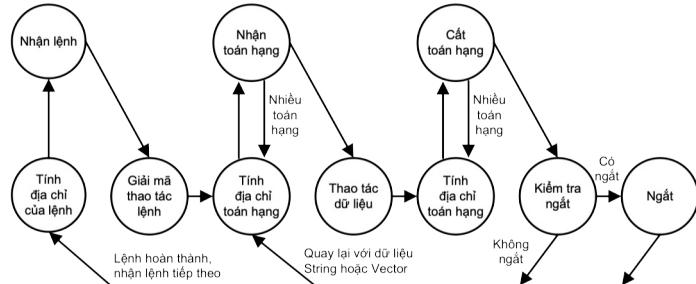
- Nhận lệnh
  - Giải mã lệnh
  - Nhận toán hạng
  - Thực hiện lệnh
  - Cất toán hạng
  - Ngắt

NKK-IT3030-CA2021.

CH5-Bộ xử lý

1

### Giản đồ trạng thái chu trình lệnh



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

17

### Tính địa chỉ của lệnh

- Bộ đếm chương trình PC (Program Counter) chứa địa chỉ của lệnh sẽ được nhận vào
- Với MIPS:
  - Tuần tự:  $PC = PC + 4$
  - Rẽ nhánh (đk đúng):  $PC = (PC+4) + imm \times 4$
  - Nhảy:  $PC = PC_{31-28}: (26bit địa chỉ) : 00$

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

18

### Nhận lệnh

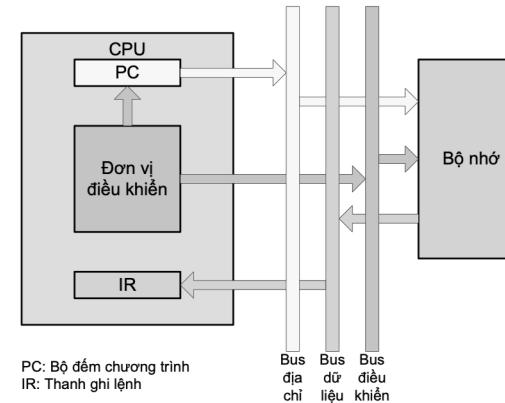
- CPU đưa địa chỉ của lệnh cần nhận từ bộ đếm chương trình PC ra bus địa chỉ đến bộ nhớ để xác định ngăn nhớ chứa lệnh đó
- CPU phát tín hiệu điều khiển đọc bộ nhớ
- Lệnh từ bộ nhớ được đặt lên bus dữ liệu và được CPU copy vào thanh ghi lệnh IR
- CPU tăng nội dung PC để trỏ sang lệnh kế tiếp

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

19

### Sơ đồ mô tả quá trình nhận lệnh

PC: Bộ đếm chương trình  
IR: Thanh ghi lệnh

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

20

### Giải mã lệnh

- Lệnh từ thanh ghi lệnh IR được đưa đến đơn vị điều khiển
- Đơn vị điều khiển tiến hành giải mã lệnh để xác định thao tác phải thực hiện
- Giải mã lệnh xảy ra bên trong CPU

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

21

### Tính địa chỉ toán hạng

- Với MIPS: lệnh lw/ lh/ lb

$$lw \quad rt, \text{imm}(rs) \#(rt) = \text{mem}[(rs) + \text{imm}]$$

Địa chỉ toán hạng = Nội dung thanh ghi rs + hằng số imm

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

22

### Nhận dữ liệu từ bộ nhớ

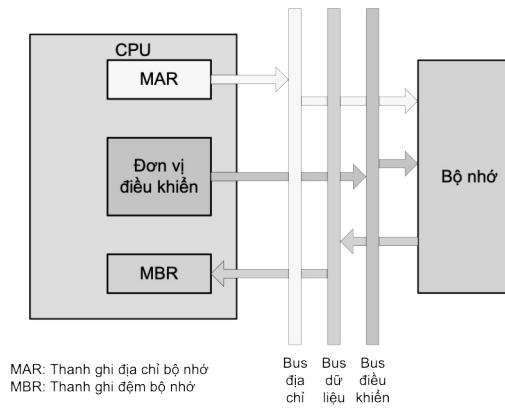
- CPU đưa địa chỉ của toán hạng ra bus địa chỉ để xác định ngăn nhớ chứa dữ liệu cần nhận
- CPU phát tín hiệu điều khiển đọc
- Toán hạng được đọc vào CPU
- Tương tự như nhận lệnh

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

23

### Sơ đồ mô tả nhận dữ liệu từ bộ nhớ



MAR: Thanh ghi địa chỉ bộ nhớ  
MBR: Thanh ghi đệm bộ nhớ

Bus  
địa  
chi  
Bus  
dữ  
liệu  
Bus  
điều  
kiện

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

24

## Thực hiện lệnh

- Có nhiều dạng tùy thuộc vào lệnh
- Có thể là:
  - Đọc/Ghi bộ nhớ
  - Vào/Ra
  - Chuyển giữa các thanh ghi
  - Phép toán số học/logic
  - Chuyển điều khiển (rẽ nhánh)
  - ...

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

25

## Tính địa chỉ toán hạng

- Với MIPS: lệnh sw/ sh/ sb

$$sw \quad rt, \text{ imm(rs)} \# \text{mem}[(rs) + \text{imm}] = rt$$

Địa chỉ toán hạng = Nội dung thanh ghi rs + hằng số imm

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

26

## Ghi toán hạng

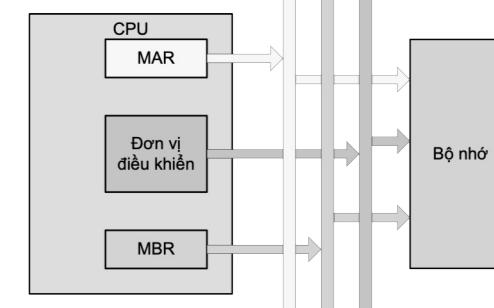
- CPU đưa địa chỉ ra bus địa chỉ
- CPU đưa dữ liệu cần ghi ra bus dữ liệu
- CPU phát tín hiệu điều khiển ghi
- Dữ liệu trên bus dữ liệu được copy đến vị trí xác định

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

27

## Sơ đồ mô tả quá trình ghi toán hạng



MAR: Thanh ghi địa chỉ bộ nhớ  
MBR: Thanh ghi đệm bộ nhớ

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

28

## Ngắt

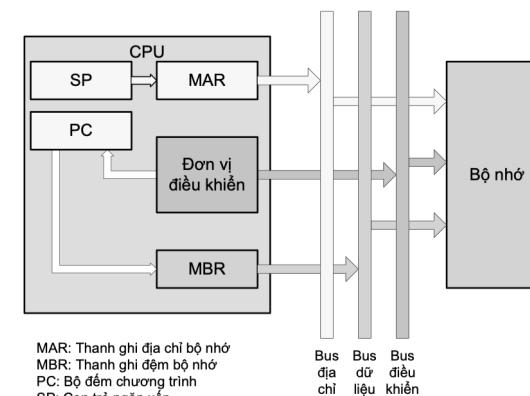
- Nội dung của bộ đếm chương trình PC (địa chỉ trở về sau khi ngắt) được đưa ra bus dữ liệu
- CPU đưa địa chỉ (thường được lấy từ con trỏ ngăn xếp SP) ra bus địa chỉ
- CPU phát tín hiệu điều khiển ghi bộ nhớ
- Địa chỉ trở về trên bus dữ liệu được ghi ra vị trí xác định (ở ngăn xếp)
- Địa chỉ lệnh đầu tiên của chương trình con điều khiển ngắt được nạp vào PC

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

29

## Sơ đồ mô tả chu trình ngắt



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

30

## 5.2. Thiết kế bộ xử lý theo kiến trúc MIPS

- Thực hiện với một số lệnh cơ bản của MIPS, nhưng chỉ ra hầu hết các khía cạnh:
  - Các lệnh tham chiếu bộ nhớ: lw, sw
  - Các lệnh số học/logic kiểu R: add, sub, and, or, slt
  - Các lệnh chuyển điều khiển: beq, j

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

31

## Tổng quan quá trình thực hiện các lệnh

- Hai bước đầu tiên với mỗi lệnh:
  - Đưa địa chỉ từ bộ đếm chương trình PC đến bộ nhớ lệnh, tìm và nhận lệnh từ bộ nhớ này
  - Sử dụng các số hiệu thanh ghi trong lệnh để chọn và đọc một hoặc hai thanh ghi:
    - Lệnh lw: đọc 1 thanh ghi
    - Các lệnh khác (không kể lệnh jump): đọc 2 thanh ghi

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

32

### Tổng quan quá trình thực hiện các lệnh (tiếp)

- Các bước tiếp theo tùy thuộc vào loại lệnh:
  - Sử dụng ALU hoặc bộ cộng Add để:
    - Tính kết quả phép toán với các lệnh số học/logic
    - So sánh các toán hạng với lệnh branch
    - Tính địa chỉ đích với các lệnh branch
    - Tính địa chỉ ngăn nhớ dữ liệu với lệnh load/store
  - Truy cập bộ nhớ dữ liệu với lệnh load/store
    - Lệnh lw: đọc dữ liệu từ bộ nhớ
    - Lệnh sw: ghi dữ liệu ra bộ nhớ
  - Ghi dữ liệu đến thanh ghi đích:
    - Các lệnh số học/logic: kết quả phép toán
    - Lệnh lw: dữ liệu được đọc từ bộ nhớ dữ liệu

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

33

### Tổng quan quá trình thực hiện các lệnh (tiếp)

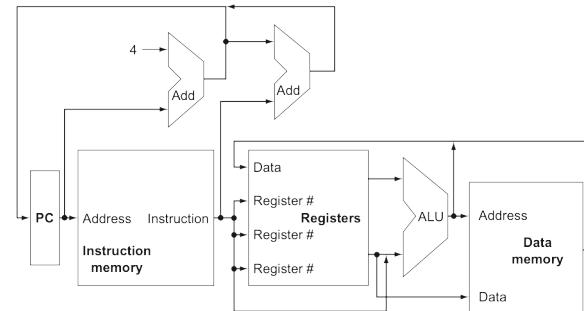
- Thay đổi nội dung bộ đếm chương trình PC:
  - Với các lệnh rẽ nhánh (branch), tùy thuộc vào kết quả so sánh:
    - Điều kiện thỏa mãn:  $PC \leftarrow$  địa chỉ đích (địa chỉ của lệnh cần rẽ tới)
    - Điều kiện không thỏa mãn:  $PC \leftarrow PC + 4$  (địa chỉ của lệnh kế tiếp)
  - Với các lệnh còn lại (không kể các lệnh jump)
    - $PC \leftarrow PC + 4$  (địa chỉ của lệnh kế tiếp)

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

34

### Sơ đồ khái quát của bộ xử lý MIPS

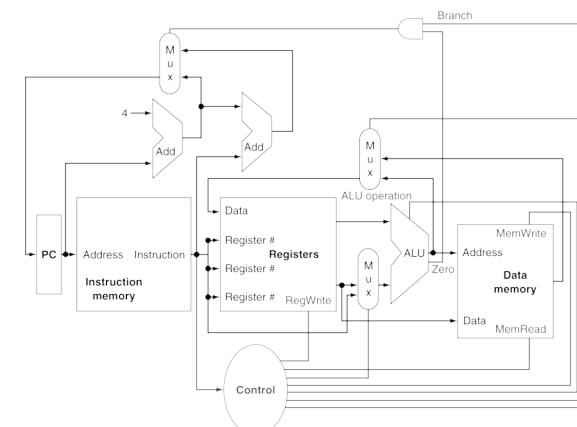


NKK-IT3030-CA2021.2

CH5-Bộ xử lý

35

### Bộ xử lý với các đường điều khiển chính



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

36

### 1. Thiết kế Datapath

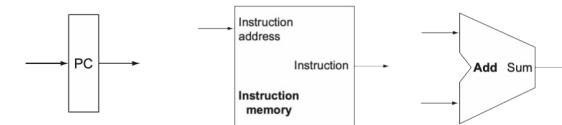
- Datapath: gồm các thành phần để xử lý dữ liệu và địa chỉ
  - Tập thanh ghi, ALUs, MUX's, bộ nhớ, ...
- Sẽ xây dựng tăng dần Datapath cho MIPS

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

37

### Các thành phần để thực hiện nhận lệnh



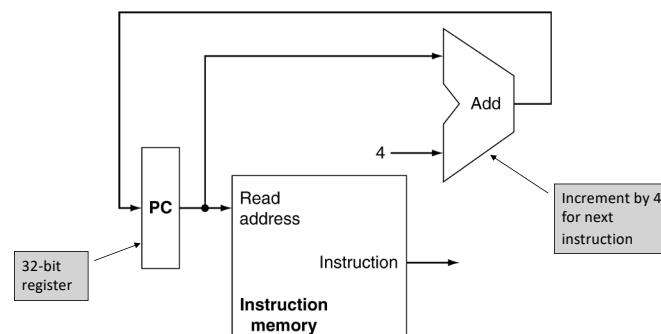
- Bộ đếm chương trình PC:
  - Thanh ghi 32-bit chứa địa chỉ của lệnh hiện tại
- Bộ nhớ lệnh (Instruction memory):
  - Chứa các lệnh của chương trình
  - Khi có địa chỉ lệnh từ PC đưa đến thì lệnh được đọc ra
- Bộ cộng (Add): được sử dụng tăng nội dung PC thêm 4 để trả tới lệnh kế tiếp

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

38

### Thực hiện phần nhận lệnh



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

39

### Thực hiện lệnh số học/logic kiểu R

| op    | rs    | rt    | rd    | shamt | funct |
|-------|-------|-------|-------|-------|-------|
| 31:26 | 25:21 | 20:16 | 15:11 | 10:6  | 5:0   |

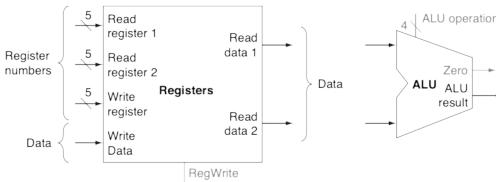
- bits 31:26: mã thao tác (opcode)
  - 000000 với các lệnh kiểu R
- bits 25:21: số hiệu thanh ghi nguồn thứ nhất rs
- bits 20:16: số hiệu thanh ghi nguồn thứ hai rt
- bits 15:11: số hiệu thanh ghi đích rd
- bits 10:6: số bit được dịch với các lệnh dịch bit
  - 00000 với các lệnh khác
- bits 5:0: mã hàm để xác định phép toán (function code)

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

40

### Các thành phần thực hiện lệnh kiểu R



- Tập thanh ghi (**Registers**): có 32 thanh ghi 32-bit, mỗi thanh ghi được xác định bởi số hiệu 5-bit
  - Read register 1, Read register 2: các đầu vào để chọn các thanh ghi cần đọc
  - Write register: đầu vào để chọn thanh ghi cần ghi
  - Read data 1, Read data 2: hai đầu ra dữ liệu đọc từ thanh ghi (32-bit)
  - Write Data: đầu vào dữ liệu ghi vào thanh ghi (32-bit)
  - RegWrite: tín hiệu điều khiển ghi dữ liệu vào thanh ghi
- ALU** để thực hiện các phép toán số học/logic

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

41

### Mô tả thực hiện lệnh số học/logic kiểu R

- Hai số hiệu thanh ghi rs và rt đưa đến hai đầu vào Read register 1, Read register 2 để chọn hai thanh ghi nguồn
- Số hiệu thanh ghi rd đưa đến đầu vào Write register để chọn thanh ghi đích
- Hai dữ liệu từ hai thanh ghi nguồn được đọc ra 2 đầu ra Read data 1 và Read data 2, rồi đưa đến đầu vào của ALU
- ALU operation (4-bit)**: tín hiệu điều khiển chọn phép toán ở ALU
- ALU thực hiện phép toán tương ứng, kết quả phép toán ở đầu ra ALU result được đưa về đầu vào Write Data của tập thanh ghi để ghi vào thanh ghi đích
- RegWrite: tín hiệu điều khiển ghi dữ liệu vào thanh ghi đích

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

42

### Thực hiện các lệnh lw/sw

| op    | rs    | rt    | imm  |
|-------|-------|-------|------|
| 31:26 | 25:21 | 20:16 | 15:0 |

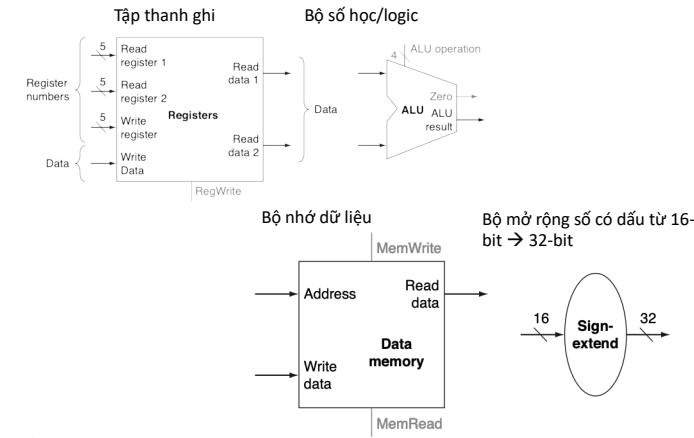
- bits 31:26 là mã thao tác
  - 100011 (35) với lệnh lw
  - 101011 (43) với lệnh sw
- bits 25:21: số hiệu thanh ghi cơ sở rs
- bits 20:16: số hiệu thanh ghi rt
  - thanh ghi đích với lệnh lw
  - thanh ghi nguồn với lệnh sw
- bits 15:0: hằng số imm có giá trị trong dải  $[-2^{15}, +2^{15} - 1]$
- Địa chỉ từ nhớ dữ liệu = nội dung thanh ghi rs + imm
- lw rt, imm(rs) #(rt) = mem[(rs)] + SignExtImm
- sw rt, imm(rs) #mem[(rs)] + SignExtImm] = (rt)

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

43

### Các thành phần thực hiện các lệnh lw/sw



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

44

### Thực hiện lệnh lw

- Số hiệu thanh ghi rs đưa đến đầu vào Read register 1 để chọn thanh ghi cơ sở rs, nội dung rs được đưa ra đầu ra Read Data 1, rồi chuyển đến ALU
- Hằng số imm 16-bit đưa đến bộ Sign-extend để mở rộng thành 32-bit, rồi chuyển đến ALU
- ALU cộng hai giá trị trên đưa ra ALU result chính là địa chỉ của dữ liệu cần đọc từ bộ nhớ dữ liệu; địa chỉ này được đưa đến đầu vào Address của bộ nhớ dữ liệu
- Số hiệu thanh ghi rt đưa đến đầu vào Write Register để chọn thanh ghi đích
- Dữ liệu 32-bit ở bộ nhớ dữ liệu, tại vị trí địa chỉ đã được tính, được đọc ra ở đầu ra Read data của bộ nhớ dữ liệu nhờ tín hiệu điều khiển MemRead, rồi đưa về đầu vào Write Data của tập thanh ghi để ghi vào thanh ghi đích rt

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

45

### Thực hiện lệnh sw

- Số hiệu thanh ghi rs đưa đến đầu vào Read register 1 để chọn thanh ghi cơ sở rs, nội dung rs được đưa ra đầu ra Read Data 1, rồi chuyển đến ALU
- Hằng số imm 16-bit đưa đến bộ Sign-extend để mở rộng thành 32-bit, rồi chuyển đến ALU
- ALU cộng hai giá trị trên đưa ra ALU result chính là địa chỉ của vị trí ở bộ nhớ dữ liệu; địa chỉ này được đưa đến đầu vào Address của bộ nhớ dữ liệu
- Số hiệu thanh ghi rt đưa đến đầu vào Read register 2 để chọn thanh ghi dữ liệu nguồn rt, nội dung rt được đưa ra đầu ra Read data 2
- Dữ liệu 32-bit này sẽ được đưa đến đầu vào Write data của bộ nhớ dữ liệu và được ghi vào vị trí nhớ có địa chỉ đã được tính, nhờ tín hiệu điều khiển MemWrite

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

46

### Thực hiện lệnh Branch (beq, bne)

| op    | rs    | rt    | imm  |
|-------|-------|-------|------|
| 31:26 | 25:21 | 20:16 | 15:0 |

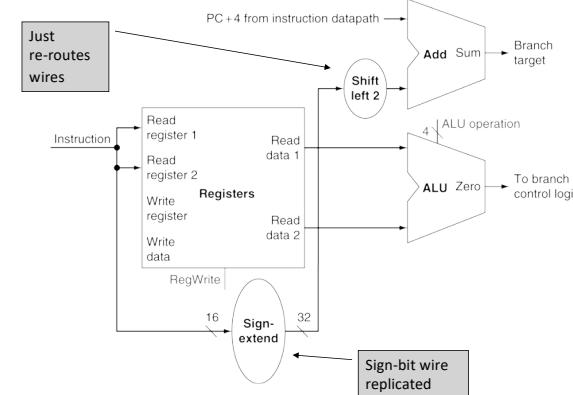
- bits 31:26 mã thao tác (op)
  - 000100 (4) với lệnh beq
  - 000101 (5) với lệnh bne
- bits 25:21 số hiệu thanh ghi nguồn rs
- bits 20:16 số hiệu thanh ghi nguồn rt
- bits 15:0 hằng số imm có giá trị trong dải  $[-2^{15}, +2^{15} - 1]$
- So sánh nội dung hai thanh ghi rs và rt:
  - Nếu điều kiện đúng: rẽ nhánh đến nhãn đích
    - $PC \leftarrow (PC + 4) + \text{hằng số} \times 4$
  - Nếu điều kiện sai: chuyển sang thực hiện lệnh kế tiếp
    - $PC \leftarrow PC + 4$

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

47

### Các thành phần thực hiện lệnh Branch



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

48

### Thực hiện lệnh Branch (beq, bne)

- Nhờ các số hiệu thanh ghi rs và rt, hai toán hạng nguồn được đọc ra đưa đến ALU
- ALU trừ hai toán hạng và thiết lập giá trị ở đầu ra “Zero”
  - Hiệu = 0 → đầu ra Zero = 1
  - Hiệu <> 0 → đầu ra Zero = 0
  - Đầu ra Zero này được đưa đến mạch logic điều khiển rẽ nhánh
- Bộ cộng Add tính địa chỉ đích rẽ nhánh
  - Hằng số imm 16-bit được mở rộng theo kiểu có dấu thành 32-bit, rồi dịch trái 2 bit
  - Cộng với PC (PC đã được tăng 4)
  - Địa chỉ đích =  $(PC+4) + (\text{hằng số imm} \text{ đã mở rộng 32-bit, dịch trái 2 bit})$
- Điều kiện đúng:  $PC \leftarrow \text{địa chỉ đích rẽ nhánh}$  (rẽ nhánh xảy ra)
- Điều kiện sai:  $PC \leftarrow PC+4$  (chuyển sang lệnh kế tiếp)

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

49

### Hợp các thành phần cho các lệnh

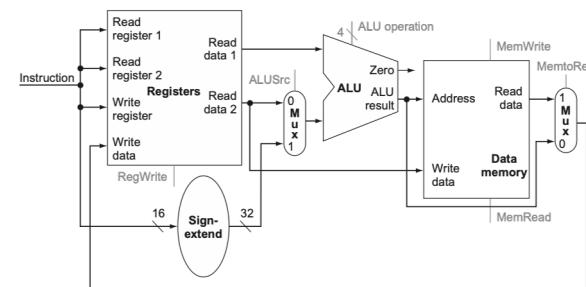
- Datapath cho các lệnh thực hiện trong 1 chu kỳ
  - Mỗi phần tử của datapath chỉ có thể làm một chức năng trong mỗi chu kỳ
  - Do đó, cần tách rời bộ nhớ lệnh và bộ nhớ dữ liệu
- Sử dụng các bộ chọn kênh để chọn dữ liệu nguồn cho các lệnh khác nhau

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

50

### Datapath cho các lệnh R-Type/Load/Store



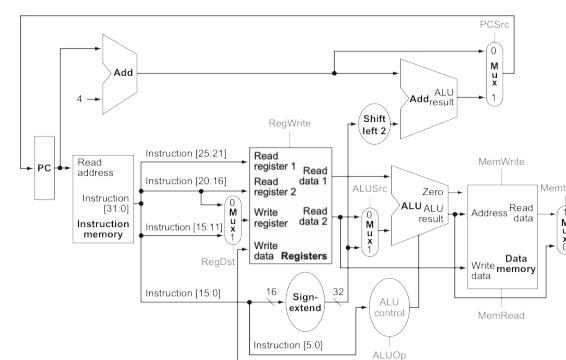
- ALUSrc: tín hiệu điều khiển chọn toán hạng đưa đến ALU:
  - Lệnh kiểu R: toán hạng từ thanh ghi nguồn thứ hai
  - Lệnh lw/sw: Hằng số imm 16-bit được mở rộng thành 32-bit (tính địa chỉ)
- MemtoReg: tín hiệu điều khiển chọn dữ liệu đưa về thanh ghi đích:
  - Lệnh kiểu R: lấy kết quả từ ALU result
  - Lệnh lw: dữ liệu đọc (Read data) từ bộ nhớ dữ liệu

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

51

### Datapath đơn giản cho các lệnh R/lw/sw/branch



- PCSrc: tín hiệu điều khiển chọn giá trị cập nhật PC
  - Không rẽ nhánh:  $PC \leftarrow PC+4$
  - Rẽ nhánh:  $PC \leftarrow (PC+4) + (\text{hằng số imm đã mở rộng thành 32-bit} \ll 2)$

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

52

## 2. Thiết kế Control Unit

- Đơn vị điều khiển có hai phần:
  - Bộ điều khiển ALU
  - Bộ điều khiển chính

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

53

## Thiết kế bộ điều khiển ALU

- ALU được sử dụng để:
  - Load/Store: F = add (xác định địa chỉ bộ nhớ dữ liệu)
  - Branch: F = subtract (so sánh)
  - Các lệnh số học/logic : F phụ thuộc vào funct code

| ALU control lines | Function         |
|-------------------|------------------|
| 0000              | AND              |
| 0001              | OR               |
| 0010              | add              |
| 0110              | subtract         |
| 0111              | set-on-less-than |
| 1100              | NOR              |

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

54

## Tín hiệu điều khiển ALU

- Bộ điều khiển ALU sử dụng mạch logic tổ hợp:
  - Đầu vào: 2-bit ALUOp được tạo ra từ opcode của lệnh và 6-bit của function code
  - Đầu ra: các tín hiệu điều khiển ALU (ALU control) gồm 4 bit

| Opcode | ALUOp | Operation        | funct  | ALU function     | ALU control |
|--------|-------|------------------|--------|------------------|-------------|
| lw     | 00    | load word        | XXXXXX | add              | 0010        |
| sw     | 00    | store word       | XXXXXX | add              | 0010        |
| beq    | 01    | branch equal     | XXXXXX | subtract         | 0110        |
| R-type | 10    | add              | 100000 | add              | 0010        |
|        |       | subtract         | 100010 | subtract         | 0110        |
|        |       | AND              | 100100 | AND              | 0000        |
|        |       | OR               | 100101 | OR               | 0001        |
|        |       | set-on-less-than | 101010 | set-on-less-than | 0111        |

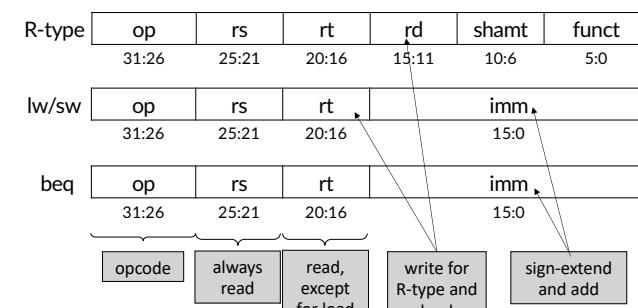
NKK-IT3030-CA2021.2

CH5-Bộ xử lý

55

## Thiết kế bộ điều khiển chính

- Các tín hiệu điều khiển được tạo ra từ lệnh

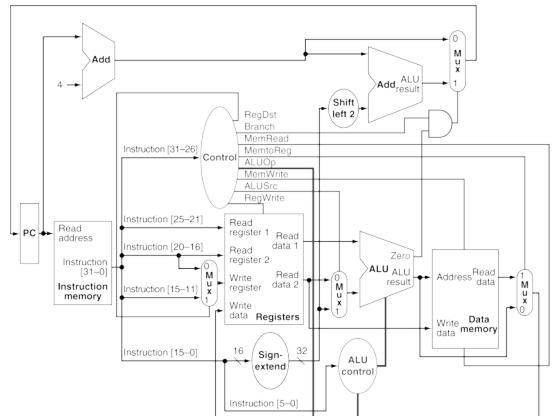


NKK-IT3030-CA2021.2

CH5-Bộ xử lý

56

### Datapath và Control Unit



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

57

### Các tín hiệu điều khiển

| Tên tín hiệu | Hiệu ứng khi tín hiệu = 0                                                                                        | Hiệu ứng khi tín hiệu = 1                                                                                  |
|--------------|------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| RegDst       | Số hiệu thanh ghi đích là các bit 20:16 (rt)                                                                     | Số hiệu thanh ghi đích là các bit 15:11 (rd)                                                               |
| Branch       | Không có lệnh rẽ nhánh beq (Branch =1) & (Zero=1): rẽ nhánh xảy ra (Branch =1) & (Zero=0): rẽ nhánh không xảy ra | Có lệnh rẽ nhánh beq (Branch =1) & (Zero=1): rẽ nhánh xảy ra (Branch =1) & (Zero=0): rẽ nhánh không xảy ra |
| RegWrite     | Không làm gì cả                                                                                                  | Ghi dữ liệu trên đầu vào Write Data ở tập thanh ghi đến thanh ghi đích                                     |
| ALUSrc       | Toán hạng thứ hai của ALU lấy từ thanh ghi nguồn thứ hai (Read data 2)                                           | Toán hạng thứ hai của ALU là giá trị 16 bit thấp của lệnh (bits 15:0) được mở rộng có dấu thành 32-bit     |
| PCSrc        | PC $\leftarrow$ PC+4                                                                                             | PC $\leftarrow$ địa chỉ đích                                                                               |

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

58

### Các tín hiệu điều khiển (tiếp)

| Tên tín hiệu | Hiệu ứng khi tín hiệu = 0                                                  | Hiệu ứng khi tín hiệu = 1                                                                                         |
|--------------|----------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| MemRead      | Không làm gì cả                                                            | Nội dung ngắn nhớ dữ liệu, được xác định bởi địa chỉ do ALU tính, được đưa ra đầu ra Read data của bộ nhớ dữ liệu |
| MemWrite     | Không làm gì cả                                                            | Dữ liệu trên đầu vào Write Data của bộ nhớ dữ liệu được ghi vào ngăn nhớ có địa chỉ do ALU tính                   |
| MemtoReg     | Giá trị được đưa đến đầu vào Write data của tập thanh ghi là từ ALU result | Giá trị được đưa đến đầu vào Write data của tập thanh ghi là từ bộ nhớ dữ liệu                                    |

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

59

### Thực hiện lệnh Jump



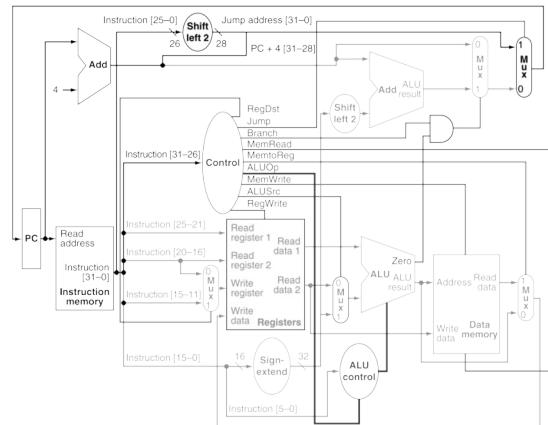
- Bits 31:26 là mã thao tác = 000010
- Bits 25:0: phần địa chỉ
- PC nhận giá trị sau:
  - Địa chỉ đích = PC31...28 : (address << 2)
    - 4 bit bên trái là của PC cũ
    - 26-bit của lệnh jump (bits 25:0)
    - 2 bit cuối là 00
  - Cần thêm tín hiệu điều khiển được giải mã từ opcode

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

60

### Datapath thêm cho lệnh jump



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

61

### 5.3. Kỹ thuật đường ống lệnh và song song mức lệnh

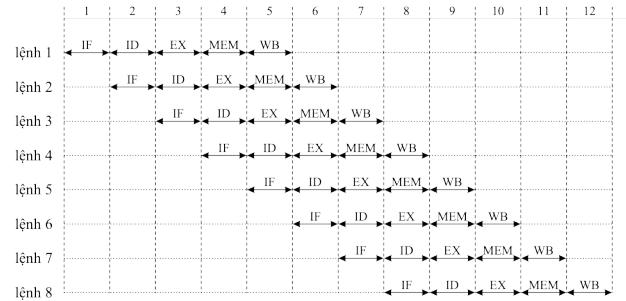
- Kỹ thuật đường ống lệnh (Instruction Pipelining):** Chia chương trình lệnh thành các công đoạn và cho phép thực hiện gối lên nhau (như dây chuyền lắp ráp)
- Bộ xử lý MIPS có 5 công đoạn:**
  - IF: Instruction fetch from memory – Nhận lệnh từ bộ nhớ
  - ID: Instruction decode & register read – Giải mã lệnh và đọc thanh ghi
  - EX: Execute operation or calculate address – Thực hiện thao tác hoặc tính toán địa chỉ
  - MEM: Access memory operand – Truy nhập operand bộ nhớ
  - WB: Write result back to register – Ghi kết quả trả về thanh ghi

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

62

### Biểu đồ thời gian của đường ống lệnh



Thời gian thực hiện 1 công đoạn = T

Thời gian thực hiện tuần tự 8 lệnh:  $8 \times 5T = 40T$ Thời gian thực hiện đường ống 8 lệnh:  $(1 \times 5T) + [(8-1) \times T] = 12T$ 

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

63

### Các mối trở ngại (Hazard) của đường ống lệnh

- Hazard:** Tình huống ngăn cản bắt đầu của lệnh tiếp theo ở chu kỳ tiếp theo
  - Hazard cấu trúc: do tài nguyên được yêu cầu đang bận
  - Hazard dữ liệu: cần phải đợi để lệnh trước hoàn thành việc đọc/ghi dữ liệu
  - Hazard điều khiển: do rẽ nhánh gây ra

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

64

### Hazard cấu trúc

- Xung đột khi sử dụng tài nguyên
- Trong đường ống của MIPS với một bộ nhớ dùng chung
  - Lệnh Load/store yêu cầu truy cập dữ liệu
  - Nhận lệnh cần trì hoãn cho chu kỳ đó
- Bởi vậy, datapath kiểu đường ống yêu cầu bộ nhớ lệnh và bộ nhớ dữ liệu tách rời (hoặc cache lệnh/cache dữ liệu tách rời)

NKK-IT3030-CA2021.2

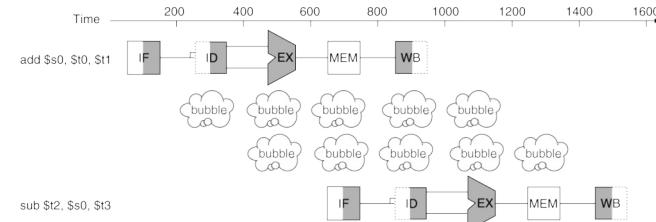
CH5-Bộ xử lý

65

### Hazard dữ liệu

- Lệnh phụ thuộc vào việc hoàn thành truy cập dữ liệu của lệnh trước đó

```
add $s0, $t0, $t1 # $s0 = $t0+$t1
sub $t2, $s0, $t3 # $t2 = $s0-$t3
```



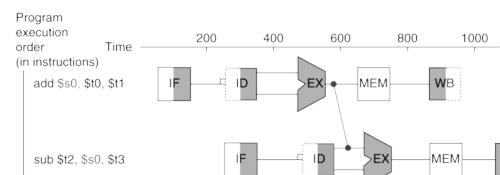
NKK-IT3030-CA2021.2

CH5-Bộ xử lý

66

### Forwarding (gửi vượt trước)

- Sử dụng kết quả ngay sau khi nó được tính
  - Không đợi đến khi kết quả được lưu đến thanh ghi
  - Yêu cầu có đường kết nối thêm trong datapath



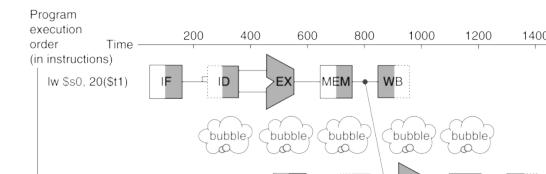
NKK-IT3030-CA2021.2

CH5-Bộ xử lý

67

### Hazard dữ liệu với lệnh load

- Không phải luôn luôn có thể tránh trì hoãn bằng cách forwarding
  - Nếu giá trị chưa được tính khi cần thiết
  - Không thể chuyển ngược thời gian
  - Cần chèn bước trì hoãn (stall hay bubble)



NKK-IT3030-CA2021.2

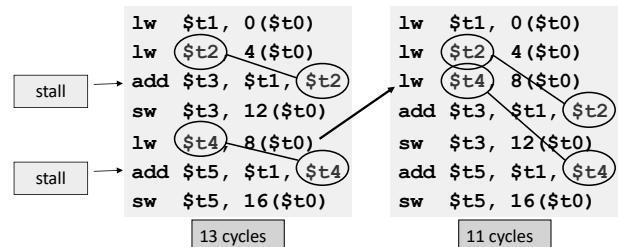
CH5-Bộ xử lý

68

### Lập lịch mã để tránh trì hoãn

- Thay đổi trình tự mã để tránh sử dụng kết quả load ở lệnh tiếp theo
- Mã C:

$$a = b + e; \quad c = b + f;$$



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

69

### Hazard điều khiển

- Rẽ nhánh xác định luồng điều khiển
  - Nhận lệnh tiếp theo phụ thuộc vào kết quả rẽ nhánh
  - Đường ống không thể luôn nhận đúng lệnh
    - Vẫn đang làm ở công đoạn giải mã lệnh (ID) của lệnh rẽ nhánh
- Với đường ống của MIPS
  - Cần so sánh thanh ghi và tính địa chỉ đích sớm trong đường ống
  - Thêm phần cứng để thực hiện việc đó trong công đoạn ID

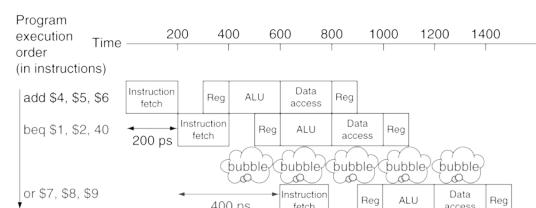
NKK-IT3030-CA2021.2

CH5-Bộ xử lý

70

### Trì hoãn khi rẽ nhánh

- Đợi cho đến khi kết quả rẽ nhánh đã được xác định trước khi nhận lệnh tiếp theo



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

71

### Dự đoán rẽ nhánh

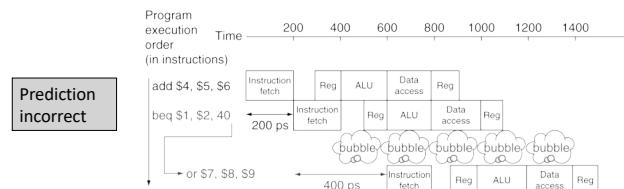
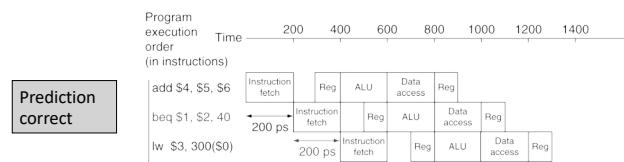
- Những đường ống dài hơn không thể sớm xác định dễ dàng kết quả rẽ nhánh
  - Cách trì hoãn không đáp ứng được
- Dự đoán kết quả rẽ nhánh
  - Chỉ trì hoãn khi dự đoán là sai
- Với MIPS
  - Có thể dự đoán rẽ nhánh không xảy ra
  - Nhận lệnh ngay sau lệnh rẽ nhánh (không làm trễ)

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

72

## MIPS với dự đoán rẽ nhánh không xảy ra



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

73

## Đặc điểm của đường ống

- Kỹ thuật đường ống cải thiện hiệu năng bằng cách tăng số lệnh thực hiện
  - Thực hiện nhiều lệnh đồng thời
  - Mỗi lệnh có cùng thời gian thực hiện
- Các dạng hazard:
  - Cấu trúc, dữ liệu, điều khiển
- Thiết kế tập lệnh ảnh hưởng đến độ phức tạp của việc thực hiện đường ống

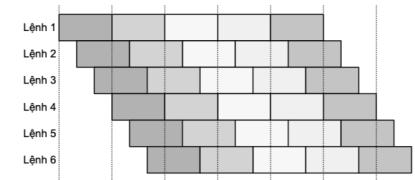
NKK-IT3030-CA2021.2

CH5-Bộ xử lý

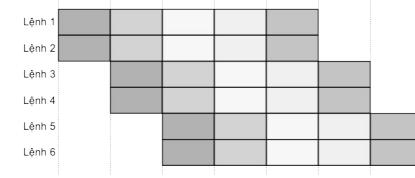
74

## Tăng cường khả năng song song mức lệnh

### ▪ Tăng số công đoạn của đường ống



### ▪ Siêu vô hướng (Superscalar)



NKK-IT3030-CA2021.2

CH5-Bộ xử lý

75

## Kiến trúc máy tính

Hết chương 5

NKK-IT3030-CA2021.2

CH5-Bộ xử lý

76



## Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

2

## Kiến trúc máy tính

### Chương 6 BỘ NHỚ MÁY TÍNH

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

3

## Nội dung

- 6.1. Tổng quan hệ thống nhớ
- 6.2. Bộ nhớ chính
- 6.3. Bộ nhớ đệm (cache)
- 6.4. Bộ nhớ ngoài
- 6.5. Bộ nhớ ảo

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

4

## 6.1. Tổng quan hệ thống nhớ

### 1. Các đặc trưng của bộ nhớ

#### ▪ Vị trí

- Bên trong CPU:
  - tập thanh ghi
- Bộ nhớ trong:
  - bộ nhớ chính
  - bộ nhớ đệm (cache)
- Bộ nhớ ngoài:
  - các thiết bị lưu trữ

#### ▪ Dung lượng

- Độ dài từ nhớ (tính bằng bit)
- Số lượng từ nhớ

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

5

## Các đặc trưng của bộ nhớ (tiếp)

### ▪ Đơn vị truyền

- Từ nhớ
- Khối nhớ

### ▪ Phương pháp truy nhập

- Truy nhập tuần tự (băng từ)
- Truy nhập trực tiếp (các loại đĩa)
- Truy nhập ngẫu nhiên (bộ nhớ bán dẫn)
- Truy nhập liên kết (cache)

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

6

## Các đặc trưng của bộ nhớ (tiếp)

### ▪ Hiệu năng (performance)

- Thời gian truy nhập
- Chu kỳ nhớ
- Tốc độ truyền

### ▪ Kiểu vật lý

- Bộ nhớ bán dẫn
- Bộ nhớ từ
- Bộ nhớ quang

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

7

## Các đặc trưng của bộ nhớ (tiếp)

### ▪ Các đặc tính vật lý

- Khả biến / Không khả biến (volatile / nonvolatile)
- Xoá được / không xoá được

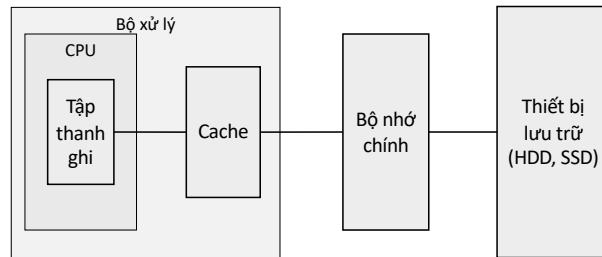
### ▪ Tổ chức

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

8

## 2. Phân cấp bộ nhớ



Từ trái sang phải:

- dung lượng tăng dần
- tốc độ giảm dần
- giá thành cùng dung lượng giảm dần

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

9

## Công nghệ bộ nhớ

| Công nghệ bộ nhớ | Thời gian truy nhập | Giá thành/GiB (2012) |
|------------------|---------------------|----------------------|
| SRAM             | 0,5 – 2,5 ns        | \$500 – \$1000       |
| DRAM             | 50 – 70 ns          | \$10 – \$20          |
| Flash memory     | 5000 – 50 000 ns    | \$0,75 – \$1         |
| HDD              | 5 – 20 ms           | \$0,05 – \$0,1       |

### Bộ nhớ lý tưởng

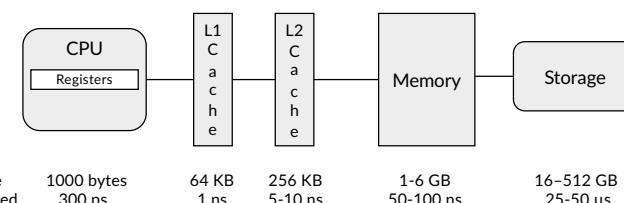
- Thời gian truy nhập như SRAM
- Dung lượng và giá thành như ổ đĩa cứng

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

10

## Phân cấp bộ nhớ cho thiết bị di động



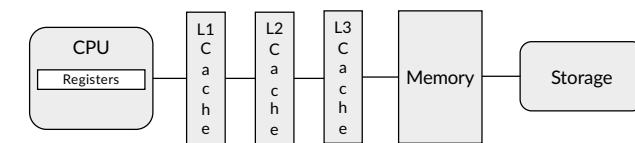
| Size       | Speed  | Size  | Speed | Size   | Speed   | Size   | Speed     |           |          |
|------------|--------|-------|-------|--------|---------|--------|-----------|-----------|----------|
| 1000 bytes | 300 ps | 64 KB | 1 ns  | 256 KB | 5-10 ns | 1-6 GB | 50-100 ns | 16-512 GB | 25-50 μs |

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

11

## Phân cấp bộ nhớ cho máy tính PC



| Laptop | Size       | Speed  | Size  | Speed | Size   | Speed   | Size   | Speed    | Size    | Speed     |
|--------|------------|--------|-------|-------|--------|---------|--------|----------|---------|-----------|
|        | 1000 bytes | 300 ps | 64 KB | 1 ns  | 256 KB | 3-10 ns | 4-8 MB | 10-20 ns | 4-16 GB | 50-100 ns |

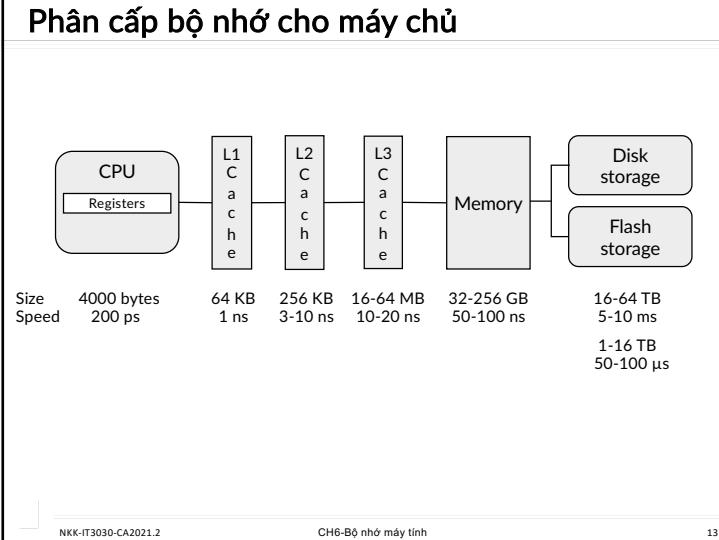
| Desktop | Size       | Speed  | Size  | Speed | Size   | Speed   | Size    | Speed    | Size    | Speed     |
|---------|------------|--------|-------|-------|--------|---------|---------|----------|---------|-----------|
|         | 2000 bytes | 300 ps | 64 KB | 1 ns  | 256 KB | 3-10 ns | 8-32 MB | 10-20 ns | 8-64 GB | 50-100 ns |

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

12

### Phân cấp bộ nhớ cho máy chủ



### Nguyên lý cục bộ hóa tham chiếu bộ nhớ

- Trong một khoảng thời gian đủ nhỏ CPU thường chỉ tham chiếu các thông tin trong một khối nhớ cục bộ
- Ví dụ:
  - Cấu trúc chương trình tuần tự
  - Vòng lặp có thân nhỏ
  - Cấu trúc dữ liệu mảng

NKK-IT3030-CA2021.2 CH6-Bộ nhớ máy tính 14

## 6.2. Bộ nhớ chính

### 1. Bộ nhớ bán dẫn

| Kiểu bộ nhớ                         | Tiêu chuẩn             | Khả năng xoá              | Cơ chế ghi | Tính khả biến  |
|-------------------------------------|------------------------|---------------------------|------------|----------------|
| Read Only Memory (ROM)              | Bộ nhớ chỉ đọc         | Không xoá được            | Mặt nạ     |                |
| Programmable ROM (PROM)             |                        |                           |            |                |
| Erasable PROM (EPROM)               | Bộ nhớ hầu như chỉ đọc | băng tia cực tím, cả chip |            |                |
| Electrically Erasable PROM (EEPROM) |                        | băng điện, mức từng byte  | Băng điện  | Không khả biến |
| Flash memory                        | Bộ nhớ đọc-ghi         | băng điện, từng khối      |            |                |
| Random Access Memory (RAM)          |                        | băng điện, mức từng byte  | Băng điện  | Khả biến       |

NKK-IT3030-CA2021.2 CH6-Bộ nhớ máy tính 15

### ROM (Read Only Memory)

- Bộ nhớ không khả biến
- Lưu trữ các thông tin sau:
  - Thư viện các chương trình con
  - Các chương trình điều khiển hệ thống (BIOS)
  - Các bảng chức năng
  - Vì chương trình

NKK-IT3030-CA2021.2 CH6-Bộ nhớ máy tính 16

### Các kiểu ROM

- ROM mặt nạ:
  - thông tin được ghi khi sản xuất
- PROM (Programmable ROM)
  - Cần thiết bị chuyên dụng để ghi
  - Chỉ ghi được một lần
- EPROM (Erasable PROM)
  - Cần thiết bị chuyên dụng để ghi
  - Xóa được bằng tia tử ngoại
  - Ghi lại được nhiều lần
- EEPROM (Electrically Erasable PROM)
  - Có thể ghi theo từng byte
  - Xóa bằng điện

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

17

### Bộ nhớ Flash

- Ghi theo khối
- Xóa bằng điện
- Dung lượng lớn

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

18

### RAM (Random Access Memory)

- Bộ nhớ đọc-ghi (Read/Write Memory)
- Khả biến
- Lưu trữ thông tin tạm thời
- Có hai loại: SRAM và DRAM  
(Static and Dynamic)

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

19

### SRAM (Static) - RAM tĩnh

- Các bit được lưu trữ bằng các Flip-Flop  
→ thông tin ổn định
- Cấu trúc phức tạp
- Dung lượng chip nhỏ
- Tốc độ nhanh
- Đắt tiền
- Dùng làm bộ nhớ cache

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

20

### DRAM (Dynamic) – RAM động

- Các bit được lưu trữ trên tụ điện  
→ cần phải có mạch làm tươi
- Cấu trúc đơn giản
- Dung lượng lớn
- Tốc độ chậm hơn
- Rẻ tiền hơn
- Dùng làm bộ nhớ chính

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

21

### Một số DRAM tiên tiến thông dụng

- Cải tiến để tăng tốc độ
- Synchronous DRAM (SDRAM): làm việc được đồng bộ bởi xung clock
- DDR-SDRAM (Double Data Rate SDRAM)
- DDR3, DDR4

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

22

### 2. Các đặc trưng cơ bản của bộ nhớ chính

- Chứa các chương trình đang thực hiện và các dữ liệu đang được sử dụng
- Tồn tại trên mọi hệ thống máy tính
- Bao gồm các ngăn nhớ được đánh địa chỉ trực tiếp bởi CPU
- Dung lượng của bộ nhớ chính nhỏ hơn không gian địa chỉ bộ nhớ mà CPU quản lý.
- Việc quản lý logic bộ nhớ chính tuỳ thuộc vào hệ điều hành

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

23

### Tổ chức bộ nhớ đan xen (interleaved memory)

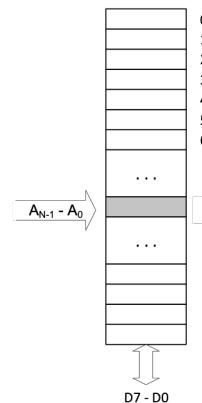
- Độ rộng của bus dữ liệu để trao đổi với bộ nhớ:  $m = 8, 16, 32, 64, 128 \dots$  bit
- Các ngăn nhớ được tổ chức theo byte  
→ tổ chức bộ nhớ vật lý khác nhau

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

24

$m=8\text{bit} \rightarrow$  một băng nhớ tuyến tính

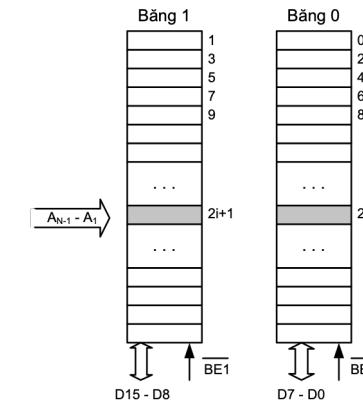


NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

25

$m = 16\text{bit} \rightarrow$  hai băng nhớ đan xen

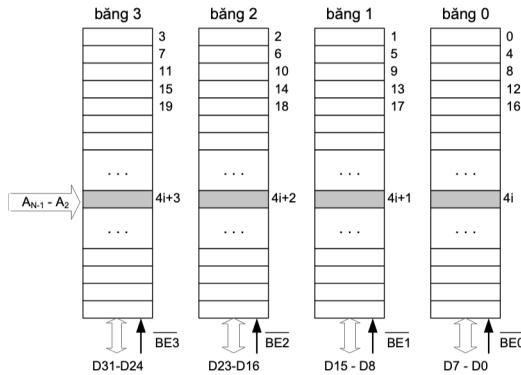


NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

26

$m = 32\text{bit} \rightarrow$  bốn băng nhớ đan xen

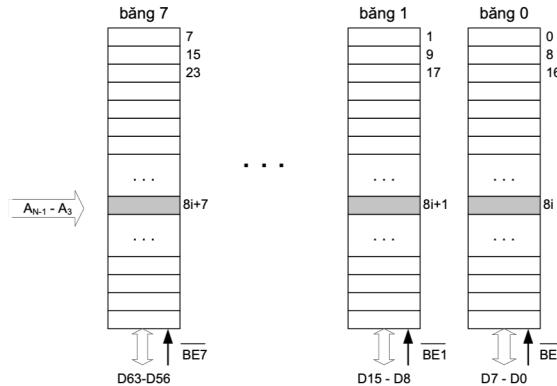


NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

27

$m = 64\text{bit} \rightarrow$  tám băng nhớ đan xen



NKK-IT3030-CA2021.2

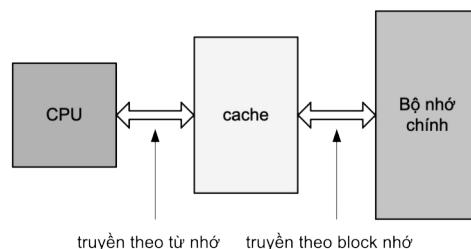
CH6-Bộ nhớ máy tính

28

### 6.3. Bộ nhớ đệm (cache memory)

#### 1. Nguyên tắc chung của cache

- Cache có tốc độ nhanh hơn bộ nhớ chính
- Cache được đặt giữa CPU và bộ nhớ chính nhằm tăng tốc độ CPU truy cập bộ nhớ
- Cache có thể được đặt trên chip CPU



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

29

#### Ví dụ về thao tác của cache

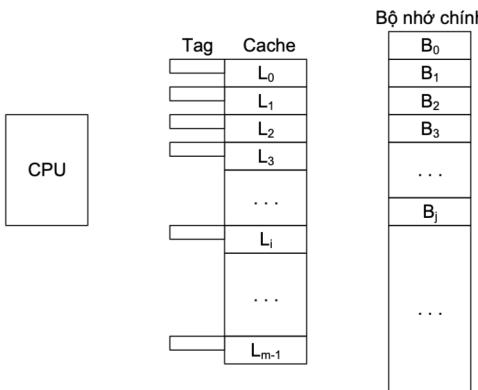
- CPU yêu cầu nội dung của ngăn nhớ
- CPU kiểm tra trên cache với dữ liệu này
- Nếu có, CPU nhận dữ liệu từ cache (nhanh)
- Nếu không có, đọc Block nhớ chứa dữ liệu từ bộ nhớ chính vào cache
- Tiếp đó chuyển dữ liệu từ cache vào CPU

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

30

#### Cấu trúc chung của cache / bộ nhớ chính



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

31

#### Cấu trúc chung của cache / bộ nhớ chính (tiếp)

- Bộ nhớ chính có  $2^N$  byte nhớ
- Bộ nhớ chính và cache được chia thành các khối có kích thước bằng nhau
  - Bộ nhớ chính:  $B_0, B_1, B_2, \dots, B_{p-1}$  ( $p$  Blocks)
  - Bộ nhớ cache:  $L_0, L_1, L_2, \dots, L_{m-1}$  ( $m$  Lines)
  - Kích thước của Block (Line) = 8,16,32,64,128 byte
- Mỗi Line trong cache có một thẻ nhớ (Tag) được gắn vào

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

32

### Cấu trúc chung của cache / bộ nhớ chính (tiếp)

- Một số Block của bộ nhớ chính được nạp vào các Line của cache
- Nội dung Tag (thẻ nhớ) cho biết Block nào của bộ nhớ chính hiện đang được chứa ở Line đó
- Nội dung Tag được cập nhật mỗi khi Block từ bộ nhớ chính nạp vào Line đó
- Khi CPU truy nhập (đọc/ghi) một từ nhớ, có hai khả năng xảy ra:
  - Từ nhớ đó có trong cache (cache hit)
  - Từ nhớ đó không có trong cache (cache miss).

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

33

### 2. Các phương pháp ánh xạ

(Chính là các phương pháp tổ chức bộ nhớ cache)

- Ánh xạ trực tiếp  
(Direct mapping)
- Ánh xạ liên kết toàn phần  
(Fully associative mapping)
- Ánh xạ liên kết tập hợp  
(Set associative mapping)

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

34

### Ánh xạ trực tiếp

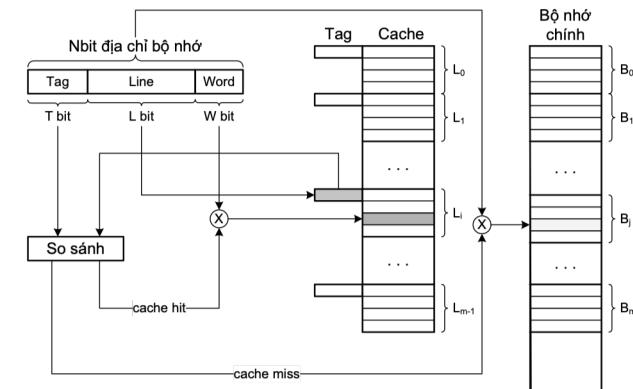
- Mỗi Block của bộ nhớ chính chỉ có thể được nạp vào một Line của cache:
  - $B_0 \rightarrow L_0$
  - $B_1 \rightarrow L_1$
  - ....
  - $B_{m-1} \rightarrow L_{m-1}$
  - $B_m \rightarrow L_0$
  - $B_{m+1} \rightarrow L_1$
  - ....
- **Tổng quát**
  - $B_j$  chỉ có thể nạp vào  $L_{j \bmod m}$
  - $m$  là số Line của cache.

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

35

### Ánh xạ trực tiếp (tiếp)



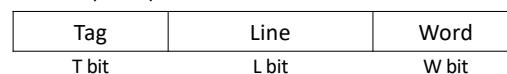
NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

36

### Ánh xạ trực tiếp (tiếp)

- Địa chỉ N bit của bộ nhớ chính chia thành ba trường:
  - Trường Word gồm W bit xác định một từ nhớ trong Block hay Line:  $2^W$  = kích thước của Block hay Line
  - Trường Line gồm L bit xác định một trong số các Line trong cache:  $2^L$  = số Line trong cache = m
  - Trường Tag gồm T bit:  $T = N - (W+L)$



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

37

### Ánh xạ trực tiếp (tiếp)

- Mỗi thẻ nhớ (Tag) của một Line chứa được T bit
- Khi Block từ bộ nhớ chính được nạp vào Line của cache thì Tag ở đó được cập nhật giá trị là T bit địa chỉ bên trái của Block đó
- Khi CPU muốn truy nhập một từ nhớ thì nó phát ra một địa chỉ N bit cụ thể
  - Nhờ vào giá trị L bit của trường Line sẽ tìm ra Line tương ứng
  - Đọc nội dung Tag ở Line đó (T bit), rồi so sánh với T bit bên trái của địa chỉ vừa phát ra
    - Giống nhau: cache hit
    - Khác nhau: cache miss
- Ưu điểm:** Bộ so sánh đơn giản
- Nhược điểm:** Xác suất cache hit thấp

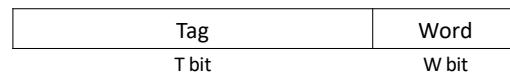
NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

38

### Ánh xạ liên kết toàn phần

- Mỗi Block có thể nạp vào bất kỳ Line nào của cache
- Địa chỉ của bộ nhớ chính chia thành hai trường:
  - Trường Word gồm W bit xác định một từ nhớ trong Block hay Line:  $2^W$  = kích thước của Block hay Line
  - Trường Tag gồm T bit, dùng để xác định Block của bộ nhớ chính:  $T = N - W$

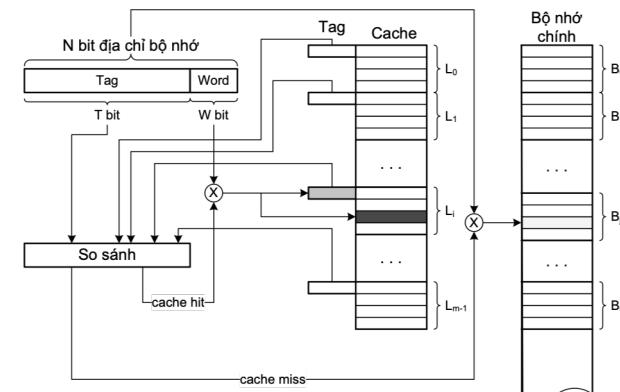


NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

39

### Ánh xạ liên kết toàn phần (tiếp)



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

40

### Ánh xạ liên kết toàn phần (tiếp)

- Mỗi thẻ nhớ (Tag) của một Line chứa được T bit
- Khi Block từ bộ nhớ chính được nạp vào Line của cache thì Tag ở đó được cập nhật giá trị là T bit địa chỉ bên trái của Block đó
- Khi CPU muốn truy nhập một từ nhớ thì nó phát ra một địa chỉ N bit cụ thể
  - So sánh T bit bên trái của địa chỉ vừa phát ra với lần lượt nội dung của các Tag trong cache
    - Nếu gặp giá trị bằng nhau: cache hit xảy ra ở Line đó
    - Nếu không có giá trị nào bằng: cache miss
- **Ưu điểm:** Xác suất cache hit cao
- **Nhược điểm:**
  - So sánh đồng thời với tất cả các Tag → mất nhiều thời gian
  - Bộ so sánh phức tạp
- Ít sử dụng

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

41

### Ánh xạ liên kết tập hợp

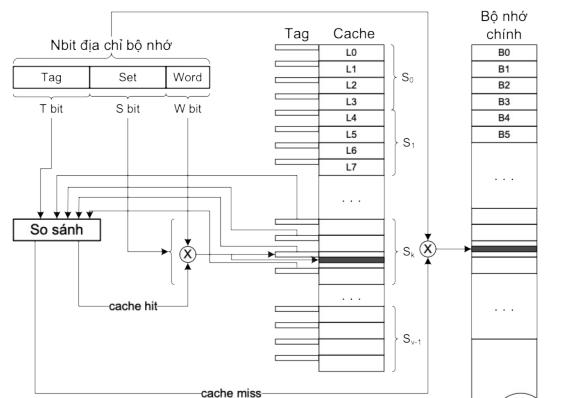
- Dung hòa cho hai phương pháp trên
- Cache được chia thành các Tập (Set)
- Mỗi một Set chứa một số Line
- **Ví dụ:**
  - 4 Line/Set → 4-way associative mapping
- **Ánh xạ theo nguyên tắc sau:**
  - $B_0 \rightarrow S_0$
  - $B_1 \rightarrow S_1$
  - $B_2 \rightarrow S_2$
  - .....

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

42

### Ánh xạ liên kết tập hợp (tiếp)



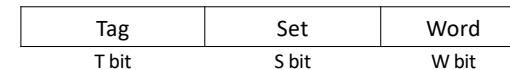
NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

43

### Ánh xạ liên kết tập hợp (tiếp)

- Địa chỉ N bit của bộ nhớ chính chia thành ba trường:
  - Trường Word gồm W bit, xác định một từ nhớ trong Block hay Line:  
 $2^W$  = kích thước của Block hay Line
  - Trường Set gồm S bit, xác định một trong số các Set trong cache:  
 $2^S$  = số Set trong cache
  - Trường Tag gồm T bit:  
 $T = N - (S+W)$



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

44

### Ánh xạ liên kết tập hợp (tiếp)

- Khi CPU muốn truy nhập một từ nhớ thì nó phát ra một địa chỉ  $N$  bit cụ thể
  - Nhờ vào giá trị  $S$  bit của trường Set sẽ tìm ra Set tương ứng
  - So sánh  $T$  bit bên trái của địa chỉ vừa phát ra với lần lượt nội dung của các Tag trong Set đó
    - Nếu gặp giá trị bằng nhau: cache hit xảy ra ở Line tương ứng
    - Nếu không có giá trị nào bằng: cache miss
- **Tổng quát cho cả hai phương pháp trên**
- Thông dụng với: 2,4,8,16 Lines/Set

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

45

### Ví dụ về ánh xạ địa chỉ

- Giả sử máy tính đánh địa chỉ cho từng byte
- Không gian địa chỉ bộ nhớ chính = 4GiB
- Dung lượng bộ nhớ cache là 256KiB
- Kích thước Line (Block) = 32byte.
- Xác định số bit của các trường địa chỉ cho ba trường hợp tổ chức:
  - Ánh xạ trực tiếp
  - Ánh xạ liên kết toàn phần
  - Ánh xạ liên kết tập hợp 4 đường

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

46

### Với ánh xạ trực tiếp

- Bộ nhớ chính = 4GiB =  $2^{32}$  byte  $\rightarrow$  Số bit địa chỉ của bộ nhớ chính là:  $N = 32$  bit
- Cache = 256 KiB =  $2^{18}$  byte
- Kích thước Line = 32 byte =  $2^5$  byte  $\rightarrow$  số bit địa chỉ của trường Word là:  $W = 5$  bit
- Số Line trong cache =  $2^{18} / 2^5 = 2^{13}$  Line  $\rightarrow$  số bit địa chỉ trường Line là:  $L = 13$  bit
- Số bit địa chỉ của trường Tag là:  

$$T = 32 - (13 + 5) = 14 \text{ bit}$$

|        |        |       |
|--------|--------|-------|
| Tag    | Line   | Word  |
| 14 bit | 13 bit | 5 bit |

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

47

### Với ánh xạ liên kết toàn phần

- Bộ nhớ chính = 4GiB =  $2^{32}$  byte  $\rightarrow$  số bit địa chỉ của bộ nhớ chính là:  $N = 32$  bit
- Kích thước Line = 32 byte =  $2^5$  byte  $\rightarrow$  số bit địa chỉ của trường Word là:  $W = 5$  bit
- Số bit địa chỉ của trường Tag là:  

$$T = 32 - 5 = 27 \text{ bit}$$

|        |       |
|--------|-------|
| Tag    | Word  |
| 27 bit | 5 bit |

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

48

### Với ánh xạ liên kết tập hợp 4 đường

- Bộ nhớ chính =  $4\text{GiB} = 2^{32}$  byte → số bit địa chỉ của bộ nhớ chính là:  $N = 32$  bit
- Cache =  $256\text{ KiB} = 2^{18}$  byte
- Kích thước Line =  $32\text{ byte} = 2^5$  byte → số bit địa chỉ của trường Word là:  $W = 5$  bit
- Số Line trong cache =  $2^{18}/2^5 = 2^{13}$  Line
- Một Set có 4 Line =  $2^2$  Line
- số Set trong cache =  $2^{13}/2^2 = 2^{11}$  Set
- số bit địa chỉ của trường Set là:  $S = 11$  bit
- Số bit địa chỉ của trường Tag là:

$$T = 32 - (11 + 5) = 16 \text{ bit}$$

| Tag    | Set    | Word  |
|--------|--------|-------|
| 16 bit | 11 bit | 5 bit |

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

49

### 3. Thay thế block trong cache

Với ánh xạ trực tiếp:

- Không phải lựa chọn
- Mỗi Block chỉ ánh xạ vào một Line xác định
- Thay thế Block ở Line đó

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

50

### Thay thế block trong cache (tiếp)

- Với ánh xạ liên kết: cần có thuật giải thay thế:
- Random: Thay thế ngẫu nhiên
  - FIFO (First In First Out): Thay thế Block nào nằm lâu nhất ở trong Set đó
  - LFU (Least Frequently Used): Thay thế Block nào trong Set có số lần truy nhập ít nhất trong cùng một khoảng thời gian
  - LRU (Least Recently Used): Thay thế Block ở trong Set tương ứng có thời gian lâu nhất không được tham chiếu tới
  - Tối ưu nhất: LRU

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

51

### 4. Phương pháp ghi dữ liệu khi cache hit

- Ghi xuyên qua (Write-through):

- ghi cả cache và cả bộ nhớ chính
- tốc độ chậm

- Ghi trả sau (Write-back):

- chỉ ghi ra cache
- tốc độ nhanh
- khi Block trong cache bị thay thế cần phải ghi trả cả Block về bộ nhớ chính

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

52

#### 6.4. Bộ nhớ ngoài

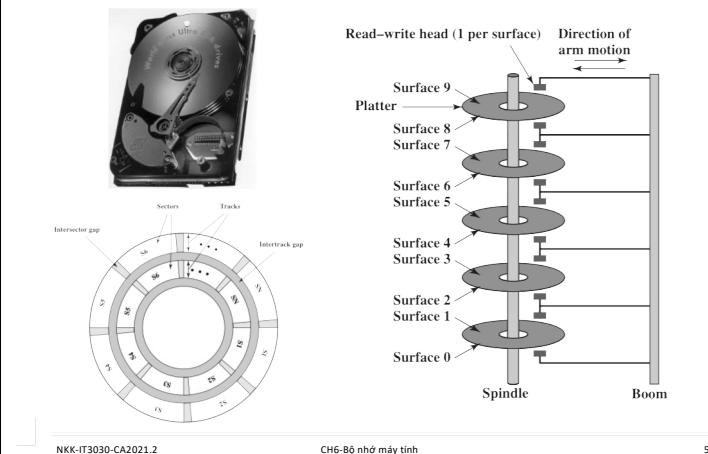
- Tồn tại dưới dạng các thiết bị lưu trữ
- Các kiểu bộ nhớ ngoài
  - Băng từ: ít sử dụng
  - Đĩa từ: Ổ đĩa cứng HDD (Hard Disk Drive)
  - Đĩa quang: CD, DVD
  - Bộ nhớ Flash:
    - Ổ nhớ thẻ rắn SSD (Solid State Drive)
    - USB flash
    - Thẻ nhớ

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

53

#### Ổ đĩa cứng (HDD – Hard Disk Drive)



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

54

#### HDD

- Dung lượng lớn
- Tốc độ đọc/ghi chậm
- Tốn năng lượng
- Dễ bị lỗi cơ học
- Rẻ tiền

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

55

#### Ổ SSD (Solid State Drive)

- Bộ nhớ bán dẫn flash
- Không khả biến
- Tốc độ nhanh
- Tiêu thụ năng lượng ít
- Gồm nhiều chip nhớ flash và cho phép truy cập song song
- Ít bị lỗi
- Đắt tiền



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

56

## Đĩa quang

- CD (Compact Disc)
  - Dung lượng thông dụng 650MB
- DVD
  - Digital Video Disc hoặc Digital Versatile Disk
  - Ghi một hoặc hai mặt
  - Một hoặc hai lớp trên một mặt
  - Thông dụng: 4,7GB/lớp

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

57

## Hệ thống lưu trữ dung lượng lớn: RAID

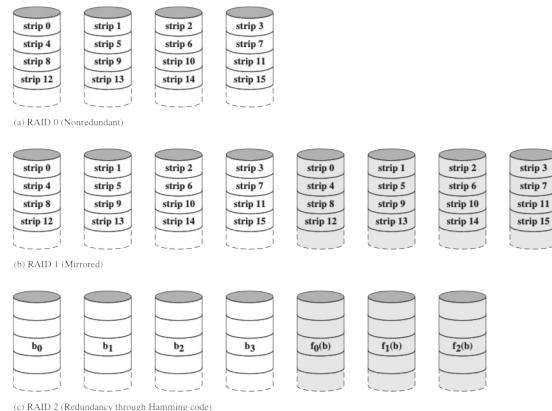
- Redundant Array of Inexpensive Disks
- (Redundant Array of Independent Disks)
- Tập các ổ đĩa cứng vật lý được OS coi như một ổ logic duy nhất → dung lượng lớn
- Dữ liệu được lưu trữ phân tán trên các ổ đĩa vật lý → truy cập song song (nhanh)
- Lưu trữ thêm thông tin dư thừa, cho phép khôi phục lại thông tin trong trường hợp đĩa bị hỏng → an toàn thông tin
- Các loại RAID: (RAID 0 – 6)

NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

58

## RAID 0, 1, 2

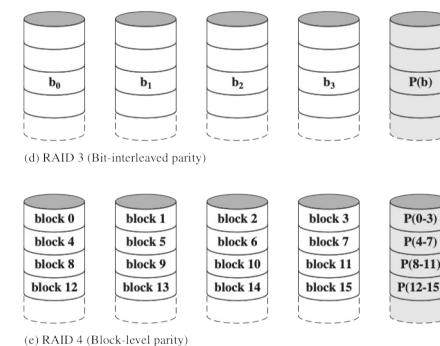


NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

59

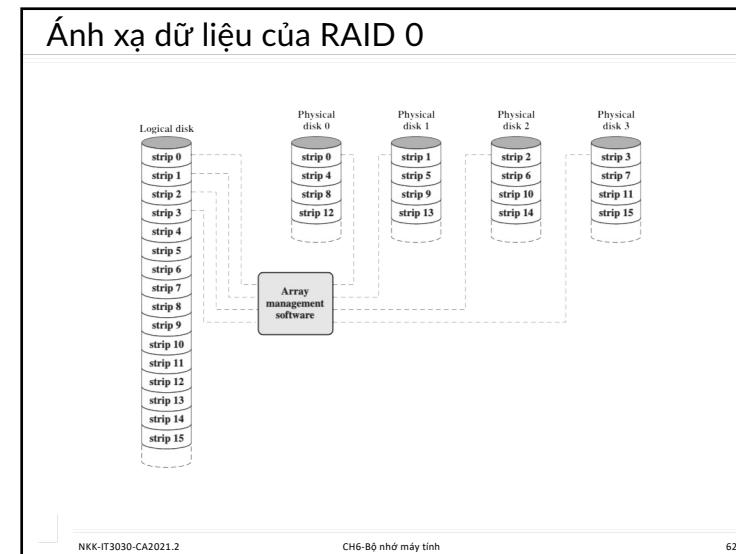
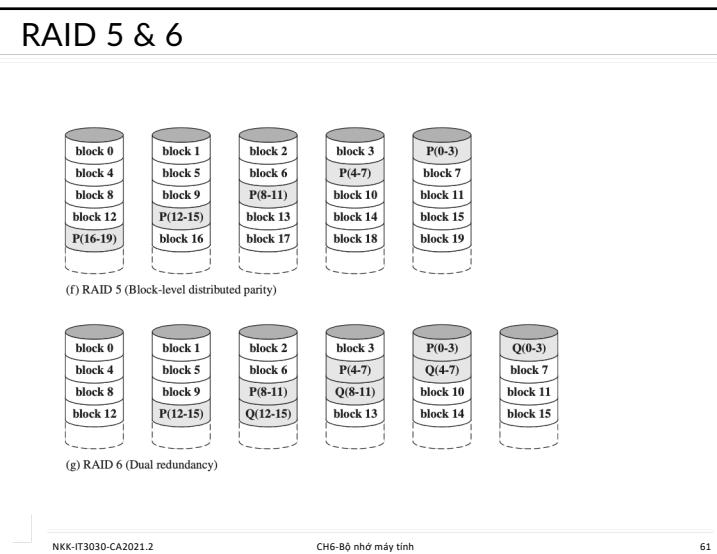
## RAID 3 & 4



NKK-IT3030-CA2021.2

CH6-Bộ nhớ máy tính

60



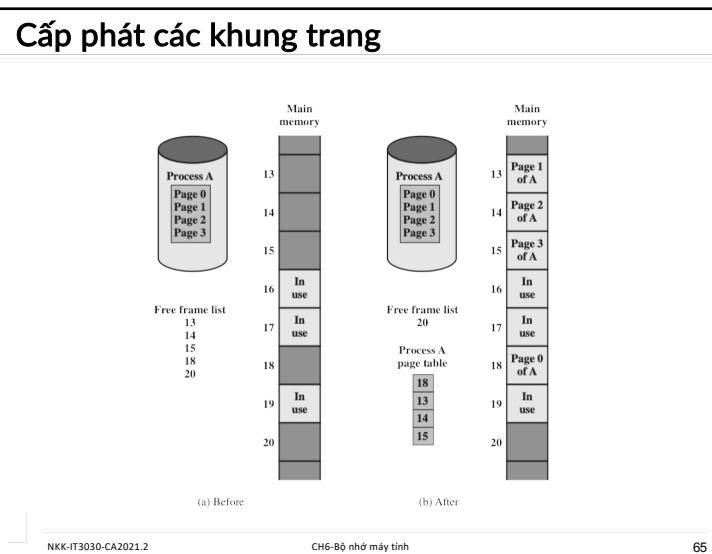
## 6.5. Bộ nhớ ảo (Virtual Memory)

- Khái niệm bộ nhớ ảo: gồm bộ nhớ chính và bộ nhớ ngoài mà được CPU coi như là một bộ nhớ duy nhất (bộ nhớ chính).
- Các kỹ thuật thực hiện bộ nhớ ảo:
  - Kỹ thuật phân trang: Chia không gian địa chỉ bộ nhớ thành các trang nhớ có kích thước bằng nhau và nằm liền kề nhau  
Thông dụng: kích thước trang = 4KiB
  - Kỹ thuật phân đoạn: Chia không gian nhớ thành các đoạn nhớ có kích thước thay đổi, các đoạn nhớ có thể gối lên nhau.

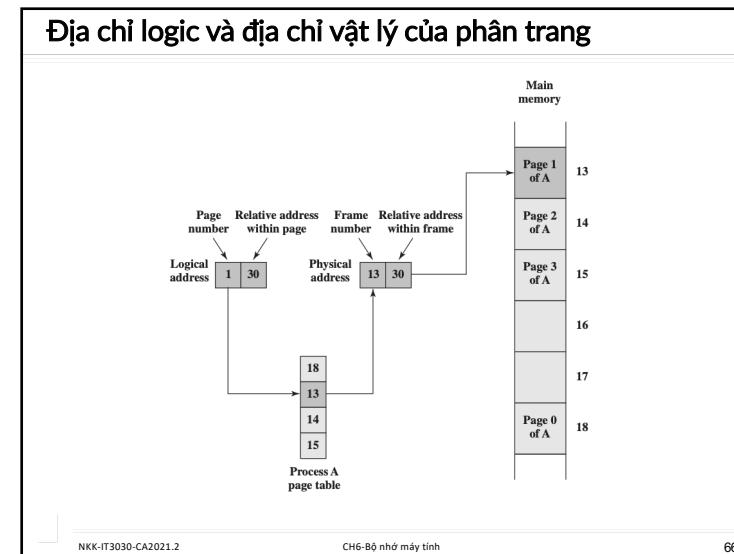
## Phân trang

- Phân chia bộ nhớ thành các phần có kích thước bằng nhau gọi là các khung trang
- Chia chương trình (tiến trình) thành các trang
- Cấp phát số hiệu khung trang yêu cầu cho tiến trình
- OS duy trì danh sách các khung trang nhớ trống
- Tiến trình không yêu cầu các khung trang liên tiếp
- Sử dụng bảng trang để quản lý

## Cấp phát các khung trang



## **Địa chỉ logic và địa chỉ vật lý của phân trang**



## **Nguyên tắc làm việc của bộ nhớ ảo phân trang**

- Phân trang theo yêu cầu
    - Không yêu cầu tất cả các trang của tiến trình nằm trong bộ nhớ
    - Chỉ nạp vào bộ nhớ những trang được yêu cầu
  - Lỗi trang
    - Trang được yêu cầu không có trong bộ nhớ
    - HĐH cần hoán đổi trang yêu cầu vào
    - Có thể cần hoán đổi một trang nào đó ra để lấy chỗ
    - Cần chọn trang để đưa ra

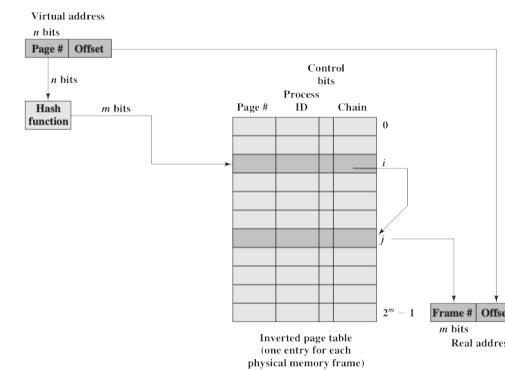
Thất bại

- Quá nhiều tiến trình trong bộ nhớ quá nhỏ
  - OS tiêu tốn toàn bộ thời gian cho việc hoán đổi
  - Có ít hoặc không có công việc nào được thực hiện
  - Đĩa luôn luôn sáng
  - Giải pháp:
    - Giảm bớt số tiến trình đang chạy
    - Thêm bộ nhớ
    - Thuật toán thay trang

### Lợi ích

- Không cần toàn bộ tiến trình nằm trong bộ nhớ để chạy
- Có thể hoán đổi trang được yêu cầu
- Như vậy có thể chạy những tiến trình lớn hơn tổng bộ nhớ sẵn dùng
- Bộ nhớ chính được gọi là bộ nhớ thực
- Người dùng cảm giác bộ nhớ lớn hơn bộ nhớ thực

### Cấu trúc bảng trang



### Kiến trúc máy tính

Hết chương 6



## Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

2

## Kiến trúc máy tính

### Chương 7 HỆ THỐNG VÀO-RA

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

3

## Nội dung

- 7.1. Tổng quan về hệ thống vào-ra
- 7.2. Các phương pháp điều khiển vào-ra
- 7.3. Nối ghép thiết bị vào-ra

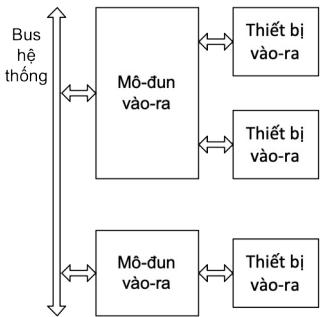
NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

4

### 7.1. Tổng quan về hệ thống vào-ra

- Chức năng: Trao đổi thông tin giữa máy tính với bên ngoài
- Các thao tác cơ bản:
  - Vào dữ liệu (Input)
  - Ra dữ liệu (Output)
- Các thành phần chính:
  - Các thiết bị vào-ra
  - Các mô-đun vào-ra



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

5

### Đặc điểm của hệ thống vào-ra

- Tồn tại đa dạng các thiết bị vào-ra khác nhau về:
    - Nguyên tắc hoạt động
    - Tốc độ
    - Khuôn dạng dữ liệu
  - Tất cả các thiết bị vào-ra đều chậm hơn CPU và RAM
- Cần có các mô-đun vào-ra để nối ghép các thiết bị với CPU và bộ nhớ chính

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

6

### Thiết bị vào-ra

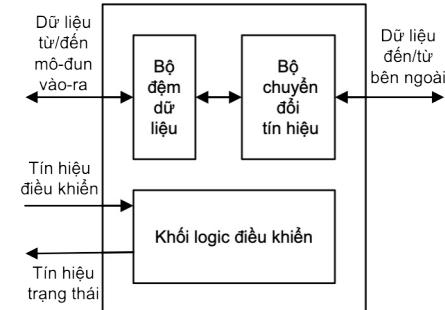
- Còn gọi là thiết bị ngoại vi (Peripherals)
- Chức năng: chuyển đổi dữ liệu giữa bên trong và bên ngoài máy tính
- Phân loại:
  - Thiết bị vào (Input Devices)
  - Thiết bị ra (Output Devices)
  - Thiết bị lưu trữ (Storage Devices)
  - Thiết bị truyền thông (Communication Devices)
- Giao tiếp:
  - Người - máy
  - Máy - máy

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

7

### Cấu trúc chung của thiết bị vào-ra



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

8

## Mô-đun vào-ra

### ▪ Chức năng:

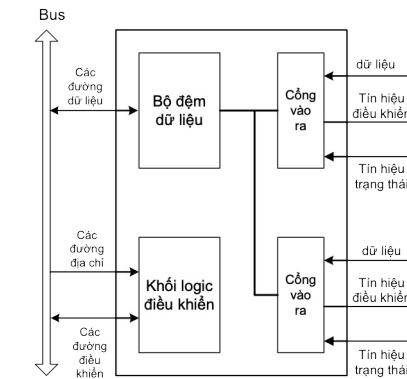
- Điều khiển và định thời
- Trao đổi thông tin với CPU hoặc bộ nhớ chính
- Trao đổi thông tin với thiết bị vào-ra
- Đem giữa bên trong máy tính với thiết bị vào-ra
- Phát hiện lỗi của thiết bị vào-ra

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

9

## Cấu trúc của mô-đun vào-ra



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

10

## 4. Địa chỉ hóa cổng vào-ra (IO addressing)

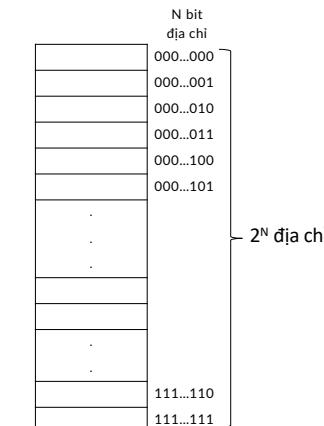
- Hầu hết các bộ xử lý chỉ có một không gian địa chỉ chung cho cả các ngăn nhớ và các cổng vào-ra → được gọi là không gian địa chỉ bộ nhớ
  - Các bộ xử lý 680x0 của Motorola
  - Các bộ xử lý theo kiến trúc RISC: MIPS, ARM, ...
- Một số bộ xử lý có hai không gian địa chỉ tách biệt:
  - Không gian địa chỉ bộ nhớ
  - Không gian địa chỉ vào-ra
  - Ví dụ: Intel x86

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

11

## Không gian địa chỉ bộ nhớ (dùng chung)

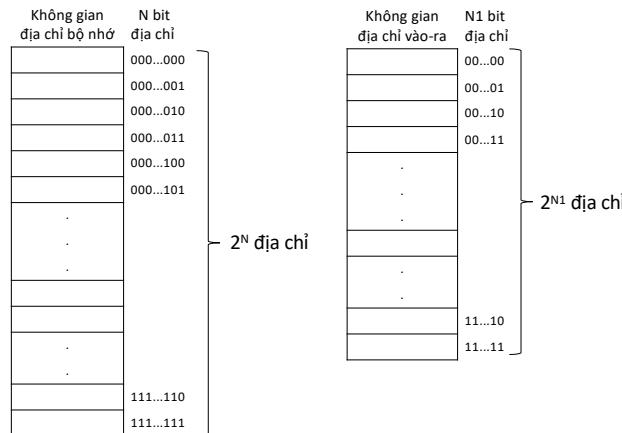


NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

12

### Không gian địa chỉ tách biệt



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

13

### Các phương pháp địa chỉ hóa cổng vào-ra

- Vào-ra theo bản đồ bộ nhớ  
(Memory mapped IO)
- Vào-ra riêng biệt  
(Isolated IO hay IO mapped IO)

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

14

### Vào-ra theo bản đồ bộ nhớ

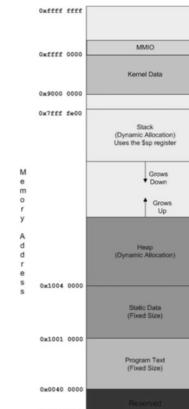
- Cổng vào-ra được đánh địa chỉ theo không gian địa chỉ bộ nhớ
- CPU coi cổng vào-ra như ngăn nhớ
- Lập trình trao đổi dữ liệu với cổng vào-ra bằng các lệnh truy nhập dữ liệu bộ nhớ
- Có thể thực hiện trên mọi hệ thống
- Ví dụ: Bộ xử lý MIPS
  - 32-bit địa chỉ cho một không gian địa chỉ chung cho cả các ngăn nhớ và các cổng vào-ra
  - Các cổng vào-ra được gắn các địa chỉ thuộc vùng địa chỉ dự trữ
  - Vào/rã dữ liệu: sử dụng lệnh load/store

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

15

### Bản đồ bộ nhớ



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

16

### Ví dụ lập trình vào-ra cho MIPS

- Ví dụ: Có hai cổng vào-ra được gán địa chỉ:

- Cổng 1: 0xFFFFFFFF4
- Cổng 2: 0xFFFFFFFF8

- Ghi giá trị 0x41 ra cổng 1

```
addi $t0, $0, 0x41 # đưa giá trị 0x41
sw $t0, 0xFFFF4($0) # ra cổng 1
```

*Chú ý: giá trị 16-bit 0xFFFF4 được sign-extended thành 32-bit 0xFFFFFFFF4*

- Đọc dữ liệu từ cổng 2 đưa vào \$t3

```
lw $t3, 0xFFFF8($0) # đọc dữ liệu cổng 2 đưa vào $t3
```

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

17

### Vào-ra riêng biệt (Isolated IO)

- Cổng vào-ra được đánh địa chỉ theo không gian địa chỉ vào-ra riêng

- Lập trình trao đổi dữ liệu với cổng vào-ra bằng các lệnh vào-ra chuyên dụng

- Ví dụ: Intel x86

- Dùng 8-bit hoặc 16-bit địa chỉ cho không gian địa chỉ vào-ra riêng
- Có hai lệnh vào-ra chuyên dụng
  - Lệnh IN: nhận dữ liệu từ cổng vào
  - Lệnh OUT: đưa dữ liệu đến cổng ra

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

18

### 7.2. Các phương pháp điều khiển vào-ra

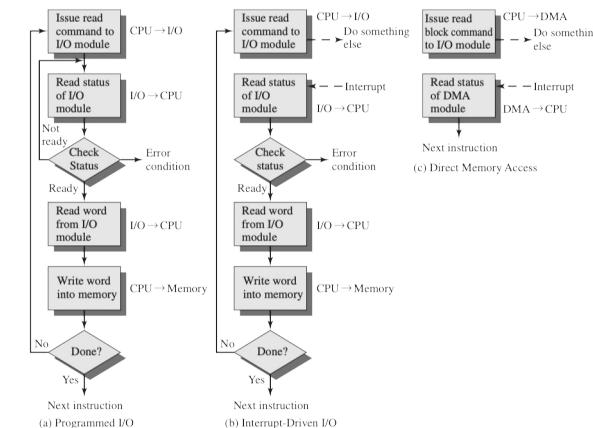
- Vào-ra bằng chương trình  
(Programmed IO)
- Vào-ra điều khiển bằng ngắt  
(Interrupt Driven IO)
- Truy nhập bộ nhớ trực tiếp - DMA  
(Direct Memory Access)

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

19

### Ba kỹ thuật thực hiện vào một khối dữ liệu



NKK-IT3030-CA2021.2

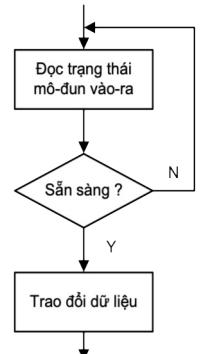
CH7-Hệ thống vào-ra

20

## 1. Vào-ra bằng chương trình

### ▪ Nguyên tắc chung:

- CPU điều khiển trực tiếp vào-ra bằng chương trình → cần phải lập trình vào-ra để trao đổi dữ liệu giữa CPU với mô-đun vào-ra
- CPU nhanh hơn thiết bị vào-ra rất nhiều lần, vì vậy trước khi thực hiện lệnh vào-ra, chương trình cần đọc và kiểm tra trạng thái sẵn sàng của mô-đun vào-ra



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

21

## Các tín hiệu điều khiển vào-ra

- Tín hiệu điều khiển (Control): kích hoạt thiết bị vào-ra
- Tín hiệu kiểm tra (Test): kiểm tra trạng thái của mô-đun vào-ra và thiết bị vào-ra
- Tín hiệu điều khiển đọc (Read): yêu cầu mô-đun vào-ra nhận dữ liệu từ thiết bị vào-ra và đưa vào bộ đệm dữ liệu, rồi CPU nhận dữ liệu đó
- Tín hiệu điều khiển ghi (Write): yêu cầu mô-đun vào-ra lấy dữ liệu trên bus dữ liệu đưa đến bộ đệm dữ liệu rồi chuyển ra thiết bị vào-ra

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

22

## Các lệnh vào-ra

- Với vào-ra theo bản đồ bộ nhớ: sử dụng các lệnh trao đổi dữ liệu với bộ nhớ để trao đổi dữ liệu với cổng vào-ra
- Với vào-ra riêng biệt: sử dụng các lệnh vào-ra chuyên dụng (IN, OUT)

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

23

## Đặc điểm

- Vào-ra do ý muốn của người lập trình
- CPU trực tiếp điều khiển trao đổi dữ liệu giữa CPU với mô-đun vào-ra
- CPU đợi trạng thái sẵn sàng của mô-đun vào-ra (qua vòng lặp) → tiêu tốn nhiều thời gian của CPU

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

24

## 2. Vào-ra điều khiển bằng ngắt

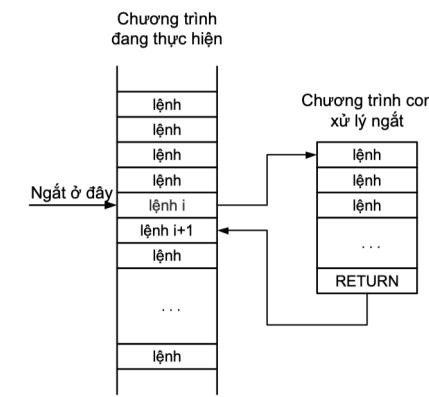
- Nguyên tắc chung:
  - CPU không phải đợi trạng thái sẵn sàng của mô-đun vào-ra, CPU thực hiện một chương trình nào đó
  - Khi mô-đun vào-ra sẵn sàng thì nó phát tín hiệu ngắt CPU
  - CPU thực hiện chương trình con xử lý ngắt vào-ra tương ứng để trao đổi dữ liệu
  - CPU trở lại tiếp tục thực hiện chương trình đang bị ngắt

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

25

## Chuyển điều khiển đến chương trình con ngắt



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

26

### Hoạt động vào dữ liệu: nhìn từ mô-đun vào-ra

- Mô-đun vào-ra nhận tín hiệu điều khiển đọc từ CPU
- Mô-đun vào-ra nhận dữ liệu từ thiết bị vào-ra, trong khi đó CPU làm việc khác
- Khi đã có dữ liệu → mô-đun vào-ra phát tín hiệu ngắt CPU
- CPU yêu cầu dữ liệu
- Mô-đun vào-ra chuyển dữ liệu đến CPU

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

27

### Hoạt động vào dữ liệu: nhìn từ CPU

- Phát tín hiệu điều khiển đọc
- Làm việc khác
- Cuối mỗi chu trình lệnh, kiểm tra tín hiệu yêu cầu ngắt
- Nếu bị ngắt:
  - Cắt ngữ cảnh (nội dung các thanh ghi liên quan)
  - Thực hiện chương trình con xử lý ngắt để vào dữ liệu
  - Khôi phục ngữ cảnh của chương trình đang thực hiện

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

28

### Các vấn đề nảy sinh khi thiết kế

- Làm thế nào để xác định được mô-đun vào-ra nào phát tín hiệu ngắt ?
- CPU làm như thế nào khi có nhiều yêu cầu ngắt cùng xảy ra ?

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

29

### Các phương pháp nối ghép ngắt

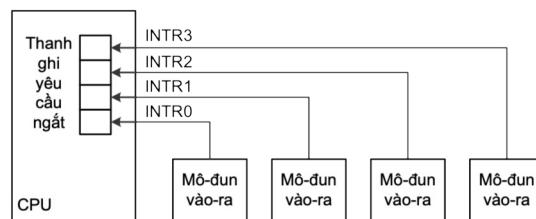
- Sử dụng nhiều đường yêu cầu ngắt
- Hỏi vòng bằng phần mềm (Software Poll)
- Hỏi vòng bằng phần cứng (Daisy Chain or Hardware Poll)
- Sử dụng bộ điều khiển ngắt (PIC)

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

30

### Nhiều đường yêu cầu ngắt



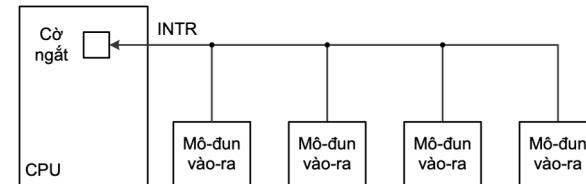
- Mỗi mô-đun vào-ra được nối với một đường yêu cầu ngắt
- CPU phải có nhiều đường tín hiệu yêu cầu ngắt
- Hạn chế số lượng mô-đun vào-ra
- Các đường ngắt được qui định mức ưu tiên

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

31

### Hỏi vòng bằng phần mềm



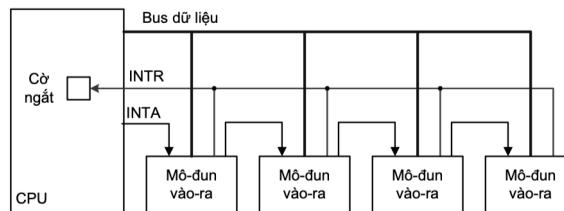
- CPU thực hiện phần mềm hỏi lần lượt từng mô-đun vào-ra
- Chậm
- Thứ tự các mô-đun được hỏi vòng chính là thứ tự ưu tiên

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

32

### Hỏi vòng bằng phần cứng



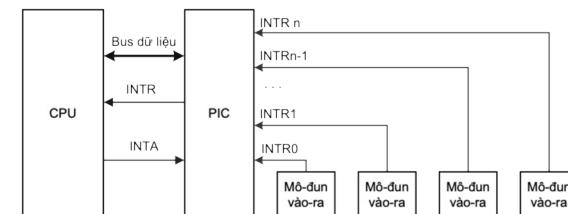
- CPU phát tín hiệu chấp nhận ngắt (INTA) đến mô-đun vào-ra đầu tiên
- Nếu mô-đun vào-ra đó không gây ra ngắt thì nó gửi tín hiệu đến mô-đun kế tiếp, cho đến khi xác định được mô-đun gây ngắt
- Thứ tự các mô-đun vào-ra kết nối trong chuỗi xác định thứ tự ưu tiên

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

33

### Bộ điều khiển ngắt lập trình được



- PIC – Programmable Interrupt Controller

- PIC có nhiều đường vào yêu cầu ngắt có qui định mức ưu tiên
- PIC chọn một yêu cầu ngắt không bị cấm có mức ưu tiên cao nhất gửi tới CPU

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

34

### Đặc điểm của vào-ra điều khiển bằng ngắt

- Có sự kết hợp giữa phần cứng và phần mềm
  - Phần cứng: gây ngắt CPU
  - Phần mềm: trao đổi dữ liệu giữa CPU với mô-đun vào-ra
- CPU trực tiếp điều khiển vào-ra
- CPU không phải đợi mô-đun vào-ra, do đó hiệu quả sử dụng CPU tốt hơn

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

35

### 3. DMA (Direct Memory Access)

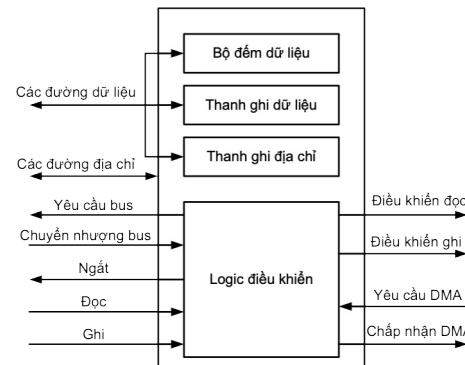
- Vào-ra bằng chương trình và bằng ngắt do CPU trực tiếp điều khiển:
  - Chiếm thời gian của CPU
- Để khắc phục dùng kỹ thuật DMA
  - Sử dụng mô-đun điều khiển vào-ra chuyên dụng, gọi là DMAC (Controller), điều khiển trao đổi dữ liệu giữa mô-đun vào-ra với bộ nhớ chính

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

36

### Sơ đồ cấu trúc của DMAC



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

37

### Các thành phần của DMAC

- Thanh ghi dữ liệu: chứa dữ liệu trao đổi
- Thanh ghi địa chỉ: chứa địa chỉ ngăn nhớ dữ liệu
- Bộ đếm dữ liệu: chứa số từ dữ liệu cần trao đổi
- Logic điều khiển: điều khiển hoạt động của DMAC

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

38

### Hoạt động DMA

- CPU “nói” cho DMAC
  - Vào hay Ra dữ liệu
  - Địa chỉ thiết bị vào-ra (cổng vào-ra tương ứng)
  - Địa chỉ đầu của mảng nhớ chứa dữ liệu → nạp vào thanh ghi địa chỉ
  - Số từ dữ liệu cần truyền → nạp vào bộ đếm dữ liệu
- CPU làm việc khác
- DMAC điều khiển trao đổi dữ liệu
- Sau khi truyền được một từ dữ liệu thì:
  - nội dung thanh ghi địa chỉ tăng
  - nội dung bộ đếm dữ liệu giảm
- Khi bộ đếm dữ liệu = 0, DMAC gửi tín hiệu ngắt CPU để báo kết thúc DMA

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

39

### Các kiểu thực hiện DMA

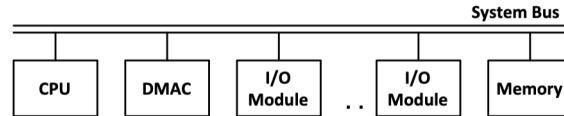
- DMA truyền theo khối (Block-transfer DMA): DMAC sử dụng bus để truyền xong cả khối dữ liệu
- DMA lấy chu kỳ (Cycle Stealing DMA): DMAC cưỡng bức CPU treo tạm thời từng chu kỳ bus, DMAC chiếm bus thực hiện truyền một từ dữ liệu.
- DMA trong suốt (Transparent DMA): DMAC nhận biết những chu kỳ nào CPU không sử dụng bus thì chiếm bus để trao đổi một từ dữ liệu.

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

40

### Cấu hình DMA (1)



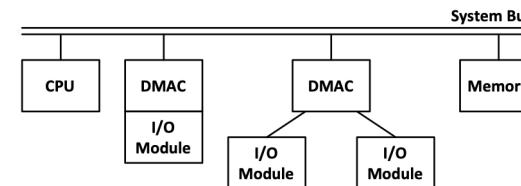
- Mỗi lần trao đổi một dữ liệu, DMAC sử dụng bus hai lần
  - Giữa mô-đun vào-ra với DMAC
  - Giữa DMAC với bộ nhớ

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

41

### Cấu hình DMA (2)



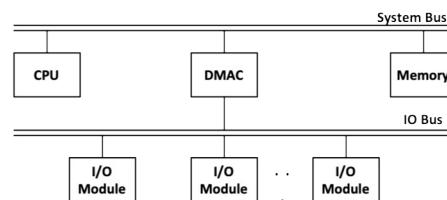
- DMAC điều khiển một hoặc vài mô-đun vào-ra
- Mỗi lần trao đổi một dữ liệu, DMAC sử dụng bus một lần
  - Giữa DMAC với bộ nhớ

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

42

### Cấu hình DMA (3)



- Bus vào-ra tách rời hỗ trợ tất cả các thiết bị cho phép DMA
- Mỗi lần trao đổi một dữ liệu, DMAC sử dụng bus một lần
  - Giữa DMAC với bộ nhớ

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

43

### Đặc điểm của DMA

- CPU không tham gia trong quá trình trao đổi dữ liệu
- DMAC điều khiển trao đổi dữ liệu giữa bộ nhớ chính với mô-đun vào-ra (hoàn toàn bằng phần cứng) → tốc độ nhanh
- Phù hợp với các yêu cầu trao đổi mảng dữ liệu có kích thước lớn

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

44

#### 4. Bộ xử lý vào-ra

- Việc điều khiển vào-ra được thực hiện bởi một bộ xử lý vào-ra chuyên dụng
- Bộ xử lý vào-ra hoạt động theo chương trình của riêng nó
- Chương trình của bộ xử lý vào-ra có thể nằm trong bộ nhớ chính hoặc nằm trong một bộ nhớ riêng

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

45

#### 7.3. Nối ghép thiết bị vào-ra

##### 1. Các kiểu nối ghép vào-ra

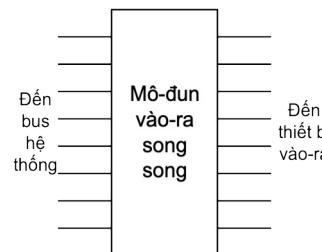
- Nối ghép song song
- Nối ghép nối tiếp

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

46

#### Nối ghép song song



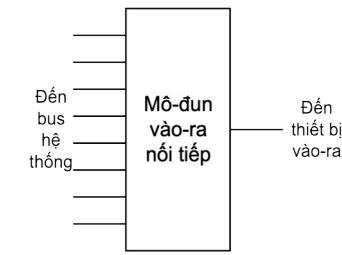
- Truyền nhiều bit song song
- Tốc độ nhanh
- Cần nhiều đường truyền dữ liệu

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

47

#### Nối ghép nối tiếp



- Truyền lần lượt từng bit
- Cần có bộ chuyển đổi từ dữ liệu song song sang nối tiếp hoặc/và ngược lại
- Tốc độ chậm hơn
- Cần ít đường truyền dữ liệu

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

48

## 2. Các cấu hình nối ghép

### ▪ Điểm tới điểm (Point to Point)

- Thông qua một cổng vào-ra nối ghép với một thiết bị
- Cổng được thiết kế chuyên dụng cho thiết bị tương ứng
- Hạn chế số lượng thiết bị vào-ra

### ▪ Điểm tới đa điểm (Point to Multipoint)

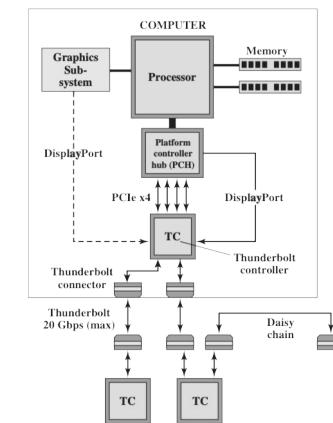
- Thông qua một cổng vào-ra cho phép nối ghép được với nhiều thiết bị
- Ví dụ:
  - USB (Universal Serial Bus): 127 thiết bị
  - IEEE 1394 (FireWire): 63 thiết bị
  - Thunderbolt

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

49

## Thunderbolt



NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

50

## Kiến trúc máy tính

Hết chương 7

NKK-IT3030-CA2021.2

CH7-Hệ thống vào-ra

51



## Nội dung học phần

- Chương 1. Giới thiệu chung
- Chương 2. Hệ thống máy tính
- Chương 3. Số học máy tính
- Chương 4. Kiến trúc tập lệnh
- Chương 5. Bộ xử lý
- Chương 6. Bộ nhớ máy tính
- Chương 7. Hệ thống vào-ra
- Chương 8. Các kiến trúc song song

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

2

## Kiến trúc máy tính

### **Chương 8 CÁC KIẾN TRÚC SONG SONG**

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

3

## Nội dung

- 8.1. Phân loại kiến trúc máy tính
- 8.2. Đa xử lý bộ nhớ dùng chung
- 8.3. Đa xử lý bộ nhớ phân tán
- 8.4. Bộ xử lý đồ họa đa dụng

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

4

## 8.1. Phân loại kiến trúc máy tính

### Phân loại kiến trúc máy tính (Michael Flynn -1966)

- SISD - Single Instruction Stream, Single Data Stream
- SIMD - Single Instruction Stream, Multiple Data Stream
- MISD - Multiple Instruction Stream, Single Data Stream
- MIMD - Multiple Instruction Stream, Multiple Data Stream

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

5

## SISD



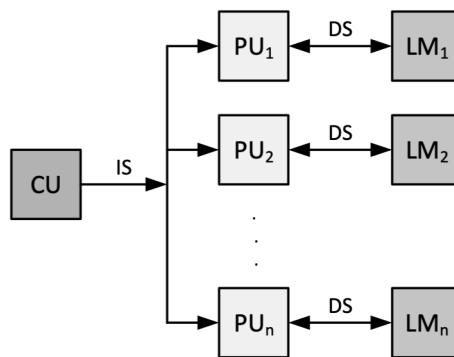
- CU: Control Unit
- PU: Processing Unit
- MU: Memory Unit
- Một bộ xử lý
- Đơn dòng lệnh
- Dữ liệu được lưu trữ trong một bộ nhớ
- Chính là Kiến trúc von Neumann (tuần tự)

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

6

## SIMD



NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

7

## SIMD (tiếp)

- Đơn dòng lệnh điều khiển đồng thời các đơn vị xử lý PUs
- Mỗi đơn vị xử lý có một bộ nhớ dữ liệu riêng LM (local memory)
- Mỗi lệnh được thực hiện trên một tập các dữ liệu khác nhau
- Các mô hình SIMD
  - Vector Computer
  - GPU (Graphic Processing Unit)

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

8

## MISD

- Một luồng dữ liệu cùng được truyền đến một tập các bộ xử lý
- Mỗi bộ xử lý thực hiện một dãy lệnh khác nhau.
- Chưa tồn tại máy tính thực tế
- Có thể có trong tương lai

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

9

## MIMD

- Tập các bộ xử lý
- Các bộ xử lý đồng thời thực hiện các dãy lệnh khác nhau trên các dữ liệu khác nhau
- Các mô hình MIMD
  - Multiprocessors (Shared Memory Multiprocessors)
  - Multicomputers (Distributed Memory Multiprocessors)

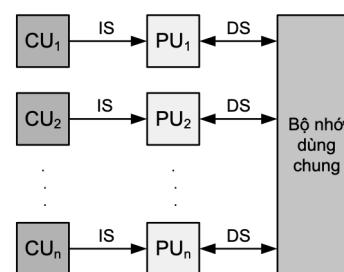
NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

10

## MIMD - Shared Memory

Đa xử lý bộ nhớ dùng chung  
(shared memory multiprocessors)



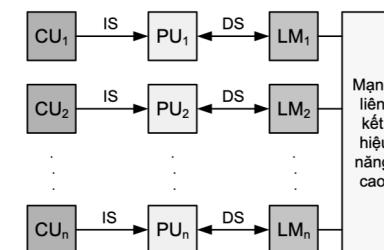
NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

11

## MIMD - Distributed Memory

Đa xử lý bộ nhớ phân tán  
(distributed memory multiprocessors or multicomputers)



NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

12

### Phân loại các kỹ thuật song song

- Song song mức lệnh (Instruction-Level Parallelism)
  - pipeline
  - superscalar
- Song song mức dữ liệu (Data-Level Parallelism)
  - SIMD
- Song song mức luồng (Thread-Level Parallelism)
  - MIMD
- Song song mức yêu cầu (Request-Level Parallelism)
  - Cloud computing

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

13

### 8.2. Đa xử lý bộ nhớ dùng chung

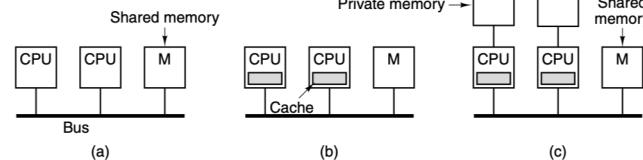
- Hệ thống đa xử lý đối xứng (SMP- Symmetric Multiprocessors)
- Hệ thống đa xử lý không đối xứng (NUMA - Non-Uniform Memory Access)
- Bộ xử lý đa lõi (Multicore Processors)

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

14

### SMP hay UMA (Uniform Memory Access)

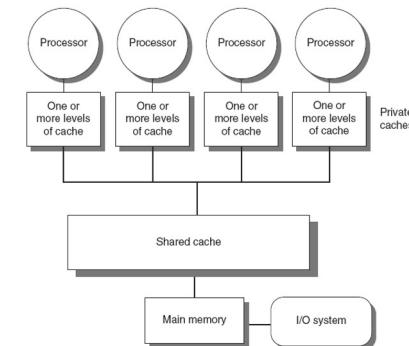


NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

15

### SMP hay UMA (Uniform Memory Access)



NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

16

### SMP (tiếp)

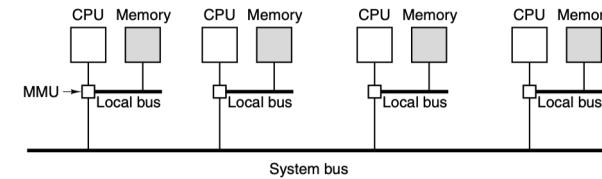
- Một máy tính có  $n \geq 2$  bộ xử lý giống nhau
- Các bộ xử lý dùng chung bộ nhớ và hệ thống vào-rà
- Thời gian truy cập bộ nhớ là bằng nhau với các bộ xử lý
- Các bộ xử lý có thể thực hiện chức năng giống nhau
- Hệ thống được điều khiển bởi một hệ điều hành phân tán
- Hiệu năng: Các công việc có thể thực hiện song song
- Khả năng chịu lỗi

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

17

### NUMA (Non-Uniform Memory Access)



- Có một không gian địa chỉ chung cho tất cả CPU
- Mỗi CPU có thể truy cập từ xa sang bộ nhớ của CPU khác
- Truy nhập bộ nhớ từ xa chậm hơn truy nhập bộ nhớ cục bộ

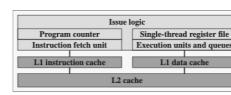
NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

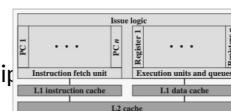
18

### Bộ xử lý đa lõi (multicores)

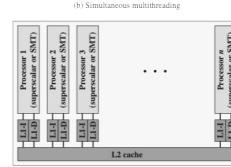
- Thay đổi của bộ xử lý:
  - Tuần tự
  - Pipeline
  - Siêu vô hướng
  - Đa luồng
  - Đa lõi: nhiều CPU trên một chip



(a) Superscalar



(b) Simultaneous multithreading



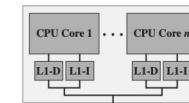
(c) Multicore

NKK-IT3030-CA2021.2

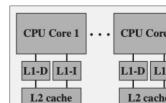
CH8-Các kiến trúc song song

19

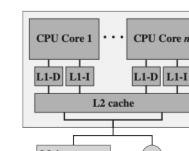
### Các dạng tổ chức bộ xử lý đa lõi



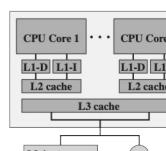
(a) Dedicated L1 cache



(b) Dedicated L2 cache



(c) Shared L2 cache



(d) Shared L3 cache

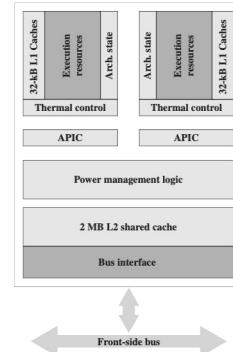
NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

20

### Intel - Core Duo

- 2006
- Two x86 superscalar, shared L2 cache
- Dedicated L1 cache per core
  - 32KiB instruction cache and 32KiB data cache
- 2MiB shared L2 cache

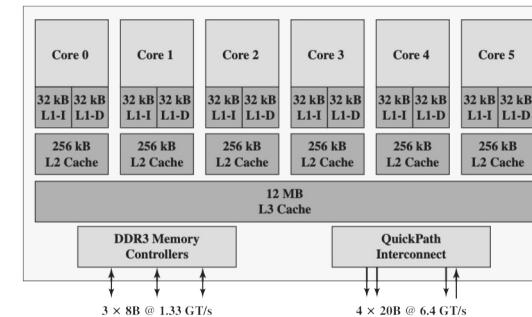


NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

21

### Intel Core i7-990X

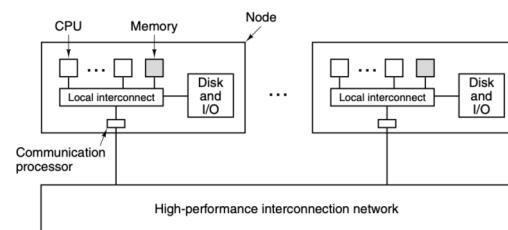


NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

22

### 8.3. Đa xử lý bộ nhớ phân tán



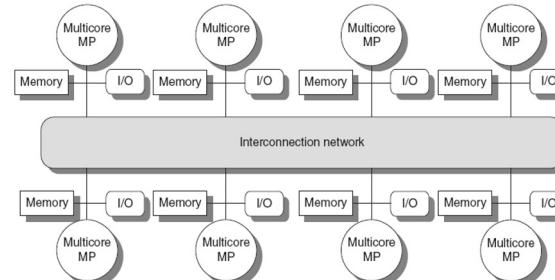
- Máy tính qui mô lớn (Warehouse Scale Computers or Massively Parallel Processors – MPP)
- Máy tính cụm (clusters)

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

23

### Đa xử lý bộ nhớ phân tán

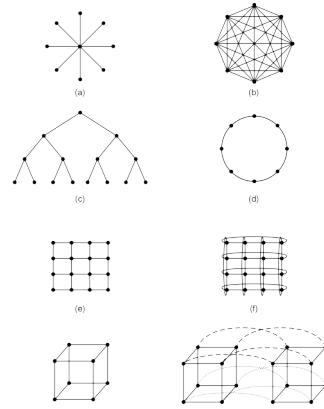


NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

24

### Mạng liên kết



NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

25

### Massively Parallel Processors

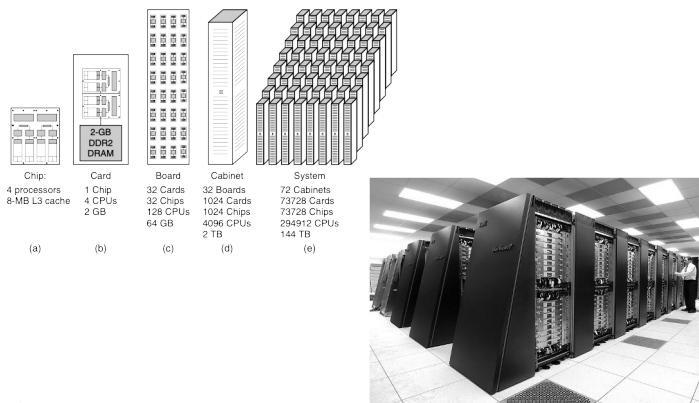
- Hệ thống qui mô lớn
- Đắt tiền: nhiều triệu USD
- Dùng cho tính toán khoa học và các bài toán có số phép toán và dữ liệu rất lớn
- Siêu máy tính

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

26

### IBM Blue Gene/P



NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

27

### Cluster

- Nhiều máy tính được kết nối với nhau bằng mạng liên kết tốc độ cao (~ Gbps)
- Mỗi máy tính có thể làm việc độc lập (PC hoặc SMP)
- Mỗi máy tính được gọi là một node
- Các máy tính có thể được quản lý làm việc song song theo nhóm (cluster)
- Toàn bộ hệ thống có thể coi như là một máy tính song song
- Tính sẵn sàng cao
- Khả năng chịu lỗi lớn

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

28

#### 8.4. Bộ xử lý đồ họa đa dụng

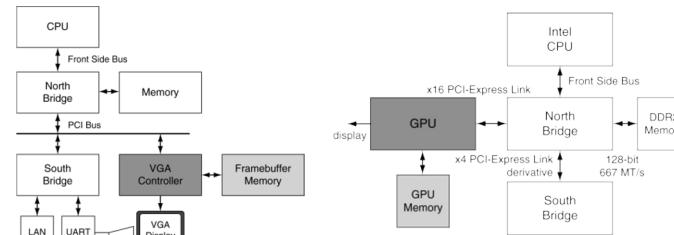
- Kiến trúc SIMD
- Xuất phát từ bộ xử lý đồ họa GPU (Graphic Processing Unit) hỗ trợ xử lý đồ họa 2D và 3D: xử lý dữ liệu song song
- GPGPU – General purpose Graphic Processing Unit
- Hệ thống lai CPU/GPGPU
  - CPU là host: thực hiện theo tuần tự
  - GPGPU: tính toán song song

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

29

#### Bộ xử lý đồ họa trong máy tính

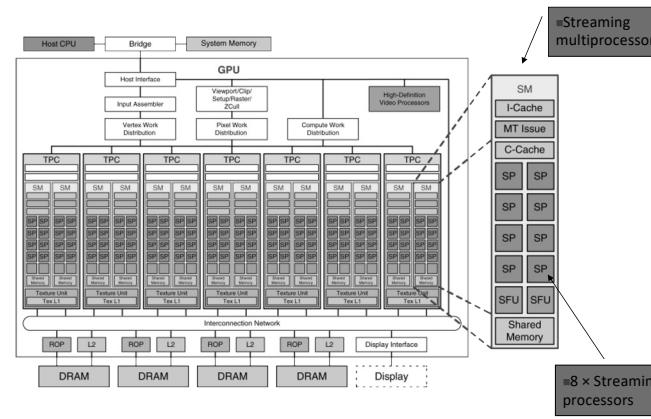


NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

30

#### GPGPU: NVIDIA Tesla

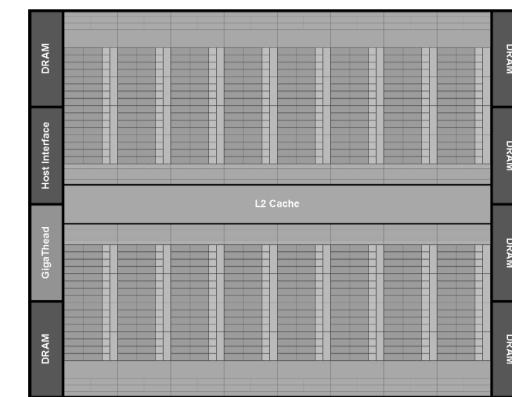


NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

31

#### GPGPU: NVIDIA Fermi



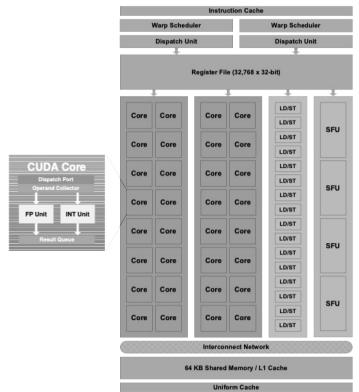
NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

32

### NVIDIA Fermi

- Có 16 Streaming Multiprocessors (SM)
- Mỗi SM có 32 CUDA cores.
- Mỗi CUDA core (Compute Unified Device Architecture) có 01 FPU và 01 IU

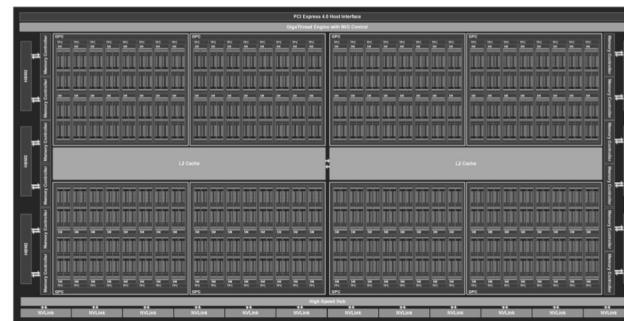


NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

33

### NVIDIA A100 Tensor Core GPU Architecture



NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

34

### GA100 Streaming Multiprocessor



NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

35

### Kiến trúc máy tính

Hết chương 8

NKK-IT3030-CA2021.2

CH8-Các kiến trúc song song

36