

BAN HỌC TẬP CÔNG NGHỆ PHẦN MỀM

TRAINING CUỐI KỲ HỌC KỲ I NĂM HỌC 2023 – 2024



Sharing is learning



 **BAN HỌC TẬP**

Khoa Công nghệ Phần mềm

Trường Đại học Công nghệ Thông tin

Đại học Quốc gia thành phố Hồ Chí Minh

 **CONTACT**

bht.cnpm.uit@gmail.com

fb.com/bhtcnpm

fb.com/groups/bht.cnpm.uit

TRAINING

TỔ CHỨC & CẤU TRÚC MÁY TÍNH II

- ⌚ **Thời gian:** 19:30 thứ 7 ngày 06/01/2024
- 📍 **Địa điểm:** Microsofts Team – w2dsy1q
- 👤 **Trainers:** Đặng Ngọc Trường Giang – KTPM2023.1
Võ Xuân Ngọc – KTPM2023.2



Sharing is learning

Nội dung Training

Chương 6: Kiến trúc tập lệnh

- 6.1. Giới thiệu về kiến trúc tập lệnh
- 6.2. Định nghĩa và phân loại kiến trúc tập lệnh
- 6.3. Lệnh
- 6.4. Tập lệnh MIPS
- 6.5. Toán hạng thanh ghi
- 6.6. Toán hạng bộ nhớ và số tức thời
- 6.7. Định lệnh dạng R



Sharing is learning

Nội dung Training

6.8. Định lệnh dạng I, J

6.9. Các mô hình địa chỉ

6.10. Nhóm lệnh số học và luận lý

6.11. Nhóm lệnh truyền dữ liệu

6.12. Nhóm lệnh điều khiển



Sharing is learning

Nội dung Training

Chương 7: Biên dịch chương trình

- 7.1. Giới thiệu Trình biên dịch
- 7.2. Trình biên dịch hợp ngữ
- 7.3. Biên dịch ngược

Chương 8: Bộ xử lý

- 8.1. Giới thiệu tổng quan về bộ xử lý
- 8.2. Giới thiệu về Datapath
- 8.3. Công đoạn tìm nạp lệnh



Sharing is learning

Nội dung Training

- 8.4. Công đoạn giải mã lệnh
- 8.5. Công đoạn ALU
- 8.6. Công đoạn truy xuất vùng nhớ
- 8.7. Công đoạn ghi lại kết quả
- 8.8. Khối control
- 8.9. Khối ALU
- 8.10. Thực thi lệnh số học và logic
- 8.11. Thực thi lệnh lw (load word)



Sharing is learning

Nội dung Training

8.12. Thực thi lệnh sw (store word)

8.13. Thực thi lệnh beq

Chương 9: Hiệu suất máy tính

9.1. Hiệu suất

9.2. Các yếu tố ảnh hưởng đến hiệu suất máy tính



Sharing is learning

Kiến trúc tập lệnh

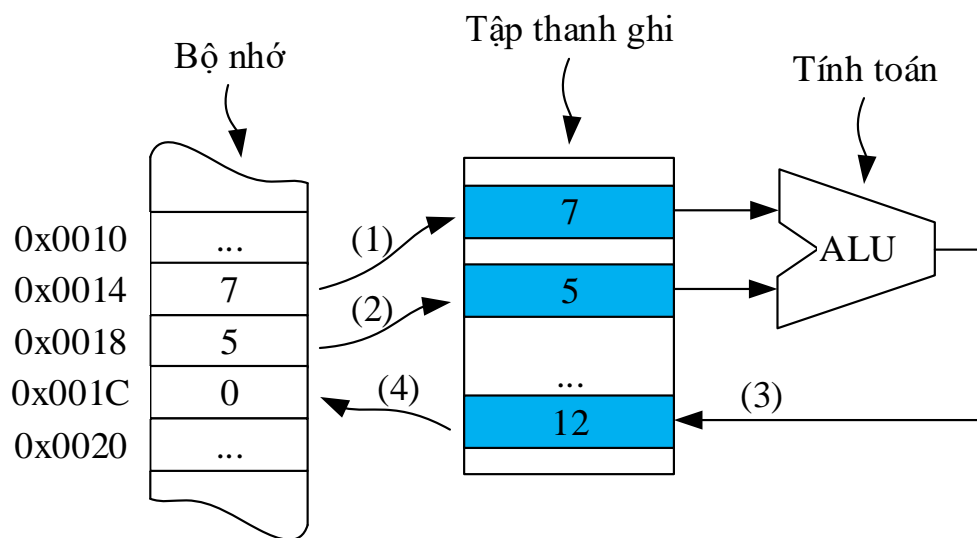
- Định nghĩa :
 - Lệnh: là chỉ dẫn để máy tính thực hiện công việc.
 - Tập lệnh: là tập hợp các lệnh.
 - Những máy tính khác nhau có tập lệnh khác nhau.
 - Nhưng! Các tập lệnh đều có điểm chung.
 - Kiến trúc tập lệnh = tập lệnh + biểu diễn lệnh.
- Biểu diễn lệnh :
 - op (opcode): Mã lệnh
 - Toán hạng: các toán hạng cần thiết để thực thi lệnh.
 - Các trường khác.



Sharing is learning

Kiến trúc tập lệnh

- Kiểu thanh ghi – thanh ghi:



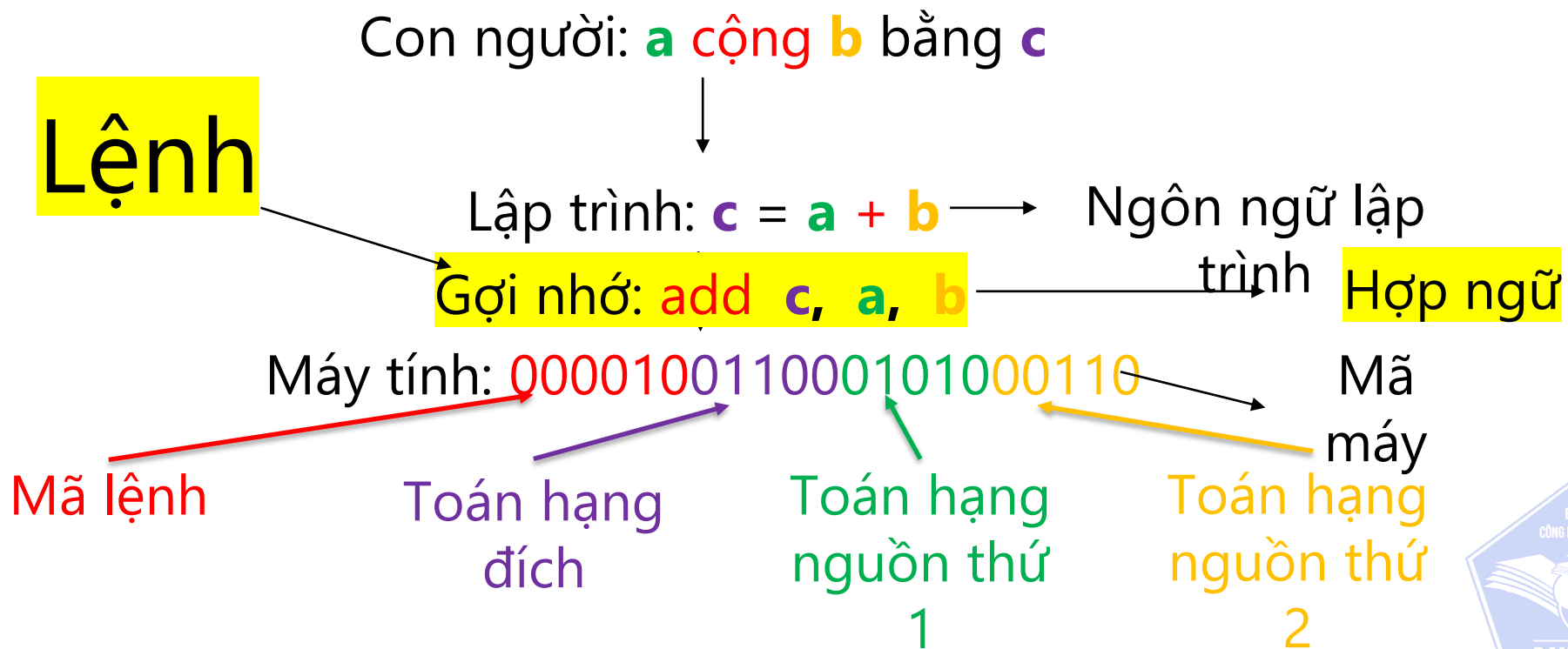
- Dữ liệu được lưu trữ ở bộ nhớ
- Tính toán trên thanh ghi (không tính toán trên bộ nhớ)
- Cần **nạp** dữ liệu từ bộ nhớ vào thanh ghi để tính toán
- Cần **lưu** giá trị thanh ghi vào bộ nhớ sau khi tính toán



Sharing is learning

Kiến trúc Tập lệnh

- Lệnh



Kiến trúc Tập lệnh

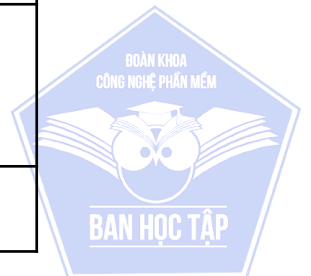
- Tập lệnh MIPS
 - Thiết kế theo kiến trúc thanh ghi – thanh ghi
 - Độ rộng lệnh: Cố định 32 bit cho tất cả các lệnh
 - Định dạng lệnh: R, I, J
 - Tập thanh ghi: 32 thanh ghi 32 bit, thanh ghi \$zero luôn bằng 0
 - Kiểu dữ liệu: Byte (8 bit), halfword (16 bit), word (32 bit)
 - Chế độ định địa chỉ: 5 chế độ
 - Toán hạng: Thanh ghi, số tức thời (bù 2), bộ nhớ
 - Định địa chỉ theo byte



Sharing is learning

Kiến trúc Tập lệnh - Tập lệnh MIPS

Loại	Lệnh	Ví dụ	Ý nghĩa	ĐD
Số học	Cộng	add \$s1,\$s2,\$s3	$\$s1 = \$s2 + \$s3$	R
	Trừ	sub \$s1,\$s2,\$s3	$\$s1 = \$s2 - \$s3$	R
	Cộng tức thì	addi \$s1,\$s2,20	$\$s1 = \$s2 + 20$	I
Truyền dữ liệu	Nạp word	lw \$s1,20(\$s2)	$\$s1 = \text{Mem}[\$s2 + 20]$	I
	Lưu word	sw \$s1,20(\$s2)	$\text{Mem}[\$s2 + 20] = \$s1$	I
Luận lý	NOR luận lý	nor \$s1,\$s2,\$s3	$\$s1 = \sim(\$s2 \mid \$s3)$	R
	Dịch phải luận lý	srl \$s1,\$s2,10	$\$s1 = \$s2 \gg 10$	R
Rẽ nhánh	Nhảy nếu bằng	beq \$s1,\$s2, label	Nếu ($\$s1 == \$s2$) đi đến label	I
Nhảy	Nhảy	j label	Đi đến label	J



Sharing is learning

Kiến trúc Tập lệnh

- Toán hạng
 - Toán hạng là một dữ liệu được dùng để tính toán
 - MIPS có 3 loại toán hạng:
 - Toán hạng thanh ghi: Dữ liệu nằm trong thanh ghi
 - Toán hạng bộ nhớ: Dữ liệu nằm trong bộ nhớ
 - Toán hạng số tức thời: Dữ liệu nằm ngay trong lệnh



Sharing is learning

- Toán hạng thanh ghi

- Kiến trúc thanh ghi – thanh ghi: Tính toán trên thanh ghi
- MIPS có 32 thanh ghi 32 bit
 - Sử dụng cho truy xuất dữ liệu tạm
 - Được đánh số từ 0 đến 31
 - Kiểu dữ liệu 32 bit (word)
- Tên gợi nhớ: Tiền tố \$ theo sau là chỉ số hoặc tên (\$2 hay \$sp)
 - \$t0, \$t1, ..., \$t9 cho các dữ liệu tạm
 - \$s0, \$s1, ..., \$s7 cho lưu trữ các biến
 - \$v0, \$v1, \$k1, ... cho các mục đích đặc biệt khác



Sharing is learning

Toán hạng thanh ghi

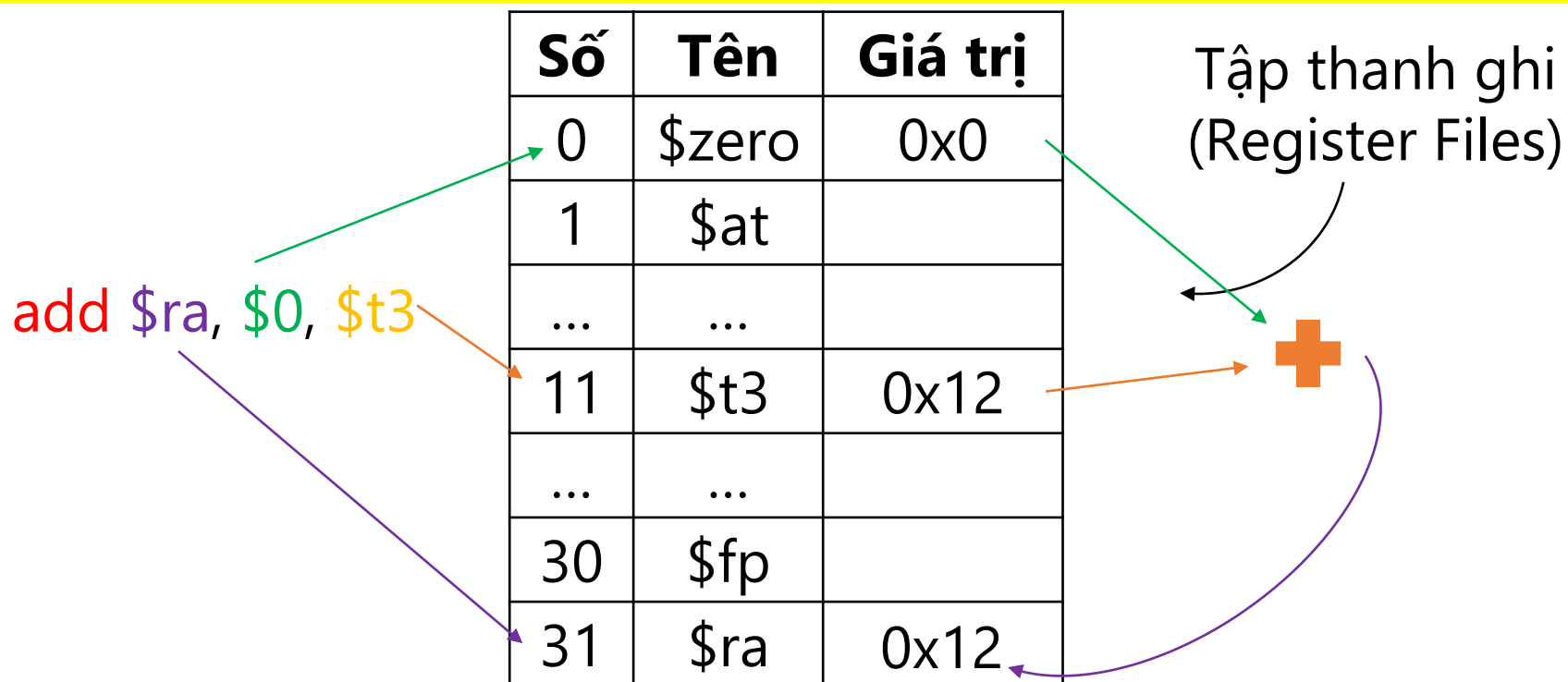
NAME	NUMBER	USE	PRESERVED ACROSS A CALL?
\$zero	0	The Constant Value 0	N.A.
\$at	1	Assembler Temporary	No
\$v0-\$v1	2-3	Values for Function Results and Expression Evaluation	No
\$a0-\$a3	4-7	Arguments	No
\$t0-\$t7	8-15	Temporaries	No
\$s0-\$s7	16-23	Saved Temporaries	Yes
\$t8-\$t9	24-25	Temporaries	No
\$k0-\$k1	26-27	Reserved for OS Kernel	No
\$gp	28	Global Pointer	Yes
\$sp	29	Stack Pointer	Yes
\$fp	30	Frame Pointer	Yes
\$ra	31	Return Address	Yes



Sharing is learning

Toán hạng thanh ghi - Ví dụ:

Tên hoặc số của thanh ghi là gợi nhớ cho địa chỉ của thanh ghi trong tập thanh ghi

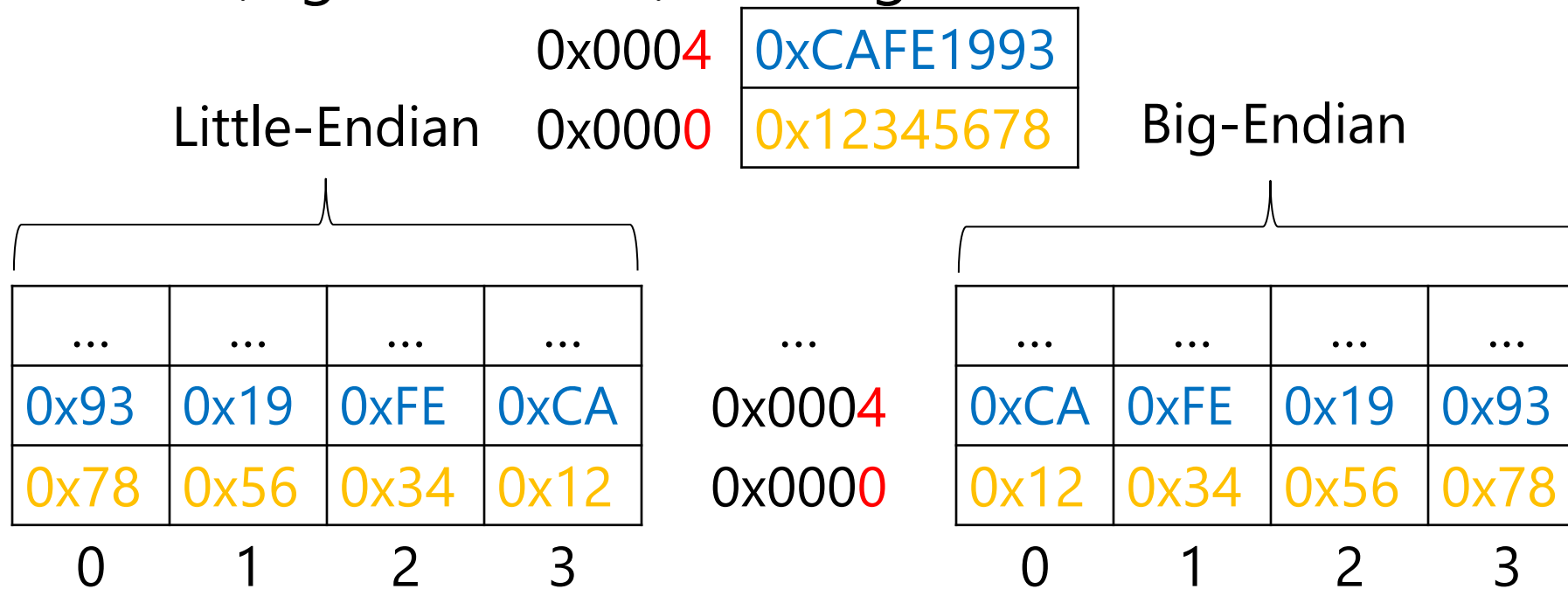


Sharing is learning

Toán hạng Bộ nhớ

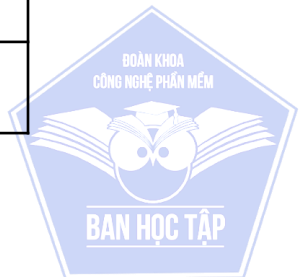
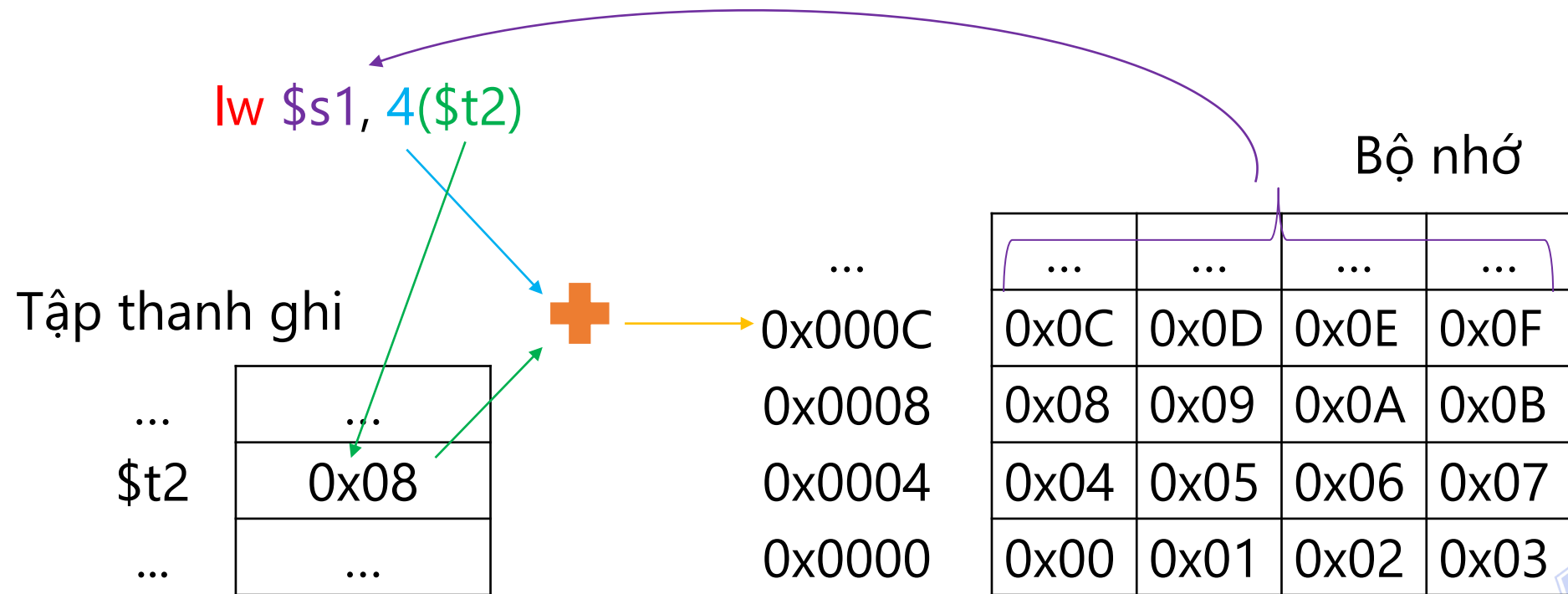
- Bộ nhớ được đánh địa chỉ theo byte
- MIPS quy định địa chỉ bộ nhớ phải là bội số của 4
- MIPS sử dụng mô hình địa chỉ Big-Endian

} 1 word
= 4
byte



Sharing is learning

Toán hạng Bộ nhớ – Ví dụ:



Sharing is learning

Toán hạng số tức thời

- Dữ liệu hằng số được chỉ định ngay trong lệnh
 - Sử dụng dữ liệu ngay mà không cần tìm kiếm như thanh ghi và bộ nhớ
 - Không cần phải nạp dữ liệu từ bộ nhớ!!!
 - Nhưng giá trị thường nhỏ
- Ví dụ:
 - `addi $s3, $s2, 4`
 - `addi $t2, $t1, -7`
- MIPS có thanh ghi số 0 (\$zero) luôn luôn là một hằng số 0
 - Sao chép giá trị: `add $t2, $t1, $zero`



Sharing is learning

Kiến trúc tập lệnh

- Định dạng R

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

■ Các trường lệnh

- op (opcode): Mã lệnh
- rs: Địa chỉ toán hạng thanh ghi nguồn thứ nhất
- rt: Địa chỉ toán hạng thanh ghi nguồn thứ hai
- rd: Địa chỉ toán hạng thanh ghi đích
- shamt (shift amount): Lượng dịch (mặc định là 00000)
- funct (function code): Mã lệnh mở rộng cho op



Sharing is learning

Kiến trúc tập lệnh

- Định dạng R

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

add \$t0, \$s1, \$s2

op (add)	\$s1	\$s2	\$t0	0	funct (add)
----------	------	------	------	---	-------------

0	17	18	8	0	0x20
---	----	----	---	---	------

000000	10001	10010	01000	00000	100000
--------	-------	-------	-------	-------	--------

00000010001100100100000000100000 0x02324020



Sharing is learning

Kiến trúc tập lệnh

- Định dạng I

op	rs	rt	immediate
6 bit	5 bit	5 bit	16 bit

- Các trường lệnh:

- op (opcode): Mã lệnh
- rs: Địa chỉ toán hạng thanh ghi nguồn thứ nhất
- rt: Địa chỉ toán hạng thanh ghi nguồn thứ hai hoặc thanh ghi đích
- immediate: Số tức thời 16 bit (biểu diễn dạng bù 2)
 - Thường là độ dời
 - Quy ước: Nếu sử dụng số tức thời lớn hơn 16 bit thì sẽ gây lỗi biên dịch



Sharing is learning

Kiến trúc tập lệnh

- Ví dụ: Biểu diễn lệnh **lw** \$a0, 12 (\$s2)

op (lw)	\$s2	\$a0	12
0x23	18	4	12
100011	10010	00100	000000000000001100

→ 1000111001000100000000000000001100 hay 0x8E44000C



Sharing is learning

Kiến trúc tập lệnh

- Định dạng J:

op	address
6 bit	26 bit

- Các trường lệnh:
 - op (opcode): Mã lệnh
 - address: Số tức thời 26 bit (biểu diễn dạng bù 2)
 - Bit [27:2] của địa chỉ nhảy tới



Sharing is learning

Kiến trúc tập lệnh

- Ví dụ:

0xCAFEBAB8: j sub_pro

...

0xCAFEBAFC: sub_pro:

op (j)	0xCAFEBAFC = {(PC+4)[31:28], address, 2'b0}
0x2	0x2BFAEBF
000010	10101111111010111010111111

→ 00001010101111111010111010111111 hay 0x0ABFAEBF



Sharing is learning

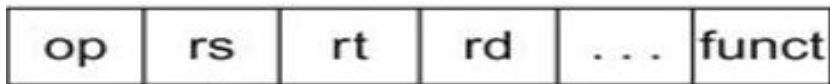
Kiến trúc tập lệnh

- Các mô hình địa chỉ:

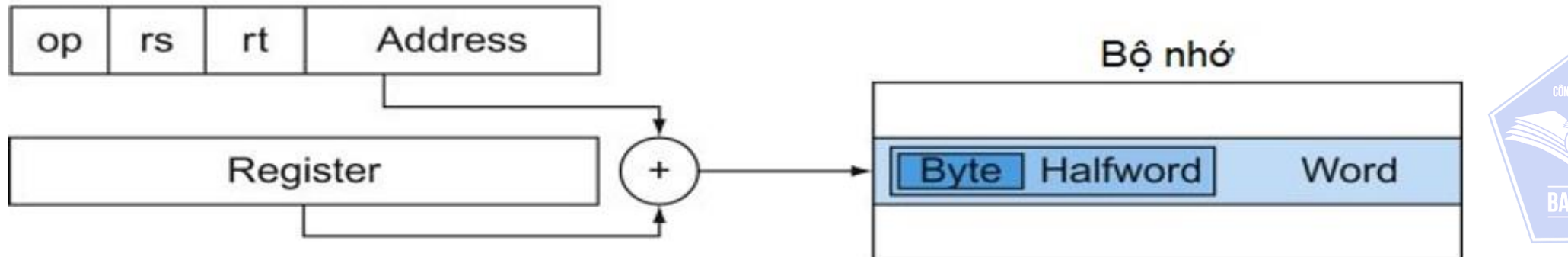
1. Định địa chỉ tức thời



2. Định địa chỉ thanh ghi



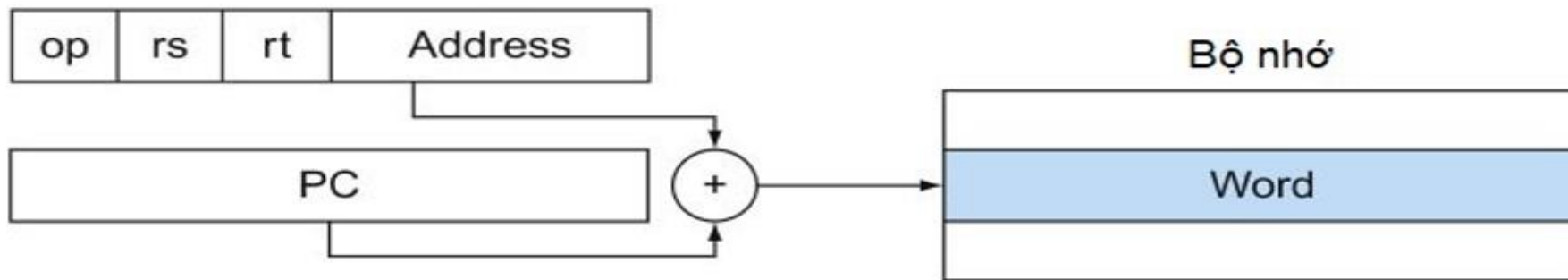
3. Định địa chỉ nền



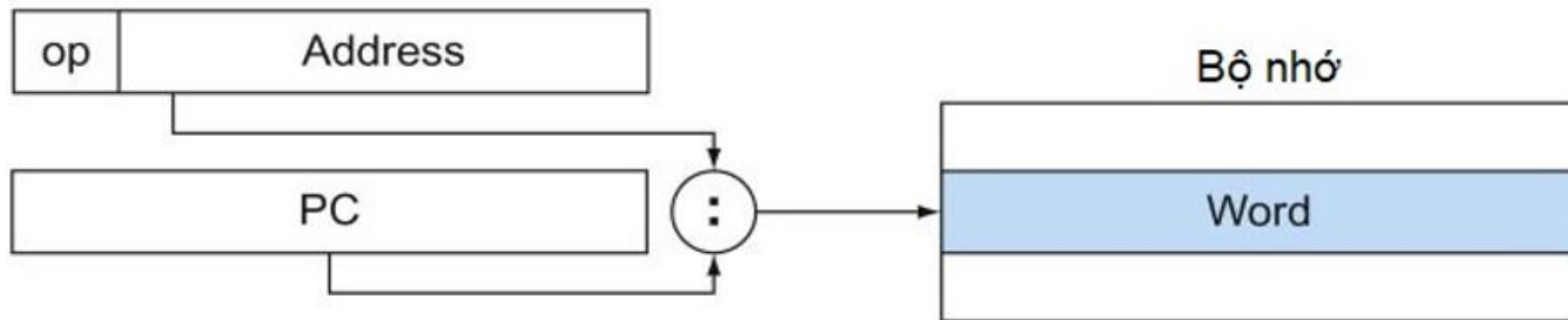
Sharing is learning

Kiến trúc tập lệnh

4. Định địa chỉ PC tương đối



5. Định địa chỉ gián tiếp



Kiến trúc tập lệnh

- Nhóm lệnh luận lý và số học

Thao tác	MIPS	Định dạng
Cộng	add/addu, addi/addiu	R, I
Trừ	sub, subu	R
Thiết lập nếu nhỏ hơn	slt/sltu, slti/sltiu	R, I
Dịch trái	sll	R
Dịch phải	srl	R
Bitwise AND	and, andi	R, I
Bitwise OR	or, ori	R, I
Bitwise NOT	nor	R



Kiến trúc tập lệnh

Ví dụ: **add** \$t0, \$s1, \$s2

op (add)	\$s1	\$s2	\$t0	0	funct (add)
0	17	18	8	0	0x20
000000	10001	10010	01000	00000	100000

→ 00000010001100100100000000100000 hay 0x02324020



Sharing is learning

Nhóm lệnh số học và luận lý (3/7) – Loại R

Tập thanh ghi (trước)

...	...
\$t0	0x08
....	...
\$s1	0x1993
\$s2	0x2021
...	

Tập thanh ghi (sau)

...	...
\$t0	0x39B4
....	...
\$s1	0x1993
\$s2	0x2021
...	

add \$t0, \$s1, \$s2



Sharing is learning

Kiến trúc tập lệnh

Ví dụ: **andi** \$t0, \$s2, -8

op (andi)	\$s2	\$t0	-8
0x8	18	8	-8
001000	10010	01000	11111111111111000

→ 001000100100100011111111111111000 hay 0x2248FFF8



Sharing is learning

Kiến trúc tập lệnh

Tập thanh ghi (trước)

...	...
\$t0	0x08
....	...
\$s1	0x1993
\$s2	0x2021
...	

Tập thanh ghi (sau)

...	...
\$t0	0x2020
....	...
\$s1	0x1993
\$s2	0x2021
...	



andi \$t0, \$s2, -8

0xFFFFFFFF8

Kiến trúc tập lệnh

Ví dụ: `sll $t0, $s2, 2`

op (sll)	0	\$s2	\$t0	2	funct (sll)
0x0	0	18	8	2	0x0
000000	00000	10010	01000	00010	000000

→ 000000000000010010010000000100000000 hay 0x00124080



Sharing is learning

Kiến trúc tập lệnh

Tập thanh ghi (trước)

...	...
\$t0	0x08
....	...
\$s1	0x1993
\$s2	0x2021
...	

Tập thanh ghi (sau)

...	...
\$t0	0x8084
....	...
\$s1	0x1993
\$s2	0x2021
...	

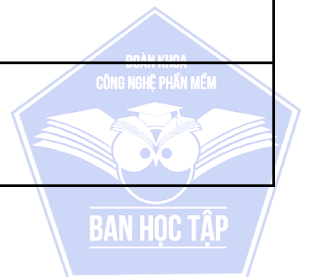


sll \$t0, \$s2, 2

Kiến trúc tập lệnh

- Nhóm lệnh truyền dữ liệu

Thao tác	MIPS	Định dạng
Nạp word	lw	I
Lưu word	sw	I
Nạp byte	lbu	I
Lưu byte	sb	I
Nạp nửa cao	lui	I



Sharing is learning

Kiến trúc tập lệnh

Ví dụ: `lw $t0, -8($s2)`

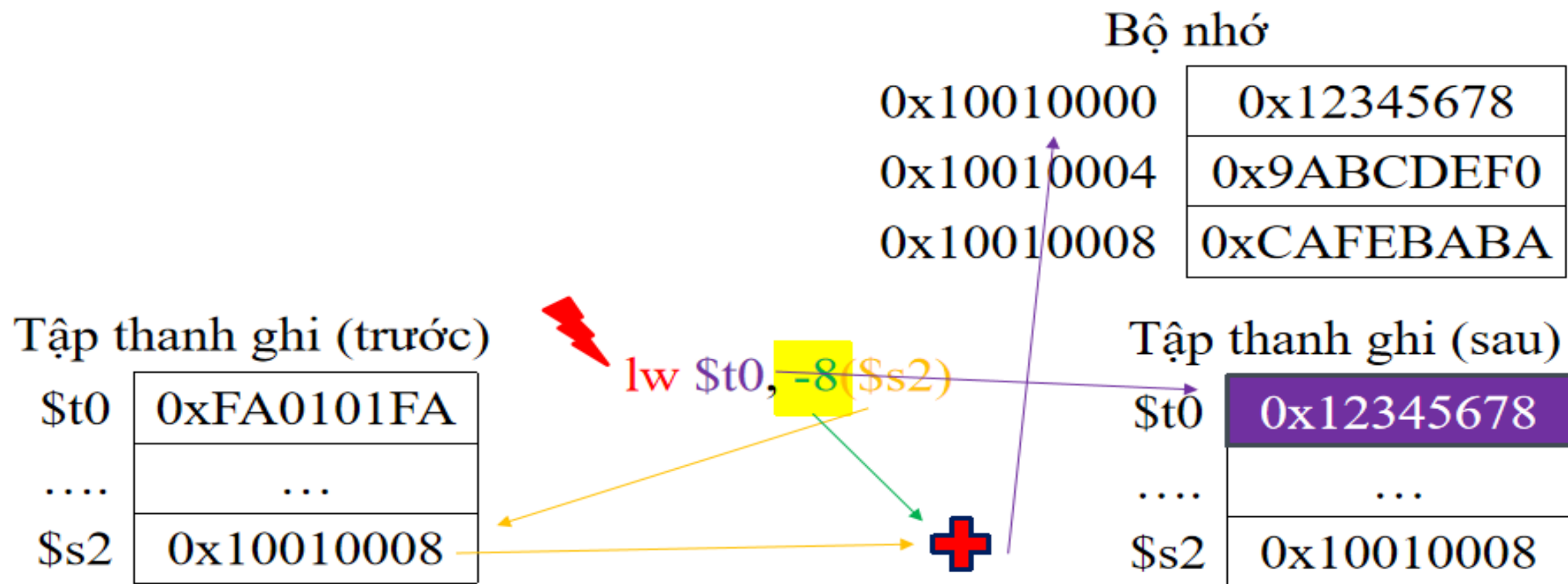
op (lw)	\$s2	\$t0	-8
0x23	18	8	-8
100011	10010	01000	11111111111111000

→ 100011100100100011111111111111000 hay 0x4E48FFF8



Sharing is learning

Kiến trúc tập lệnh



Sharing is learning

Kiến trúc tập lệnh

Bộ nhớ (trước)

0x10010000	0x12345678
0x10010004	0x9ABCDEF0
0x10010008	0xCAFEBAABA

Bộ nhớ (sau)

0x10010000	0xFA0101FA
0x10010004	0x9ABCDEF0
0x10010008	0xCAFEBAABA

Tập thanh ghi

\$t0	0xFA0101FA
....	...
\$s2	0x10010008

 **sw \$t0, -8(\$s2)**

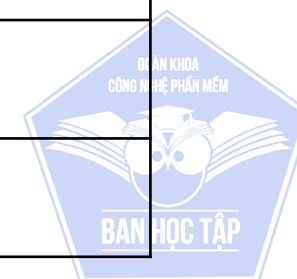


Sharing is learning

Kiến trúc tập lệnh

- Nhóm lệnh điều khiển

Thao tác	MIPS	Định dạng
Nhảy nếu bằng	beq	I
Nhảy nếu khác	bne	I
Nhảy (nhẫn)	j	J
Nhảy thanh ghi	jr	R
Nhảy và liên kết	jal	J



Kiến trúc tập lệnh

Ví dụ: `slti $t0, $s2, -8`

op (slti)	\$s2	\$t0	-8
0xA	18	8	-8
001010	10010	01000	11111111111111000

→ 001010100100100011111111111111000 hay 0x2A48FFF8



Sharing is learning

Kiến trúc tập lệnh

Tập thanh ghi (trước)

\$t1	0x3
\$t0	0x08
....	...
\$s1	0x1993
\$s2	0x2021
...	

sltiu \$t1, \$s2, -8

slti \$t0, \$s2, -8

0xFFFFFFFF8

Tập thanh ghi (sau)

\$t1	0x1
\$t0	0x0
....	...
\$s1	0x1993
\$s2	0x2021
...	

Kiến trúc tập lệnh

Ví dụ: 0x400000: **beq** \$s2, \$t0, PASS

...

0x400200: PASS:

op (beq)	\$s2	\$t0	PASS = PC + 4 + {extS(imm) << 2}
000100	10010	01000	0000000001111111

→ 00010010010010000000000001111111 hay 0x1248007F



Sharing is learning

Kiến trúc tập lệnh

PC (hiện tại) 0x400000

PC (sau) 0x400004

Tập thanh ghi

...	...
\$t0	0x08
....	?
\$s2	0x2021

0x400000: beq \$s2, \$t0, PASS

0x400200: PASS:

$$0x2021 - 0x08 = 0x2019$$
$$0x2019 \neq 0 \rightarrow PC = PC + 4$$



Sharing is learning

Kiến trúc tập lệnh

PC (hiện tại) 0x400000

PC (sau) 0x400200

Tập thanh ghi 0x400000: bne \$s2, \$t0, PASS

...	...
\$t0	0x08
....	?
\$s2	0x2021

...
0x400200: PASS:

$$0x2021 - 0x08 = 0x2019$$
$$0x2019 \neq 0 \rightarrow PC = 0x400200$$



Sharing is learning

Kiến trúc tập lệnh

0x400000: **j** **PASS**

...

0x400200: **PASS:**

0x400204: **jr** **\$t0**

...

0x400400: **FAIL:**

0x400404: **jal** **END**

...

0x400800: **END:**

PC

Hiện tại	Sau
0x400000	0x400200

PC

Hiện tại	Sau
0x400204	\$t0

PC

Hiện tại	Sau
0x400404	0x400800
\$ra	0x400408

x



Sharing is learning

Chương trình hợp ngữ-Cấu trúc chương trình

Chú thích bằng đầu bằng ký tự

.data # Khai báo dữ liệu ngay sau hàng này

 # Khai báo dữ liệu

.text # Viết chương trình ngay sau hàng này

main: # Nhãn thể hiện bắt đầu chương trình

 # Viết chương trình



Sharing is learning

Chương trình hợp ngữ-Cấu trúc chương trình

- Ký số: Hệ thập phân (17), Hệ thập lục phân thêm tiền tố 0x (0x17)
- Ký tự: Đặt trong cặp nháy đơn ('c')
- Chuỗi: Đặt trong cặp nháy kép ("PH002")

[<nhãn>:] .<kiểu dữ liệu> <danh sách giá trị>

Ví dụ:

```
var1: .word 3          # biến nguyên var1 có kích thước 1 word
arr1: .byte 'a', 'k'   # mảng arr1 có 2 phần tử, mỗi phần tử 1 byte
arr2: .space 40        # mảng arr2 có kích thước 40 byte liên tục
str1: .asciiz "ahihi"  # chuỗi str1 có kết thúc bởi null
```



Sharing is learning

Biên dịch chương trình

Giới thiệu trình biên dịch

Trình biên dịch có chức năng chuyển chương trình được viết bởi ngôn ngữ lập trình cấp cao thành chương trình hợp ngữ.



Sharing is learning

Biên dịch chương trình

Trình biên dịch hợp ngữ

Trình biên dịch hợp ngữ có chức năng chuyển chương trình được viết bởi hợp ngữ thành mã máy:

- Mã máy là các chuỗi bit (0, 1) có thể được thực thi trên máy tính.
- Có thể sử dụng lệnh giả (pseudo instruction) để viết chương trình hợp ngữ nhằm đơn giản hơn cho lập trình viên.



Sharing is learning

Biên dịch chương trình

Trình biên dịch hợp ngữ

Ví dụ: Biên dịch chương trình được viết bằng hợp ngữ MIPS bên cạnh sang mã máy: **add** \$s2, \$t1, \$t2

Giải:

Add

add R R[rd] = R[rs] + R[rt]

(1) 0 / 20_{hex}

opcode	rs	rt	rd	shamt	funct
000000	01001	01010	10010	00000	010100

\$t0-\$t7	8-15
\$s0-\$s7	16-23

➤ Mã máy: **000000****01001****01010****10010****00000****010100**



Sharing is learning

Biên dịch chương trình

Biên dịch ngược

Biên dịch ngược là quá trình khôi phục mã máy thành chương trình hợp ngữ.

Ví dụ: Mã máy: 000000 00101 01111 10010 00000 100000

➤ MIPS: add \$s2, \$a1, \$t7



Sharing is learning

Bộ xử lý

- Vi kiến trúc
 - Kiến trúc Máy tính bao gồm 3 thành phần chính:
 - Kiến trúc tập lệnh (ISA)
 - Vi kiến trúc (Tổ chức Phần cứng Máy tính)
 - Hệ thống Máy tính



Sharing is learning

Bộ xử lý

- Vi kiến trúc
 - Về chức năng, Vi kiến trúc là một tổ chức phần cứng dùng để hiện thực tập lệnh của một máy tính.
 - Về cấu tạo, Vi kiến trúc được chia thành 2 khối:
 - Khối đường dữ liệu (datapath): Thực thi lệnh
 - Khối điều khiển (control unit): Điều khiển datapath hoạt động



Sharing is learning

Bộ xử lý

- Chu kỳ thực thi lệnh



- Bộ nhớ lệnh
- Tập thanh ghi
- ALU
- Bộ nhớ dữ liệu
- Tập thanh ghi
- PC
- Bộ nhớ dữ liệu
- Bộ so sánh
- Mở rộng dấu

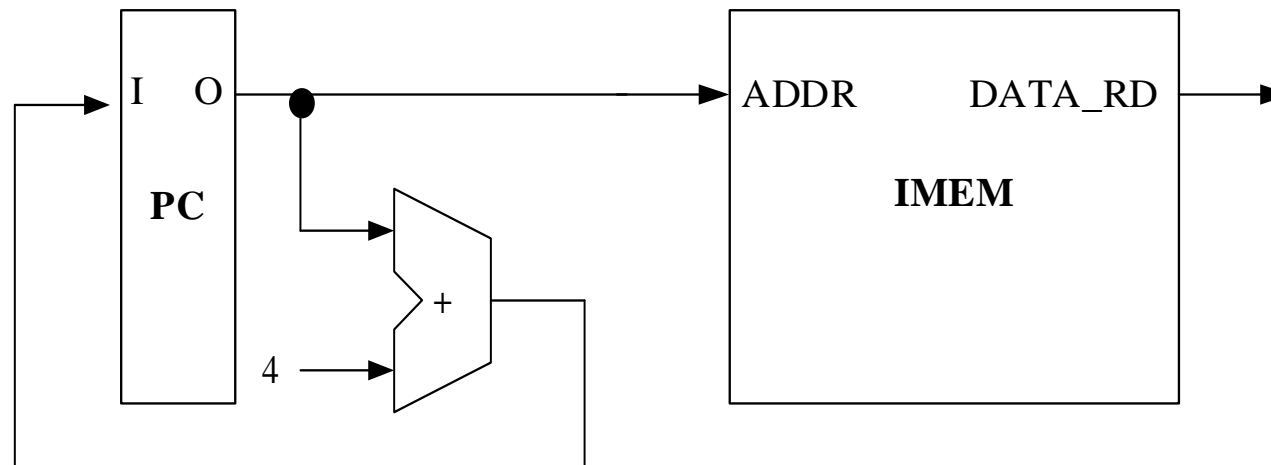


Sharing is learning

Bộ xử lý

I. Nạp lệnh

- Lệnh cần nạp lưu trong Bộ nhớ lệnh
- Địa chỉ của lệnh cần nạp lưu trong thanh ghi PC
- Tăng PC lên 4 để chuẩn bị nạp lệnh tiếp theo → cần thêm bộ cộng

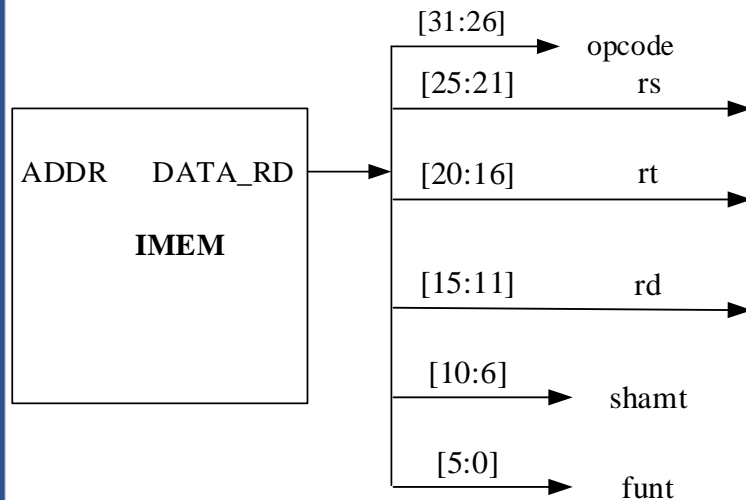


Sharing is learning

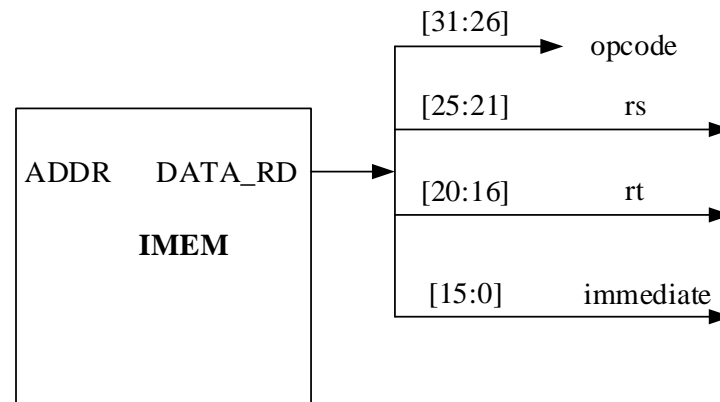
Bộ xử lý

II. Giải mã lệnh – Định dạng lệnh

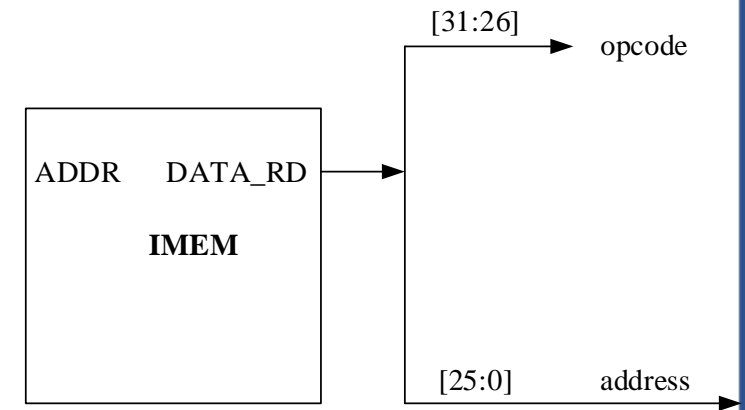
- Dựa vào opcode để xác định định dạng lệnh



Định dạng R



Định dạng I



Định dạng J

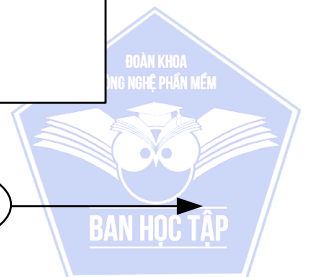
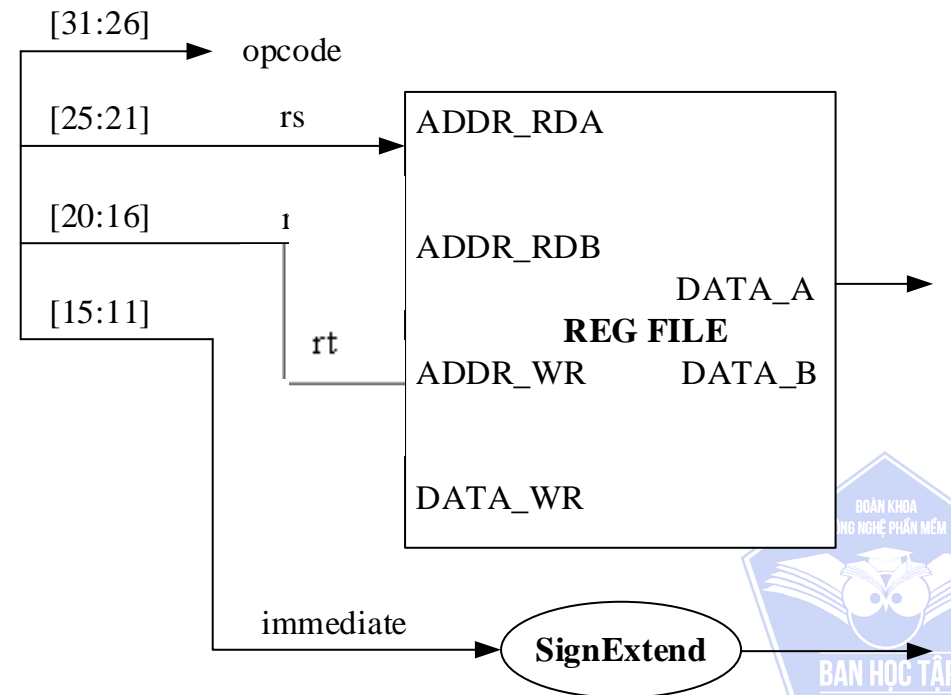
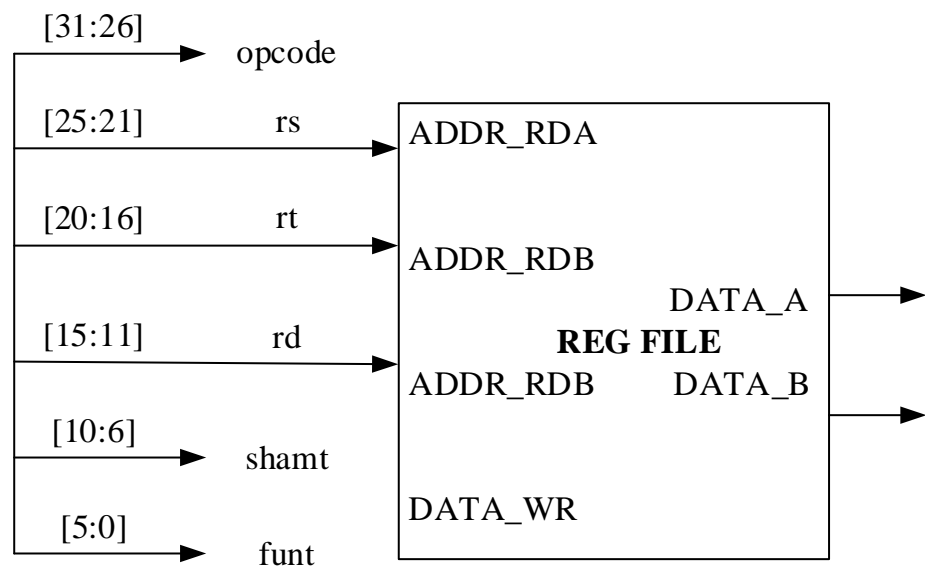


Sharing is learning

Bộ xử lý

II. Giải mã lệnh – Nạp toán hạng

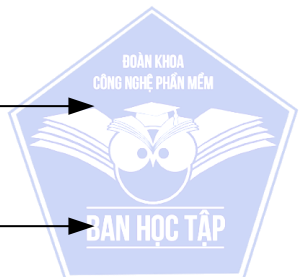
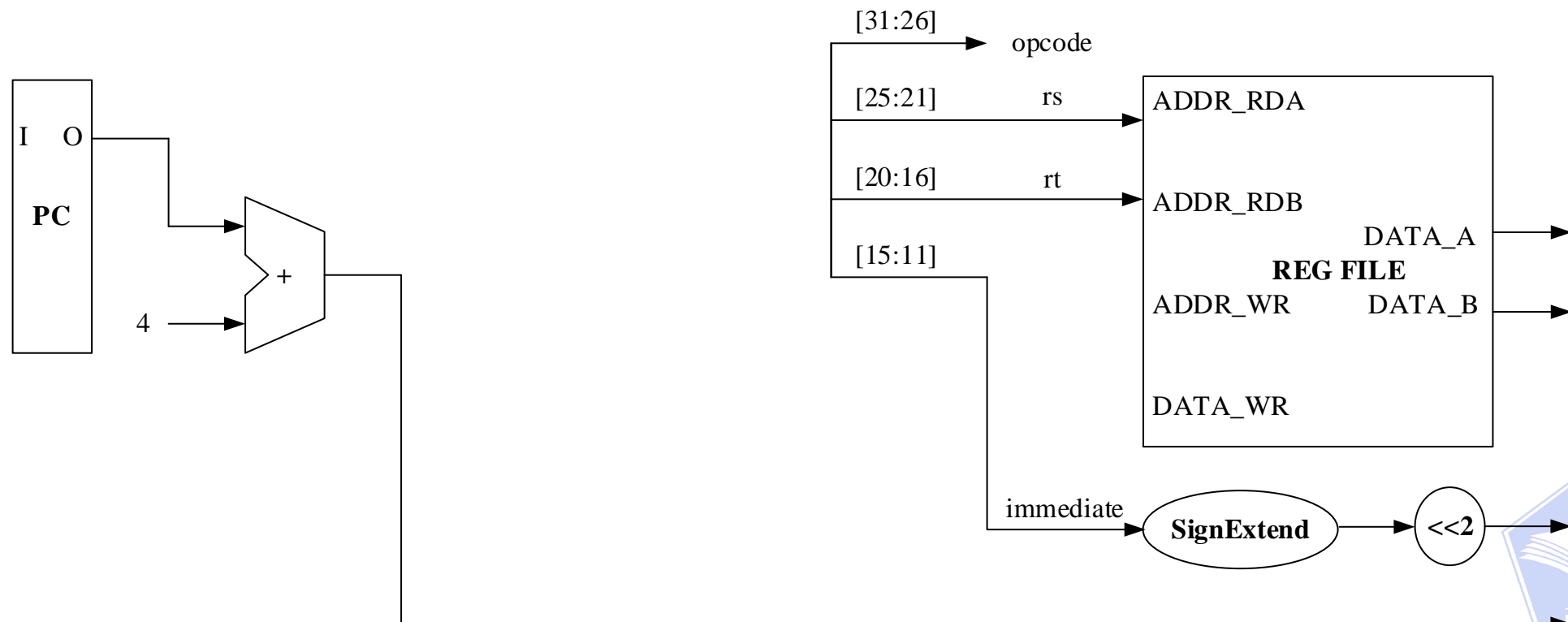
- Dựa vào định dạng lệnh mà nạp toán hạng tương ứng



Sharing is learning

Bộ xử lý

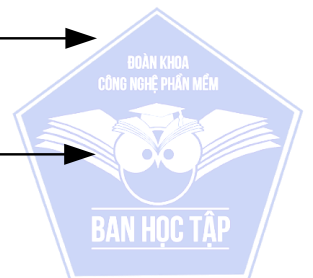
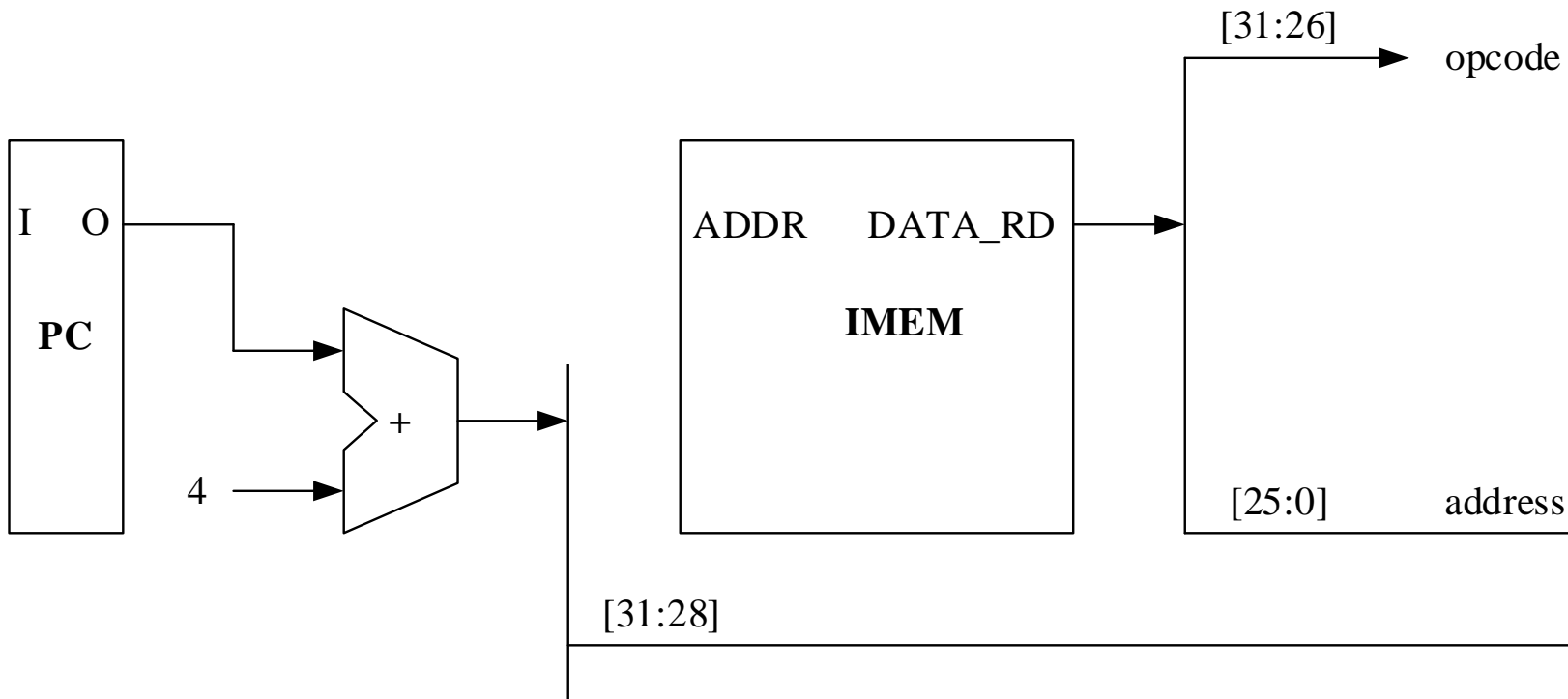
II. Giải mã lệnh – Nạp toán hạng: beq/bne



Sharing is learning

Bộ xử lý

II. Giải mã lệnh – Nạp toán hạng: j

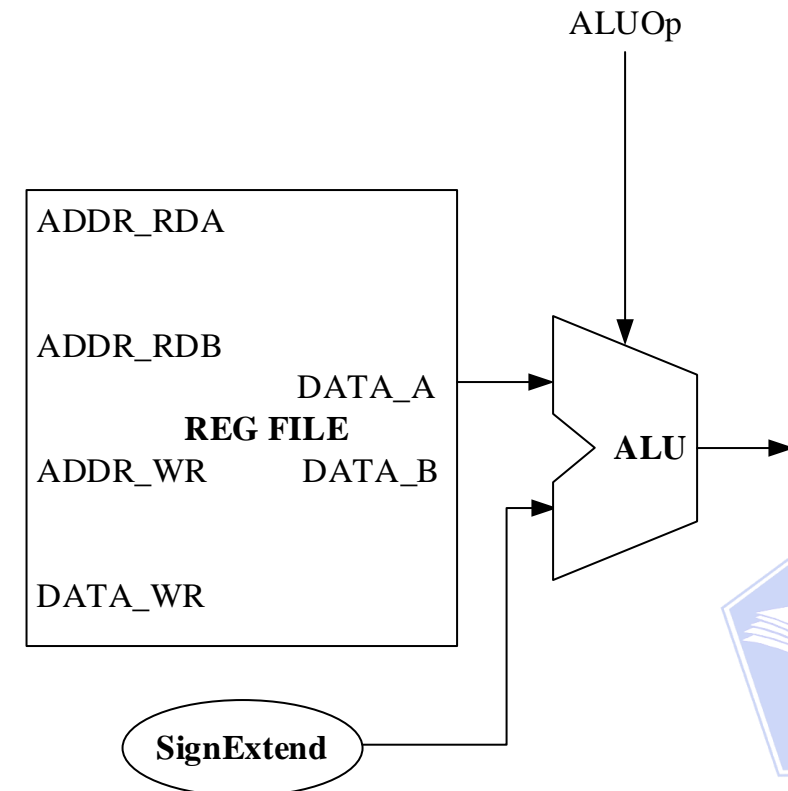
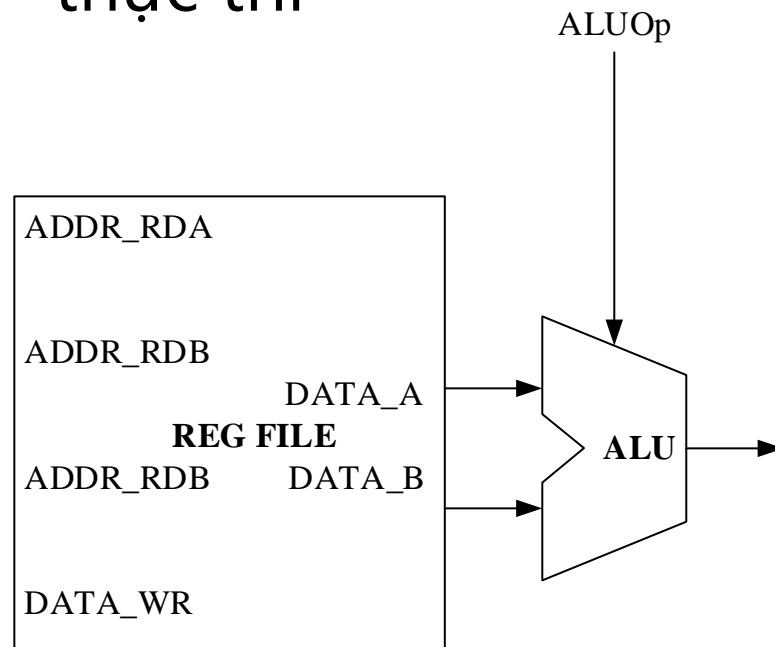


Sharing is learning

Bộ xử lý

III. Thực thi

- Dựa vào opcode và funct mà quyết định thao tác gì sẽ được thực thi

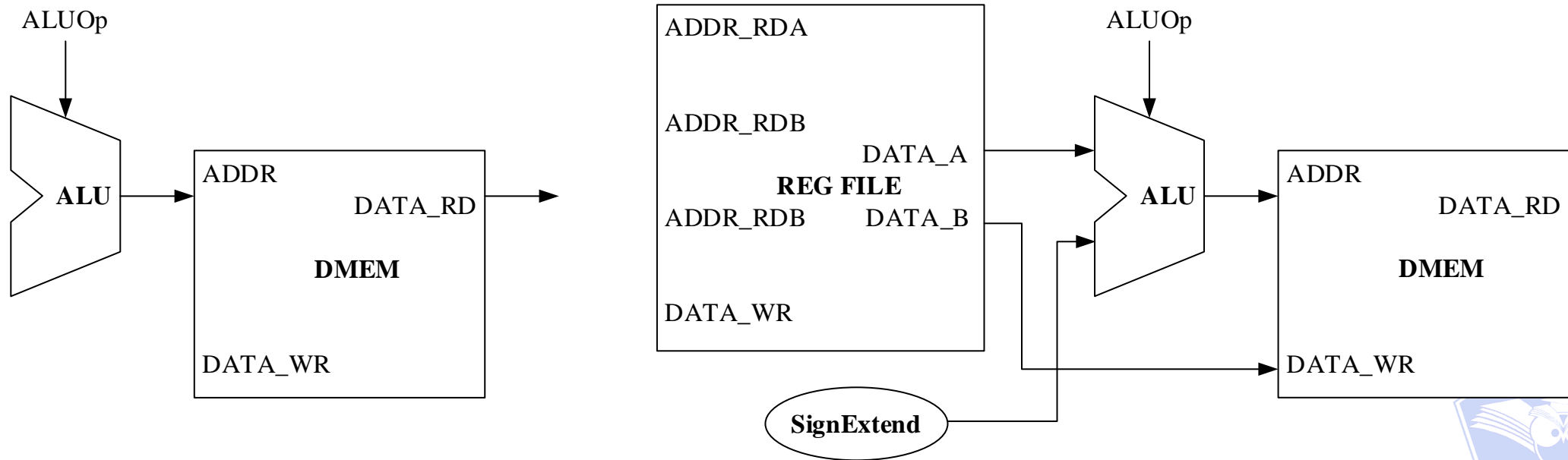


Sharing is learning

Bộ xử lý

IV. Truy xuất bộ nhớ

- Đọc dữ liệu từ bộ nhớ dữ liệu đối với các lệnh nạp
- Ghi dữ liệu tới bộ nhớ dữ liệu đối với các lệnh lưu

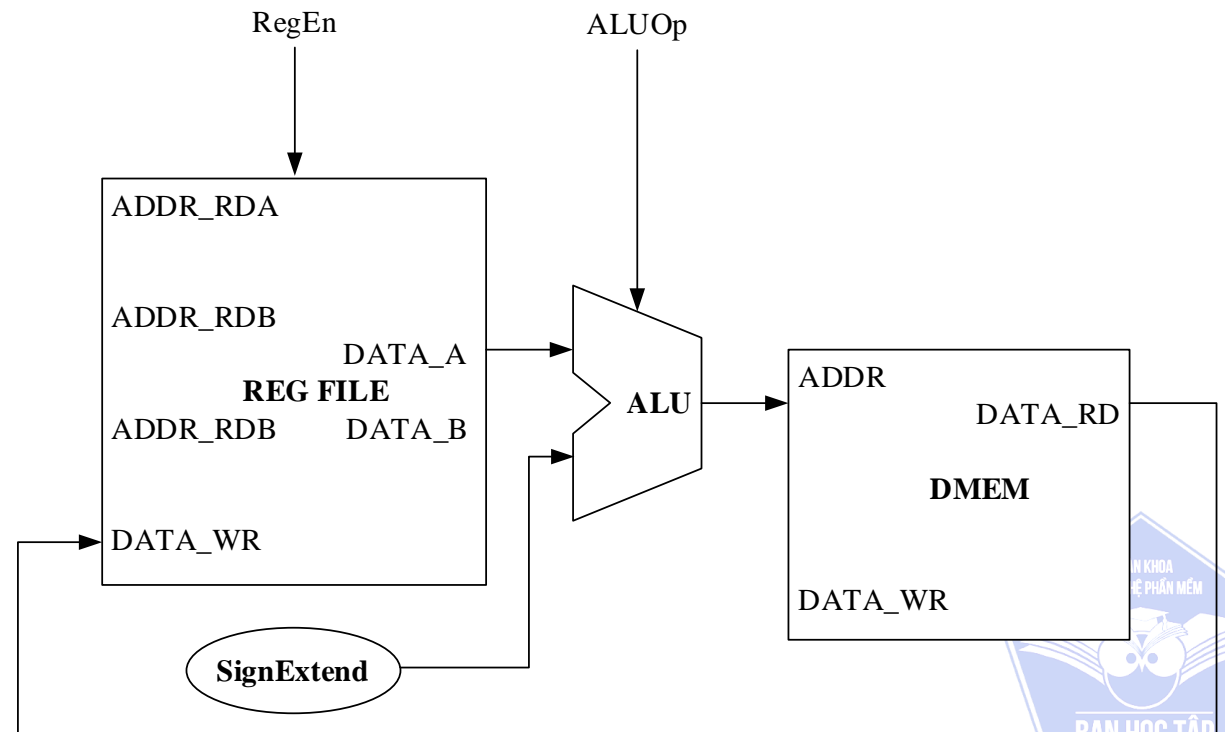
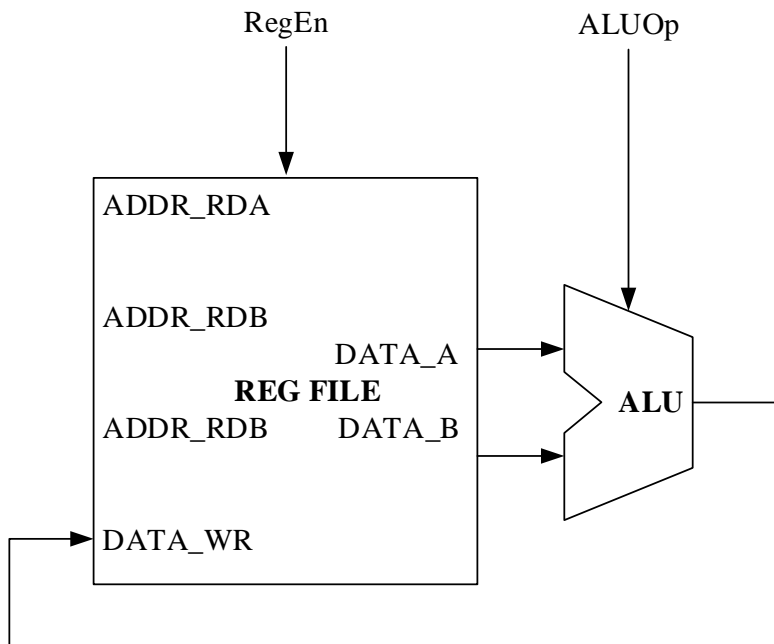


Sharing is learning

Bộ xử lý

V. Lưu kết quả

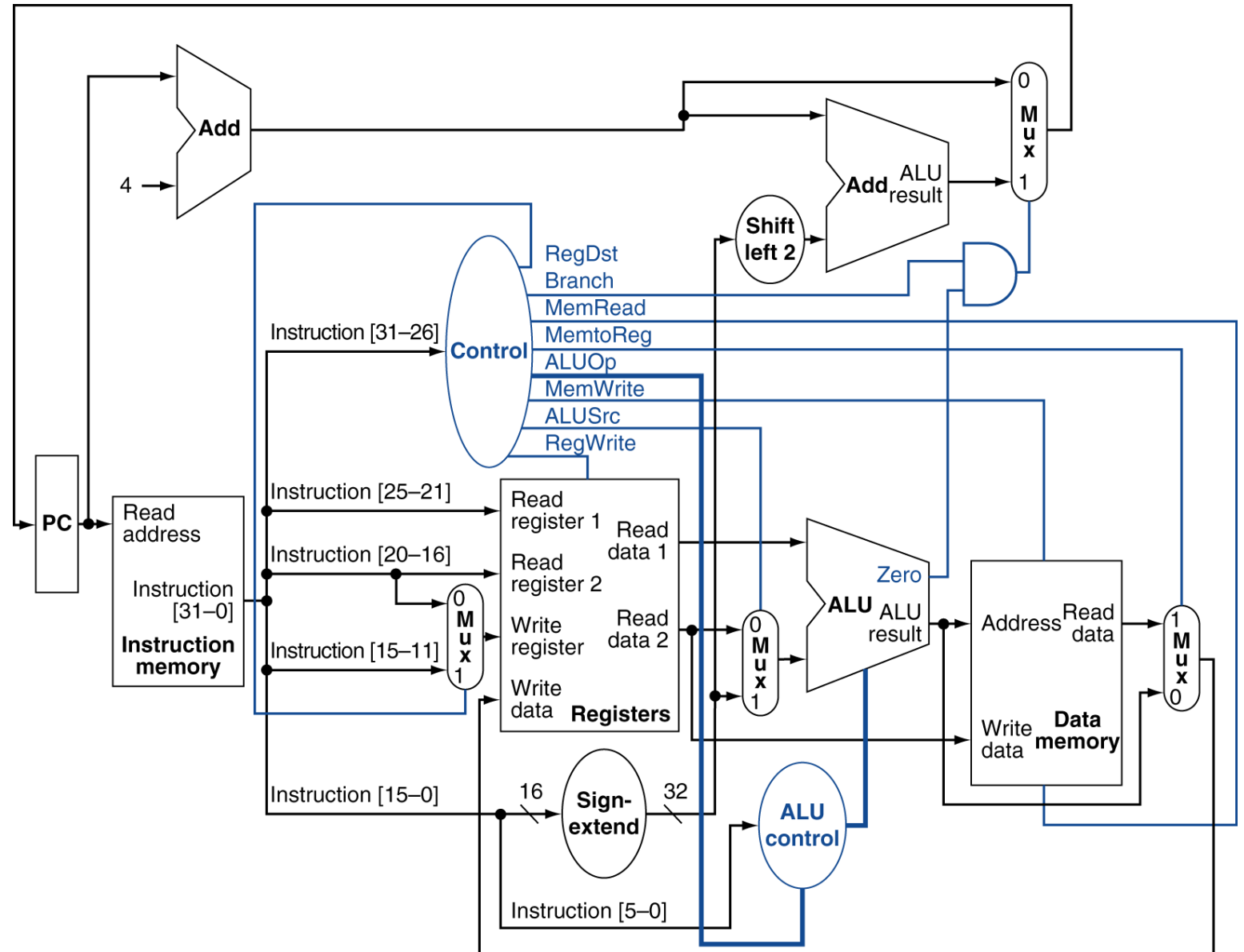
- Có thể ghi dữ liệu về lại Tập thanh ghi



Bộ xử lý

- Khối điều khiển

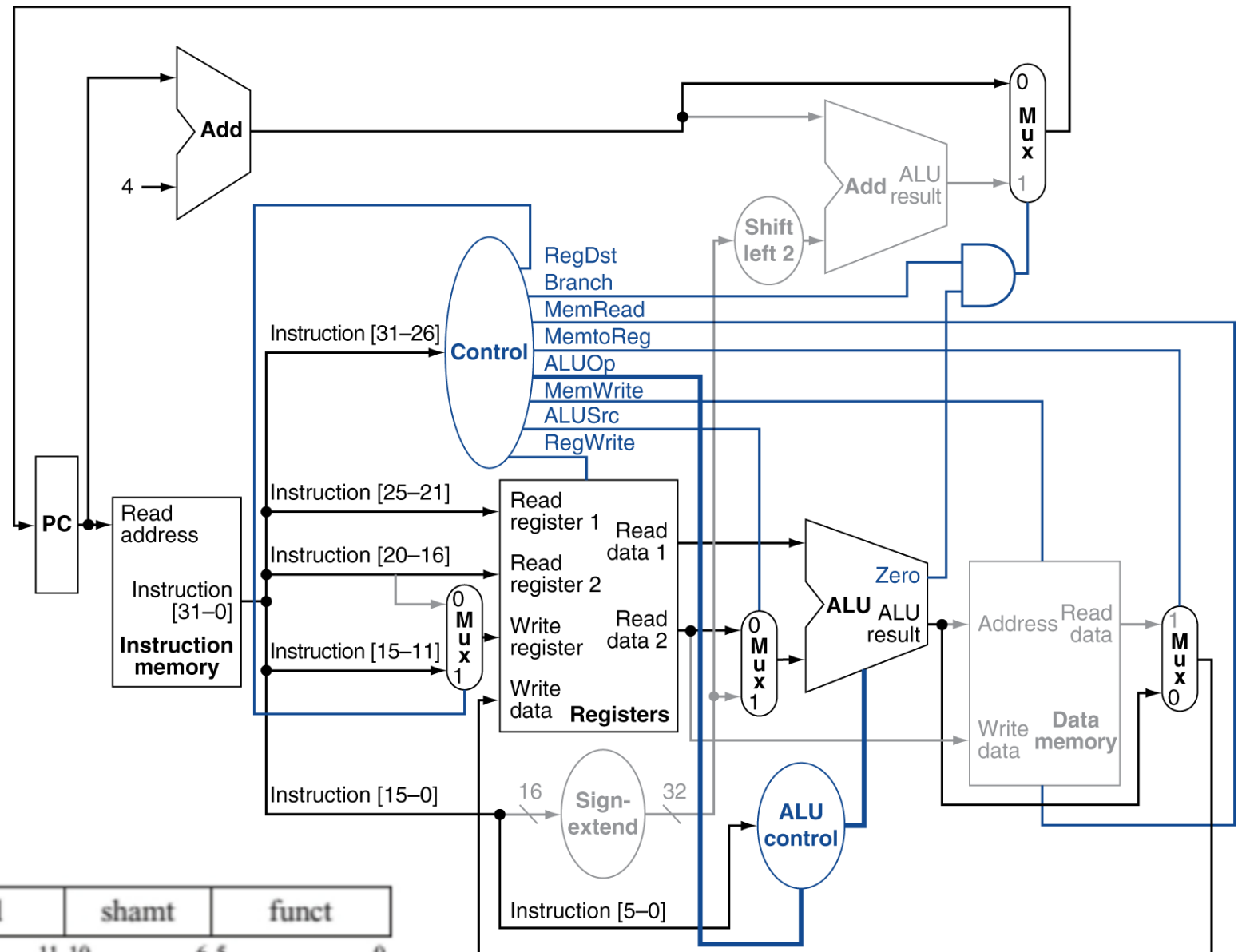
➤ Khối điều khiển (Control) là một mạch tổ hợp điều khiển datapath hoạt động



Bộ xử lý

- Khối điều khiển
 - Loại R

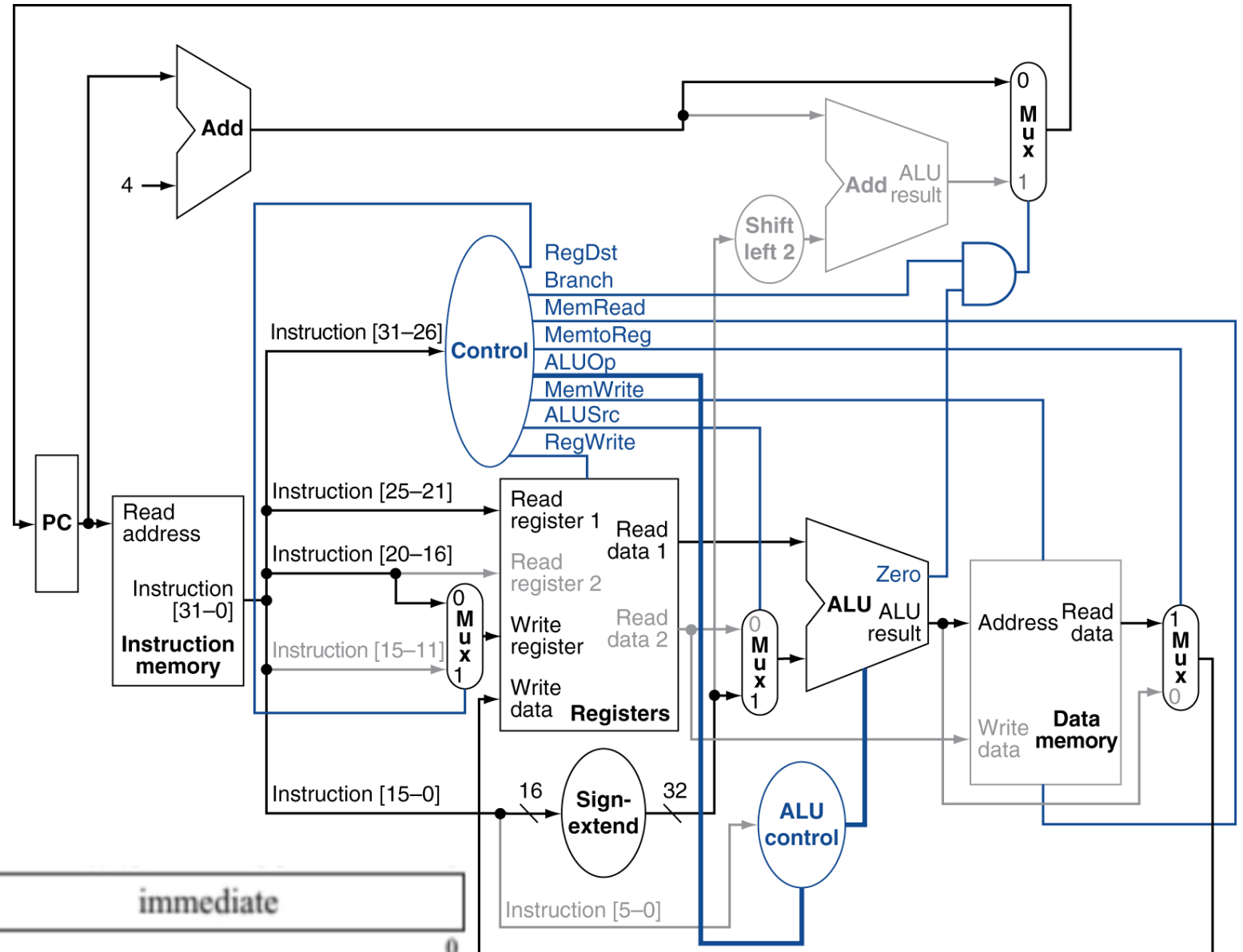
RegDst	1
Branch	0
MemRead	0
MemtoReg	0
ALUOp	10
MemWrite	0
ALUSrc	0
Regwrite	1



Bộ xử lý

- Khối điều khiển
 - lw

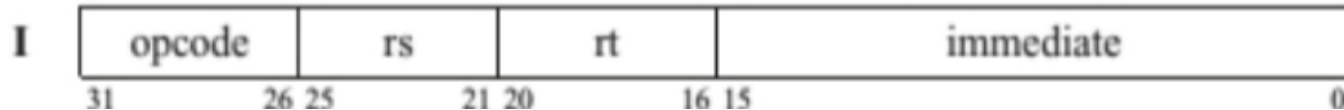
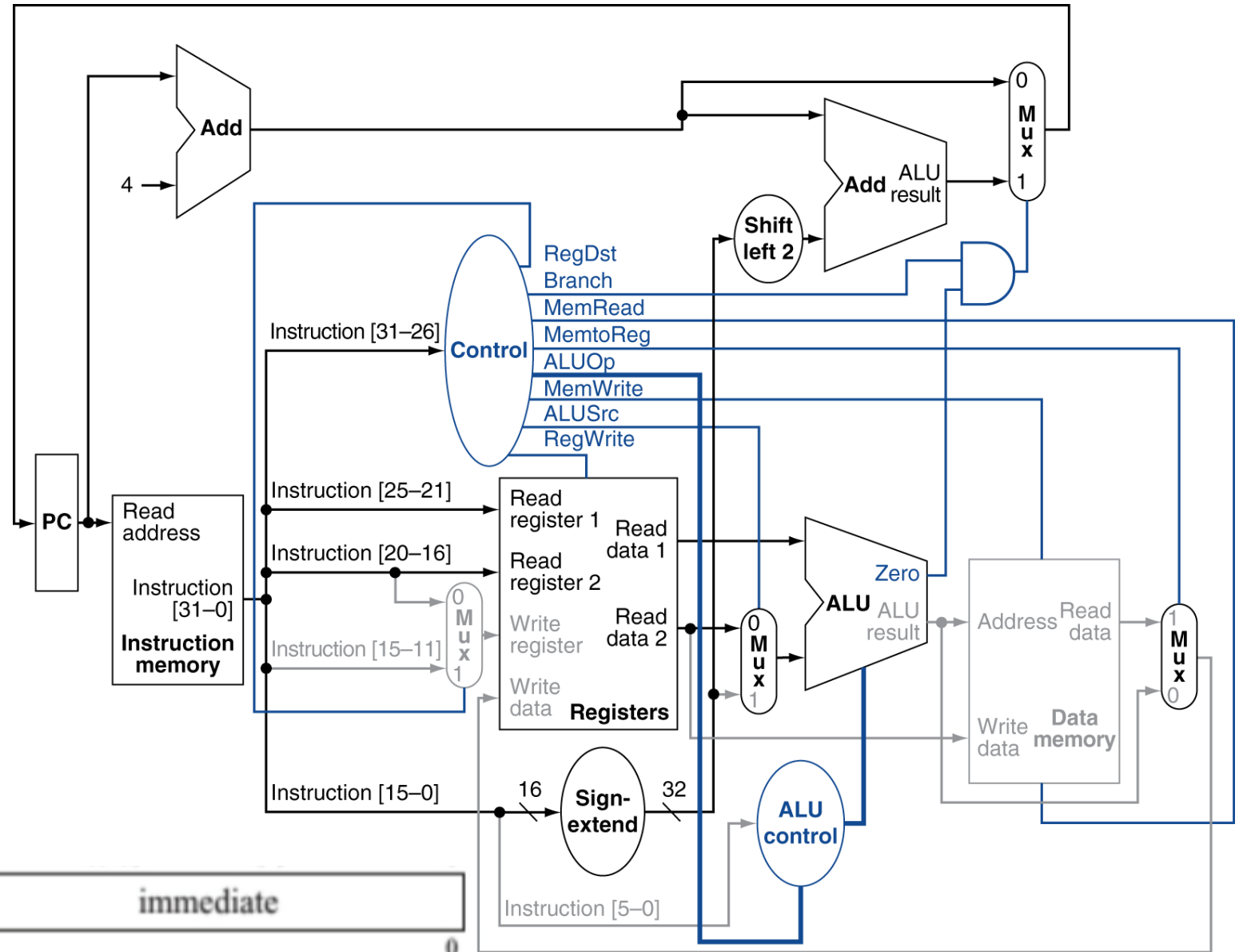
RegDst	0
Branch	0
MemRead	1
MemtoReg	1
ALUOp	00
MemWrite	0
ALUSrc	1
Regwrite	1



Bộ xử lý

- Khối điều khiển
 - beq

RegDst	X
Branch	1
MemRead	0
MemtoReg	X
ALUOp	01
MemWrite	0
ALUSrc	0
Regwrite	0



Bộ xử lý

- Khối điều khiển – ALU Control
 - ALU chỉ cần thực hiện các phép toán: Cộng, trừ, AND, OR, Thiết lập nếu nhỏ hơn
 - ALU Control là một mạch tổ hợp điều khiển ALU có bảng chân trị sau:

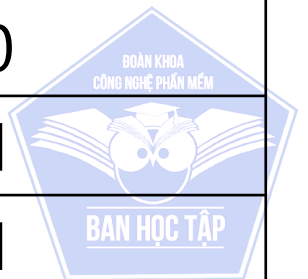


Sharing is learning

Bộ xử lý

- Khối điều khiển – ALU Control

Opcode	ALUOp	Lệnh	Funct	Phép toán ALU	ALU Control
lw	00	Nạp word	XXXXXX	Cộng	0010
sw	00	Lưu word	XXXXXX	Cộng	0010
beq	01	Nhảy nếu bằng	XXXXXX	Trừ	0110
Loại R	10	Cộng	100000	Cộng	0010
		Trừ	100010	Trừ	0110
		And	100100	And	0000
		Or	100101	Or	0001
		slt	101010	slt	0111



Sharing is learning

Hiệu suất máy tính

Các tiêu chí đánh giá hiệu suất

1. Thời gian thực thi và hiệu suất

Thời gian thực thi: tổng thời gian để hoàn thành một tác vụ (truy cập bộ nhớ, ổ đĩa,...).

Hiệu suất máy tính: số lượng tác vụ hoàn thành trong một đơn vị thời gian.

$$\text{Hiệu suất} = \frac{1}{\text{Thời gian thực thi}}$$



Sharing is learning

Hiệu suất máy tính

Các tiêu chí đánh giá hiệu suất

2. Clock

- Máy tính cần một clock để xác định khi nào một thao tác được thực hiện trong phần cứng.
- Khối tạo ra các khoảng thời gian định thời cho máy tính làm việc này được gọi là khối tạo clock.
- Hai khái niệm liên quan đến clock:
 - + Chu kỳ (Clock cycle)
 - + Tần số (Clock rate hoặc clock frequency)

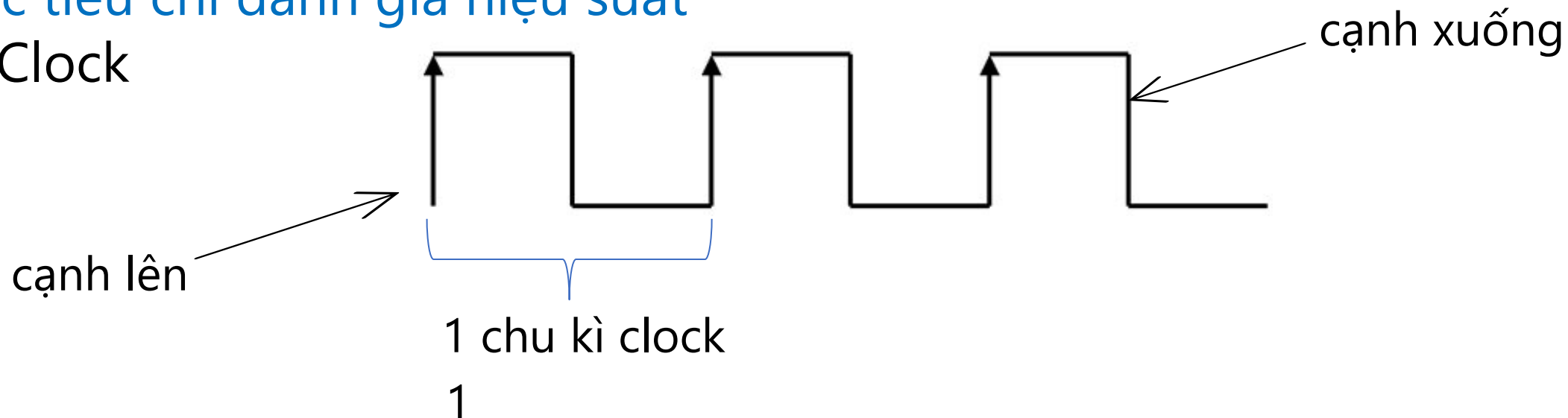


Sharing is learning

Hiệu suất máy tính

Các tiêu chí đánh giá hiệu suất

2. Clock



$$\text{Tần số clock} = \frac{1}{\text{Chu kì clock}}$$



Sharing is learning

Hiệu suất máy tính

Các tiêu chí đánh giá hiệu suất

3. CPI

CPI (cycles per instruction) là số chu kì clock trung bình trên mỗi lệnh của một chương trình hoặc đoạn chương trình.

Tổng số chu kì clock = Tổng số lệnh * CPI

Thời gian thực thi = Tổng số chu kì clock * chu kì clock

$$= \frac{\text{Tổng số chu kì clock}}{\text{Tần số clock}}$$



Sharing is learning

Hiệu suất máy tính

Các tiêu chí đánh giá hiệu suất

3. CPI

Ví dụ:

Máy tính A có chu kỳ clock = 200ps, CPI = 2, máy tính B có tần số clock = 10 Ghz, CPI = 4.0. Biết cả 2 máy đều có cùng kiến trúc tập lệnh. Máy tính nào nhanh hơn?

$$T = 200ps = 200 \cdot 10^{-12}s \Rightarrow f = \frac{1}{T} = 5 \cdot 10^9 Hz = 5Ghz$$

$$t = \frac{n \cdot CPI}{f} \Rightarrow \frac{t_A}{t_B} = \frac{CPI_A}{CPI_B} \cdot \frac{f_B}{f_A} = \frac{2}{4} \cdot \frac{10}{5} = 1$$

\Rightarrow 2 máy nhanh như nhau



Sharing is learning

Hiệu suất máy tính

Các tiêu chí đánh giá hiệu suất

3. CPI

Một tập lệnh có thể có nhiều nhóm lệnh khác nhau

➤ Tính CPI trung bình

$$\text{Chu kì clock} = \sum_{i=1}^n (\text{CPI}_i * \text{Số lệnh trong nhóm } i)$$

$$\text{CPI}_{\text{trung bình}} = \frac{\text{Chu kì clock}}{\text{Tổng số lệnh}} = \sum_{i=1}^n \left(\text{CPI}_i * \frac{\text{Số lệnh trong nhóm } i}{\text{Tổng số lệnh}} \right)$$



Sharing is learning

Hiệu suất máy tính

Các tiêu chí đánh giá hiệu suất

3. CPI

Ví dụ: Bộ xử lý P1 có 2 lớp lệnh A và B. Với một chương trình có 100 lệnh lớp A và 200 lệnh lớp B, tổng số chu kì clock là 300 thì CPI trung bình là bao nhiêu?

$$\text{CPI}_{\text{trung bình}} = \frac{\text{Chu kì clock}}{\text{Tổng số lệnh}} = \frac{300}{100 + 200} = 1$$



Sharing is learning

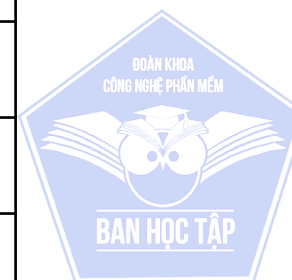
Hiệu suất máy tính

Các yếu tố ảnh hưởng đến hiệu suất

Thành phần của hiệu suất:

- Thời gian thực thi
- Tổng số lệnh
- Tần số
- CPI

Yếu tố	Tác động đến
Thuật toán	Tổng số lệnh, có thể có CPI
Ngôn ngữ lập trình	Tổng số lệnh, CPI
Trình biên dịch	Tổng số lệnh, CPI
Kiến trúc tập lệnh	Tổng số lệnh, tần số, CPI



Sharing is learning

Hiệu suất máy tính

Các kỹ thuật nâng cao hiệu suất

Giảm thời gian thực thi

- Tăng tần số clock
- Pipeline: thực thi đồng thời nhiều lệnh bằng cách chia chu kỳ thực thi lệnh thành các stage. Tại một thời điểm, một lệnh chỉ được thực thi một stage.
- Tiên đoán: dự đoán các lệnh nhảy có xảy ra hay không.
- Multicore: thực thi đồng thời nhiều chương trình bằng cách tăng số lượng bộ xử lý.
- Multithread: thực thi đồng thời nhiều tác vụ bằng cách tăng số lượng đơn vị xử lý.
- Phân cấp bộ nhớ: xử dụng bộ nhớ nhanh cho việc để thao tác với dữ liệu, bộ nhớ chậm để lưu trữ dữ liệu.



Sharing is learning

BAN HỌC TẬP CÔNG NGHỆ PHẦN MỀM

TRAINING CUỐI KỲ HỌC KỲ I NĂM HỌC 2023 – 2024



Sharing is learning

HẾT

**CẢM ƠN CÁC BẠN ĐÃ THEO DÕI
CHÚC CÁC BẠN CÓ KẾT QUẢ THI THẬT TỐT!**

 **BAN HỌC TẬP**

Khoa Công nghệ Phần mềm

Trường Đại học Công nghệ Thông tin

Đại học Quốc gia thành phố Hồ Chí Minh

 **CONTACT**

bht.cnpm.uit@gmail.com

fb.com/bhtcnpm

fb.com/groups/bht.cnpm.uit

QR Điểm danh

