TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN – ĐHQG HCM

KHOA KỸ THUẬT MÁY TÍNH

TỔ CHỨC VÀ CẤU TRÚC MÁY TÍNH 2

CHƯƠNG 6
KIẾN TRÚC TẬP LỆNH
(Phần 1)

PHAN ĐÌNH DUY

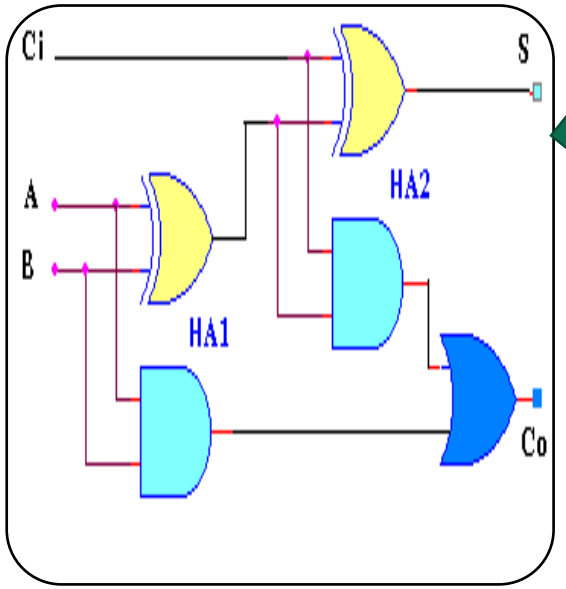
TP. Hồ Chí Minh, ngày 05 tháng 9 năm 2022

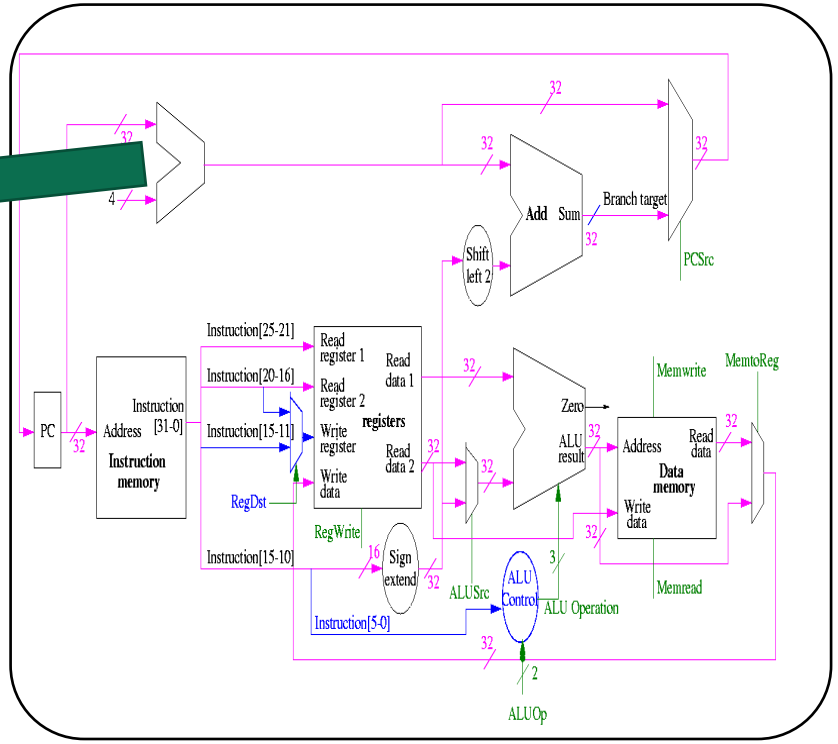
MỤC TIÊU CHƯƠNG

- Biết được các kiến trúc tập lệnh phổ biến
- Hiểu được những khái niệm cơ bản của kiến trúc MIPS
- Hiểu được các loại toán hạng trong lệnh của MIPS
- Hiểu được các định dạng lệnh của MIPS

NỘI DUNG

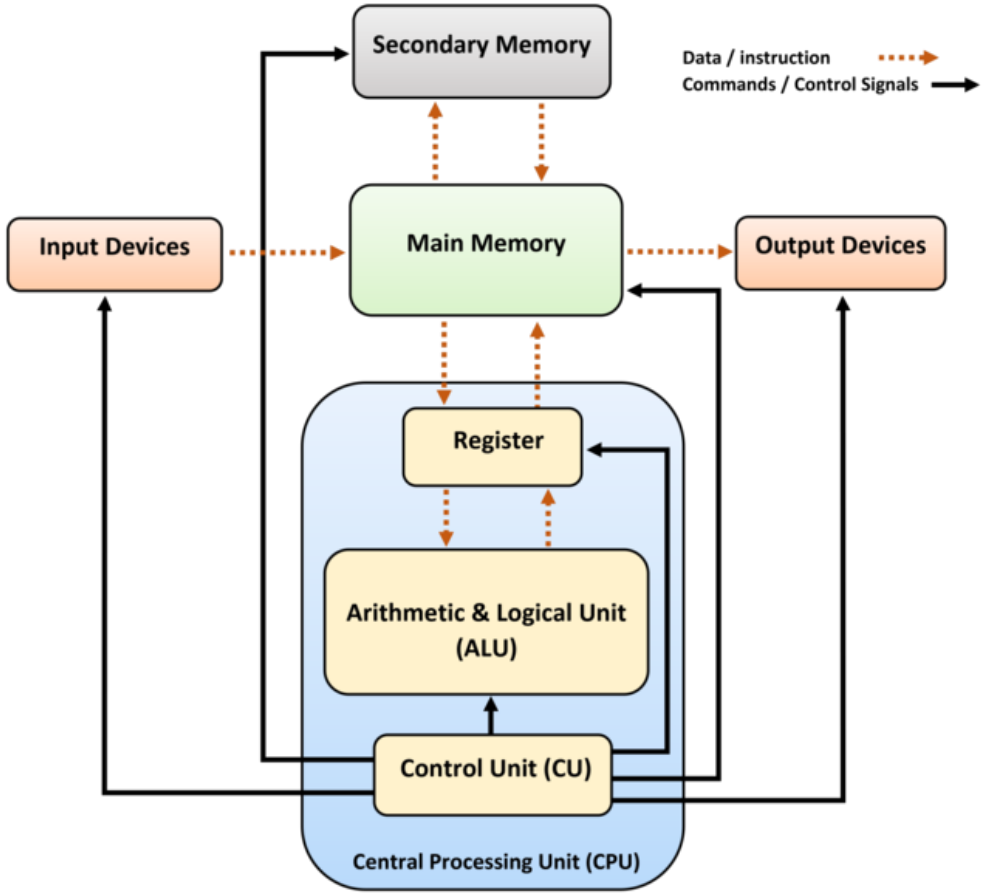
- Kiến trúc tập lệnh
- Toán hạng
- Định dạng lệnh
- Bài tập





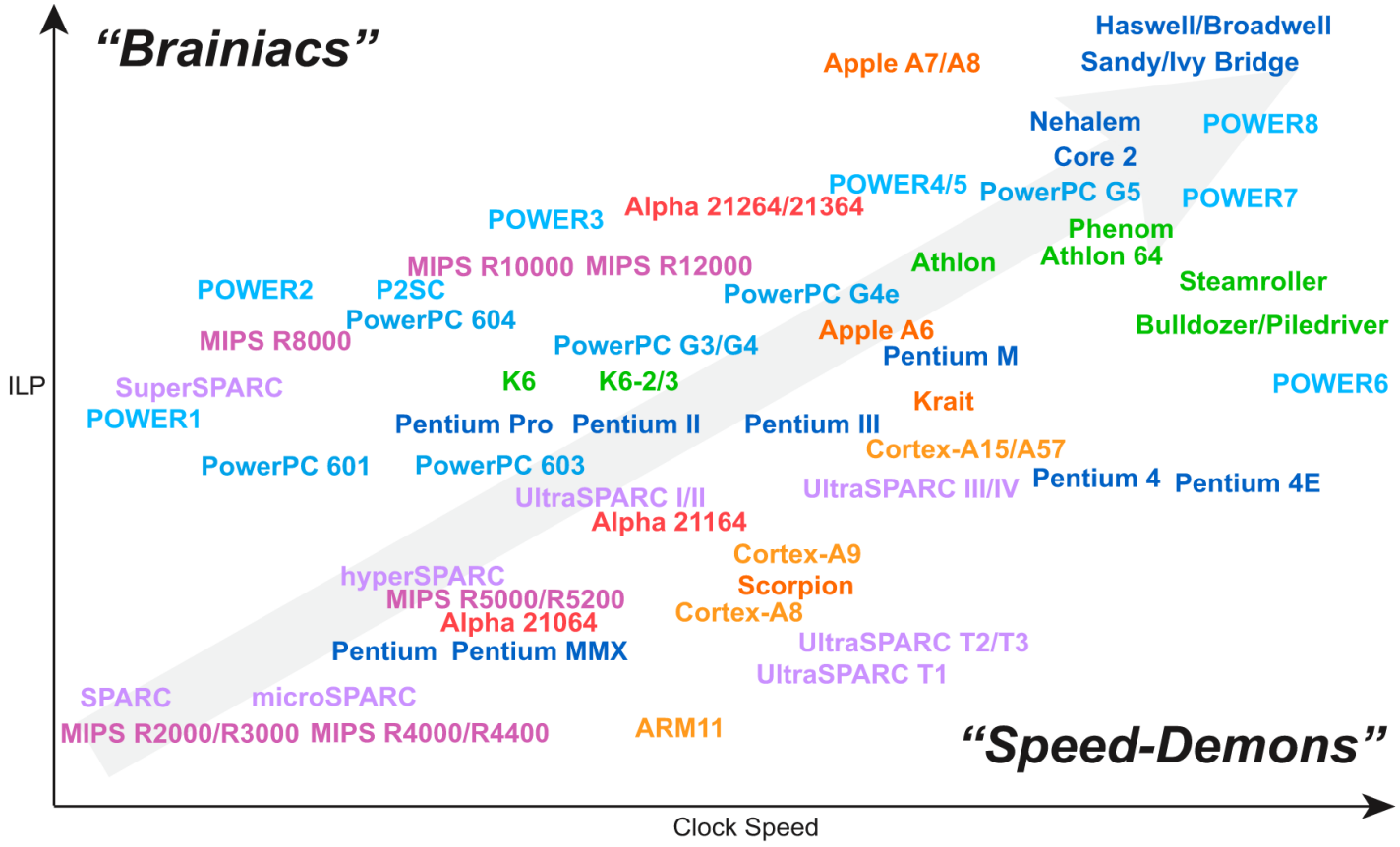
NỘI DUNG

- **Kiến trúc tập lệnh**
- Toán hạng
- Định dạng lệnh
- Bài tập



Kiến trúc Tập lệnh – Tổng quan

- ARM
- Intel (x86)
- MIPS



Kiến trúc Tập lệnh – Định nghĩa

- Lệnh (Instruction) là một chỉ dẫn để máy tính thực hiện công việc nào đó
  - Ví dụ: Lệnh ADD chỉ dẫn máy tính thực hiện phép toán cộng
- Tập lệnh (Instruction Set) là tập hợp các lệnh của máy tính
  - Tập lệnh quy định máy tính có thể làm những gì!
  - Những máy tính khác nhau sẽ có tập lệnh khác nhau!
  - NHƯNG! Các tập lệnh đều có điểm chung!!!
- Kiến trúc Tập lệnh = Tập lệnh + Biểu diễn lệnh

Kiến trúc Tập lệnh – Định nghĩa (2)

- Tập lệnh: Máy tính có thể làm những gì?

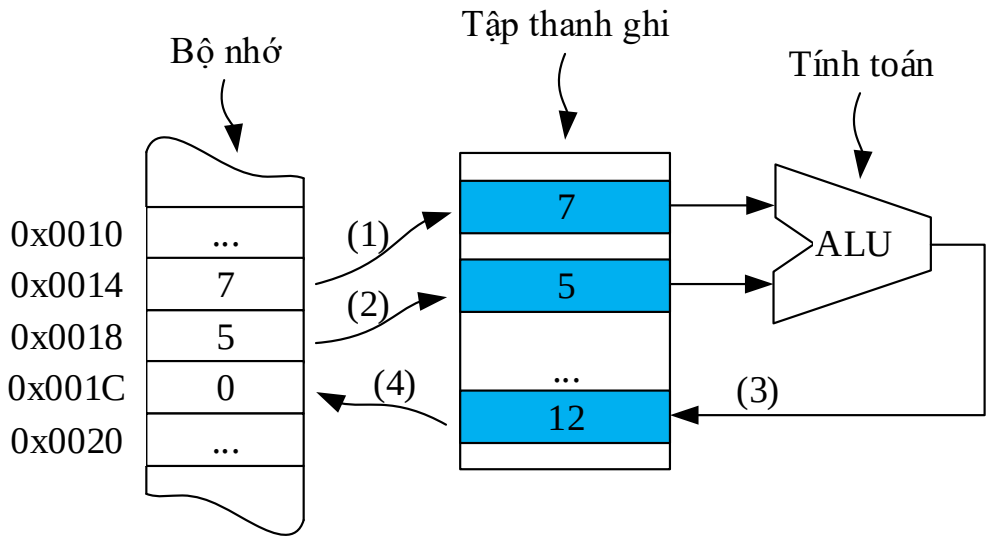
- Định dạng lệnh (biểu diễn lệnh): Mỗi lệnh được biểu diễn như thế nào?
  - Opcode (Operation Code): Mã lệnh (mã thao tác)
  - Toán hạng: Các toán hạng cần thiết để thực thi lệnh
  - Các trường khác

Kiến trúc Tập lệnh – Phân loại

- Ngăn xếp (stack)
- Bộ tích lũy (accumulator)
- Thanh ghi – Bộ nhớ (register–memory)
- Thanh ghi – thanh ghi / nạp – lưu (register-register/load-store)

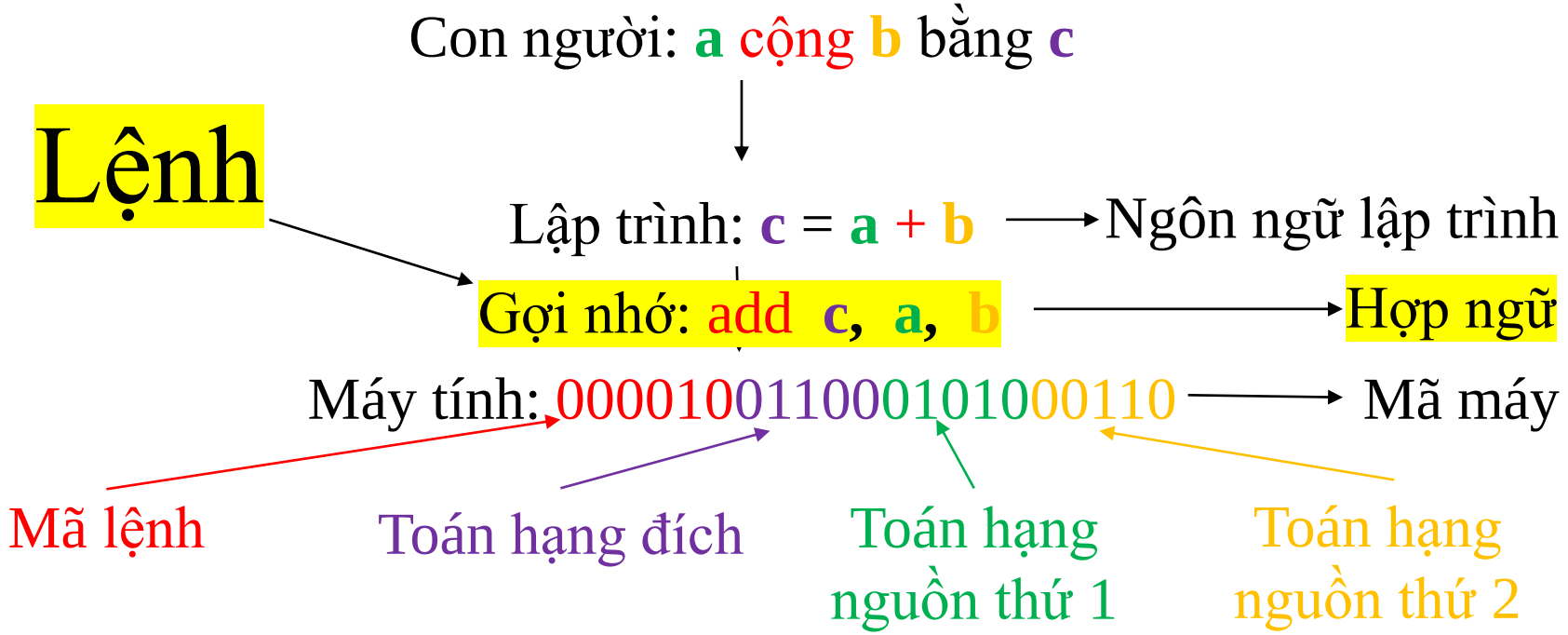
Stack	Accumulator	Register (register-memory)	Register (load-store)
Push A	Load A	Load R1,A	Load R1,A
Push B	Add B	Add R1,B	Load R2,B
Add	Store C	Store C,R1	Add R3,R1,R2
Pop C			Store C,R3

Kiến trúc Tập lệnh - 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

Kiến trúc Tập lệnh - Lệnh



Kiến trúc Tập lệnh – Quiz 1

- Đề xuất lệnh thực hiện thao tác trừ:
  - A trừ B bằng C

- Chuyển đổi lệnh trong lập trình C thành lệnh hợp ngữ

$$F = (A + B) – (C + D)$$

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

Kiến trúc Tập lệnh - Tập lệnh MIPS (2)

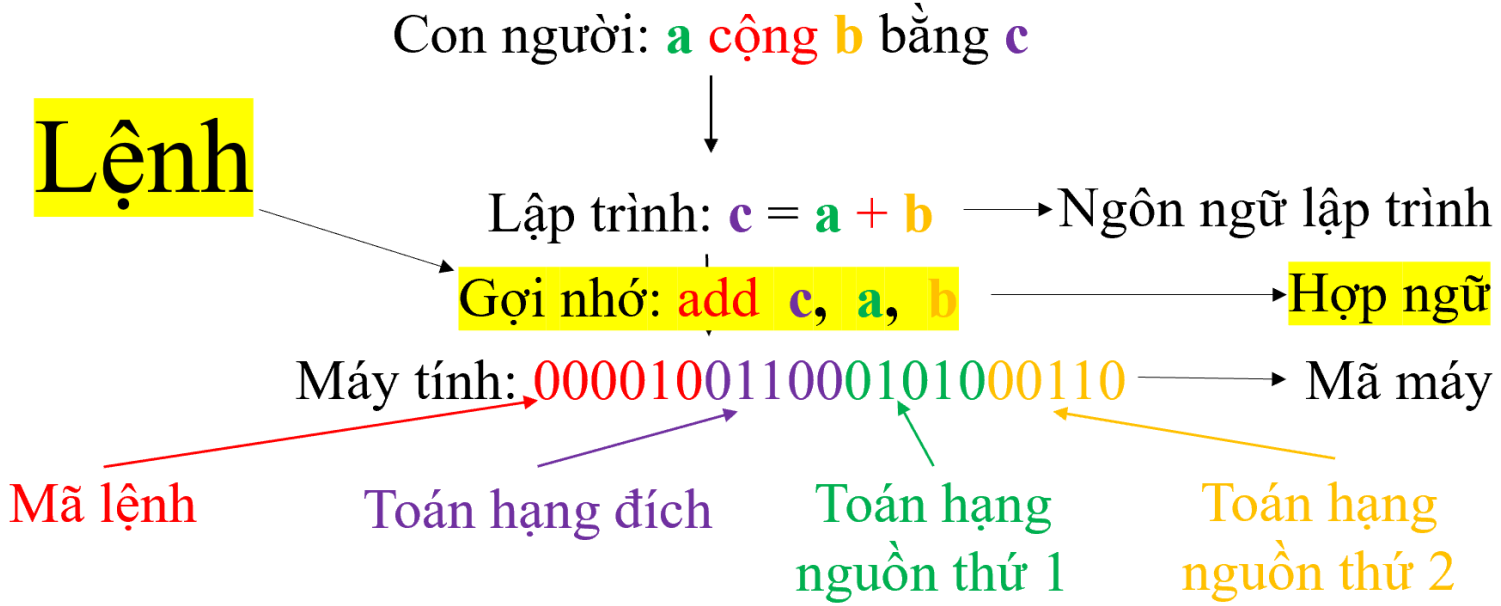
- Kiểu dữ liệu: Byte (8 bit), halfword (16 bit), word (32 bit)
- Chế độ định địa chỉ: 5 chế độ
- Toán hạng: Thanh ghi, số tức thời (bù 2), bộ nhớ
- Định địa chỉ theo byte

Kiến trúc Tập lệnh - Tập lệnh MIPS (3)

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)	$s1= Mem[$s2 + 20]	I
	Lưu word	sw $s1,20($s2)	Mem[s2 + 20]=$s1	I
Luận lý	NOR luận lý	nor $s1,$s2,$s3	$s1= ~($s2 | $s3)	R
	Dịch phải luận lý	srl $s1,$s2,10	$s1=$s2>>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

NỘI DUNG

- Kiến trúc tập lệnh
- **Toán hạng**
- Định dạng lệnh
- Bài tập



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

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

Toán hạng thanh ghi (2)

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

Toán hạng thanh ghi (3)

Số	Tên	Giá trị
0	$zero	0x0
1	$at	
…	…	
11	$t3	0x12
31	$ra	0x12

Tập thanh ghi
(Register Files)

add $ra, $0, $t3

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

Toán hạng thanh ghi - Quiz 2

Lệnh	Ví dụ	Ý nghĩa
Cộng	add $s1,$s2,$s3	$s1=$s2 + $s3
Trừ	sub $s1,$s2,$s3	$s1=$s2 - $s3

NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

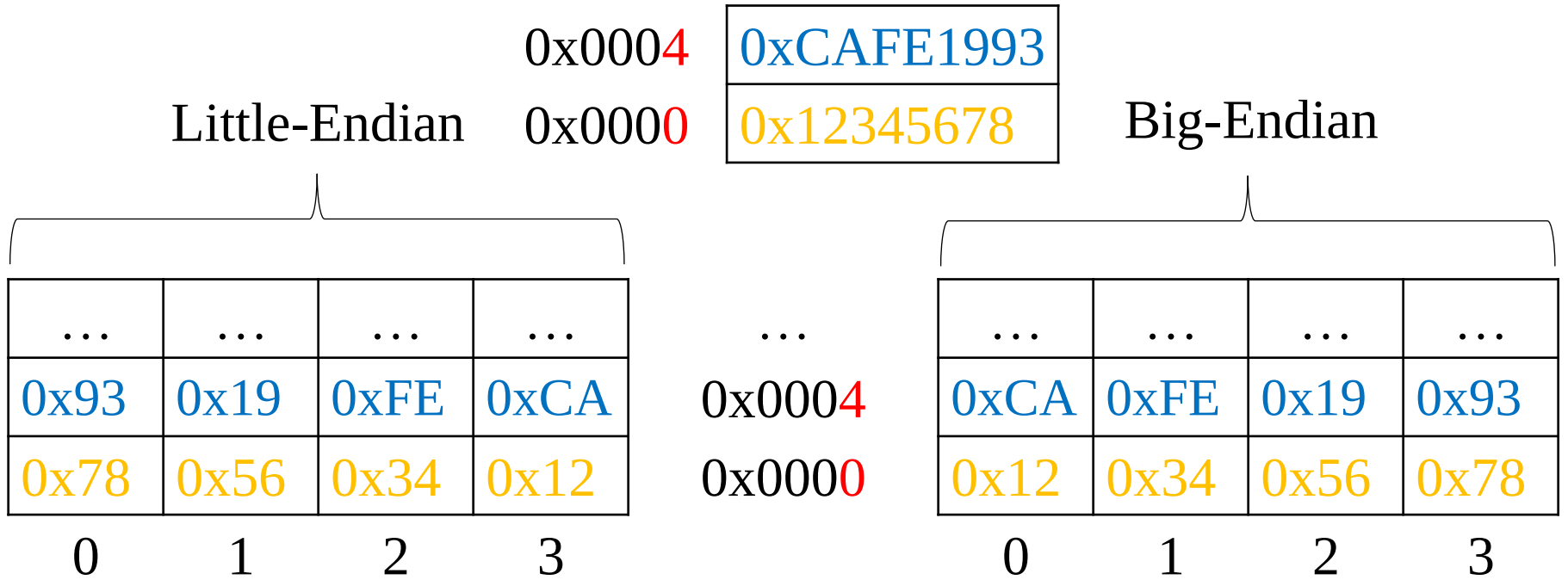
Chuyển đổi lệnh trong lập trình C thành lệnh hợp ngữ MIPS, biết rằng các biến F, A, B, C và D đều nằm trong các thanh ghi lần lượt từ t0 đến t4:

$$F = (A + B) – (C + D)$$

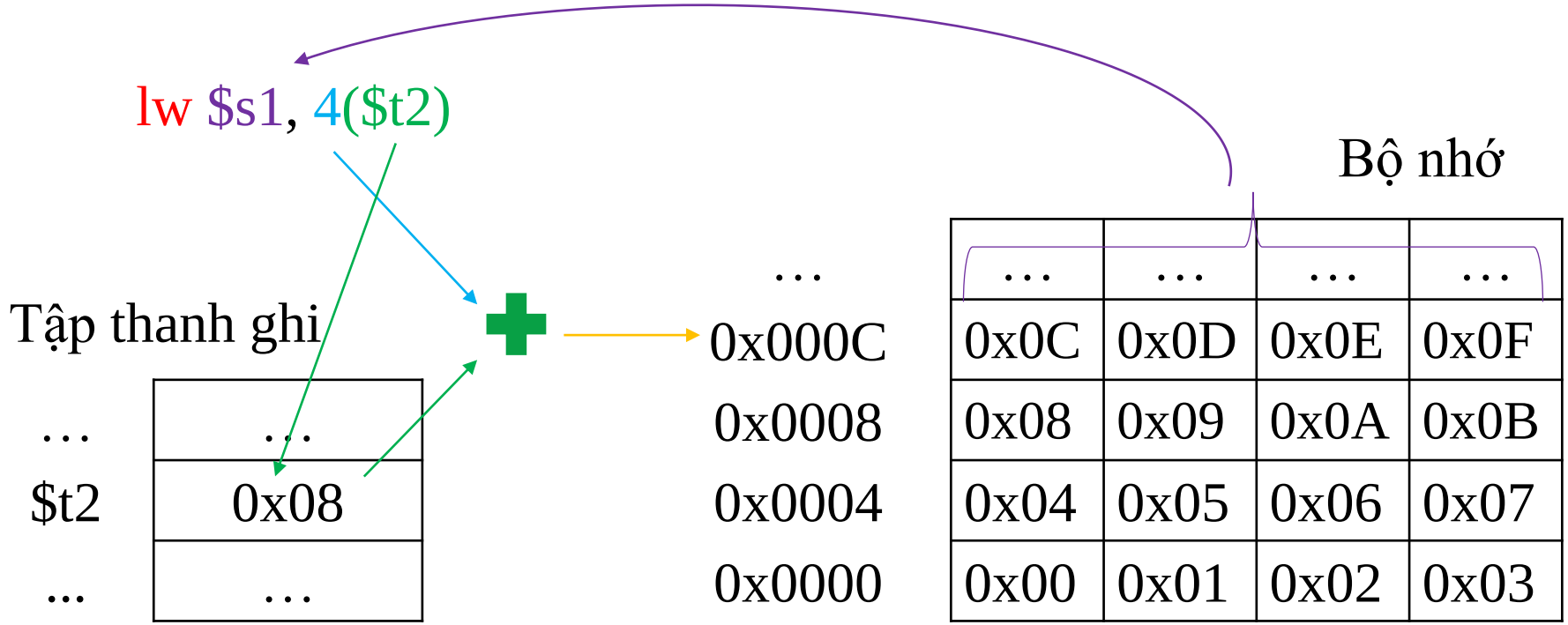
Toán hạng bộ nhớ

- Bộ nhớ được đánh địa chỉ theo byte
- MIPS quy định địa chỉ bộ nhớ phải là bội số của 4 (1 word = 4 byte)
- MIPS sử dụng mô hình địa chỉ Big-Endian

Toán hạng bộ nhớ (2)



Toán hạng bộ nhớ (3)



Toán hạng bộ nhớ - Quiz 3

Lệnh	Ví dụ	Ý nghĩa
Cộng	add $s1,$s2,$s3	$s1= $s2 + $s3
Nạp word	lw $s1,20($s2)	$s1= Mem[$s2 + 20]

Chuyển đổi lệnh trong lập trình C thành lệnh hợp ngữ MIPS, giả sử A là một biến nguyên nằm trong $a0 và mảng các số nguyên B có địa chỉ nằm trong $t0:

$$F = A + B[3]$$

NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

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

Toán hạng số tức thời - Quiz 4

Lệnh	Ví dụ	Ý nghĩa
Cộng	add $s1,$s2,$s3	$s1=$s2 + $s3
Trừ	sub $s1,$s2,$s3	$s1=$s2 - $s3
Cộng tức thì	addi $s1,$s2,5	$s1=$s2 + 5

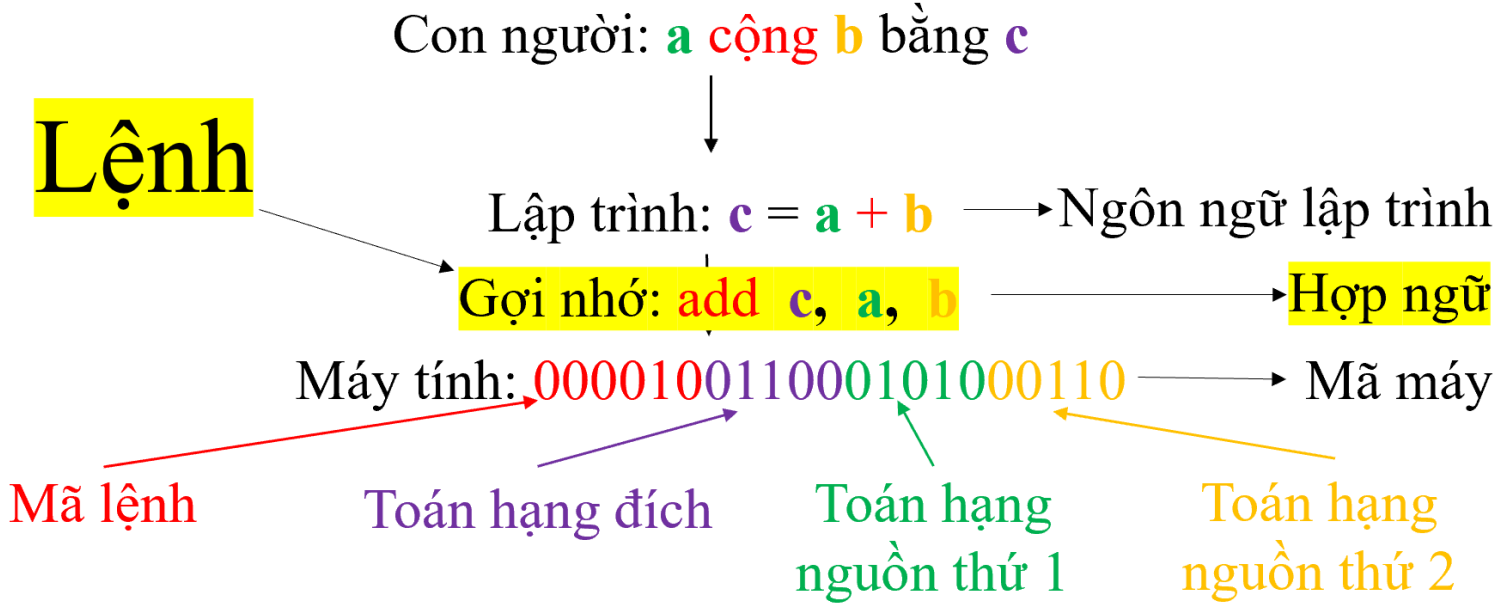
NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Chuyển đổi lệnh trong lập trình C thành lệnh hợp ngữ MIPS, biết rằng các biến F, A, B, C nằm trong các thanh ghi từ t5 đến t8:

$$F = A - (B + 7) + C$$

NỘI DUNG

- Kiến trúc tập lệnh
- Toán hạng
- **Định dạng lệnh**
- Bài tập



Định dạng lệnh

- Lệnh được biểu diễn bằng các mã nhị phân (mã máy)
- Định dạng lệnh là một hình thức biểu diễn một lệnh dưới dạng các trường mã nhị phân
  - Lệnh của MIPS đều rộng 32 bit
- MIPS có 3 định dạng lệnh:
  - Định dạng lệnh R: Cho các thao tác tuần túy trên thanh ghi
  - Định dạng lệnh I: Cho các thao tác sử dụng số tức thời có giá trị nhỏ và vừa
  - Định dạng lệnh J: Cho các thao tác sử dụng số tức thời có giá trị lớn

Định dạng lệnh - R

Các trường lệnh dạng R:

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

- 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

Định dạng lệnh – R (ví dụ)

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

Chuyển đổi lệnh ASM MIPS sang mã máy

- Chuyển câu lệnh assembly (ASM) MIPS sau sang mã máy:

  and $t3, $s0, $s2

- Thực hiện việc chuyển đổi theo các bước sau:
  - Bước 1: Tra bảng "MIPS reference data" xem lệnh and thuộc định dạng nào => R type

And	and	R	R[rd] = R[rs] & R[rt]	$0 / 24_{hex}$

op	rs	rt	rd	shamt	funct

Chuyển đổi lệnh ASM MIPS sang mã máy (2)

- Chuyển câu lệnh assembly (ASM) MIPS sau sang mã máy:

  and $t3, $s0, $s2

- Thực hiện việc chuyển đổi theo các bước sau:
  - Bước 1: Tra bảng "MIPS reference data" xem lệnh and thuộc định dạng nào => R type
  - Bước 2: Tra các trường opcode và function

And	and	R	R[rd] = R[rs] & R[rt]	$0 / 24_{hex}$

op	rs	rt	rd	shamt	funct
000000					100100

Chuyển đổi lệnh ASM MIPS sang mã máy (3)

- Chuyển câu lệnh assembly (ASM) MIPS sau sang mã máy:

  and $t3, $s0, $s2 => $t3 = $s0 & $s2

- Thực hiện việc chuyển đổi theo các bước sau:
  - Bước 1: Tra bảng "MIPS reference data" xem lệnh and thuộc định dạng nào => R type
  - Bước 2: Tra các trường opcode và function
  - Bước 3: Tra vị trí và chỉ số các thanh ghi

And	and	R R[rd] = R[rs] & R[rt]	$0 / 24_{hex}$

op	rs	rt	rd	shamt	funct
000000	10000	10010	01011		100100

NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Chuyển đổi lệnh ASM MIPS sang mã máy (4)

- Chuyển câu lệnh assembly (ASM) MIPS sau sang mã máy:

  and $t3, $s0, $s2 => $t3 = $s0 & $s2

- Thực hiện việc chuyển đổi theo các bước sau:
  - Bước 1: Tra bảng "MIPS reference data" xem lệnh and thuộc định dạng nào => R type
  - Bước 2: Tra các trường opcode và function
  - Bước 3: Tra vị trí và chỉ số các thanh ghi
  - Bước 4: Điền trường shamt và hoàn thành mã máy của lệnh

op	rs	rt	rd	shamt	funct
000000	10000	10010	01011	00000	100100

Định dạng lệnh - R - Quiz 5

Biểu diễn các lệnh sau:

- add $a0, $t1, $sp
- sll $t1, $t5, 7

NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Định dạng lệnh - I

Các trường lệnh dạng I:

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

- 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

Định dạng lệnh – I (ví dụ)

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

lw $t0, -8($s2)

op (lw)	$s2	$t0	-8
0x23	18	8	-8
100011	10010	01000	1111111111111000

10001110010010000111111111111000 0x4E48FFF8

Định dạng lệnh - I - Quiz 6

Biểu diễn lệnh lw $a0, 48($sp)

NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Định dạng lệnh - J

op	address
6 bit	26 bit

Các trường lệnh dạng J:

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

Định dạng lệnh – J (ví dụ)

op	address
6 bit	26 bit

0xCAFEBAB8: j sub_pro

…

0xCAFEBAFC: sub_pro:

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

00001010101111111010111010111111 0x0ABFAEBF

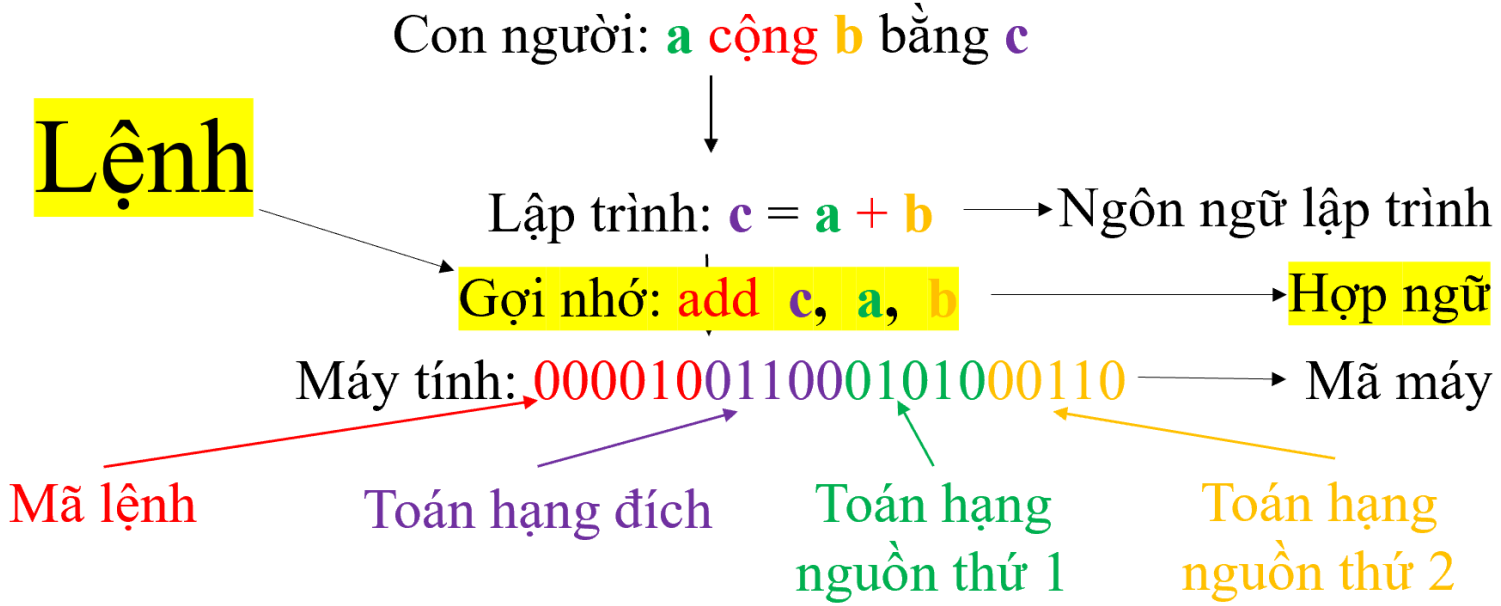
Định dạng lệnh - J - Quiz 7

Biểu diễn lệnh j IT012

- Lệnh j ở địa chỉ 0x00CAFE00
- Nhãn IT012 có địa chỉ tương ứng là 0x0000A5B0

TỔNG KẾT

- Kiến trúc tập lệnh
- Toán hạng
- Định dạng lệnh
- Bài tập



Bài tập 1

Lệnh	Ví dụ	Ý nghĩa
Cộng	add $s1,$s2,$s3	$s1=$s2 + $s3
Nạp word	lw $s1,20($s2)	$s1= Mem[$s2 + 20]
Cộng tức thì	addi $s1,$s2,5	$s1=$s2 + 5

NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Chuyển đổi lệnh trong lập trình C thành lệnh hợp ngữ MIPS, giả sử các biến F, B, G nằm trong các thanh ghi và mảng các số nguyên A có địa chỉ lưu trong $t1:

$$F = A[B + 4]$$

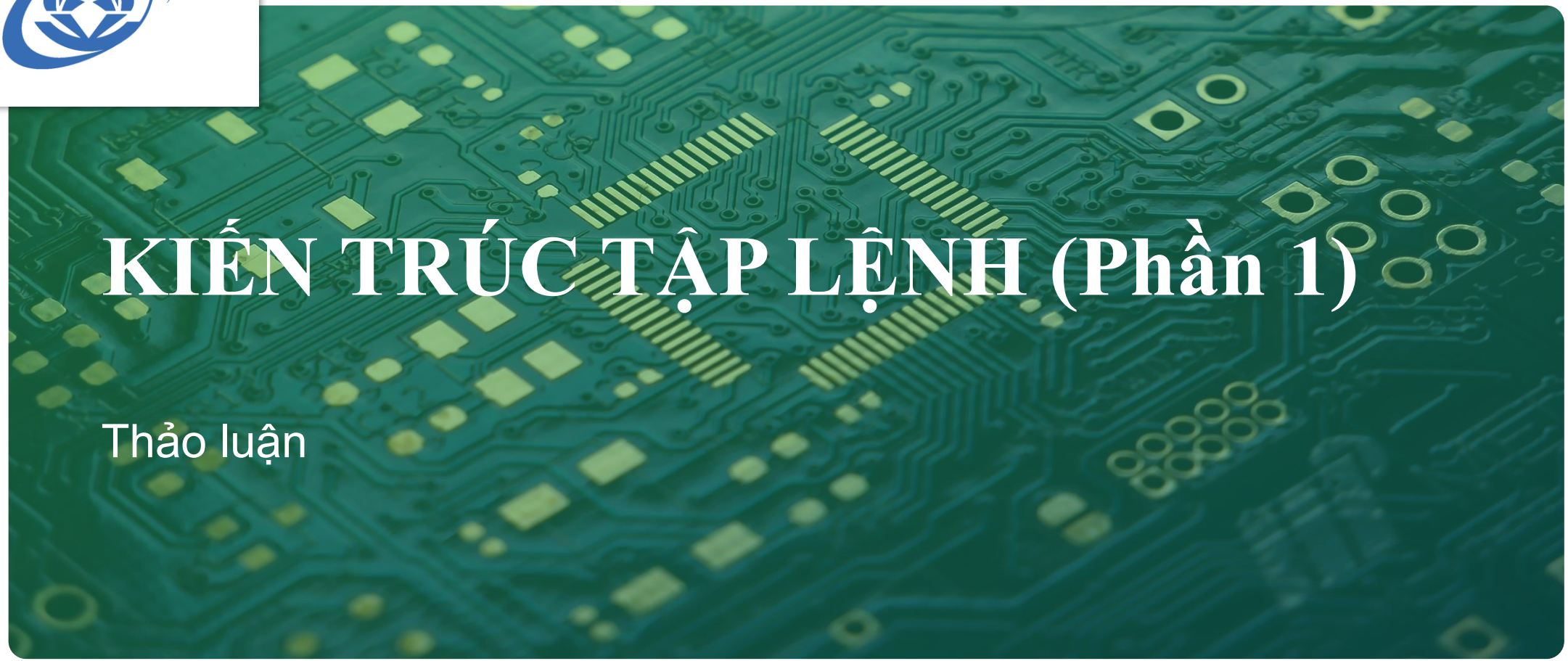
$$G = A[16 - C] + A[B - 4]$$

Bài tập 2

Biểu diễn các lệnh sau:

- sub $s1, $s2, $s3
- lw $t7, 20($k0)
- sw $v1, 20($gp)
- nor $at, $ra, $a2
- j ABC
  - Lệnh j đang ở địa chỉ 0xFEC0
  - Nhãn ABC có địa chỉ tương ứng là 0x2500

NAME	NUMBER
$zero	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31



KIẾN TRÚC TẬP LỆNH (Phần 1)

Thảo luận