HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÀI GIẢNG MÔN

KIẾN TRÚC MÁY TÍNH VÀ HỆ ĐIỀU HÀNH

Giảng viên: ThS. Nguyễn Thị Ngọc Vinh

Bộ môn: Khoa học máy tính- Khoa CNTT1

Email: ntngocvinh@yahoo.com

CHƯƠNG 2: KHỐI XỬ LÝ TRUNG TÂM

1. Khối xử lý trung tâm
   - Sơ đồ khối tổng quát
   - Chu kỳ xử lý lệnh
   - Thanh ghi
   - Khối điều khiển (CU)
   - Khối số học và logic (ALU)
   - Bus trong CPU

2. Tập lệnh máy tính

- Khái niệm lệnh, tập lệnh
- Chu kỳ và các pha thực hiện lệnh
- Các dạng toán hạng
- Các chế độ địa chỉ
- Một số dạng lệnh thông dụng

2.1 KHỐI XỬ LÝ TRUNG TÂM

CPU – SƠ ĐỒ KHỐI TỔNG QUÁT

CU: (Control Unit) Khối điều khiển

IR: (Instruction Register) Thanh ghi lệnh

PC: (Program Counter) Bộ đếm chương trình

MAR: (Memory Address Register) Thanh ghi địa chỉ bộ nhớ

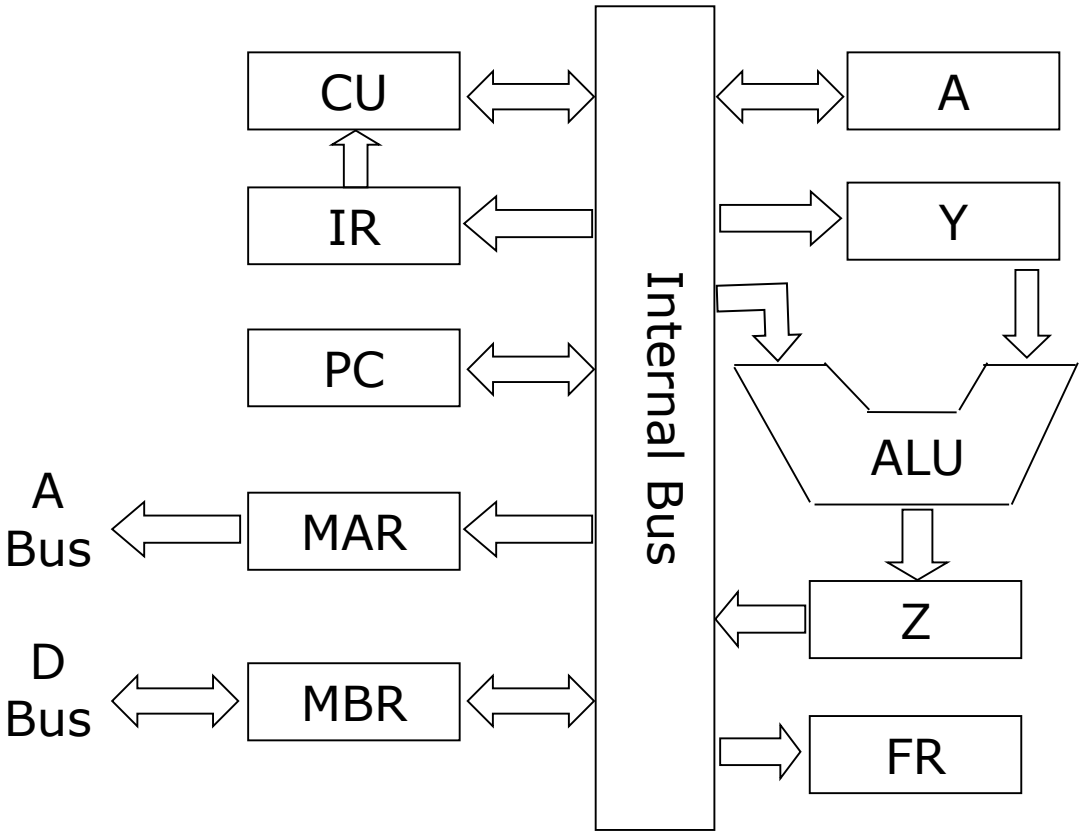
MBR: (Memory Buffer Register) Thanh ghi nhớ đệm

A: (Accumulator Register) Thanh ghi tích lũy

Y, Z: (Temporary Register) Thanh ghi tạm thời

FR: (Flag Register) Thanh ghi cờ

ALU: (Arithmetic and Logic Unit) Khối tính toán số học -logic



CHU KỲ XỬ LÝ LỆNH

1. Khi một chương trình được chạy, hệ điều hành tải mã chương trình vào bộ nhớ trong
2. Địa chỉ lệnh đầu tiên của chương trình được đưa vào thanh ghi PC
3. Địa chỉ của ô nhớ chứa lệnh được chuyển tới bus A qua thanh ghi MAR
4. Tiếp theo, bus A truyền địa chỉ tới khối quản lý bộ nhớ MMU (Memory Management Unit)
5. MMU chọn ô nhớ và sinh ra tín hiệu READ

CHU KỲ XỬ LÝ LỆNH

6. Lệnh chứa trong ô nhớ được chuyển tới thanh ghi MBR qua bus D
7. MBR chuyển lệnh tới thanh ghi IR. Sau đó IR lại chuyển lệnh tới CU
8. CU giải mã lệnh và sinh ra các tín hiệu xử lý cho các đơn vị khác, ví dụ như ALU để thực hiện lệnh
9. Địa chỉ trong PC được tăng lên để trỏ tới lệnh tiếp theo của chương trình sẽ được thực hiện
10. Thực hiện lại các bước 3->9 để chạy hết các lệnh của chương trình

THANH GHI

- Thanh ghi là thành phần nhớ ở bên trong CPU:
  - Lưu trữ tạm thời lệnh và dữ liệu cho CPU xử lý
  - Dung lượng nhỏ, số lượng ít
  - Tốc độ rất cao (bằng tốc độ CPU)
- Các CPU thế hệ cũ (80x86) có 16 – 32 thanh ghi. CPU thế hệ mới (Intel Pentium 4, Core 2 Duo) có hàng trăm thanh ghi
- Kích thước thanh ghi phụ thuộc vào thiết kế CPU: 8, 16, 32, 64, 128 và 256 bit
  - 8086 và 80286: 8 và 16 bit
  - 80386, Pentium II: 16 – 32 bit
  - Pentium IV, Core Duo: 32, 64 và 128 bit

THANH GHI TÍCH LŨY A (Accumulator)

- Thanh ghi tích lũy hay thanh ghi A là một trong những thanh ghi quan trọng nhất của CPU
  - Lưu trữ các toán hạng đầu vào
  - Lưu kết quả đầu ra
- Kích thước của thanh ghi A tương ứng với độ dài từ xử lý của CPU: 8, 16, 32, 64 bit
- Cũng được sử dụng để trao đổi dữ liệu với các thiết bị vào ra

BỘ ĐẾM CHƯƠNG TRÌNH PC

- Program Counter hay Instruction Pointer lưu địa chỉ bộ nhớ của lệnh tiếp theo
- PC chứa địa chỉ ô nhớ chứa lệnh đầu tiên của chương trình khi nó được kích hoạt và được tải vào bộ nhớ
- Sau khi CPU chạy xong 1 lệnh, địa chỉ ô nhớ chứa lệnh tiếp theo được tải vào PC
- Kích thước của PC phụ thuộc vào thiết kế CPU: 8, 16, 32, 64 bit

THANH GHI TRẠNG THÁI FR

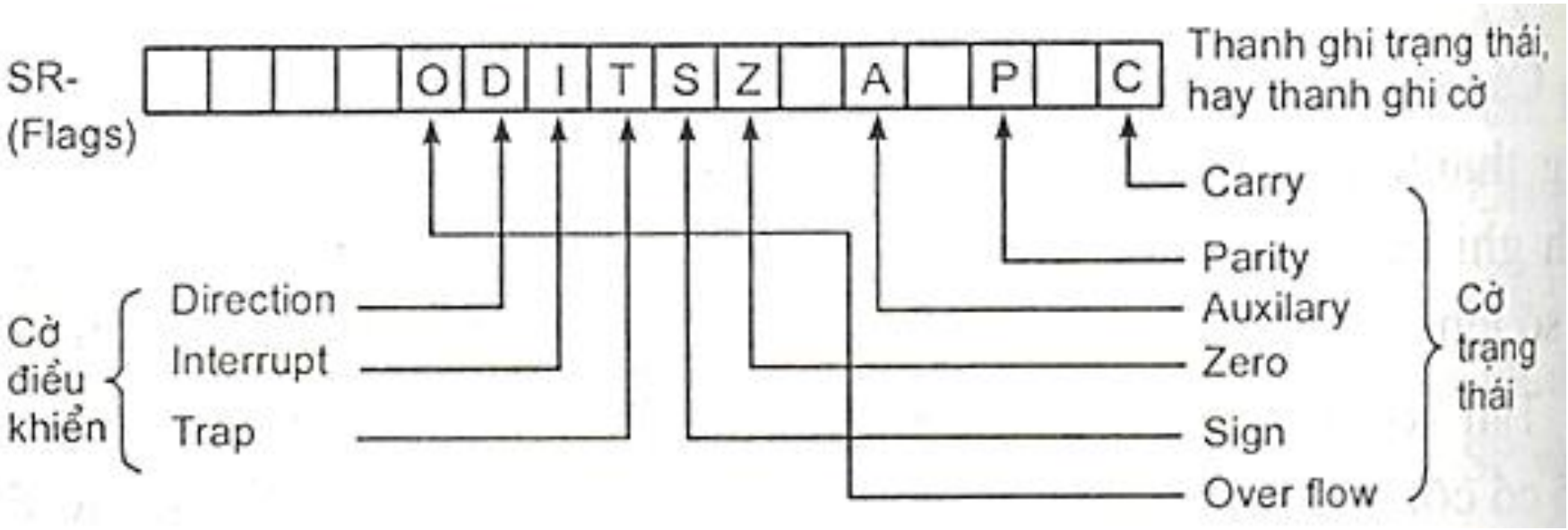
- Mỗi bit của thanh ghi cờ lưu trữ trạng thái kết quả phép tính được ALU thực hiện
- Có 2 kiểu cờ:
  - Cờ trạng thái: CF, OF, AF, ZF, PF, SF
  - Cờ điều khiển: IF, TF, DF
- Các bit cờ thường được dùng là các điều kiện rẽ nhánh lệnh tạo logic chương trình
- Kích thước FR phụ thuộc thiết kế CPU

THANH GHI TRẠNG THÁI FR

Flag	ZF	SF	CF	AF	IF	OF	PF	1
Bit No	7	6	5	4	3	2	1	0

- ZF: Zero Flag, ZF=1 nếu kết quả =0 và ZF=0 nếu kết quả <>0.
- SF: Sign Flag, SF=1 nếu kết quả âm và SF=0 nếu kết quả dương
- CF: Carry Flag, CF=1 nếu có nhớ/mượn ở bit trái nhất
- AF: Auxiliary Flag, AF=1 nếu có nhớ ở bit trái nhất của nibble
- OF: Overflow Flag, OF=1 nếu có tràn, OF=0 ngược lại
- PF: Parity Flag, PF=1 nếu tổng số bit 1 trong kết quả là số lẻ, PF=0 ngược lại
- IF: Interrupt Flag, IF=1: ngắt được phép, IF=0: cấm ngắt

THANH GHI TRẠNG THÁI CỦA 8086



CON TRỎ NGĂN XẾP (SP: Stack Pointer)

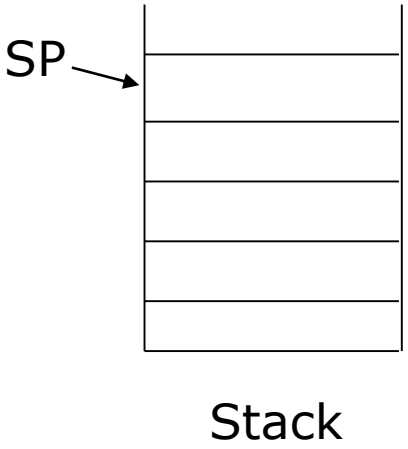
- Ngăn xếp là 1 đoạn bộ nhớ đặc biệt hoạt động theo nguyên tắc vào sau ra trước (LIFO)
- Con trỏ ngăn xếp là thanh ghi luôn trỏ tới đỉnh của ngăn xếp
- 2 thao tác với ngăn xếp:
  - Push: đẩy dữ liệu vào ngăn xếp

    SP ← SP + 1

    {SP} ← Data
  - Pop: lấy dữ liệu ra khỏi ngăn xếp

    Register ← {SP}

    SP ← SP - 1



CÁC THANH GHI ĐA NĂNG

- Có thể sử dụng cho nhiều mục đích:
  - Lưu các toán hạng đầu vào
  - Lưu các kết quả đầu ra
- Ví dụ: CPU 8086 có 4 thanh ghi đa năng
  - AX: Accumulator Register
  - BX: Base Register
  - CX: Counter Register
  - DX: Data Register

THANH GHI LỆNH IR

- Lưu trữ lệnh đang được xử lý
- IR lấy lệnh từ MBR và chuyển nó tới CU để giải mã lệnh

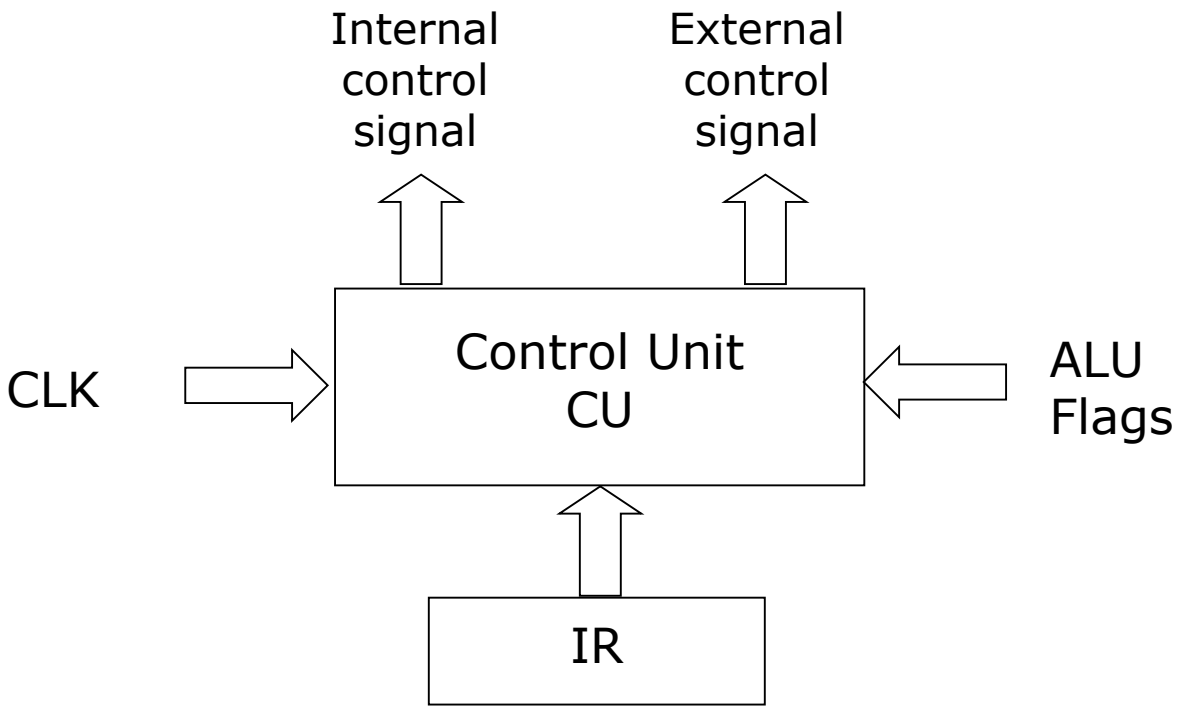
THANH GHI MBR VÀ MAR

- MAR: thanh ghi địa chỉ bộ nhớ
  - Giao diện giữa CPU và bus địa chỉ
  - Nhận địa chỉ bộ nhớ của lệnh tiếp theo từ PC và chuyển nó tới bus địa chỉ
- MBR: thanh ghi đệm bộ nhớ
  - Giao diện giữa CPU và bus dữ liệu
  - Nhận lệnh từ bus dữ liệu và chuyển nó tới IR

CÁC THANH GHI TẠM THỜI

- CPU thường sử dụng một số thanh ghi tạm thời để:
  - Lưu trữ các toán hạng đầu vào
  - Lưu các kết quả đầu ra
  - Hỗ trợ xử lý song song (tại một thời điểm chạy nhiều hơn 1 lệnh)
  - Hỗ trợ thực hiện lệnh theo cơ chế thực hiện tiên tiến kiểu không trật tự (OOO – Out Of Order execution)

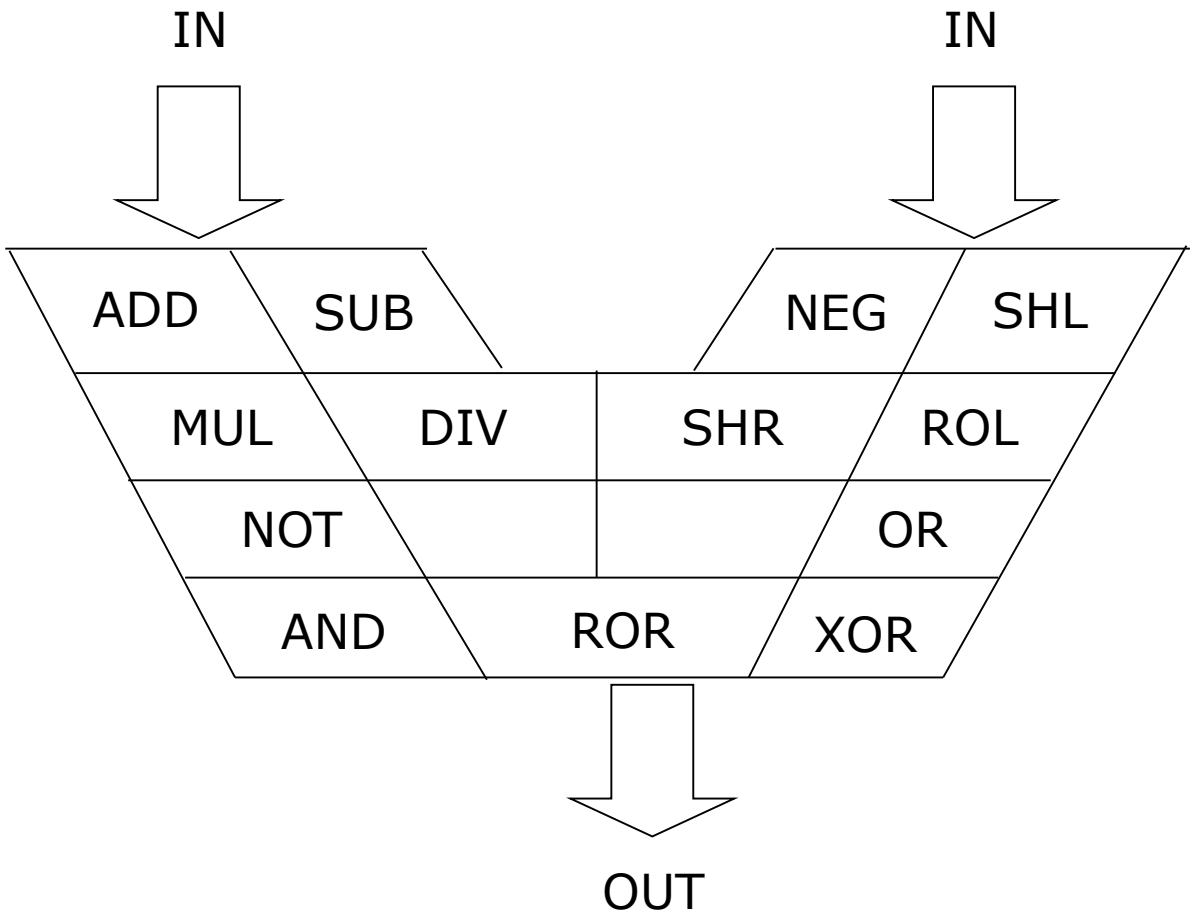
KHỐI ĐIỀU KHIỂN CU



KHỐI ĐIỀU KHIỂN CU

- Điều khiển tất cả các hoạt động của CPU theo xung nhịp đồng hồ
- Nhận 3 tín hiệu đầu vào:
  - Lệnh từ IR
  - Giá trị các cờ trạng thái
  - Xung đồng hồ
- CU sinh 2 nhóm tín hiệu đầu ra:
  - Nhóm tín hiệu điều khiển các bộ phận bên trong CPU
  - Nhóm tín hiệu điều khiển các bộ phận bên ngoài CPU
- Sử dụng nhịp đồng hồ để đồng bộ hóa các đơn vị bên trong CPU và giữa CPU với các thành phần bên ngoài

KHỐI SỐ HỌC VÀ LOGIC ALU



KHỐI SỐ HỌC VÀ LOGIC ALU

- Bao gồm các đơn vị chức năng con để thực hiện các phép toán số học và logic:
  - Bộ cộng (ADD), bộ trừ (SUB), bộ nhân (MUL), bộ chia (DIV), …
  - Các bộ dịch (SHIFT) và quay (ROTATE)
  - Bộ phủ định (NOT), bộ và (AND), bộ hoặc (OR), và bộ hoặc loại trừ (XOR)
- ALU có:
  - 2 cổng IN để nhận đầu vào từ các thanh ghi
  - 1 cổng OUT được nối với bus trong để gửi kết quả tới các thanh ghi

BUS TRONG

- Bus trong là kênh liên lạc của tất cả các thành phần trong CPU
- Hỗ trợ liên lạc 2 chiều
- Bus trong có giao diện để trao đổi thông tin với bus ngoài (bus hệ thống)
- Bus trong luôn có băng thông lớn và tốc độ nhanh hơn so với bus ngoài

2.2 TẬP LỆNH MÁY TÍNH

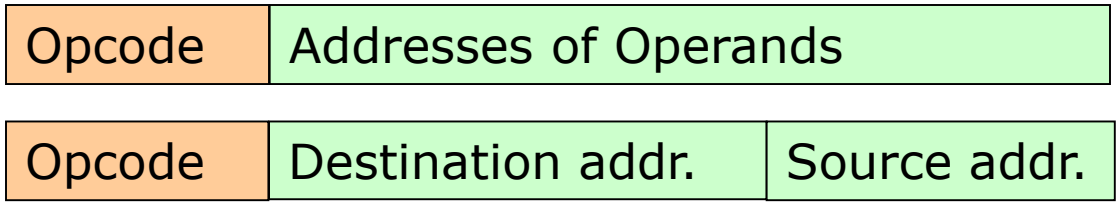
GIỚI THIỆU CHUNG

- Lệnh máy tính là một từ nhị phân (binary word) mà thực hiện một nhiệm vụ cụ thể:
  - Lệnh được lưu trong bộ nhớ
  - Lệnh được đọc từ bộ nhớ vào CPU để giải mã và thực hiện
  - Mỗi lệnh có chức năng riêng của nó
- Tập lệnh gồm nhiều lệnh, có thể được chia thành các nhóm theo chức năng:
  - Chuyển dữ liệu (data movement)
  - Tính toán (computational)
  - Điều kiện và rẽ nhánh (conditioning & branching)
  - Các lệnh khác …

- Quá trình thực hiện/ chạy lệnh được chia thành các pha hay giai đoạn (stage). Mỗi lệnh có thể được thực hiện theo 4 giai đoạn:
  - Đọc lệnh IF(Instruction Fetch): lệnh được đọc từ bộ nhớ vào CPU
  - Giải mã lệnh ID(Instruction Decode): CPU giải mã lệnh
  - Chay lệnh IE(Instruction Execution): CPU thực hiện lệnh
  - Ghi WB(Write Back): kết quả lệnh (nếu có) được ghi vào thanh ghi hoặc bộ nhớ

KHUÔN DẠNG LỆNH

- Khuôn dạng lệnh thông thường bao gồm 2 phần:
  - Mã lệnh (opcode): mỗi lệnh đều có riêng một mã
  - Địa chỉ các toán hạng (addresses of operands): số lượng toán hạng phụ thuộc vào lệnh. Có thể có các dạng địa chỉ toán hạng sau:
    - 3 địa chỉ
    - 2 địa chỉ
    - 1 địa chỉ
    - 1.5 địa chỉ
    - 0 địa chỉ



TOÁN HẠNG 3 ĐỊA CHỈ

- Khuôn dạng:
  - opcode addr1, addr2, addr3
  - Mỗi địa chỉ addr1, addr2, addr3: tham chiếu tới một ô nhớ hoặc 1 thanh ghi
- Ví dụ
  1. ADD R_1, R_2, R_3; $R_2 + R_3 \rightarrow R_1$

     R_2 cộng R_3 sau đó kết quả đưa vào R_1

     R_i là các thanh ghi CPU
  2. ADD A, B, C; M[B]+M[C] → M[A]

     A, B, C là các vị trí trong bộ nhớ

TOÁN HẠNG 2 ĐỊA CHỈ

- Khuôn dạng:
  - opcode addr1, addr2
  - Mỗi địa chỉ addr1, addr2: tham chiếu tới 1 thanh ghi hoặc 1 vị trí trong bộ nhớ
- Ví dụ
  1. ADD R_1, R_2; $R_1 + R_2 \rightarrow R_1$

     R_1 cộng R_2 sau đó kết quả đưa vào R_1

     R_i là các thanh ghi CPU
  2. ADD A, B; M[A]+M[B] → M[A]

     A, B là các vị trí trong bộ nhớ

TOÁN HẠNG 1 ĐỊA CHỈ

- Khuôn dạng:
  - opcode addr
  - addr: tham chiếu tới 1 thanh ghi hoặc 1 vị trí trong bộ nhớ
  - Khuôn dạng này sử dụng R_{acc} (thanh ghi tích lũy) mặc định cho địa chỉ thứ 2
- Ví dụ
  1. ADD R_1; $R_1 + R_{acc} \rightarrow R_{acc}$
     R_1 cộng R_{acc} sau đó kết quả đưa vào R_{acc}
     R_i là các thanh ghi CPU
  2. ADD A; $M[A]+R_{acc} \rightarrow R_{acc}$
     A là vị trí trong bộ nhớ

TOÁN HẠNG 1.5 ĐỊA CHỈ

- Khuôn dạng:
  - opcode addr1, addr2
  - Một địa chỉ tham chiếu tới 1 ô nhớ và địa chỉ còn lại tham chiếu tới 1 thanh ghi
  - Là dạng hỗn hợp giữa các toán hạng thanh ghi và vị trí bộ nhớ
- Ví dụ
  1. ADD R_1, B; M[B] + R_1 → R_1

CÁC CHẾ ĐỘ ĐỊA CHỈ

- Chế độ địa chỉ là cách thức CPU tổ chức các toán hạng
  - Chế độ địa chỉ cho phép CPU kiểm tra dạng và tìm các toán hạng của lệnh
- Một số chế độ địa chỉ tiêu biểu:
  - Chế độ địa chỉ tức thì (Immediate)
  - Chế độ địa chỉ trực tiếp (Direct)
  - Chế độ địa chỉ gián tiếp qua thanh ghi (Register Indirect)
  - Chế độ địa chỉ gián tiếp qua bộ nhớ (Memory Indirect)
  - Chế độ địa chỉ chỉ số (Indexed)
  - Chế độ địa chỉ tương đối (Relative)

CHẾ ĐỘ ĐỊA CHỈ TỨC THÌ

- Giá trị của toán hạng nguồn có sẵn trong lệnh (hằng số)
- Toán hạng đích có thể là thanh ghi hoặc một vị trí bộ nhớ
- Ví dụ:

LOAD R_1, #1000; 1000 → R_1

giá trị 1000 được tải vào thanh ghi R1

LOAD B, #500; 500 → M[B]

Giá trị 500 được tải vào vị trí B trong bộ nhớ

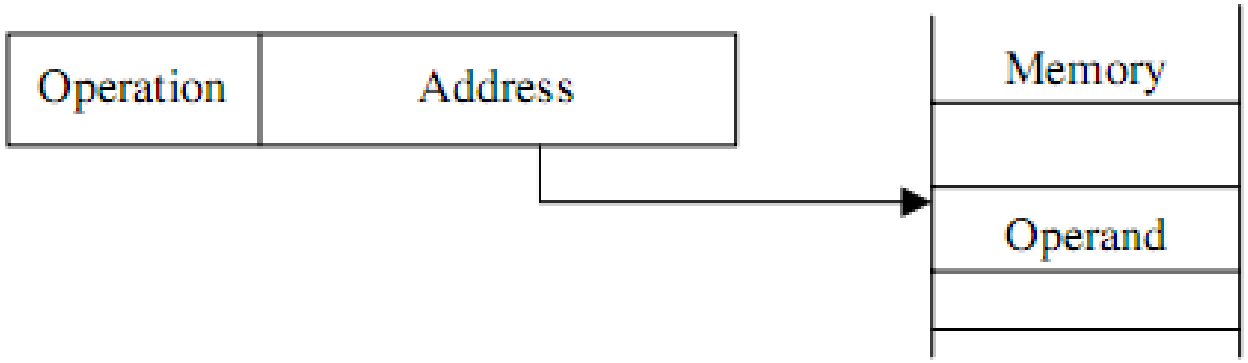
CHẾ ĐỘ ĐỊA CHỈ TRỰC TIẾP/ TUYỆT ĐỐI

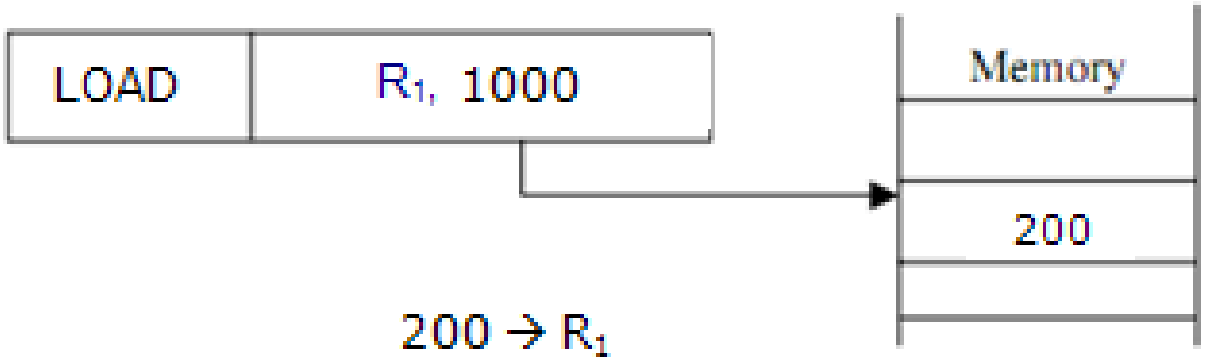
- Một toán hạng là địa chỉ của một vị trí trong bộ nhớ chứa dữ liệu
- Toán hạng kia là thanh ghi hoặc 1 địa chỉ ô nhớ
- Ví dụ:

LOAD R_1, 1000; M[1000] → R_1

giá trị lưu trong vị trí 1000 ở bộ nhớ được tải vào thanh ghiR1

CHẾ ĐỘ ĐỊA CHỈ TRỰC TIẾP/ TUYỆT ĐỐI





CHẾ ĐỘ ĐỊA CHỈ GIÁN TIẾP

- Một thanh ghi hoặc một vị trí trong bộ nhớ được sử dụng để lưu địa chỉ của toán hạng
  - Gián tiếp thanh ghi:

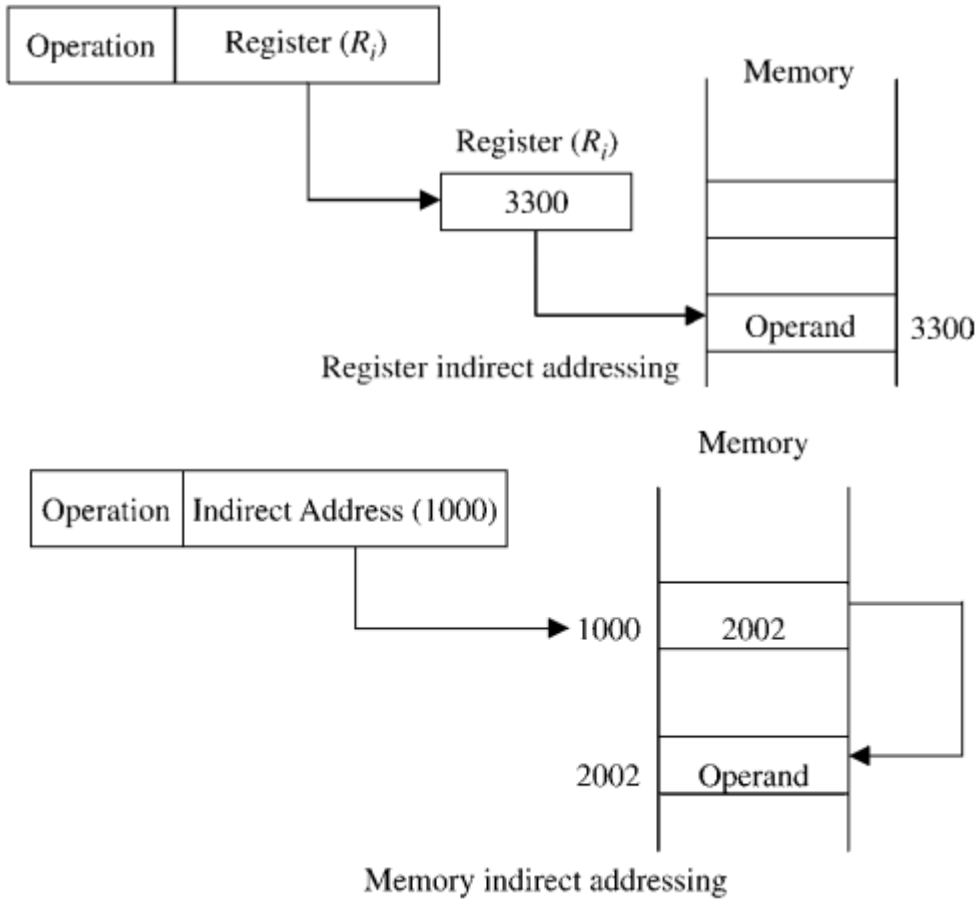
    LOAD R_j ,(R_i); M[R_i] → R_j

    Tải giá trị tại vị trí bộ nhớ có địa chỉ được lưu trong Ri vào thanh ghi Rj
  - Gián tiếp bộ nhớ:

    LOAD R_i , (1000); M[M[1000]] → R_i

    Giá trị của vị trí bộ nhớ có địa chỉ được lưu tại vị trí 1000 vào Ri

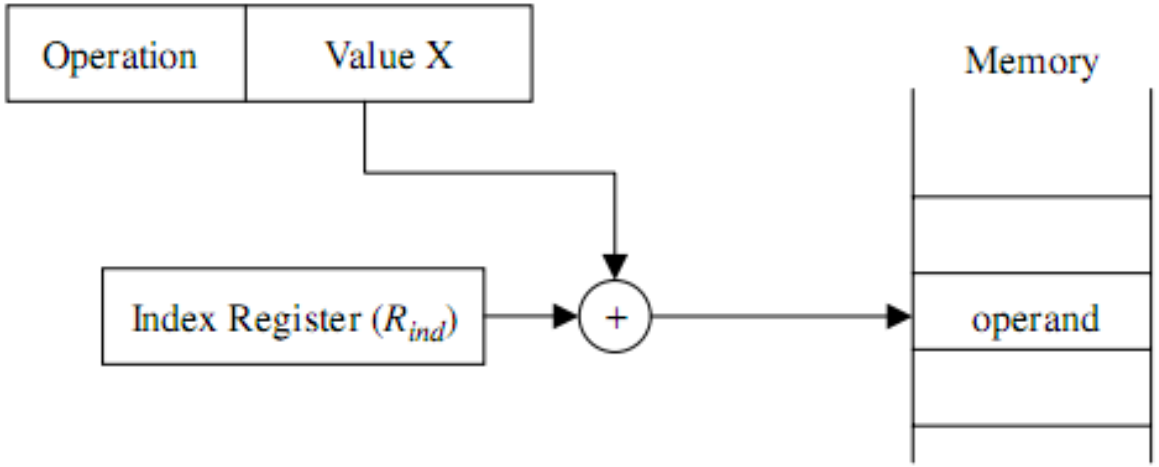
CHẾ ĐỘ ĐỊA CHỈ GIÁN TIẾP



CHẾ ĐỘ ĐỊA CHỈ CHỈ SỐ

- Địa chỉ của toán hạng có được bằng cách cộng thêm hằng số vào nội dung của một thanh ghi, là thanh ghi chỉ số
- Ví dụ

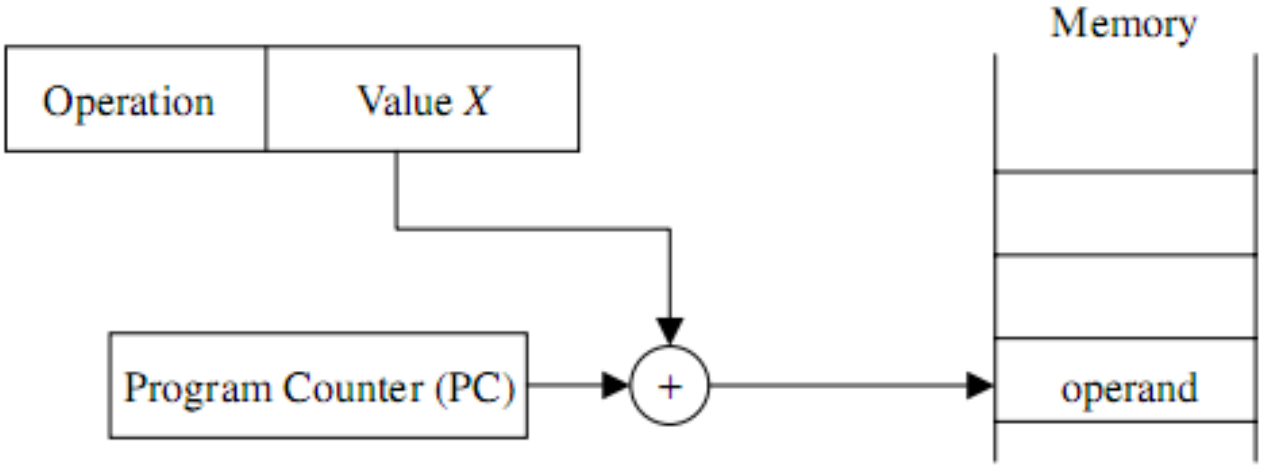
  LOAD R_i, $X(R_{ind})$; $M[X+R_{ind}] \rightarrow R_i$



CHẾ ĐỘ ĐỊA CHỈ TƯƠNG ĐỐI

- Địa chỉ của toán hạng có được bằng cách cộng thêm hằng số vào nội dung của một thanh ghi, là thanh ghi con đếm chương trình PC
- Ví dụ

  LOAD R_i, X(PC); M[X+PC] → R_i



TỔNG KẾT CÁC CHẾ ĐỘ ĐỊA CHỈ

Chế độ địa chỉ	Ý nghĩa	Ví dụ	Thực hiện
Tức thì	Giá trị của toán hạng được chứa trong lệnh	LOAD Ri, #1000	Ri ← 1000
Trực tiếp	Địa chỉ của toán hạng được chứa trong lệnh	LOAD Ri, 1000	Ri ← M[1000]
Gián tiếp thanh ghi	Giá trị của thanh ghi trong lệnh là địa chỉ bô nhớ chứa toán hạng	LOAD Ri, (Rj)	Ri ← M[Rj]
Gián tiếp bộ nhớ	Địa chỉ bộ nhớ trong lệnh chứa địa chỉ bộ nhớ của toán hạng	LOAD Ri, (1000)	Ri ← M[M[1000]]
Chỉ số	Địa chỉ của toán hạng là tổng của hằng số (trong lệnh) và giá trị của một thanh ghi chỉ số	LOAD Ri, X(Rind)	Ri ← M[X+ Rind]
Tương đối	Địa chỉ của toán hạng là tổng của hằng số và giá trị của thanh ghi con đếm chương trình	LOAD Ri, X(PC)	Ri ← M[X+ PC]

MỘT SỐ DẠNG LỆNH THÔNG DỤNG

- Các lệnh vận chuyển dữ liệu
- Các lệnh số học và logic
- Các lệnh điều khiển chương trình
- Các lệnh vào/ ra

- Chuyển dữ liệu giữa các phần của máy tính
  - Giữa các thanh ghi trong CPU

    MOVE Ri, Rj ; Rj -> Ri
  - Giữa thanh ghi CPU và một vị trí trong bộ nhớ

    MOVE Rj,1000; M[1000] -> Rj
  - Giữa các vị trí trong bộ nhớ

    MOVE 1000, (Rj) ; M[Rj] -> M[1000]

MỘT SỐ LỆNH VẬN CHUYỂN DỮ LIỆU THÔNG DỤNG

- MOVE: chuyển dữ liệu giữa thanh ghi – thanh ghi, ô nhớ - thanh ghi, ô nhớ - ô nhớ
- LOAD: nạp nội dung 1 ô nhớ vào 1 thanh ghi
- STORE: lưu nội dung 1 thanh ghi ra 1 ô nhớ
- PUSH: đẩy dữ liệu vào ngăn xếp
- POP: lấy dữ liệu ra khỏi ngăn xếp

LỆNH SỐ HỌC VÀ LOGIC

- Thực hiện các thao tác số học và logic giữa các thanh ghi và nội dung ô nhớ
- Ví dụ:

ADD R1, R2, R3; R2 + R3 -> R1

SUBSTRACT R1, R2, R3; R2 – R3 -> R1

CÁC LỆNH TÍNH TOÁN SỐ HỌC THÔNG DỤNG

- ADD: cộng 2 toán hạng
- SUBSTRACT: trừ 2 toán hạng
- MULTIPLY: nhân 2 toán hạng
- DIVIDE: chia số học
- INCREMENT: tăng 1
- DECREMENT: giảm 1

CÁC LỆNH LOGIC THÔNG DỤNG

- NOT: phủ định
- AND: và
- OR: hoặc
- XOR: hoặc loại trừ
- COMPARE: so sánh
- SHIFT: dịch
- ROTATE: quay

LỆNH ĐIỀU KHIỂN/ TUẦN TỰ

- Được dùng để thay đổi trình tự các lệnh được thực hiện:
  - Các lệnh rẽ nhánh (nhẩy) có điều kiện (conditional branching/ jump)
  - Các lệnh rẽ nhánh (nhẩy) không điều kiện (unconditional branching/ jump)
  - CALL và RETURN: lệnh gọi thực hiện và trở về từ chương trình con
- Đặc tính chung của các lệnh này là quá trình thực hiện lệnh của chúng làm thay đổi giá trị PC
- Sử dụng các cờ ALU để xác định các điều kiện

MỘT SỐ LỆNH ĐIỀU KHIỂN THÔNG DỤNG

- BRANCH – IF – CONDITION: chuyển đến thực hiện lệnh ở địa chỉ mới nếu điều kiện là đúng
- JUMP: chuyển đến thực hiện lệnh ở địa chỉ mới
- CALL: chuyển đến thực hiện chương trình con
- RETURN: trở về (từ chương trình con) thực hiện tiếp chương trình gọi

MỘT SỐ LỆNH ĐIỀU KHIỂN THÔNG DỤNG

```
        LOAD  R1, #100
LAP:
        ADD R0, (R2)
        DECREMENT  R1
        BRANCH_IF >0 LAP
```

CÁC LỆNH VÀO/ RA

- Được dùng để truyền dữ liệu giữa máy tính và các thiết bị ngoại vi
- Các thiết bị ngoại vi giao tiếp với máy tính thông qua các cổng. Mỗi cổng có một địa chỉ dành riêng
- Hai lệnh I/O cơ bản được sử dụng là các lệnh INPUT và OUTPUT
  - Lệnh INPUT được dùng để chuyển dữ liệu từ thiết bị ngoại vi vào tới bộ vi xử lý
  - Lệnh OUTPUT dùng để chuyển dữ liệu từ VXL ra thiết bị đầu ra

CÁC VÍ DỤ

```
    CLEAR R0;                   R0← 0
    MOVE R1, #100;          R1← 100
    CLEAR R2;                   R2← 0
LAP:
    ADD R0, 1000(R2);           R0←R0+ M[R2+1000]
    INCREMENT R2;               R2← R2+1
    DECREMENT R1;               R1← R1-1
    BRANCH_IF>0  LAP;       go to LAP if R1>0
    STORE 2000, R0;         M[2000]← R0
```

BÀI TẬP

1. Cho đoạn lệnh sau:

   ADD R2, (R0);

   SUBSTRACT R2, (R1);

   MOVE 500(R0), R2;

   LOAD R2, #5000;

   STORE 100(R2), R0;

Biết R0=1500, R1=4500, R2=1000, M[1500]=3000, M[4500]=500

Hãy chỉ ra giá trị của thanh ghi và tại vị trí trong bộ nhớ qua mỗi lệnh thực hiện.

BÀI TẬP

2. Cho đoạn lệnh sau:

```
    MOVE R0, #100;
    CLEAR R1;
    CLEAR R2;
LAP:
    ADD R1, 2000(R2);
    ADD R2, #2;
    DECREMENT R0;
    BRANCH_IF>0 LAP;
    STORE 3000, R1;
```

a. Hãy giải thích ý nghĩa của từng lệnh
b. Chỉ ra chế độ địa chỉ của từng lệnh (đối với các lệnh có 2 toán hạng)
c. Đoạn lệnh trên thực hiện công việc gì?

BÀI TẬP

- Cho một mảng gồm 10 số, được lưu trữ liên tiếp nhau trong bộ nhớ, bắt đầu từ vị trí ô nhớ 1000. Viết đoạn chương trình tính tổng các số dương trong mảng đó và lưu kết quả vào ô nhớ 2000.