

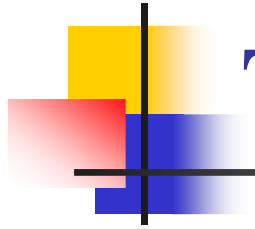


Bài 4

KIẾN TRÚC TẬP CHỈ THỊ

ISA (Instruction Set Architecture)

Nguyễn Hồng Sơn



Tập chỉ thị

- Tập hợp các chỉ thị khác nhau mà bộ xử lý có thể thực thi



Các đặc trưng

- Các thành phần của một chỉ thị
- Biểu diễn chỉ thị
- Loại chỉ thị
- Số địa chỉ
- Đặc trưng thiết kế



Các thành phần

- Mã lệnh (Operation code)
- Toán hạng nguồn (Source operand)
- Toán hạng đích (Result operand)
 - Bộ nhớ chính hay bộ nhớ ảo
 - Thanh ghi
 - Thiết bị I/O
- Tham chiếu chỉ thị kế (Next instruction reference)
 - Tường minh
 - Không tường minh



Biểu diễn chỉ thị

- Một tuần tự bit
- Biểu diễn mỗi thành phần như thế nào

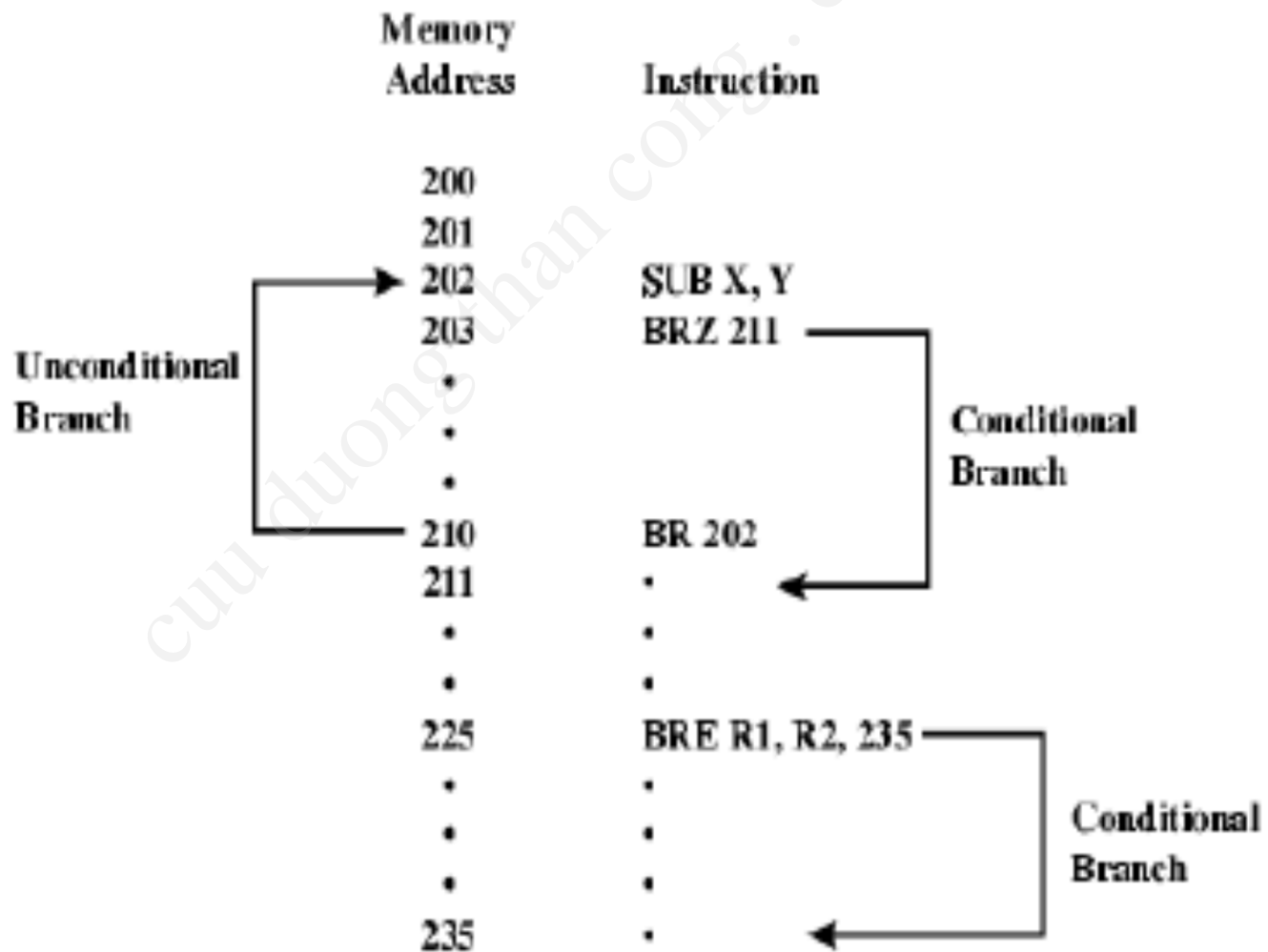
Opcode	Operand	Operand
--------	---------	---------

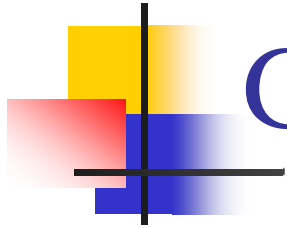


Loại chỉ thị

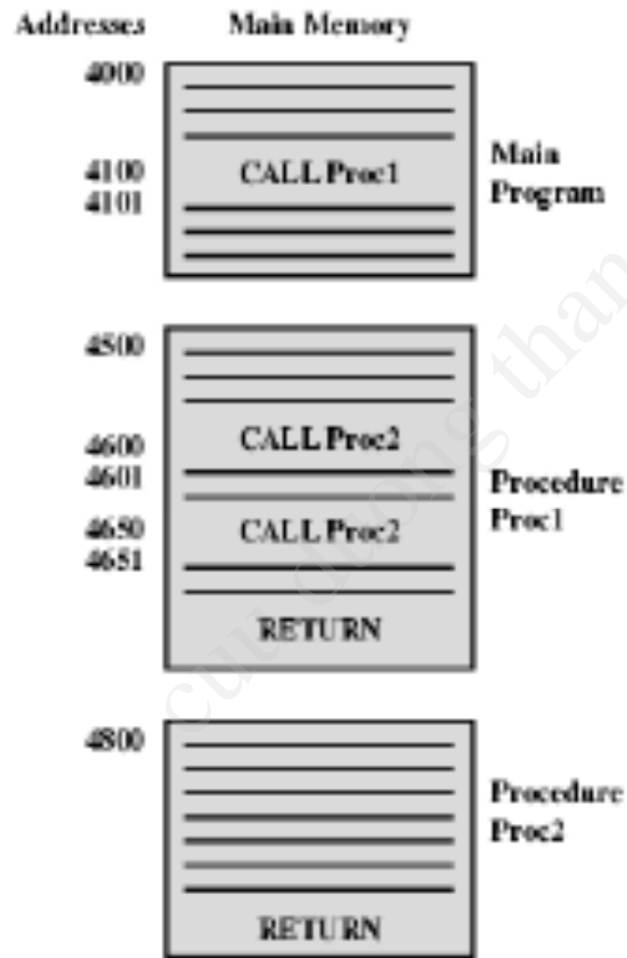
- Tính toán
 - Số học
 - Luận lý
- Lưu trữ dữ liệu: memory
- Di chuyển dữ liệu: I/O
- Điều khiển: kiểm tra, rẽ nhánh, chuyển điều khiển

Ví dụ rẽ nhánh

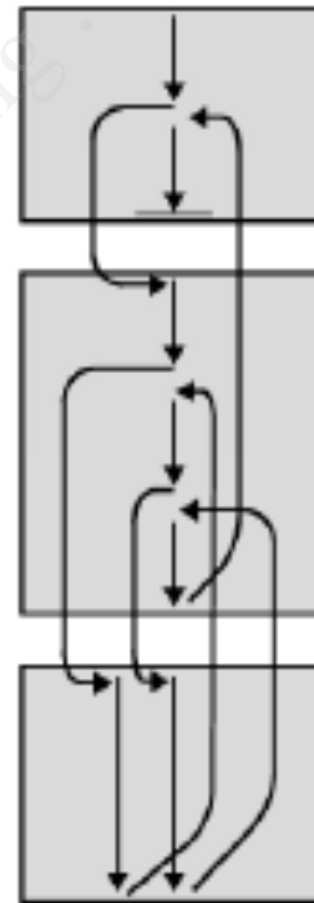




Gọi thủ tục



(a) Calls and returns



(b) Execution sequence



Số lượng địa chỉ

- Bao nhiêu địa chỉ được chứa trong một chỉ thị
- Địa chỉ được biểu diễn tường minh hay không tường minh
 - địa chỉ bộ nhớ
 - thanh ghi, bộ tích lũy (accumulator)
 - chỉ thị không địa chỉ



Các đặc trưng thiết kế

- Tác vụ của lệnh
- Kiểu toán hạng
- Các thanh ghi
- Chế độ địa chỉ



Các kiểu toán hạng

- Địa chỉ
- Con số
 - nguyên (integer)
 - dấu chấm động (floating point)
 - thập phân, nhị phân
- Ký tự
- Luận lý



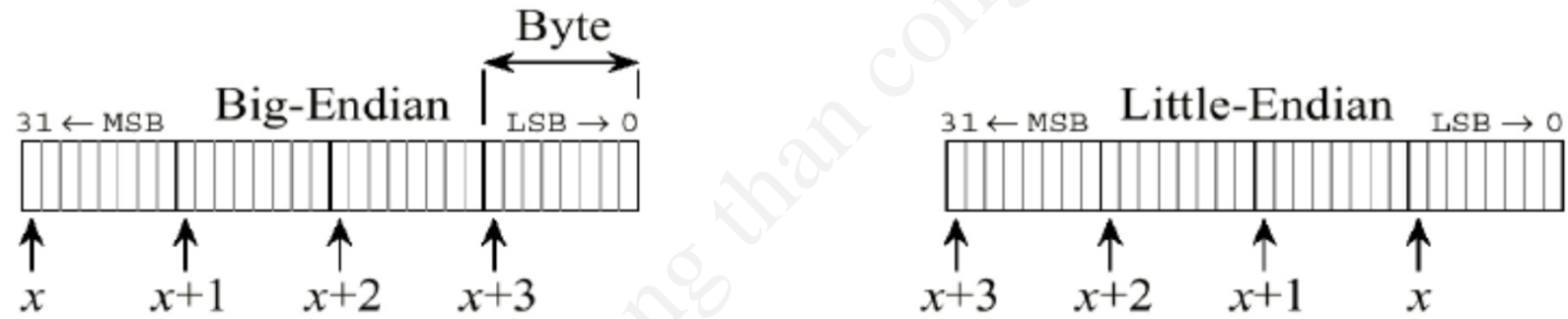
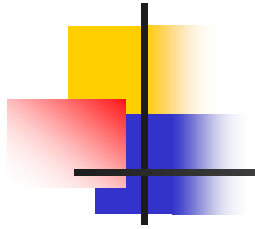
Tập chỉ thị được đo lường qua vài yếu tố

- Kích thước lưu trữ mà chương trình cần
- Độ phức tạp của tập chỉ thị cũng như độ phức tạp của các tác vụ
- Chiều dài của chỉ thị
- Tổng số chỉ thị
- Bao nhiêu thanh ghi và tổ chức các thanh ghi như thế nào



Các cân nhắc thiết kế

- Chỉ thị ngắn hay dài
- Chiều dài cố định hay thay đổi (cố định dễ giải mã nhưng lãng phí)
- Tổ chức bộ nhớ (địa chỉ hóa theo byte hay không)
- Chỉ thị có chiều dài cố định không nhất thiết phải cố định số toán hạng (expanding opcode)
- Chế độ địa chỉ hóa
- Thứ tự lưu giữ các byte của các từ có nhiều byte như thế nào (little endian và big endian)



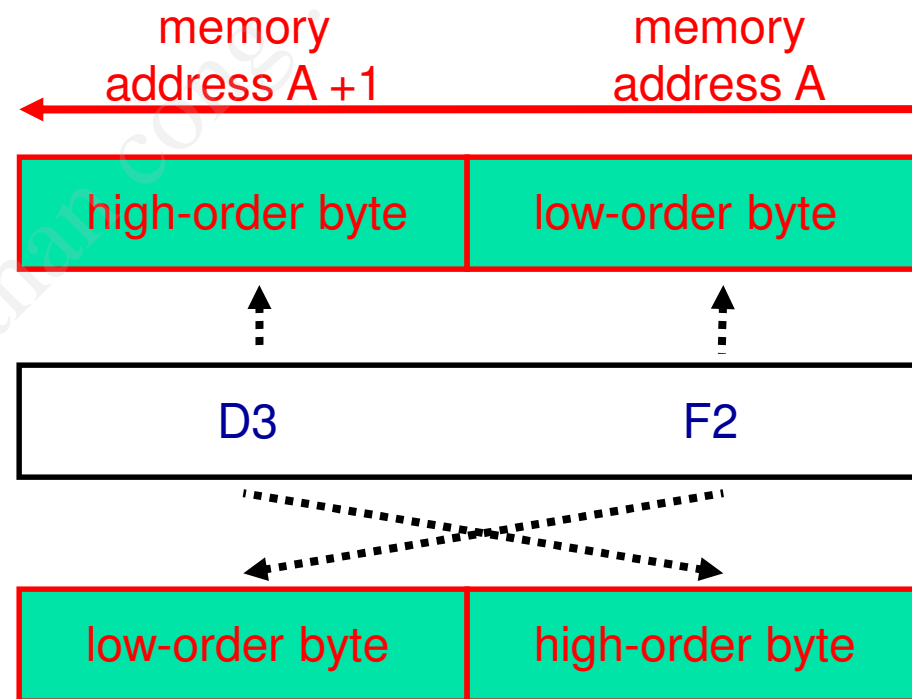
Địa chỉ từ là x cho cả hai minh họa

Thứ tự byte của số nguyên

Lưu ở dạng little-endian

Integer (2 byte)

Lưu ở dạng big-endian





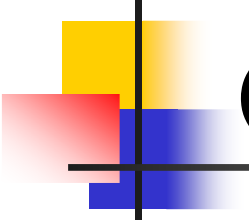
Hỗ trợ lưu trữ bên trong CPU

- Kiến trúc stack: dùng một stack để thực thi chỉ thị, các toán hạng được ngầm định ở đỉnh stack, không thể truy xuất ngẫu nhiên, khó sinh mã hiệu quả.
- Kiến trúc accumulator: một toán hạng ngầm định (không tường minh) trong accumulator, tối thiểu độ phức tạp nhưng lưu lượng bộ nhớ lớn.
- Kiến trúc thanh ghi mục đích tổng quát (GPR: General Purpose Register): dùng một số thanh ghi, truy xuất nhanh. Có hai đặc trưng chính: số lượng toán hạng và cách thức địa chỉ hóa toán hạng



Chế độ địa chỉ (addressing mode)

- ☐ Immediate addressing
- ☐ Direct addressing
- ☐ Indirect addressing
- ☐ Register addressing
- ☐ Register indirect addressing
- ☐ Displacement addressing
- ☐ Based addressing
- ☐ Indexed addressing
- ☐ Stack addressing
- ☐ Các chế độ cải tiến



Địa chỉ tức thời (Immediate addressing)

- Giá trị tham chiếu nằm ngay trong chỉ thị
- Không có tham chiếu bộ nhớ để lấy dữ liệu
- Nhanh
- Ví dụ

ADD #5

- Cộng 5 vào nội dung của thanh ghi AC
- 5 là toán hạng

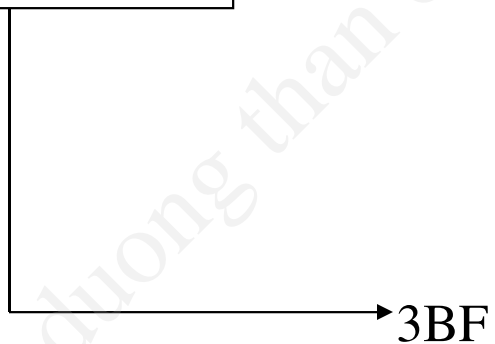
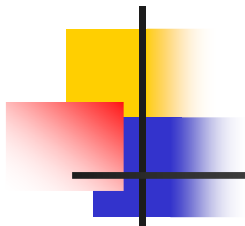


Địa chỉ trực tiếp (Direct addressing)

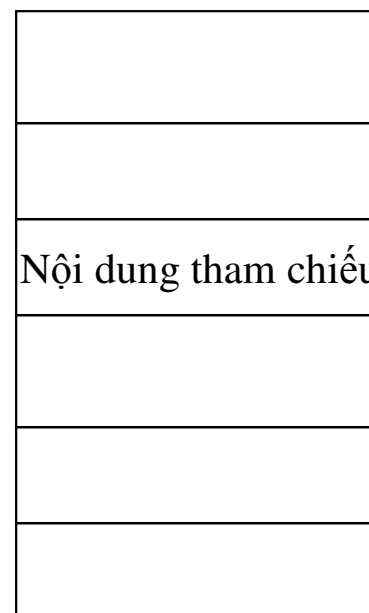
- Toán hạng là địa chỉ của giá trị tham chiếu
- Tham chiếu một vị trí bộ nhớ để truy xuất dữ liệu
- Ví dụ

ADD 3BF

- Cộng nội dung của ô nhớ 3BF với nội dung trong AC
- 3BF là địa chỉ hiệu quả (effective address)



Memory



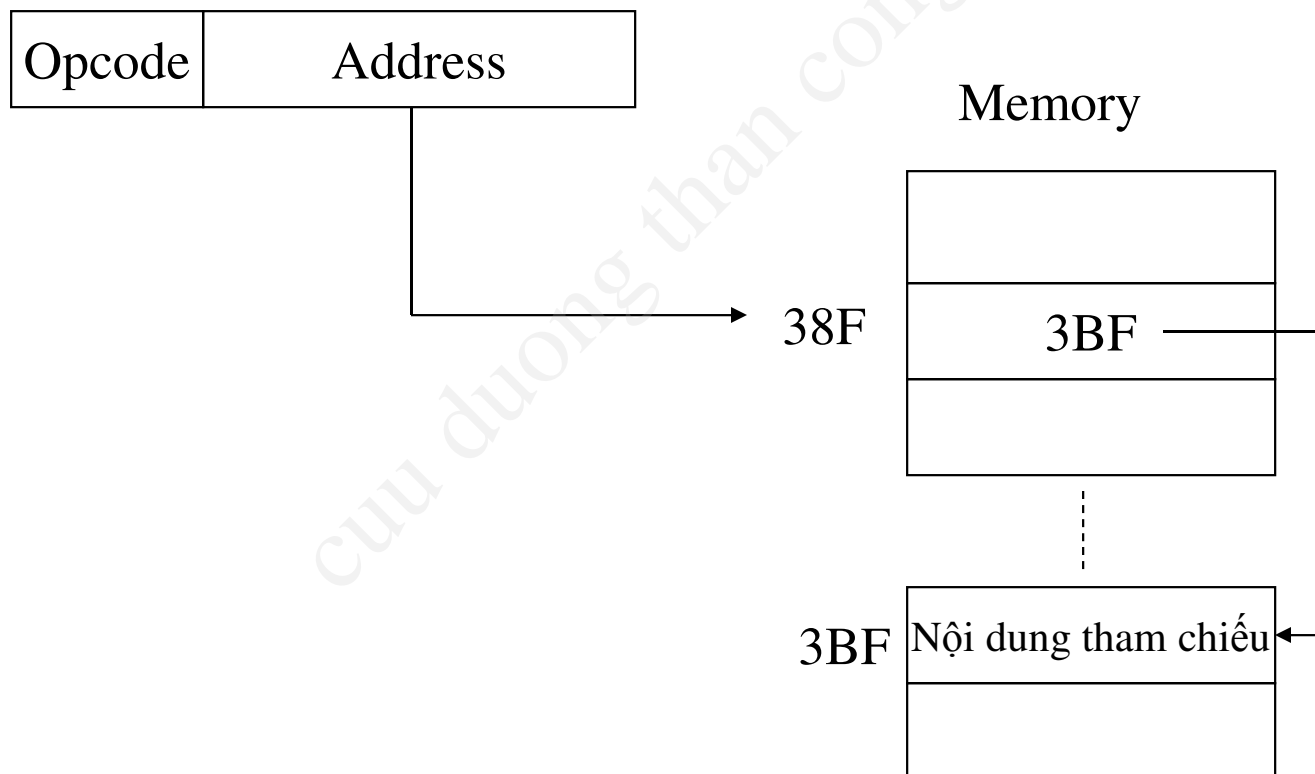
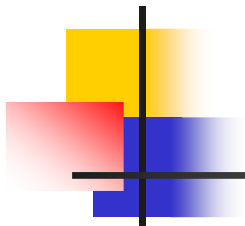


Địa chỉ gián tiếp (indirect addressing)

- Toán hạng là địa chỉ của con trỏ chỉ đến dữ liệu
- Địa chỉ hiệu quả chính là con trỏ
- Ví dụ

ADD 38F

- Cộng nội dung tại ô nhớ có địa chỉ được chứa trong ô nhớ 38F với nội dung trong AC
- 38F không phải là địa chỉ hiệu quả.

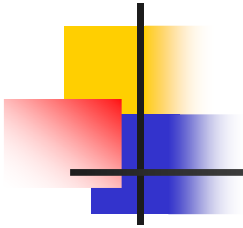




Địa chỉ thanh ghi (Register addressing)

- Toán hạng là tên thanh ghi (địa chỉ thanh ghi)
- Địa chỉ hiệu quả là thanh ghi, nội dung thanh ghi chứa giá trị tham chiếu.
- Truy xuất nhanh
- Ví dụ

ADD R1 cộng nội dung thanh ghi R1 với nội dung của AC

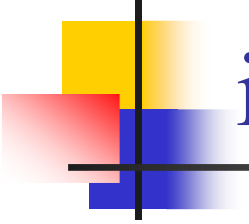


opcode	Địa chỉ thanh ghi
--------	-------------------



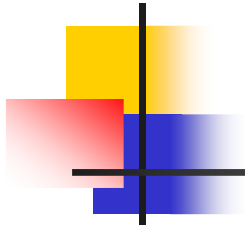
Các thanh ghi

Nội dung tham chiếu

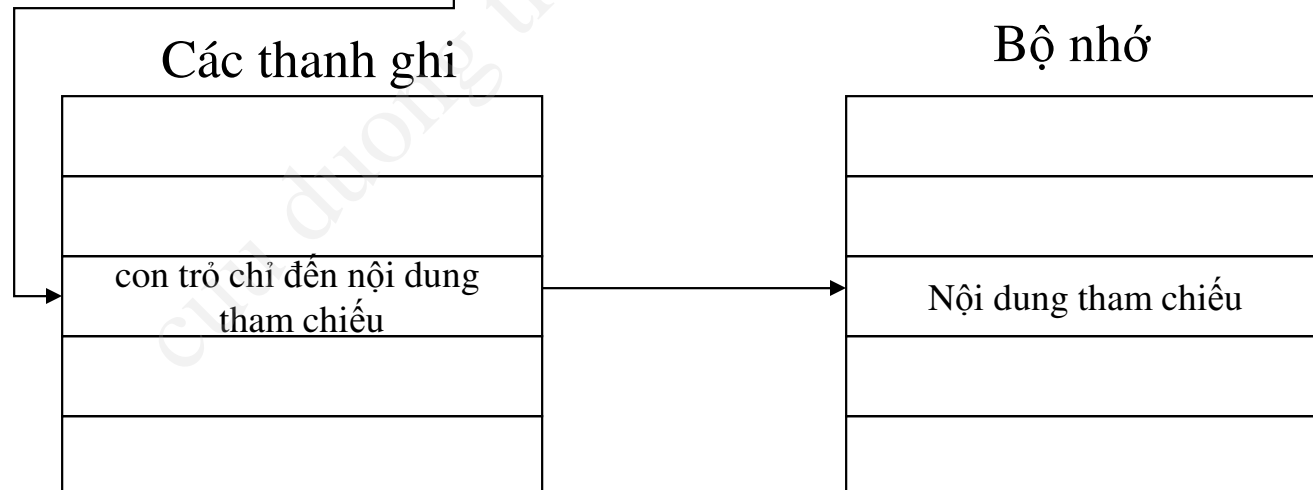


Địa chỉ gián tiếp thanh ghi (register indirect addressing)

- Kết hợp địa chỉ thanh ghi và địa chỉ gián tiếp
- Dùng thanh ghi để chứa con trỏ chỉ đến vị trí chứa giá trị tham chiếu



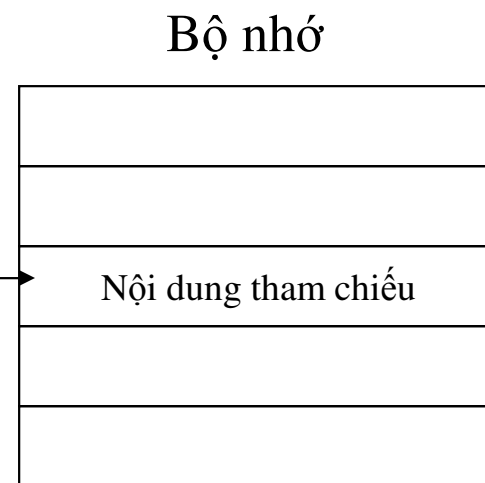
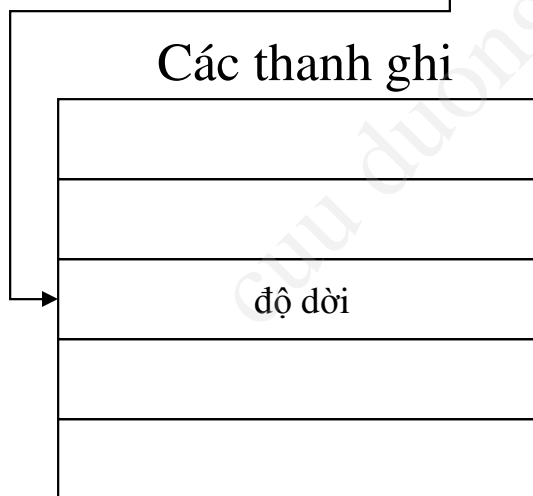
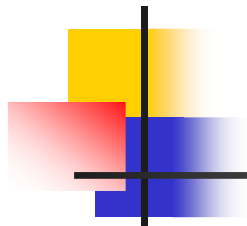
opcode	Địa chỉ thanh ghi
--------	-------------------





Địa chỉ dùng độ dời (displacement addressing)

- Phần toán hạng chứa hai giá trị:
 - Địa chỉ
 - Thanh ghi giữ độ dời (offset)
- Địa chỉ hiệu quả = địa chỉ + độ dời





Relative addressing

- Từ displacement addressing nếu thanh ghi là PC thì gọi là địa chỉ quan hệ (relative addressing);
- Lấy nội dung từ ô nhớ tại vị trí "địa chỉ" tính từ vị trí hiện hành được chỉ ra trong thanh ghi PC.
- Ví dụ LD A
nạp nội dung từ ô nhớ $A+(PC)$ vào thanh ghi AC



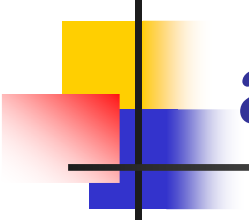
Địa chỉ dùng thanh ghi nền (Base-register addressing)

- Giá trị thứ nhất là độ dời
- Thanh ghi chứa con trỏ chỉ đến địa chỉ nền
- Thanh ghi có thể là tường minh hay không tường minh (ngầm)



Indexed addressing

- Giá trị địa chỉ trong chỉ thị chứa địa chỉ nền
- Thanh ghi chứa độ dời (offset)
- Địa chỉ hiệu quả = địa chỉ nền + độ dời
- Thích hợp cho truy xuất mảng
 - Địa chỉ truy xuất = địa chỉ nền + độ dời trong thanh ghi R
 - R++



Địa chỉ ngăn xếp (stack addressing)

- Các toán hạng được ngầm định trên đỉnh stack

- Ví dụ ADD

Lấy hai giá trị trên đỉnh stack và thực hiện cộng hai giá trị với nhau



Các chế độ địa chỉ cải tiến

- Có thể kết hợp các chế độ địa chỉ với nhau.
- Ví dụ indirect indexed addressing, indirect based register addressing ...



Bài tập

1. Giải thích và cho ví dụ các chế độ địa chỉ cải tiến
 - Indirect indexed addressing
 - Indirect based register addressing
2. Tìm hiểu các chế độ địa chỉ trong máy Pentium