ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

KIẾN TRÚC MÁY TÍNH

Chương 5. KIẾN TRÚC BỘ VI XỬ LÝ

- 5.1 Khái niệm
- 5.2 Kiểu toán hạng và toán tử
- 5.3 Định dạng lênh
- 5.4 Các phương pháp đánh địa chỉ

5.1 KHÁI NIỆM

- Lệnh và tập lệnh máy tính
- Biểu diễn lệnh
- Kiểu lệnh
- Số lượng địa chỉ
- Thiết kế tập lệnh

5.1 KHÁI NIỆM

- Lệnh và tập lệnh máy tính
- Biểu diễn lệnh
- Kiểu lệnh
- Số lượng địa chỉ
- Thiết kế tập lệnh

LỆNH VÀ TẬP LỆNH MÁY TÍNH

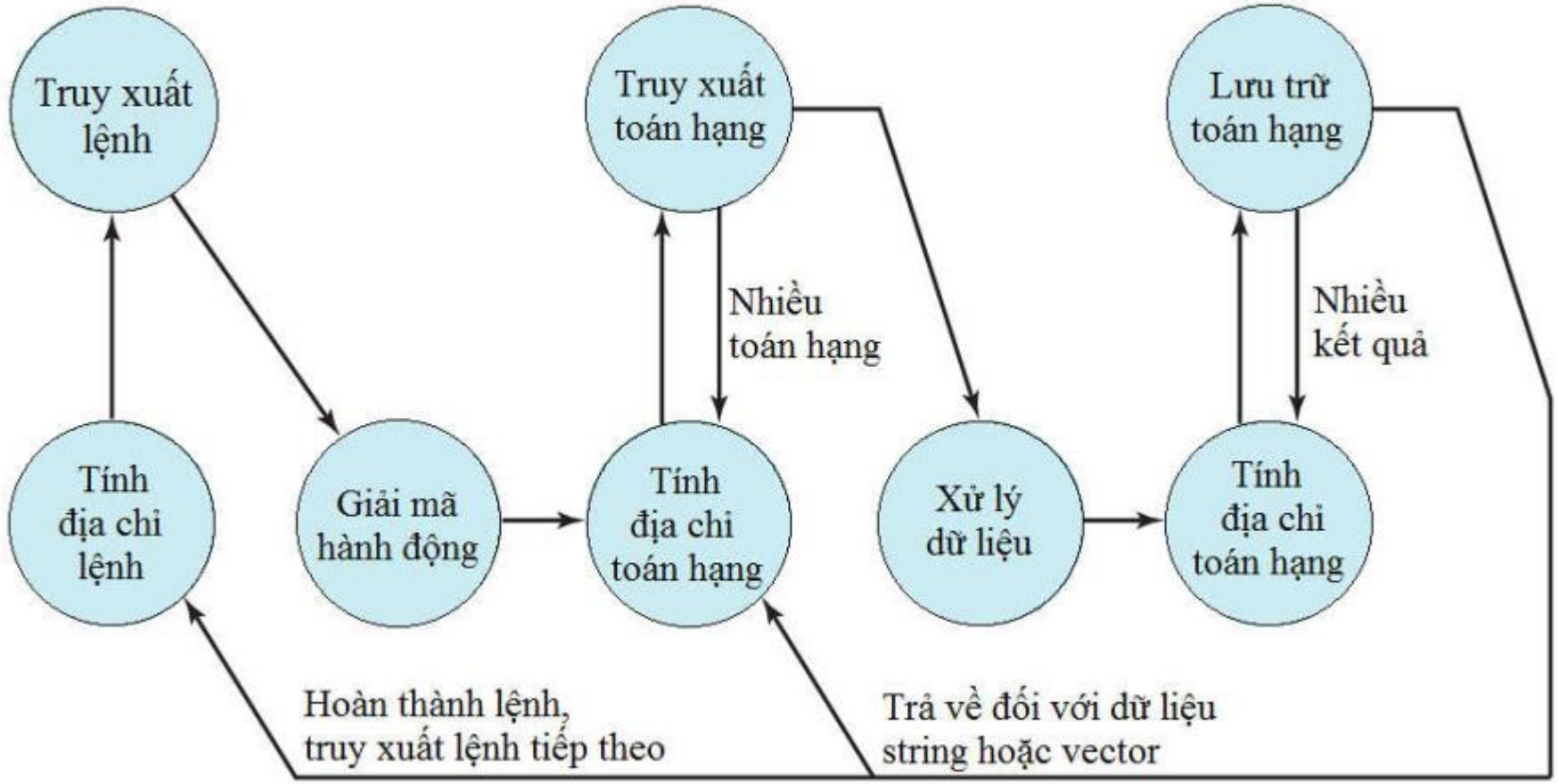
- Mỗi bộ vi xử lý có một tập lệnh xác định
- Tập lệnh thường có hàng chục đến hang trăm lệnh
- Mỗi lệnh là một chuỗi các số nhị phân mà bộ xử lý hiểu được để thực hiện một thao tác nhất định
- Mỗi lệnh phải có những thông tin cần thiết cho bộ xử lý thực hiện hành động

LỆNH VÀ TẬP LỆNH MÁY TÍNH

- Các thành phần của lệnh:
  - **Mã lệnh – (opcode):** chỉ ra hành động cần thực hiện (Ví dụ: ADD, I/O). Opcode được xác định bởi một mã nhị phân.
  - **Địa chỉ toán hạng nguồn**: một hành động có thể liên quan đến một hoặc nhiều toán hạng nguồn. Toán hạng nguồn là đầu vào của hành động.
  - **Địa chỉ toán hạng đích:** một hành động có thể tạo ra một kết quả đích
  - **Địa chỉ lệnh tiếp theo:** chỉ cho VXL nơi để lấy lệnh tiếp theo sau khi hoàn thành lệnh này

LỆNH VÀ TẬP LỆNH MÁY TÍNH

- Sơ đồ trạng thái chu kỳ lệnh

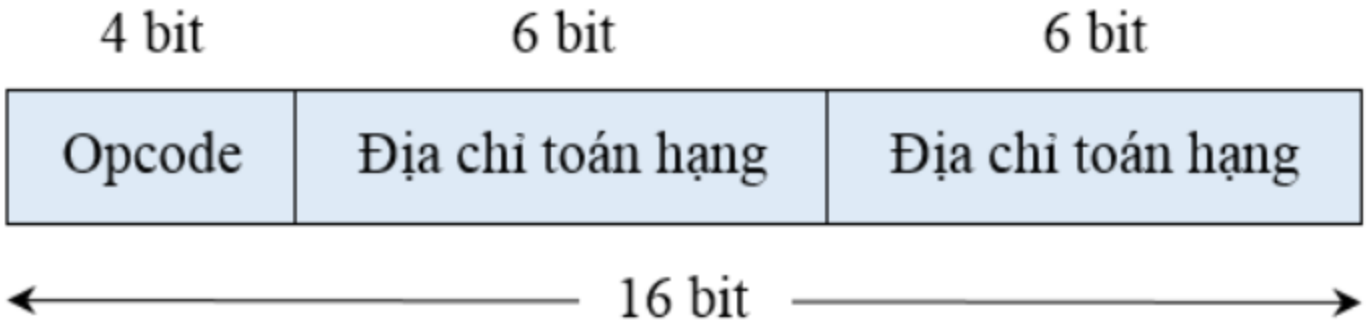


5.1 KHÁI NIỆM

- Lệnh và tập lệnh máy tính
- Biểu diễn lệnh
- Kiểu lệnh
- Số lượng địa chỉ
- Thiết kế tập lệnh

BIỂU DIỄN LỆNH

- Trong máy tính, mỗi lệnh được biểu diễn bằng một chuỗi bit
- Lệnh được chia ra thành các trường tương ứng với cánh thành phần cấu thành nên lệnh
- Để thuận tiện, sử dụng biểu diễn ký tự gợi nhớ
  - Ví dụ: ADD, SUB, LOAD
  - Ví dụ: ADD R, Y



Một định dạng lệnh đơn giản

5.1 KHÁI NIỆM

- Lệnh và tập lệnh máy tính
- Biểu diễn lệnh
- Kiểu lệnh
- Số lượng địa chỉ
- Thiết kế tập lệnh

Ví dụ:

1 lệnh ngôn ngữ bậc cao : **X=X+Y**

Giả sử: X=513, Y=514

Các lệnh máy thực hiện:

1. Tải nội dung của vị trí nhớ 513 vào 1 thanh ghi
2. Cộng nội dung của vị trí nhớ 514 vào thanh ghi đó
3. Lưu trữ nội dung của thanh ghi vào vị trí bộ nhớ 513

KIỂU LỆNH

- Xử lý dữ liệu
  - Lệnh số học cung cấp khả năng tính toán để xử lý dữ liệu số.
  - Lệnh logic (Boolean) thực hiện các bit của một từ dưới dạng bit chứ không phải là số, do đó chúng cung cấp khả năng xử lý bất kỳ loại dữ liệu nào mà người dùng muốn.
- Lưu trữ dữ liệu
  - Việc chuyển dữ liệu vào hay ra thanh ghi và/ hoặc bộ nhớ.
- Di chuyển dữ liệu
  - Lệnh vào/ra được sử dụng để truyền chương trình và dữ liệu vào bộ nhớ, truyền kết quả tính toán lại cho người dùng.
- Điều khiển
  - Lệnh kiểm tra: kiểm tra giá trị của dữ liệu hoặc trạng thái của 1 phép toán.
  - Lệnh rẽ nhánh: rẽ nhánh tới 1 tập lệnh khác dựa vào quyết định được đưa ra.

5.1 KHÁI NIỆM

- Lệnh và tập lệnh máy tính
- Biểu diễn lệnh
- Kiểu lệnh
- Số lượng địa chỉ
- Thiết kế tập lệnh

SỐ LƯỢNG ĐỊA CHỈ

- 3 địa chỉ
  - Toán hạng 1, Toán hạng 2, Kết quả
  - c= a+b;
  - Từ lệnh phải rất dài để lưu được cả 3 địa chỉ của 3 toán hạng
  - Ví dụ: Y=(A-B)/[C+ (D*E)]
    - T – vị trí tạm thời, lưu trữ kết quả trung gian

Lệnh		Giải thích
SUB	Y, A, B	Y ← A – B
MPY	T, D, E	T ← D × E
ADD	T, T, C	T ← T + C
DIV	Y, Y, T	Y ← Y ÷ T

(a) Lệnh có ba địa chỉ

Lệnh		Giải thích
MOVE	Y, A	Y ← A
SUB	Y, B	Y ← Y – B
MOVE	T, D	T ← D
MPY	T, E	T ← T × E
ADD	T, C	T ← T + C
DIV	Y, T	Y ← Y ÷ T

(b) Lệnh có hai địa chỉ

Lệnh		Giải thích
LOAD	D	AC ← D
MPY	E	AC ← AC × E
ADD	C	AC ← AC + C
STOR	Y	Y ← AC
LOAD	A	AC ← A
SUB	B	AC ← AC – B
DIV	Y	AC ← AC ÷ Y
STOR	Y	Y ← AC

(c) Lệnh có một địa chỉ

SỐ LƯỢNG ĐỊA CHỈ

- 2 địa chỉ
  - 1 địa chỉ phải nhận 2 vai trò là toán hạng nguồn và kết quả, địa chỉ còn lại là của toán hạng nguồn.
  - a = a + b
  - Giảm độ dài lệnh
  - Phát sinh thêm công việc: Lưu trữ tạm thời một số kết quả
  - Y = (A-B)/[C+(D*E)]

Instruction	Comment
MOVE Y, A	Y ← A
SUB Y, B	Y ← Y – B
MOVE T, D	T ← D
MPY T, E	T ← T x E
ADD T, C	T ← T + C
DIV Y, T	Y ← Y ÷ T

SỐ LƯỢNG ĐỊA CHỈ

❑ 1 địa chỉ

- Một toán hạng được chỉ ra trong lênh
- Một toán hạng là ngầm định → thường là 1 thanh ghi (AC - tích lũy)
- Phổ biến trong các máy tính đời đầu
- Y= (A-B)/[C+ (D*E)]

Instruction	Comment
LOAD D	AC ← D
MPY E	AC ← AC x E
ADD C	AC ← AC + C
STOR Y	Y ← AC
LOAD A	AC ← A
SUB B	AC ← AC - B
DIV Y	AC ← AC ÷ Y
STOR Y	Y ← AC

SỐ LƯỢNG ĐỊA CHỈ

- 0 địa chỉ
  - Tất cả địa chỉ toán hạng được ngầm định
  - Sử dụng ngăn xếp (stack)
    - Stack là các vị trí nhớ LIFO
  - Ví dụ:
    - push a
    - push b
    - add
    - pop c
    - Có nghĩa là: c=a+b
  - Không thông dụng

SỐ LƯỢNG ĐỊA CHỈ

- Số lượng địa chỉ bao nhiêu là đủ?
  - Nhiều địa chỉ hơn
    - Lệnh phức tạp hơn, dài hơn
    - Sử dụng nhiều thanh ghi hơn
    - Ít lệnh trên 1 chương trình
  - Ít địa chỉ hơn
    - Lệnh ít phức tạp hơn, ngắn hơn
    - Truy xuất/ thực thi lệnh nhanh hơn
    - Nhiều lệnh trên 1 chương trình → chương trình dài, phức tạp hơn

5.1 KHÁI NIỆM

- Lệnh và tập lệnh máy tính
- Biểu diễn lệnh
- Kiểu lệnh
- Số lượng địa chỉ
- Thiết kế tập lệnh

THIẾT KẾ TẬP LỆNH

- Các vấn đề thiết kế tập lệnh
  - Danh sách các hành động
    - Bao nhiêu hành động? Hành động nào? Độ phức tạp của hành động?
  - Các kiểu dữ liệu
    - Các kiểu dữ liệu mà các hành động tham chiếu đến.
  - Định dạng lệnh
    - Độ dài lệnh (bit), số lượng địa chỉ, kích thước các trường, …
  - Các thanh ghi
    - Số lượng thanh ghi của VXL mà lệnh có thể tham chiếu đến?
    - Cách sử dụng thanh ghi?
  - Các phương pháp định địa chỉ
    - Một hoặc nhiều chế độ định địa chỉ của toán hạng.
  - RISC hay CISC
    - Reduced Instruction Set Computing.
    - Complex Instruction Set Computing.

Chương 5. KIẾN TRÚC BỘ VI XỬ LÝ

- 5.1 Khái niệm
- 5.2 Kiểu toán hạng và toán tử
- 5.3 Định dạng lênh
- 5.4 Các phương pháp đánh địa chỉ
- 5.5 Tổng kết và làm bài ôn tập

KIỂU TOÁN HẠNG VÀ TOÁN TỬ

- Các kiểu dữ liệu quan trọng:
  - Địa chỉ
  - Số
  - Ký tự
  - Dữ liệu logic

KIỂU TOÁN HẠNG VÀ TOÁN TỬ

- Số
  - Tất cả các ngôn ngữ máy đều có dữ liệu dạng số
  - Số được lưu trữ trong máy tính là hữu hạn
    - Độ lớn của số biểu diễn được trên máy bị giới hạn.
    - Độ chính xác bị giới hạn (số dấy chấm động).
  - Ba kiểu dữ liệu số thông thường
    - số nguyên nhị phân hoặc số nhị phân dấu chấm tính.
    - Số nhị phân dấu chấm động.
    - Số thập phân.
  - Packed decimal
    - Mỗi chữ số thập phân được biểu diễn bởi một mã 4 bit, 2 chữ số được lưu trữ trên một byte.
    - Để tạo số, các mã 4 bit được nối với nhau, thường là bội của 8 bit.

KIỂU TOÁN HẠNG VÀ TOÁN TỬ

- Ký tự
  - Một trong những dạng dữ liệu cơ bản là văn bản (text) hoặc xâu ký tự (character strings).
  - Dữ liệu văn bản dưới dạng ký tự không thể lưu trữ hoặc truyền qua hệ thống xử lý dữ liệu và truyền thông vì các hệ thống này được thiết kế cho dữ liệu nhị phân.
  - Bảng mã mã hóa ký tự được sử dụng phổ biến nhất là bảng mã IRA (International Reference Alphabet).
    - Còn được gọi là mã ASCII (American Standard Code for Information)
  - Bảng mã EBCDIC (Exdended Binary Coded Decimal Interchange Code) được sử dụng trong các máy tính mainframe của IBM.

KIỂU TOÁN HẠNG VÀ TOÁN TỬ

- Dữ liệu logic
  - Một khối n-bit gồm n phần tử dữ liệu 1 bit, mỗi phần tử có giá trị 0 hoặc 1.
  - Hai ưu điểm của view theo bit:
    - Bộ nhớ được sử dụng hiệu quả nhất để lưu trữ một mảng các phần tử dữ liệu nhị phân/ Boolean, trong đó mỗi phần tử chỉ nhận giá trị 1 (đúng) hoặc 0 (sai)
    - Để thao tác trên các bit của một phần dữ liệu
      - Nếu phép toán dấu chấm động thực hiện trong phần mền, trong một số phép toán ta cần phải dịch các bit có nghĩa
      - Để chuyển đổi từ IRA thánh packed decimal, ta cần tricshh xuất 4 bit bên phải của mỗi byte.

Chương 5. KIẾN TRÚC BỘ VI XỬ LÝ

- 5.1 Khái niệm
- 5.2 Kiểu toán hạng và toán tử
- 5.3 Định dạng lênh
- 5.4 Các phương pháp đánh địa chỉ

5.3. ĐỊNH DẠNG LỆNH

- Độ dài lệnh
- Phân biệt các thành phần trong cấu trúc lệnh
- Lệnh có độ dài thay đổi

5.3 ĐỊNH DẠNG LỆNH

- Định nghĩa cách bố trí của các bit trong 1 lệnh (trong các trường thành phần của nó).
- Bao gồm:
  - 1 opcode
  - Không hoặc nhiều toán hạng, ngầm định hoặc rõ ràng
- Phải ngầm định hoặc rõ ràng chỉ ra chế độ địa chỉ cho từng toán hạng.
- Với mỗi tập lệnh, có thể sử dụng nhiều định dạng.

5.3. ĐỊNH DẠNG LỆNH

- Độ dài lệnh
- Phân biệt các thành phần trong cấu trúc lệnh
- Lệnh có độ dài thay đổi

ĐỘ DÀI LỆNH

- Vấn đề thiết kế cơ bản nhất
- Ảnh hưởng và bị ảnh hưởng bởi:
  - Kích thước bộ nhớ
  - Tổ chức bộ nhớ
  - Cấu trúc bus
  - Độ phức tạp của bộ xử lý
  - Tốc độ xử lý
- Trade off giữa hiệu quả của lệnh và tiết kiệm không gian

ĐỘ DÀI LỆNH

- Cần xem xét:
  - Độ dài lệnh có bằng độ dài dữ liệu bộ nhớ truyền đi. Hoặc một trong hai cái nên là bội của cái còn lại.
  - Tốc độ truyền bộ nhớ chậm hơn tốc độ VXL
  - Độ dài lệnh nên là bội của độ dài ký tự (thường là 8 bit) và bội của độ dài số dấu chấm tĩnh.

5.3. ĐỊNH DẠNG LỆNH

- Độ dài lệnh
- Phân biệt các thành phần trong cấu trúc lệnh
- Lệnh có độ dài thay đổi

PHÂN BIỆT CÁC THÀNH PHẦN CẤU TRÚC TRONG LỆNH

- Các yếu tố ảnh hưởng đến việc sử dụng bit địa chỉ trong 1 lệnh
  - Số lượng chế độ địa chỉ
  - Số lượng toán hạng
  - Thanh ghi hay bộ nhớ
  - Số lượng tập thanh ghi
  - Dải địa chỉ
  - Độ chi tiết địa chỉ

ĐỊNH DẠNG LỆNH PDP - 8

- 35 lệnh
- Từ lệnh 12 bit:
  - 3 bit opcode → 8 hành động:
    - 6 hành động cơ bản
    - 110= lệnh I/O
    - 111= tham chiếu thanh ghi
  - Lệnh tham chiếu bộ nhớ:
    - 7 bit địa chỉ
    - 2 bit điều khiển
  - Lệnh I/O:
    - 6 bit địa chỉ thiết bị
    - 3 bit mệnh lệnh I/O
  - Lệnh tham chiếu thanh ghi:
    - Mỗi bit = 1 hành động

Các lệnh tham chiếu bộ nhớ

Opcode			D/I	Z/C	Dịch chuyển						
0		2	3	4	5						11

Các lệnh vào/ra

1	1	0	Thiết bị						Opcode		
0	1	2	3					8	9		11

Các lệnh tham chiếu thanh ghi

Các vi lệnh nhóm 1

1	1	1	0	CLA	CLL	CMA	CML	RAR	RAL	BSW	IAC
0	1	2	3	4	5	6	7	8	9	10	11

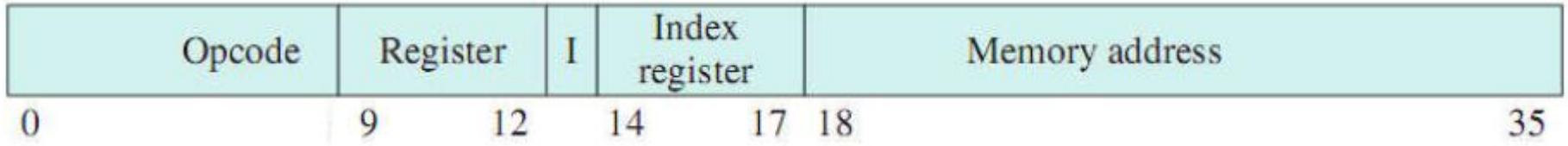
Các vi lệnh nhóm 2

1	1	1	0	CLA	SMA	SZA	SNL	RSS	OSR	HLT	0
0	1	2	3	4	5	6	7	8	9	10	11

Các vi lệnh nhóm 3

1	1	1	0	CLA	MQA	0	MQL	0	0	0	1
0	1	2	3	4	5	6	7	8	9	10	11

ĐỊNH DẠNG LỆNH PDP - 10



I= indirect bit

- Từ lệnh 36 bit, độ dài cố định
- 365 hành động
- 9 bit opcode
- Lệnh 2 địa chỉ
  - 4 bit địa chỉ thanh ghi
  - 18 bit địa chỉ tức thời/ địa chỉ bộ nhớ
- Thiết kế "xa xỉ"!

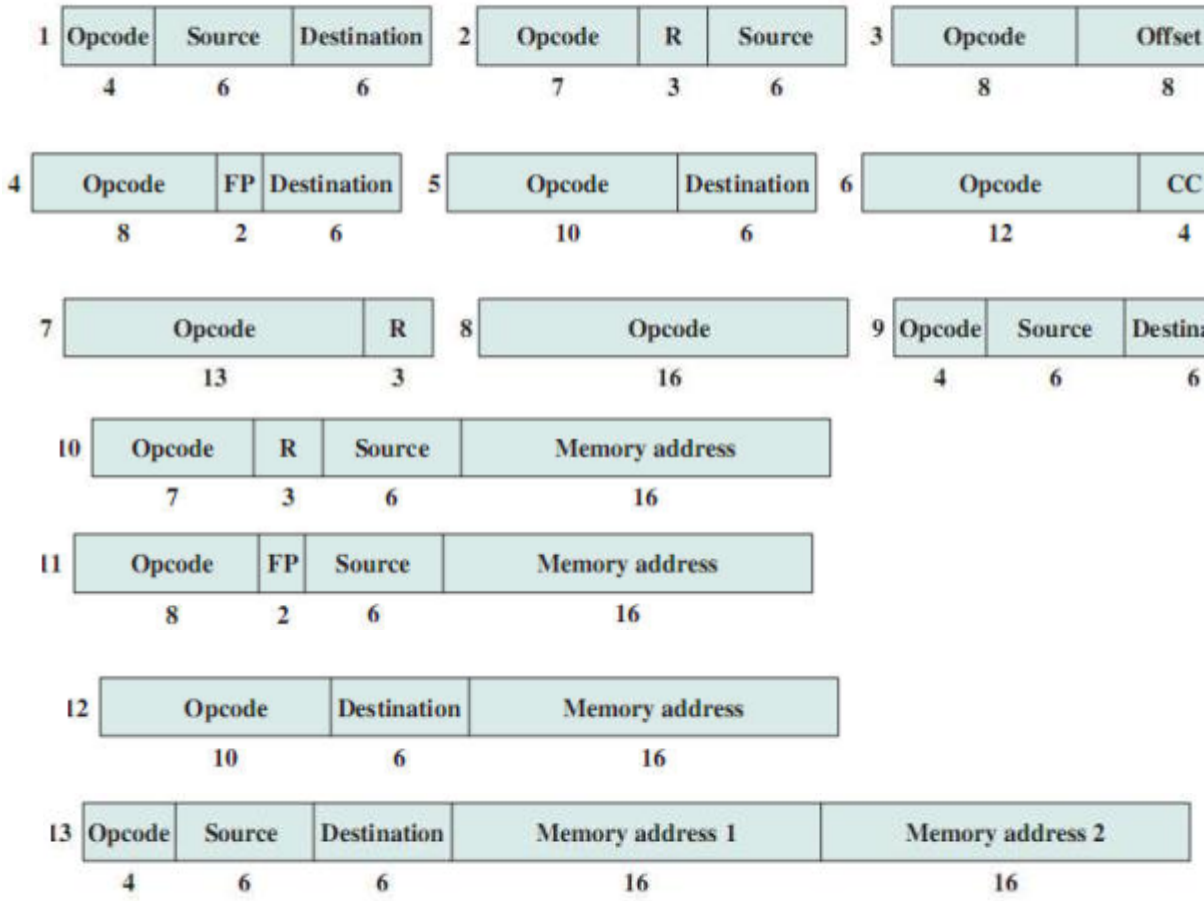
5.3. ĐỊNH DẠNG LỆNH

- Độ dài lệnh
- Phân bố các thành phần trong format lệnh
- Lệnh có độ dài thay đổi

LỆNH CÓ ĐỘ DÀI THAY ĐỔI

- Lệnh có độ dài thay đổi: Các lệnh có khuôn dạng, độ dài khác nhau
  - Độ dài opcode tối thiểu
  - Một vài opcode dùng nhiều bit hơn để chỉ định nhiều hành động hơn (ví dụ: lệnh cần ít toán hạng hơn và/hoặc địa chỉ đơn giản)
- Hiệu quả và linh hoạt hơn
- Tăng sự phức tạp của VXL
- Đòi hỏi lập trình viên hiểu biết nhiều hơn về nguyên lý thiết kế VXL

KHUÔN DẠNG LỆNH PDP-11



- ❑ Tập lệnh có độ dài thay đổi
- ❑ 13 khuôn dạng khác nhau

- ✓ Số ở dưới các trường biểu thị số bit của trường tương ứng.
- ✓ Nguồn, đích: toán hạng nguồn và toán hạng đích. Các trường này gồm 3 bit xác định chế độ địa chỉ và d3 bit xác định một thanh ghi.
- ✓ FP chỉ ra một trong bốn thanh ghi dâu chấm động
- ✓ R chỉ ra một trong các thanh ghi đa năng
- ✓ CC là trường mã điều kiện

Chương 5. KIẾN TRÚC BỘ VI XỬ LÝ

- 5.1 Khái niệm
- 5.2 Kiểu toán hạng và toán tử
- 5.3 Định dạng lênh
- 5.4 Các phương pháp đánh địa chỉ

5.4. CÁC PHƯƠNG PHÁP ĐÁNH ĐỊA CHỈ

- Chế độ địa chỉ là phương thức xác định vị trí hoặc giá trị của một toán hạng.
- Giá trị của trường chế độ quyết định chế độ địa chỉ nào được sử dụng.

5.4. CÁC PHƯƠNG PHÁP ĐÁNH ĐỊA CHỈ

- Tức thời
- Trực tiếp
- Gián tiếp thông qua bộ nhớ chính
- Gián tiếp thông qua thanh ghi
- Sử dụng stack
- Gián tiếp kết hợp dịch chuyển
- Các kiểu đánh địa chỉ của Intel, Ultraspac

CÁC CHẾ ĐỘ ĐỊNH ĐỊA CHỈ CƠ BẢN

Chế độ	Thuật toán	Ưu điểm	Nhược điểm
Tức thời	Toán hạng = A	Không cần tham chiếu bộ nhớ	Hạn chế về giá trị của toán hạng
Trực tiếp	EA = A	Đơn giản	Không gian địa chỉ hạn chế
Gián tiếp	EA = (A)	Không gian địa chỉ lớn	Tham chiếu bộ nhớ nhiều lần
Thanh ghi	EA = R	Không cần tham chiếu bộ nhớ	Không gian địa chỉ hạn chế
Gián tiếp thanh ghi	EA = A + (R)	Linh hoạt	Phức tạp
Ngăn xếp	EA= đỉnh ngăn xếp	Không cần tham chiếu bộ nhớ	Khả năng ứng dụng ít

- A= nội dung trường address trong từ lệnh
- R = tên thanh ghi được tham chiếu đến
- EA = địa chỉ hiệu dụng của vị trí chứa toán hạng (địa chỉ bộ nhớ chính hoặc thanh ghi)
- (X) = Nội dung của vị trí bộ nhớ X hoặc thanh ghi X

ĐỊA CHỈ TỨC THỜI

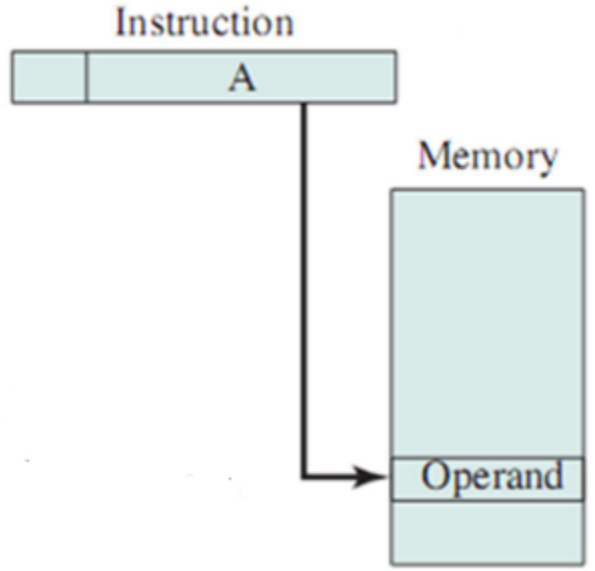
- Dạng định địa chỉ đơn giản nhất
- Toán hạng = A (nằm ngay trong trường địa chỉ)
- Sử dụng để định nghĩa hằng số hoặc thiết lập giá trị ban đầu của biến
- Ví dụ:

```
ADD 5          ; AC ← AC + 5
ADD R1, 5      ; R1 ← R1 + 5
```

- Ưu điểm:
  - Không cần tham chiếu bộ nhớ để lấy dữ liệu toán hạng
  - Truy xuất toán hạng rất nhanh → tiết kiệm một chu kỳ cache hoặc bộ nhớ trong chu kỳ lệnh.
- Nhược điểm: Kích thước của số bị giới hạn bởi kích thước của trường địa chỉ

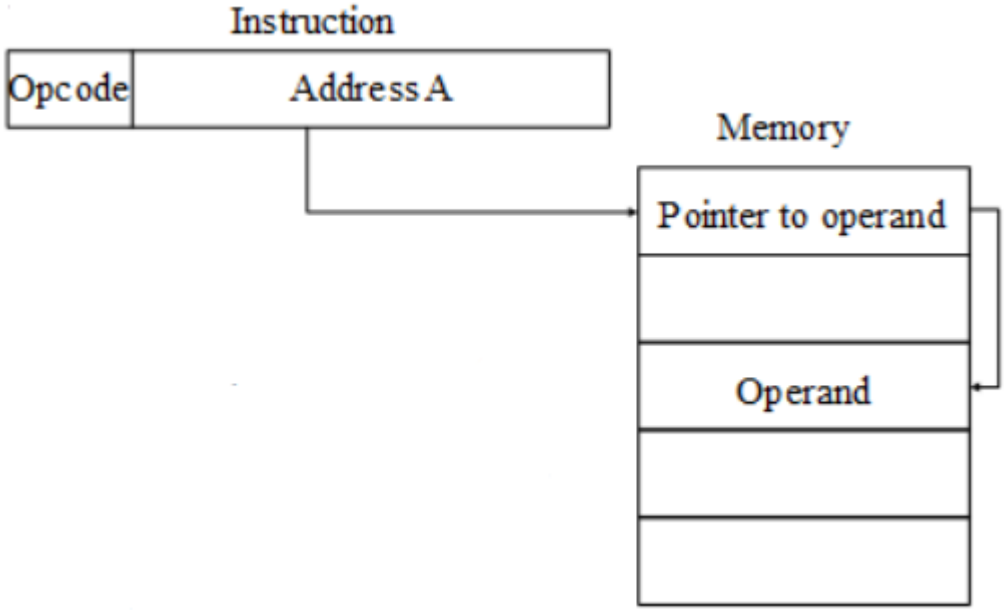
ĐỊA CHỈ TRỰC TIẾP

- Địa chỉ hiệu dụng (EA) = trường địa chỉ (A)
- Trường địa chỉ chứa địa chỉ hiệu dụng của toán hạng
- Ví dụ: ADD R1, A ; R1 ← R1 + A
  - Cộng nội dung thanh ghi R1 với nội dung của ngăn nhớ có địa chỉ là A
  - Tìm toán hạng trong bộ nhớ ở địa chỉ A
- Phổ biến trong các thế hệ máy tính trước đây
- CPU chỉ tham chiếu bộ nhớ 1 lần để lấy dữ liệu
- Không cần tính toán để tìm EA
- Hạn chế: chỉ cung cấp một không gian địa chỉ nhỏ
- Ưu điểm: lưu dữ liệu/ truy xuất dữ liệu trong bộ nhớ



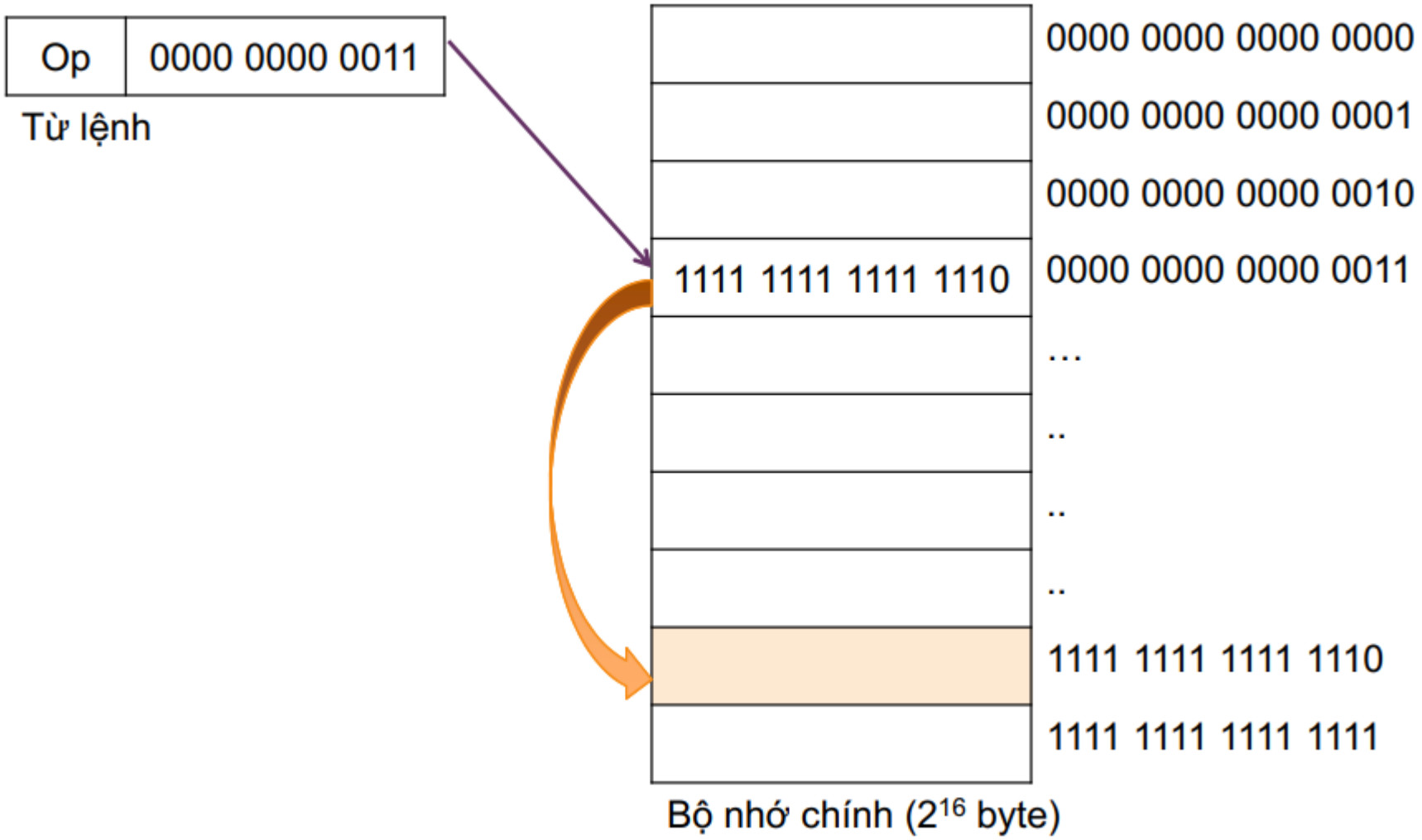
ĐỊA CHỈ GIÁN TIẾP

- Trường địa chỉ tham chiếu đến địa chỉ của một từ trong bộ nhớ chứa địa chỉ đầy đủ của toán hạng
- EA = (A)
  - Đến địa chỉ A trong bộ nhớ, lấy địa chỉ (A), tới (A) để lấy toán hạng
- Ứng dụng:
  - Cập nhật/ thiết lập phần tử
  - Truy xuất phần tử
  - Lưu /di chuyển phần tử



ĐỊA CHỈ GIÁN TIẾP

- Ví dụ:

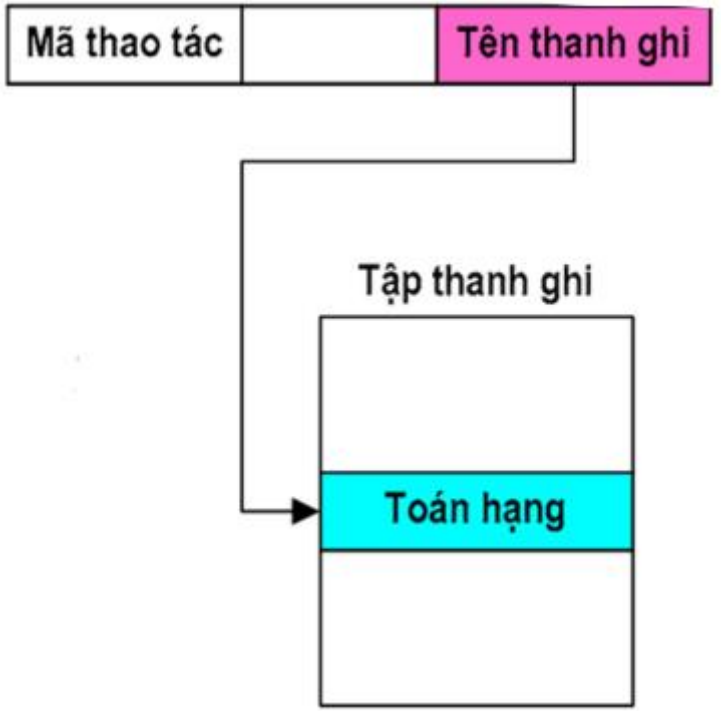


ĐỊA CHỈ GIÁN TIẾP

- Ưu điểm:
  - Không gian địa chỉ lớn: Với một từ có kích thước N cho phép một không gian địa chỉ là 2^N
- Nhược điểm:
  - Một thực thi lệnh đòi hỏi hai lần tham chiếu bộ nhớ để truy xuất toán hạng: 1 là để lấy ra địa chỉ, 2 là để lấy ra giá trị của nó

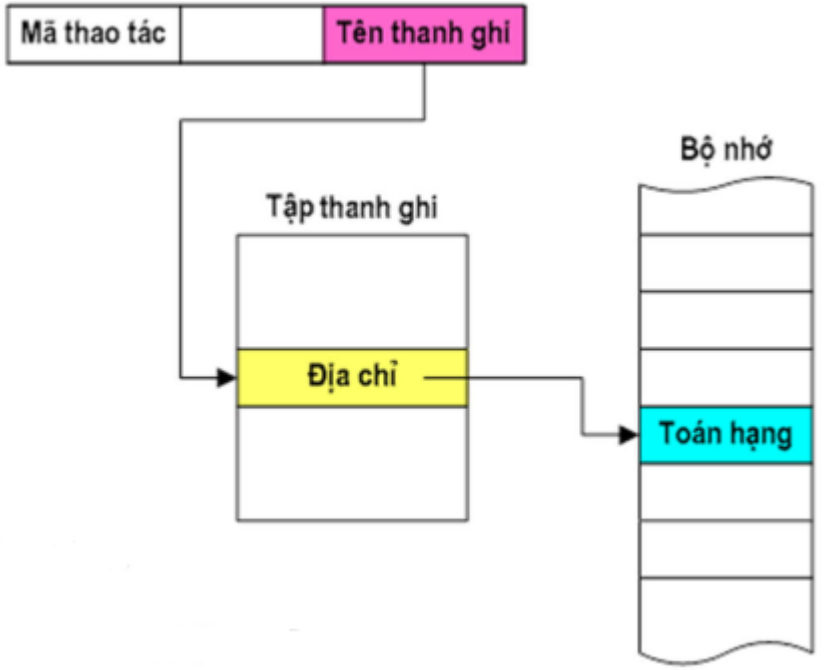
ĐỊA CHỈ THANH GHI

- Trường địa chỉ tham chiếu đến thanh ghi có chưa toán hạng
- EA= R
- Ví dụ: ADD R1, R2 ; R1 ← R1 + R2
- Ưu điểm:
  - Số lượng thanh ghi ít → trường địa chỉ cần ít bit → lệnh ngắn hơn, truy xuất lệnh nhanh hơn
  - Không cần tham chiếu bộ nhớ
- Nhược điểm:
  - Không gian địa chỉ giới hạn



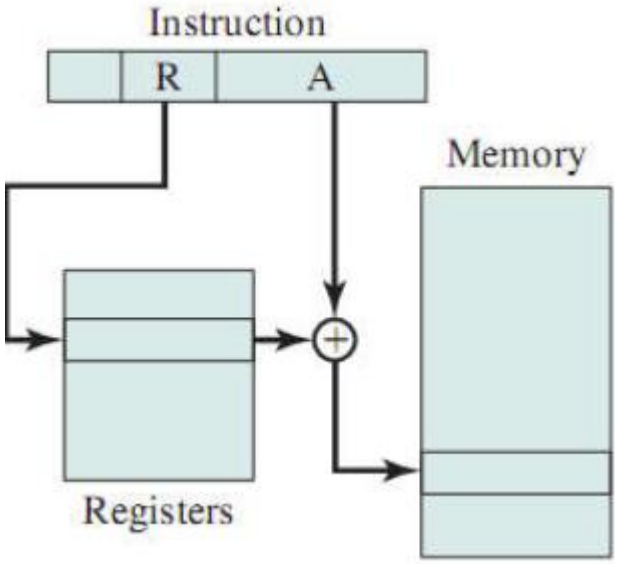
ĐỊA CHỈ GIÁN TIẾP THANH GHI

- EA = (R)
- Toán hạng nằm trong ô nhớ có địa chỉ nằm trong thanh ghi R
- Trường địa chỉ cho biết tên thanh ghi
- Thanh ghi này được gọi là thanh ghi con trỏ
- Ưu điểm:
  - Không gian địa chỉ lớn hơn (trường địa chỉ tham chiếu đến vị trí chứa địa chỉ có độ dài bằng một từ)
  - Tham chiếu bộ nhớ ít hơn 1 lần so với địa chỉ gián tiếp



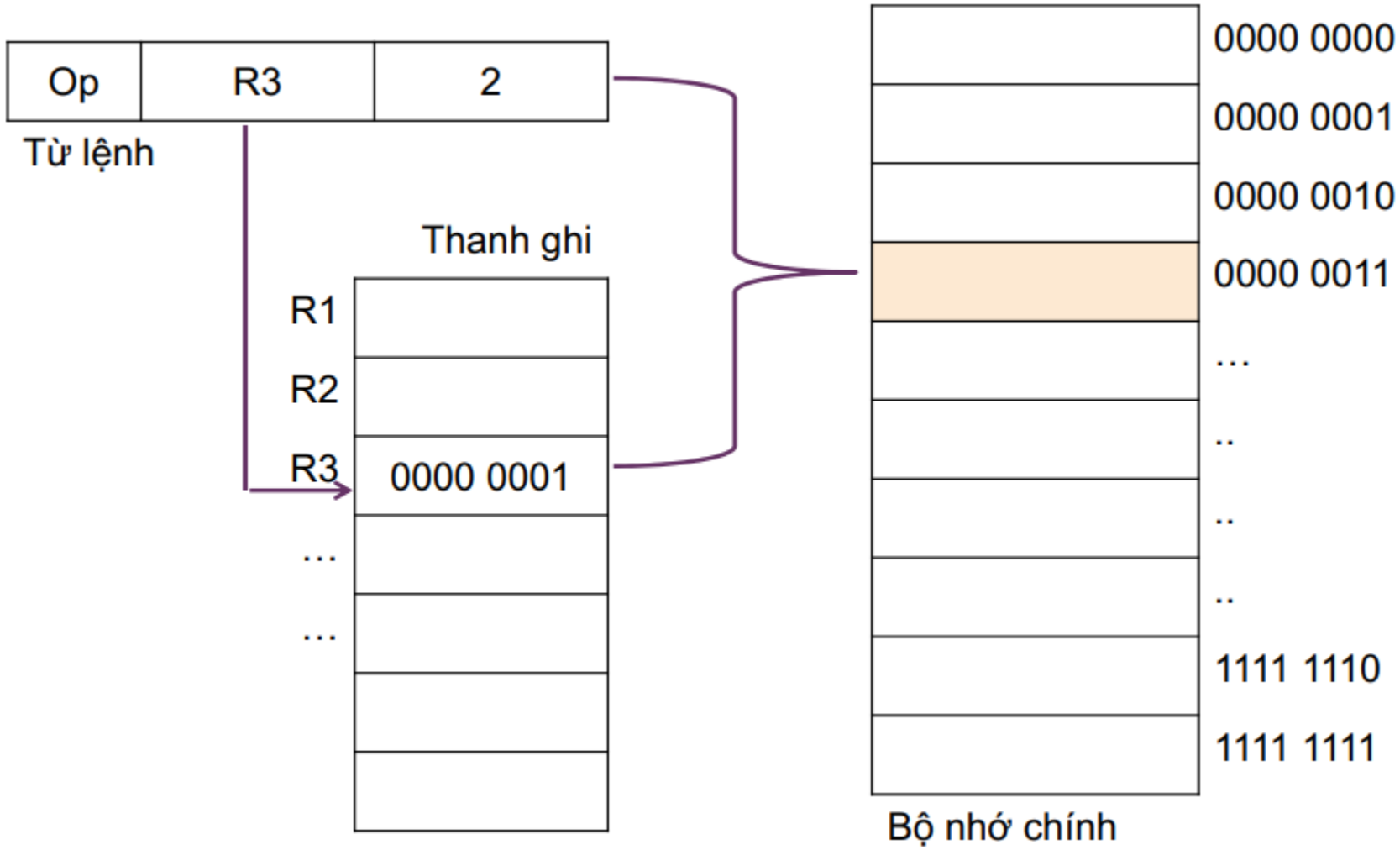
GIÁN TIẾP KẾT HỢP DỊCH CHUYỂN

- Kết hợp chế độ địa chỉ trực tiếp và địa chỉ gián tiếp thanh ghi
- Lệnh phải có 2 trường địa chỉ:
- Địa chỉ của toán hạng EA= (R) + A
- Thanh ghi có thể được ngầm định (Ví dụ: PC)
- Lợi dụng tính cục bộ của tham chiếu bộ nhớ



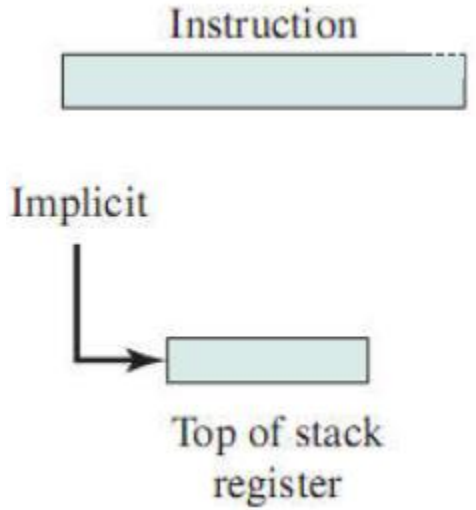
GIÁN TIẾP KẾT HỢP DỊCH CHUYỂN

❑ Ví dụ:



ĐỊA CHỈ NGĂN XẾP

- Ngăn xếp là vùng nhớ có cấu trúc LIFO
- Ngăn xếp thường dung để phục vụ cho chương trình con
- Đáy ngăn xếp là một ngăn nhớ xác định
- Đỉnh 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
- Con trỏ gắn với ngăn xếp có giá trị là địa chỉ của đỉnh ngăn xếp
  - Con trỏ ngăn xếp được duy trì trong một thanh ghi.
  - Vì vậy, tham chiếu đến các vị trí ngăn xếp trong bộ nhớ trong thực tế là địa chỉ gián tiếp thanh ghi.
- Là một dạng của địa chỉ ngầm định



5.4.7 CÁC KIỂU ĐÁNH ĐỊA CHỈ CỦA INTEL, ARM

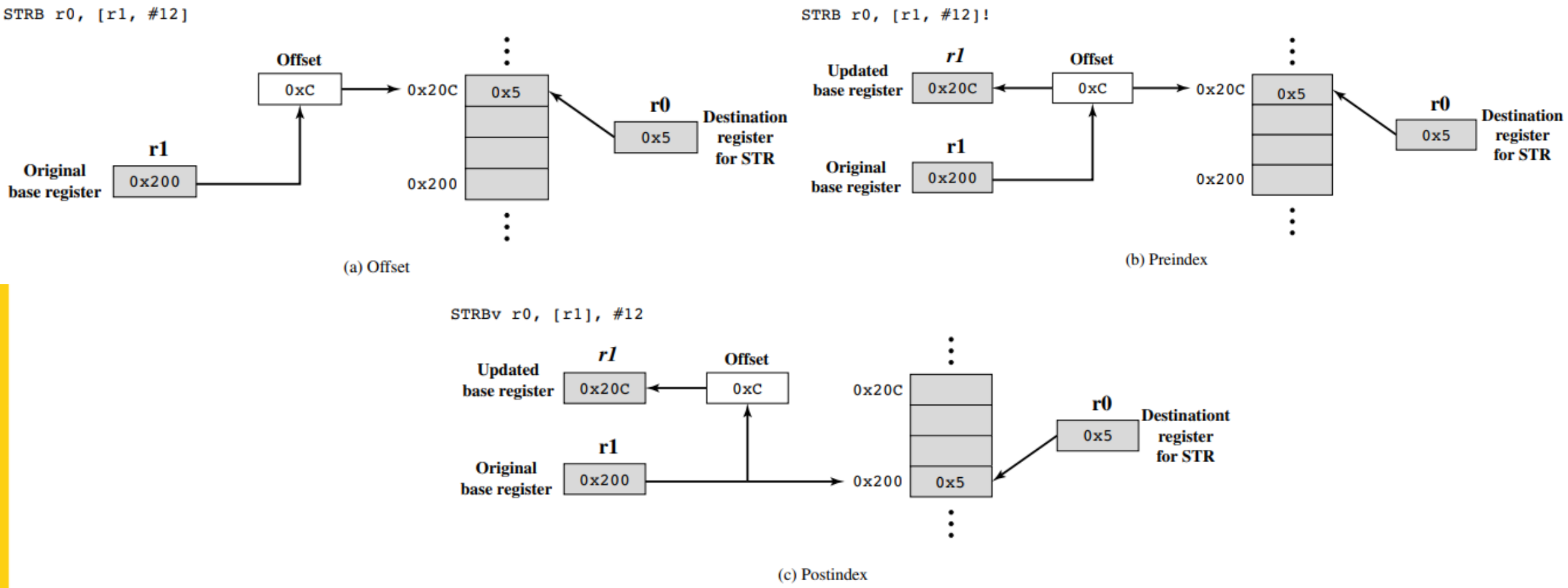
- Các chế độ địa chỉ của Intel.

Mode	**Algorithm**
Immediate	Operand = A
Register Operand	LA=R
Displacement	LA=(SR)+A
Base	LA=(SR)+(B)
Base with Displacement	LA=(SR)+ (B)+A
Scaled Index with Displacement	LA= (SR)+(I)x(S)+A
Base with Index and Displacement	LA=(SR)+(B)+(I)+A
Base with Scaled Index and Displacement	LA=(SR)+ (I)x(S) + (B) + A
Relative	LA=(PC) + A

LA = linear address
(X)=contents of X
SR= segment register
PC=program counter
A= contents of an address field in the instruction
R= register
B= base register
I= index register
S= scaling factor

5.4.7 CÁC KIỂU ĐÁNH ĐỊA CHỈ CỦA INTEL, ARM

- Chế độ địa chỉ của ARM



(a) Offset

(b) Preindex

(c) Postindex