Chương 6:

TẬP LỆNH

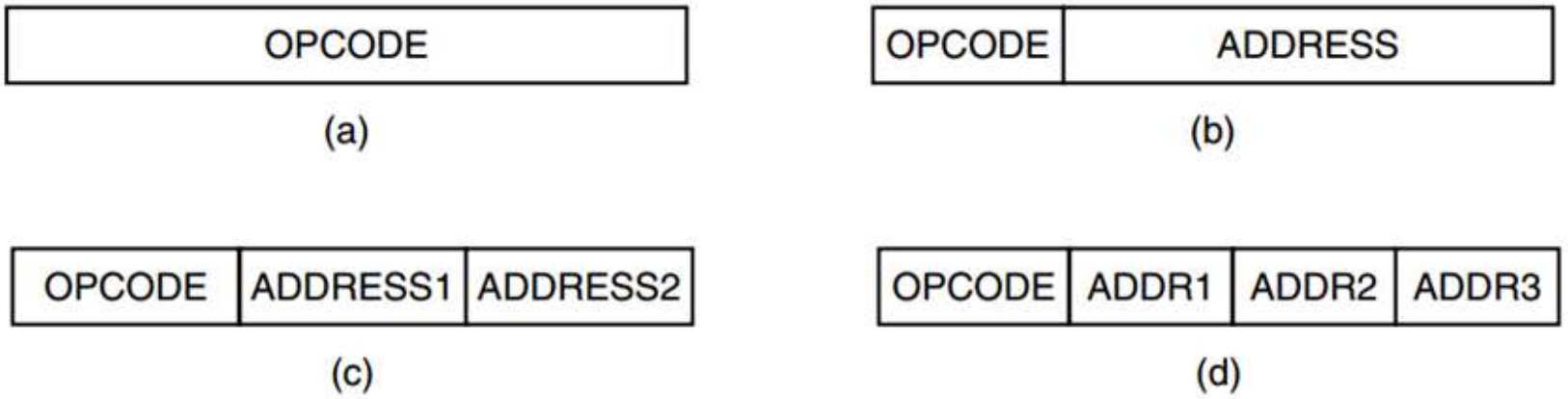
Khái Niệm

- Hoạt động của bộ xử lý được xác định bởi các lệnh mà nó thực hiện, được gọi là các lệnh máy (*machine instruction*) hoặc lệnh máy tính (*computer instructions*).
- Tập các lệnh khác nhau mà bộ xử lý có thể thực hiện được gọi là tập lệnh (*instruction set*) của bộ xử lý.

Biểu Diễn Lệnh

- Một lệnh bao gồm mã lệnh, thường là cùng với một số thông tin bổ sung như nơi các toán hạng đến từ đâu và nơi mà kết quả đi đến. Tổng quát, vấn đề xác định nơi các toán hạng (nghĩa là, địa chỉ của chúng) được gọi là định địa chỉ.
- Hình 5.1 cho thấy một số định dạng có thể cho các lệnh. Một lệnh luôn luôn có một mã lệnh để cho biết điều gì mà lệnh phải làm. Có thể không có hoặc có một, hai hay ba địa chỉ.

Biểu Diễn Lệnh

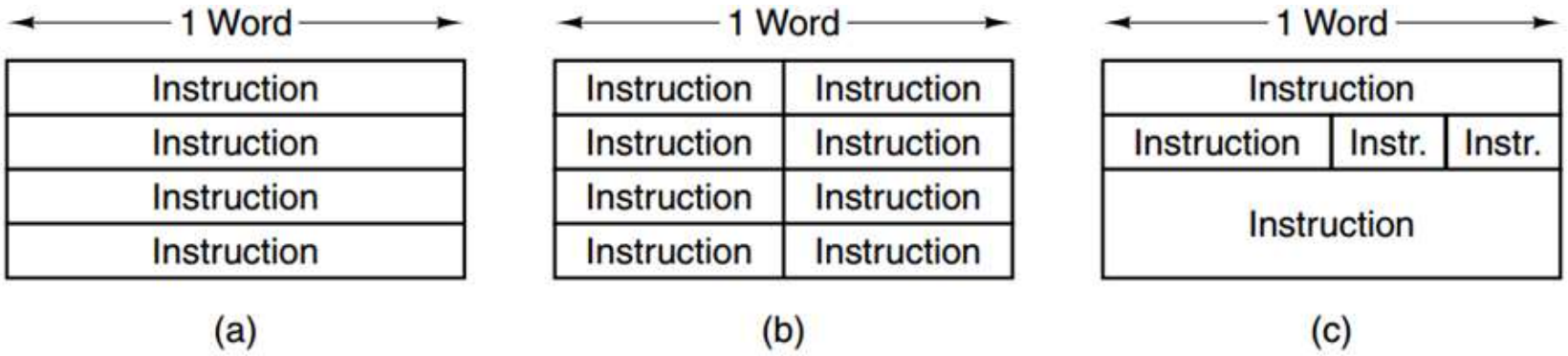


Hình 5.1: ***Bốn dạng lệnh phổ biến.(a) Lệnh không có địa chỉ.***
(b), (c), (d) Lệnh có một, hai hay ba địa chỉ.

Biểu Diễn Lệnh

- Trên một số máy, tất cả các lệnh có cùng độ dài; trên một số máy khác có thể có nhiều độ dài khác nhau.
- Lệnh có thể ngắn hơn, bằng hoặc dài hơn chiều dài của từ nhớ.
- Nếu tất cả các lệnh có cùng độ dài thì đơn giản hơn và dễ dàng giải mã hơn nhưng thường làm lãng phí không gian, vì tất cả các lệnh có cùng độ dài với lệnh dài nhất.
- Hình 5.2 cho thấy một số mối quan hệ giữa chiều dài lệnh và chiều dài từ.

Biểu Diễn Lệnh



Hình 5.2: Mối quan hệ giữa chiều dài lệnh và chiều dài từ.

Biểu Diễn Lệnh

- Trong quá trình thực hiện lệnh, lệnh được đọc vào thanh ghi lệnh (*instruction register* - IR) trong bộ vi xử lý. Bộ vi xử lý phải có khả năng trích dữ liệu từ các trường khác nhau của lệnh để thực hiện thao tác được yêu cầu.
- Chúng ta khó có thể đọc sự biểu diễn nhị phân của các lệnh máy. Vì vậy, trong thực tế người ta sử dụng biểu tượng đại diện cho mỗi lệnh máy. Các mã lệnh được thể hiện bằng các từ viết tắt, gọi là từ gợi nhớ (*mnemonics*), nó chỉ ra thao tác của lệnh đó. Ví dụ:

Biểu Diễn Lệnh

ADD	Cộng
SUB	Trừ
MUL	Nhân
DIV	Chia
LOAD	Nạp dữ liệu từ bộ nhớ
STOR	Lưu dữ liệu vào bộ nhớ

- Các toán hạng cũng được thể hiện bằng biểu tượng đại diện. Cho ví dụ, lệnh

  ADD R, Y

  có nghĩa là cộng giá trị được chứa trong vị trí dữ liệu Y với nội dung của thanh ghi R. Trong ví dụ này, Y tham chiếu đến địa chỉ của một vị trí trong bộ nhớ và R tham chiếu đến một thanh ghi cụ thể.

Biểu Diễn Lệnh

- Lưu ý rằng thao tác được thực hiện trên nội dung của một vị trí chứ không phải trên địa chỉ của nó.
- Do đó, có thể viết một chương trình ngôn ngữ máy dưới hình thức biểu tượng. Mỗi biểu tượng mã lệnh là một dãy nhị phân cố định và người lập trình chỉ định vị trí của mỗi biểu tượng toán hạng.
- Ví dụ, người lập trình có thể bắt đầu bằng một danh sách các định nghĩa:

  X = 513

  Y = 514

Các Dạng Dữ Liệu – Số Nguyên Không Dấu

- Sự biểu diễn dữ liệu trong máy tính được thể hiện bởi một chuỗi các giá trị nhị phân là 0 hay 1. Giả sử máy tính lưu trữ số nguyên trong một từ nhớ 8 bit, thì tất cả 8 bit đó được sử dụng làm giá trị nhị phân của số nguyên không dấu, có miền trị từ 0 đến 255:

```
00000000 = 0
00000001 = 1
00101001 = 41
10000000 = 128
11111111 = 255
```

Số Nguyên Không Dấu

- Một cách tổng quát, nếu một dãy n bit của các ký số nhị phân $a_{n-1}a_{n-2}...a_1a_0$ dùng để biểu diễn số nguyên A không dấu thì:
  - Giá trị của A là:

$$A = \sum_{i=0}^{n-1} 2^i a_i$$

  - Miền trị của A từ 0 đến $(2^n - 1)$.

Số Nguyên Có Dấu

- Để biểu diễn số nguyên có dấu, bit bên trái nhất của dãy bit được sử dụng làm bit dấu: 0 là số dương, 1 là số âm.
- Như vậy, nếu một số nguyên có dấu được biểu diễn trong 8 bit thì chỉ có 7 bit được sử dụng làm giá trị nhị phân vì có một bit dung làm bit dấu. Miền trị của số có dấu trong trường hợp này là từ -128 đến 127:

```
10000000 = -128
10000001 = -127
11111111 = -1
00000000 = 0
00000001 = 1
01111111 = 127
```

Số Nguyên Có Dấu

- Để biểu diễn một số nguyên có dấu, người ta thực hiện phép lấy bù hai (two's complement). Chẳng hạn, để biểu diễn số nguyên –n, đầu tiên ta biểu diễn n dạng nhị phân, lấy bù của nó – đảo ngược các bit 0 thành 1 và 1 thành 0 – rồi cộng thêm 1 vào kết quả. Ví dụ, giá trị –58 được biểu diễn như sau:
  - Số 58 dạng nhị phân 8 bit:

    `00111010`
  - Lấy bù chuỗi bit này:

    `11000101`
  - Cộng thêm 1:

    `11000110 = -58`

Số Nguyên Có Dấu

- Một cách tổng quát, nếu một dãy n bit của các ký số nhị phân $a_{n-1}a_{n-2}...a_1a_0$ dùng để biểu diễn số nguyên A có dấu thì:
  - Giá trị của A là:

$$A = -2^{n-1}a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

  - Miền trị của A từ -2^{n-1} đến $(2^{n-1} - 1)$.
- Bài tập tại lớp: Liệt kê tất cả giá trị của số nguyên không dấu và có dấu 4 bit ở dạng thập phân và nhị phân.

Số BCD

- Số BCD (*Binary Coded Decimal*) là số thập phân được mã hóa theo nhị phân)
- Mặc dù tất cả các hoạt động bên trong máy tính là nhị phân theo tự nhiên, người sử dụng hệ thống lại xử lý số thập phân.
- Vì vậy, có một điều cần thiết để chuyển đổi từ thập phân sang nhị phân trên đầu vào và từ nhị phân sang thập phân trên đầu ra. Số BCD được sử dụng trong các mạch số.
- Mỗi ký số thập phân được biểu diễn bằng một mã nhị phân 4 bit. Hai ký số được lưu trữ trong một byte. Ví dụ:

Số BCD

Số thập phân	Số BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Số BCD

- Số BCD có hai dạng:
  - Số BCD không nén (*unpacked BCD*): mỗi số BCD 4 bit được lưu trữ trong 4 bit thấp của mỗi byte. Ví dụ, số 15 được lưu trữ như sau:

  - Số BCD nén (*packed BCD*): hai số BCD được lưu trữ trong 1 byte. Số BCD mà chúng ta xét ở trên là số BCD nén.
- Số âm được thể hiện bằng cách gộp dấu vào trong 4 bit ở đầu bên trái hoặc cuối bên phải của

Số BCD

- Số BCD có hai dạng:
  - Số BCD không nén (unpacked BCD): mỗi số BCD 4 bit được lưu trữ trong 4 bit thấp của mỗi byte. Ví dụ, số 15 được lưu trữ như sau:

  - Số BCD nén (packed BCD): hai số BCD được lưu trữ trong 1 byte.
- Số âm được thể hiện bằng cách gộp dấu vào trong 4 bit ở đầu bên trái hoặc cuối bên phải của các ký số thập phân. Dấu chuẩn

Số BCD

chuẩn của các giá trị là $1100_2 = C_{16}$ cho số dương (+) và $1101_2 = D_{16}$ cho số âm (-). Ví dụ:

$$11000011_2 = +3$$

$$11010011_2 = -3$$

Số Thực

- Sự biểu diễn số thực trong bộ nhớ máy tính được chuẩn hóa vào năm 1985 bởi Viện kỹ thuật điện và điện tử (*Institute for Electrical and Electronic Engineers* – IEEE). Theo chuẩn này, sự biểu diễn nhị phân của một số thực dạng dấu chấm động như sau:

$$b_1.b_2b_3...\times 2^k$$

- Trong đó b_i là 0 hay 1 và $b_1 = 1$ (ngoại trừ số cần biểu diễn là 0).
- Dãy các số nhị phân $b_1.b_2b_3...\times 2^k$ được gọi là phần định trị (*mantissa*) hay phần phân số (*fractional*) và k là số mũ (*exponent*).

Số Thực

- Ví dụ: số 22.625 có dạng nhị phân là:

  10110.101_2

  được viết lại ở dạng dấu chấm động là:

  $1.0110101_2 \times 2^4$

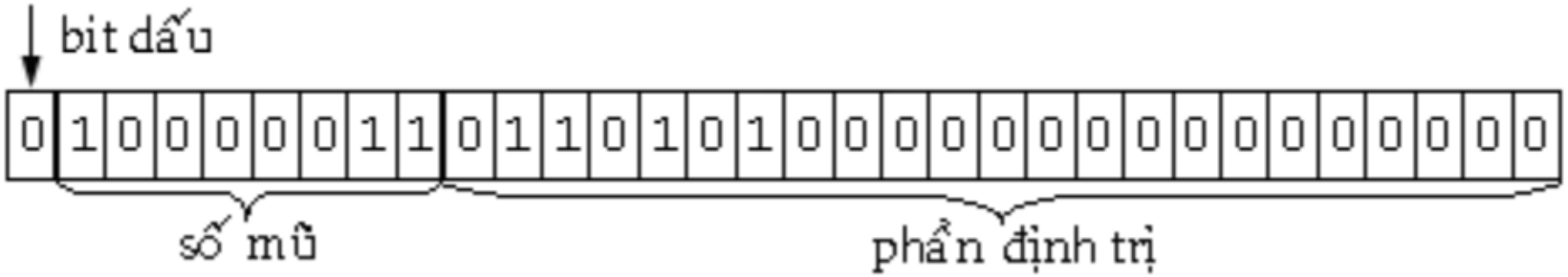
  Trong đó, số 1.0110101_2 là phần định trị và 4 là số mũ.
- Theo chuẩn IEEE, cấu trúc của một giá trị thực 32 bit (có độ chính xác đơn) là:
  - Bit bên trái nhất chứa dấu của phần định trị: 0 là số dương, 1 là số âm.
  - Tám bit kế tiếp là giá trị nhị phân của số mũ + 127. Trong đó 127 được gọi là độ lệch (*bias*). Như vậy, với tám bit này

Số Thực

có thể biểu diễn được số mũ có giá trị từ –127 đến 128.

- Với 23 bit còn lại chứa những bit bên phải dấu chấm của phần định trị (bit bên trái của phần định trị không được chứa vì nó luôn là 1).

❖ Ví dụ, với giá trị 22.625 thì số mũ được lưu trữ sẽ là $4 + 127 = 10000011_2$ và phần định trị được lưu trữ là $01101010000000000000000_2$:



Số Thực

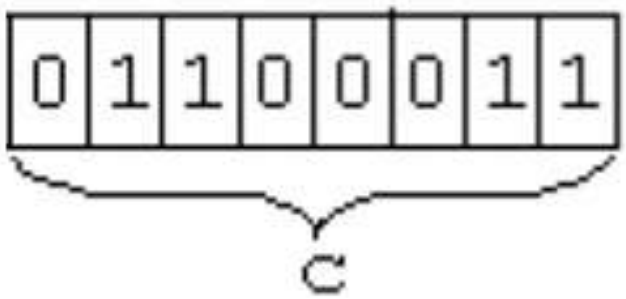
- Với một số thực 64 bit (có độ chính xác kép) thì theo chuẩn IEEE sẽ sử dụng 11 bit làm số mũ với độ lệch là 1023 và 53 bit cho phần định trị có dấu.

Ký Tự

- Các ký được biểu diễn trong máy tính bởi tập ký tự (bảng mã) ASCII (*American Standard Code for Information Interchange* – mã tiêu chuẩn Mỹ dùng trong trao đổi thông tin).
- Ký tự có thể là hoa, thường, ký số hay ký tự đặc biệt. Ví dụ: `'A'`, `'@'`, `'3'`, `'+'`.
- Các ký tự được thể hiện trong bộ nhớ bằng số và được lưu trữ dưới dạng số nguyên một byte.
- Ví dụ, các ký tự trên được biểu diễn dưới dạng số tương ứng trong tập ký tự ASCII là: 65, 64, 51 và 43.

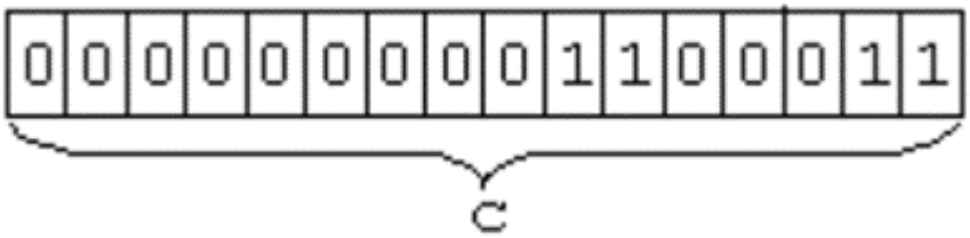
Ký Tự

- Mã ASCII là mã 7 bit, nên có $2^7 = 128$ ký tự được mã hoá thành các giá trị từ 0 đến 127, đủ để biểu thị tất cả ký tự của một bàn phím chuẩn cũng như các chức năng điều khiển.
- Tuy nhiên, nhiều máy tính dùng nhóm 8 bit (1 byte) làm đơn vị lưu trữ thông tin nên bảng mã ASCII được mở rộng thành mã 8 bit. Các ký tự mỡ rộng có mã từ 128 đến 255 dùng làm ký tự trang trí.
- Ví dụ, ký tự 'c' được biểu diễn là:



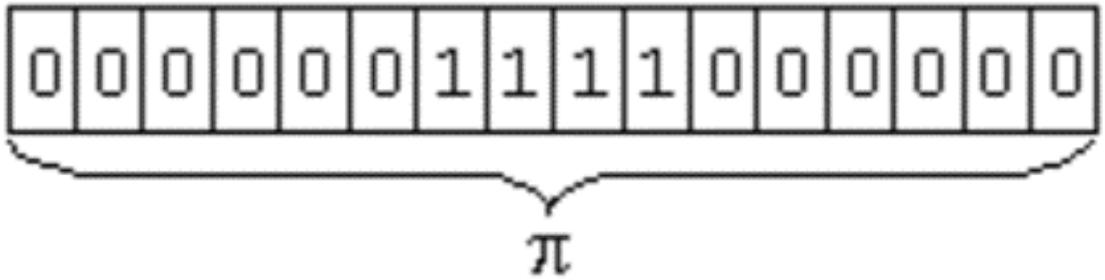
Ký Tự

- Mã Unicode được thiết kế để sử dụng cho hầu hết các ngôn ngữ chữ viết trên thế giới.
- Trong khi ASCII chỉ có thể mã hóa được 128 ký tự thì Unicode cung cấp mã cho hơn 65000 ký tự. Để thực hiện việc này nó sử dụng các mã 16 bit.
- Ví dụ, mã của ký tự c (99) cũng giống như trong bộ mã ASCII nhưng được lưu trữ trong hai byte:



Ký Tự

- Mã của những ký tự không thuộc bộ mã ASCII cũng được lưu trữ trong hai byte.
- Ví dụ, ký tự π là 960 = 0000001111000000_2:

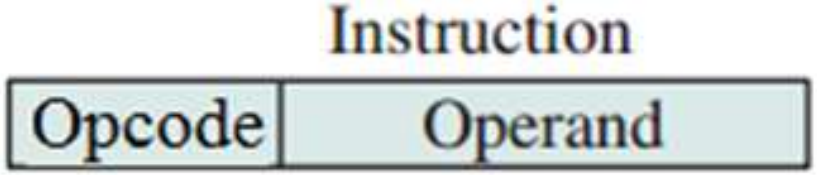


Định Địa Chỉ - Khái Niệm

- Định địa chỉ (*addressing*) là cách xác định đến các toán hạng của lệnh từ các vùng địa chỉ trên lệnh.
- Toán hạng có thể là hằng số, dữ liệu của thanh ghi hay bộ nhớ. Các kỹ thuật (hay chế độ) định địa chỉ bao gồm:
  - Tức thời (*Immediate*).
  - Trực tiếp (*Direct*).
  - Gián tiếp (*Indirect*).
  - Thanh ghi (*Register*).
  - Gián tiếp thanh ghi (*Register Indirect*)
  - Chỉ số (*Indexed/Displacement*).
  - Stack.

Định Địa Chỉ Tức Thời

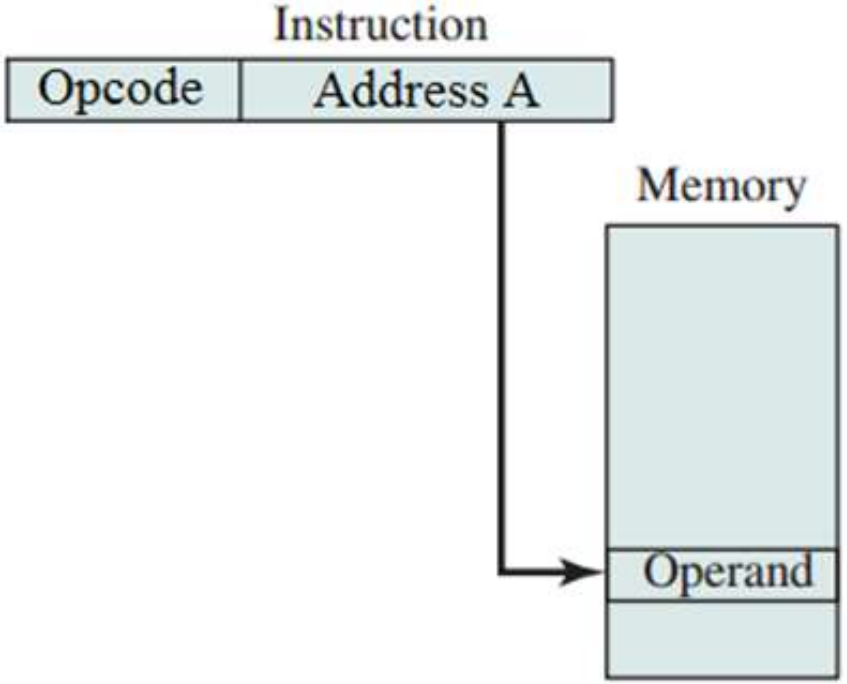
- Dạng đơn giản nhất của các chế độ định địa chỉ là định địa chỉ tức thời, nghĩa là giá trị của toán hạng được thể hiện trong lệnh:



- Ưu điểm của địa chỉ tức thời là không tham chiếu bộ nhớ, do đó tiết kiệm bộ nhớ hoặc bộ nhớ cache trong chu kỳ lệnh.
- Nhược điểm là kích thước của toán hạng bị giới hạn trong kích thước của trường địa chỉ. Trong hầu hết các tập lệnh, kích thước này nhỏ so với chiều dài từ nhớ.

Định Địa Chỉ Trực Tiếp

- Một dạng rất đơn giản trong chế độ định địa chỉ là định địa chỉ trực tiếp, nghĩa là trường địa chỉ chứa địa chỉ hiệu dụng (địa chỉ thật) của toán hạng.



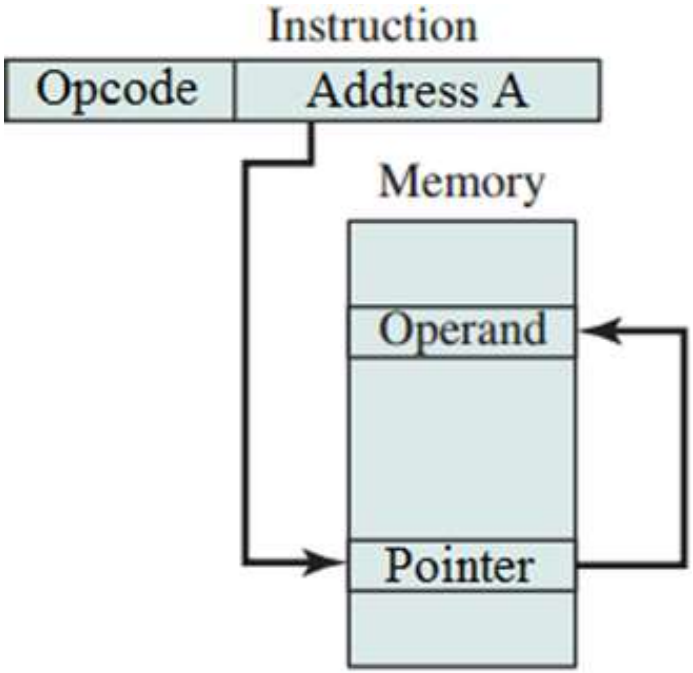
Định Địa Chỉ Trực Tiếp

- Kỹ thuật này phổ biến trong các thế hệ máy tính trước đây nhưng không phổ biến trên các kiến trúc hiện đại. Nó chỉ yêu cầu một tham chiếu bộ nhớ và không cần tính toán địa chỉ của toán hạng.
- Hạn chế rõ ràng là nó chỉ cung cấp không gian địa chỉ giới hạn.

Định Địa Chỉ Gián Tiếp

- Với chế độ định địa chỉ trực tiếp, độ dài của trường địa chỉ thường nhỏ hơn chiều dài của địa chỉ chứa toán hạng cần truy xuất, do đó bị giới hạn phạm vi địa chỉ.
- Một giải pháp là để có trường địa chỉ tham khảo đến địa chỉ của ô nhớ và ô nhớ đó có chứa một địa chỉ đầy đủ của toán hạng được gọi là định địa chỉ gián tiếp.
- Ưu điểm rõ ràng của cách tiếp cận này là với một ô nhớ có chiều dài N, sẽ có một không gian địa chỉ của 2^N địa chỉ có thể.

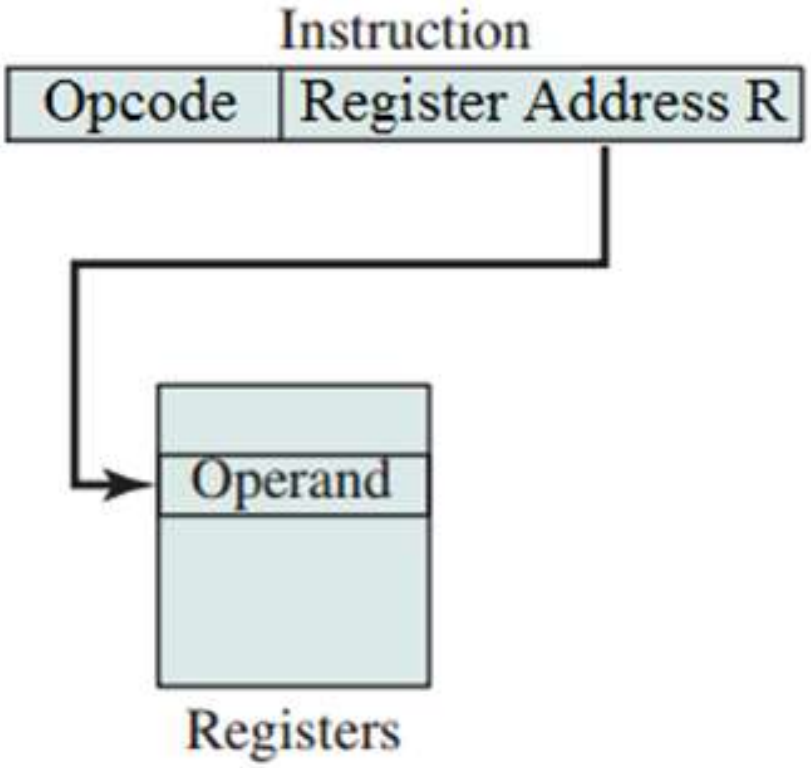
Định Địa Chỉ Gián Tiếp



- Nhược điểm của phương pháp này là việc thực hiện lệnh đòi hỏi hai lần tham chiếu bộ nhớ để lấy toán hạng: một lần để lấy địa chỉ của nó và một lần để lấy giá trị của nó.

Định Địa Chỉ Thanh Ghi

- Định địa chỉ thanh ghi tương tự như định địa chỉ trực tiếp. Sự khác biệt duy nhất là trường địa chỉ dùng để chỉ thanh ghi chứ không phải địa chỉ trong bộ nhớ chính.



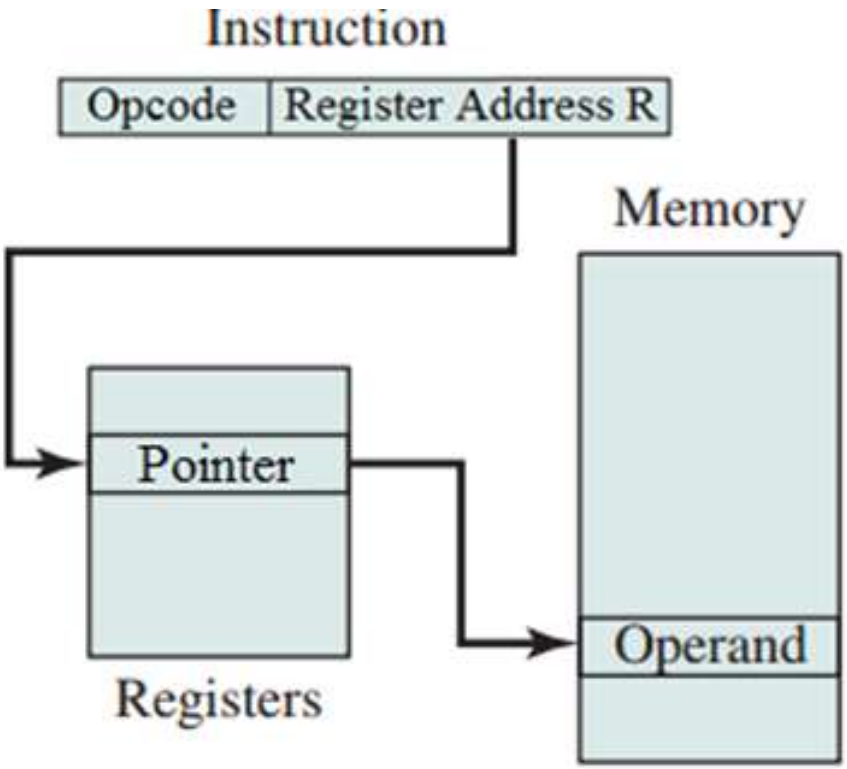
Định Địa Chỉ Thanh Ghi

- **Ví dụ, nếu nội dung của trường địa chỉ thanh ghi trong lệnh là 5, thì thanh ghi R5 là địa chỉ và giá trị toán hạng được chứa trong R5.**

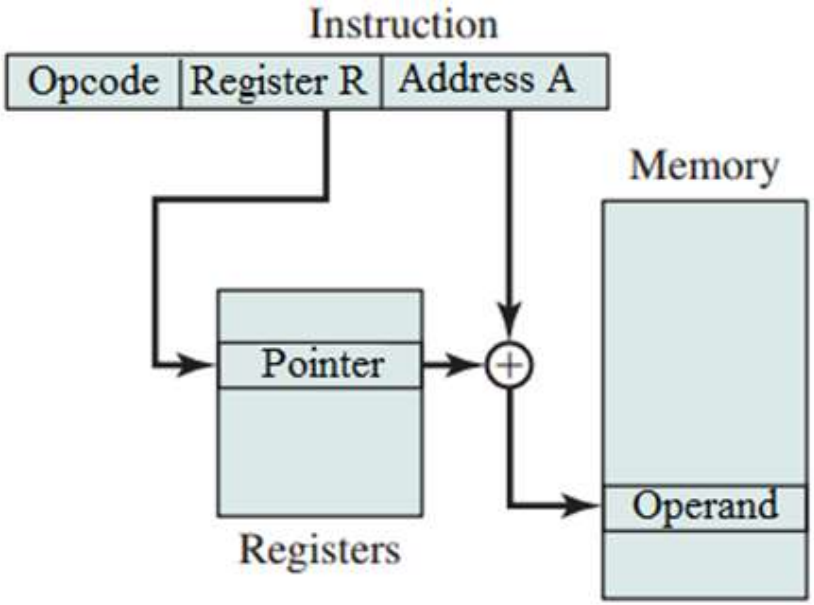
Định Địa Chỉ Gián Tiếp Thanh Ghi

- Định địa chỉ thanh ghi tương tự như định địa chỉ trực tiếp, định địa chỉ gián tiếp thanh ghi tương tự như địa chỉ gián tiếp.
- Trong cả hai trường hợp, sự khác biệt duy nhất là trường địa chỉ tham chiếu đến vị trí nhớ hay thanh ghi.
- Vùng địa chỉ của lệnh chứa tên thanh ghi, giá trị thanh ghi là địa chỉ ô nhớ lưu toán hạng.

Định Địa Chỉ Gián Tiếp Thanh Ghi



Định Địa Chỉ Chỉ Số

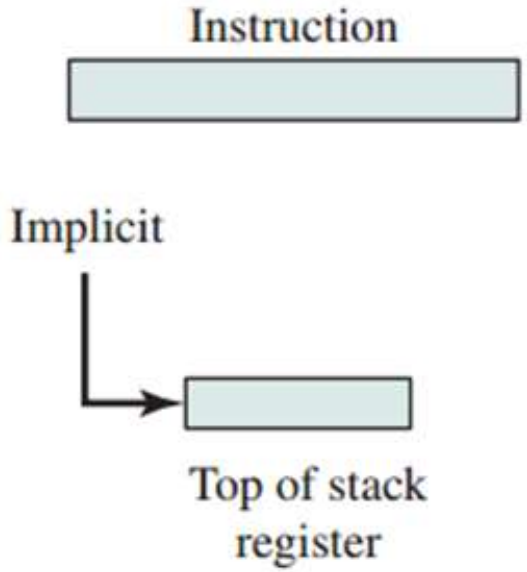


- Một chế độ định địa chỉ rất mạnh mẽ kết hợp khả năng định vị trực tiếp và định địa chỉ gián tiếp thanh ghi gọi là định địa chỉ chỉ số.

Định Địa Chỉ Chỉ Số

- Định địa chỉ chỉ số yêu cầu là các lệnh phải có hai trường địa chỉ, ít nhất một trong số đó là tường minh.
- Giá trị được chứa trong một trường địa chỉ được sử dụng trực tiếp (giá trị = Address A). Trường địa chỉ khác, hoặc tham chiếu ngầm định dựa trên mã lệnh, tham chiếu đến thanh ghi mà có nội dung được thêm vào A để sinh ra địa chỉ hiệu dụng.

Định Địa Chỉ Satck



- Chế độ địa chỉ cuối cùng mà chúng ta xét đến là định địa chỉ ngăn xếp (stack). Toán hạng được hiểu ngầm tại vị trí đỉnh stack.
- Kết hợp với ngăn xếp là một con trỏ ngăn xếp mà giá trị của nó là địa chỉ của đỉnh

Định Địa Chỉ Satck

ngăn xếp.

- Con trỏ ngăn xếp được duy trì trong thanh ghi. Vì vậy, các tham chiếu đến các vị trí ngăn xếp trong bộ nhớ thực sự là các địa chỉ gián tiếp thanh ghi.

Câu Hỏi và Bài Tập

1. Biểu diễn giá trị thập lục phân 23 ở dạng:
   a) Số BCD nén.
   b) Các ký tự ASCII.
2. Cho biết giá trị thập phân của các số BCD nén sau đây:

   ```
   0111 0011 0000 1001
   0101 1000 0010
   0100 1010 0110
   ```

3. Cho một bộ xử lý có các từ nhớ 1 byte. Hãy cho biết giá trị nguyên nhỏ nhất và lớn nhất có thể được biểu diễn ở các dạng sau:
   a) Không dấu.
   b) Có dấu.

Câu Hỏi và Bài Tập

c) Bù một.
d) Bù hai.
e) Số BCD nén không dấu.
f) Số BCD nén có dấu (giả sử 4 bit dấu nằm bên trái số đó).

4. Xét phép cộng hai số BCD nén không dấu. Nếu mỗi số BCD nén có N ký số thì nó có 4N bit. Hai số được cộng sử dụng bộ cộng nhị phân. Hãy đề xuất một quy tắc đơn giản để thực hiện phép cộng cho kết quả đúng với hai số 1698 và 1786.
5. Bù 10 của (ten's complement) của một số thập phân X được định nghĩa là $10^N - X$, trong đó N là số ký số của số X. Hãy sử

Câu Hỏi và Bài Tập

dụng bù 10 để thực hiện phép trừ thập phân $0736_{10} - 0326_{10}$.

6. Hãy biểu diễn ở dạng nhị phân 32 bit các số thực thập phân sau:

   a) `123.45`

   b) `-78.375`

7. Cho các địa chỉ sau trong bộ nhớ có chứa giá trị của toán hạng:
   - Địa chỉ 20 chứa 40.
   - Địa chỉ 30 chứa 50.
   - Địa chỉ 40 chứa 60.
   - Địa chỉ 50 chứa 70.

   Giả sử thanh ghi tích luỹ chứa toán hạng.

Câu Hỏi và Bài Tập

Hãy cho biết giá trị của toán hạng được nạp vào thanh ghi khi các lệnh sau được thực thi:

a) `LOAD IMMEDIATE 20`
b) `LOAD DIRECT 20`
c) `LOAD INDIRECT 20`
d) `LOAD IMMEDIATE 30`
e) `LOAD DIRECT 30`
f) `LOAD INDIRECT 30`

8. Trường địa chỉ trong lệnh chứa giá trị thập phân 14. Hãy cho biết nơi chứa toán hạng trong các kiểu định địa chỉ sau:
   a) Định địa chỉ tức thời.

Câu Hỏi và Bài Tập

b) Định địa chỉ trực tiếp.
c) Định địa chỉ gián tiếp.
d) Định địa chỉ thanh ghi.
e) Định địa chỉ gián tiếp thanh ghi.