**Cổng Logic và Đại số Boolean**

**1. Đại số Boolean**

**Giới thiệu khái quát:**

- Đại số Boolean hay đại số Boole được phát minh bởi nhà toán học Anh George Boole(1815-1864).

- Đại số Boolean là một cấu trúc đại số thao tác với các biến luận lý nhị phân (biến luận lý).

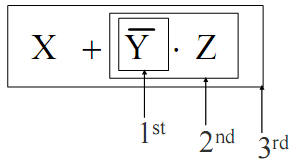
- Biến luận lý chỉ mang giá trị 1 (TRUE) hoặc 0 (FALSE).

**Các phép toán logic cơ bản trong đại số Boolean:**

* Phép cộng: **(+)** hay **OR.**
* Phép nhân: **(.)** hay **AND.**
* Phép bù: **NOT.**

**Độ ưu tiên của các phép toán**

* + Biểu thức được tính từ **trái** sang **phải**.
  + Biểu thức trong **ngoặc đơn** được đánh giá trước.
  + Thứ tự ưu tiên của các phép toán : **NOT > AND > OR**

 **Ví dụ**: Y = A + B.

**Bảng chân trị**

|  |  |  |
| --- | --- | --- |
| A | B | A AND B |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| A | B | A OR B |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

|  |  |
| --- | --- |
| A | (NOT A) |
| 0 | 1 |
| 1 | 0 |

**Các tiên đề của đại số boolean**

**Tiên đề 1**: A = 0 khi và chỉ khi A không bằng 1

A = 1 khi và chỉ khi A không bằng 0

**Tiên đề 2**: Phần tử đồng nhất

x + 0 = x

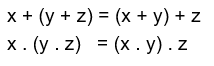
x . 1 = x

**Tiên đề 3**: Tính giao hoán

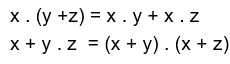
x + y = y + x

x . y = y . x

**Tiên đề 4**: Tính kết hợp



**Tiên đề 5**: Tính phân phối



**Tiên đề 6**: Tính bù



**Tính đối ngẫu**

Đại số Boolean mang tính đối ngẫu .

Đổi **AND** thành **OR** và **OR** thành **AND.**

Đổi **0** thành **1** và **1** thành **0.**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Cột 1** | **Cột 2** | **Cột 3** |
| **Dòng 1** | **1 + 1 = 1** | **1 + 0 = 0 + 1 = 1** | **0 + 0 = 0** |
| **Dòng 2** | **0 . 0 = 0** | **0 . 1 = 1 . 0 = 0** | **1 . 1 = 1** |

**Ví dụ :**

x . (y + z) = x . y + x . z **đối ngẫu** x + yz = (x + y)(x + z)

**Các định lý của đại số boolean**

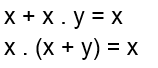
**Định lí 1**: Luật lũy đẳng



**Định lí 2** : Luật nuốt



**Định lí 3** : Luật hấp thu



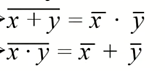
**Định lí 4 :** Luật bù kép



**Định lí 5 :**



**Định lí 6** : De Morgan



**Hàm Boolean**

- Kết hợp các biến, hằng số, toán tử, dấu ngoặc tạo thành 1 **biểu thức boolean.**

**Ví dụ**: x + yz

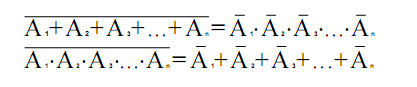
- Kết hợp theo thứ tự 1 tên hàm, 1 dấu bằng và cuối cùng là 1 biểu thức boolean sẽ cho chúng ta được một **hàm boolean** ( hàm boolean dạng chuẩn). Giá trị của hàm Boolean có thể là 0 hoặc 1.

Ví dụ: f(x,y,z) = x + yz

**Phần bù của hàm Boolean**

**Có 2 cách để tìm phần bù của hàm Boolean:**

+ Sử dụng định lí De Morgan.

****

**Ví dụ:**

F = x . y +

= = .

= ( . ( x + y + )

+ Lấy đối ngẫu và lấy bù các biến.

Đổi AND thành OR và OR thành AND

Đổi 0 thành 1 và 1 thành 0

**Ví dụ**:

**F = x . y +**

Lấy đối ngẫu : (x + y) . (

Bù các biến **= ( . ( x + y + )**

**Biểu diễn hàm boole**

**Bảng trạng thái**

- Bảng trạng thái gồm các cột, liệt kê giá trị (trạng thái) mỗi biến theo từng cột và giá trị hàm theo một cột riêng (thường là bên phải bảng).

- Bảng trạng thái còn được gọi là bảng sự thật hay bảng chân trị.

- Số hàng của bảng là , n là số các biến nhị phân được sử dụng trong hàm

**Ví dụ** bảng trạng thái của hàm f(x,y,z) = x + yz

|  |  |  |  |
| --- | --- | --- | --- |
| **x** | **y** | **z** | **f** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

- Rõ ràng, trực quan. Sau khi xác định các giá trị biến vào có thể tìm được giá trị  
đầu ra nhờ bảng trạng thái. Để giải quyết bài toán ở dạng logic thì sử dụng bảng trạng thái là hữu ích nhất. Do vậy, trong quá trình thiết kế mạch số việc đầu tiên nên làm là lập bảng trạng thái.  
- **Nhược điểm** chủ yếu của bảng trạng thái là sẽ phức tạp nếu số biến quá nhiều, không thể dùng các công thức và định lý để tính toán.

**Bài tập : Lập bảng chân trị từ biểu thức**

F = C +

|  |  |  |
| --- | --- | --- |
| B | C | F |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

F = (A + B)(AC + A) + A(B + BC)

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | F |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

**Dạng chính tắc của hàm Boolean**

Hạng tích (**minterm**): ký hiệu mi , với i = 0 đến 2n -1, là các tổ hợp gồm **tích** các biến, trong đó

* Giá trị ‘1’ được biểu diễn bằng nguyên biến (biến trực tiếp)
* Giá trị ‘0’ được biểu diễn bằng đảo biến (biến phủ định)

Hạng tổng (**Maxterm**): ký hiệu Mi , với i = 0 đến 2n -1, là các tổ hợp gồm **tổng** các biến, trong đó:

* Giá trị ‘0’ được biểu diễn bằng nguyên biến (biến trực tiếp)
* Giá trị ‘1’ được biểu diễn bằng đảo biến (biến phủ định)

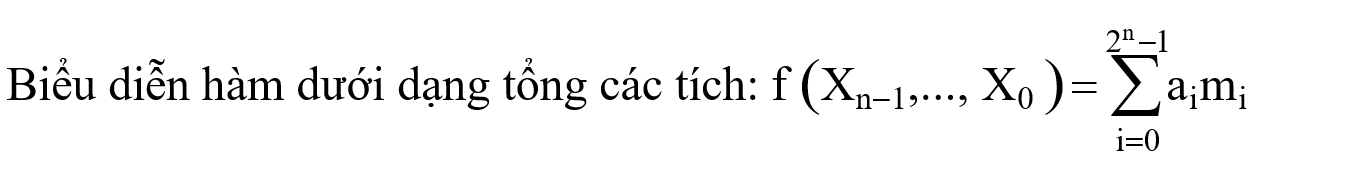
Mối quan hệ giữa minterm và Maxterm:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Biến** | | | **minterm** | | **Maxterm** | |
| **x** | **y** | **z** | **Biểu thức** | **Kí hiệu** | **Biểu thức** | **Kí hiệu** |
| 0 | 0 | 0 |  | m0 | x + y + z |  |
| 0 | 0 | 1 | z |  | x + y+ |  |
| 0 | 1 | 0 |  |  | x + + z |  |
| 0 | 1 | 1 |  |  | x + |  |
| 1 | 0 | 0 |  |  |  |  |
| 1 | 0 | 1 |  |  |  |  |
| 1 | 1 | 0 |  |  |  |  |
| 1 | 1 | 1 | x y z |  |  |  |

**Một biểu thức n biến luôn có thể biểu diện dưới 2 dạng**

+ Dạng tuyển (tổng các tích – **SOP(sum of product)):** Mỗi số hạng là một *hạng tích* hay *minterm* (mi). Mà tại đó tổ hợp hàm Boolean có giá trị là 1.

+ Dạng hội (tích các tổng – **POS(product of sum)):** Mỗi thừa số là một *hạng tổng* hay *Maxterm* (Mi). Mà tại đó tổ hợp hàm Boolean có giá trị là 0.

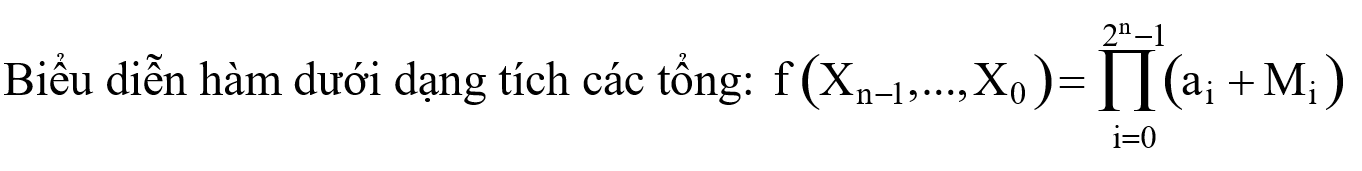


**Các bước để biểu diễn hàm Bool theo dạng tổng của các tích:**

1. Xây dựng một bảng chân trị cho hàm Boolean.

2. Hình thành một minterm cho mỗi sự kết hợp của các biến tạo ra hàm có giá trị là 1.

3. Biểu thức cuối cùng là cộng tất cả các minterm thu được từ bước 2.



**Các bước để biểu diễn hàm Bool theo dạng tích của các tổng**

1. Xây dựng một bảng chân trị cho hàm Boolean.

2. Hình thành một maxterm cho mỗi sự kết hợp của các biến với các biến này thì hàm này có giá trị là 0.

3. Biểu thức cuối cùng là nhân tất cả các maxterm thu được từ bước 2.

**Ví dụ**: Cho bảng chân trị hàm f

|  |  |  |  |
| --- | --- | --- | --- |
| x | y | z | f |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

**Biểu diễn dưới dạng tổng các tích - SOP**

Các **minterm** tương ứng là:

x x.y. x.y.z

Sau đó lấy tổng(OR) của tất cả các **minterm** ta được biểu thức hàm f

f(x,y,z) = + + x. + x.y. + x.y.z

= 001 + 011 + 100 + 110 + 111

=

=

**Biểu diễn dưới dạng tích các tổng - POS**

f(x,y,z) = (x + y + z)(x + (

Các **Maxterm** tương ứng là

(x + y + z) (x + (

Sau đó, lấy tích (AND) của tất cả các maxterm này, ta được biểu thức hàm f

f(x,y,z) = (x + y + z)(x + (

= 000 . 010 . 101

=

=

**Bài tập**: Tính biểu thức hàm f (x,y,z) = x.y + z dưới dạng tổng các tích

*Chuẩn hoá hàm về dạng chuẩn tổng các tích (chuẩn tắc tuyển):*

* Thêm các biến còn thiếu vào các hạng tích mà không làm ảnh hưởng đến kết quả bằng cách nhân hạng tích đó với ‘1’ (tổng của nguyên biến và đảo biến còn thiếu).
* Loại bỏ các hạng tích lặp lại (hạng tích thừa).

f(x,y,z) = x.y + z

= x.y( + z) + ( + x).(y + ).z

= x.y. + x.y.z + z + y.z + x.z + x.y.z

**Ghi chú** : Loại bỏ 1 xyz đi vì theo luật lũy đẳng x + x = x

= x.y + x.y.z + z + .y.z + x.z

= 110 + 111 + 001 + 011 + 101

= +   
 =

**Ví dụ**: Tính biểu thức f(x,y,z) = (x + y)(x + ) dưới dạng tích các tổng

*\* Chuẩn hoá hàm về dạng chuẩn tích các tổng (chuẩn tắc hội):*

* Thêm các biến còn thiếu vào các hạng tổng mà không làm ảnh hưởng đến kết quả bằng cách cộng hạng tổng đó với ‘0’ (tích của nguyên biến và đảo biến còn thiếu).
* Loại bỏ các hạng tổng lặp lại (hạng tổng thừa).

f(x,y,z) = (x + y)(x + )

= (x + y + z)(x + y.+ )

= (x + y + z)(x + y + )(x + y + )(x + + )

**Ghi chú:** Loại bỏ 1 x + y + z’ vì theo luật lũy đẳng x.x=x

= (x + y + z)(x + y + )(x + + )

= 000 . 001 . 011

=

**Tối thiểu hóa hàm Boole**

- Là đưa hàm Boolean về dạng biểu diễn đơn giản nhất sao cho biếu thức có chứa ít nhất các thừa số và mỗi thừa số chứa ít nhất các biến, mạch logic thực hiện có chứa ít nhất các vi mạch số.

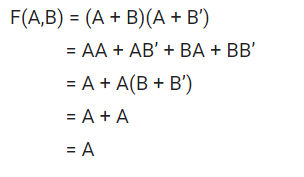
- Phương pháp sử dụng phương pháp đại số. Áp dụng các định lí tiên đề các luật để tối thiểu hàm boolean đến mức thấp nhất.

**Ví dụ**:

f(x,y,z) = x + y + xy

= x(1 + y) + y

= x + y



***Chú*** *ý: nếu trong tổng các tích, xuất hiện một biến và đảo của biến đó trong hai số hạng khác nhau, các thừa số còn lại trong hai số hạng đó tạo thành thừa số của một số hạng thứ ba thì số hạng thứ ba đó là thừa và có thể bỏ đi.*

F = C +

= C + (áp dụng đli de morgan)

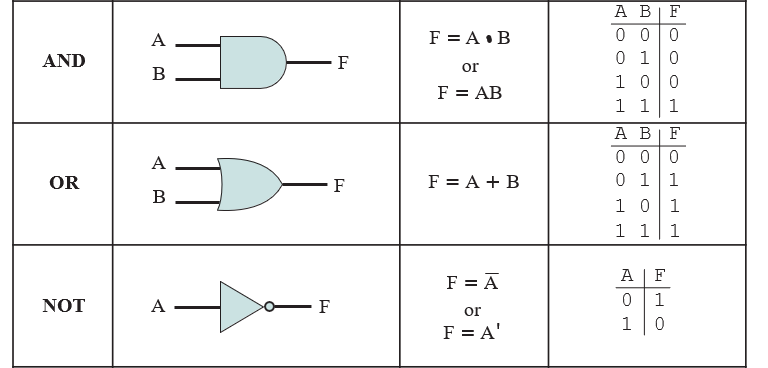
= = 1

**2. Cổng logic**

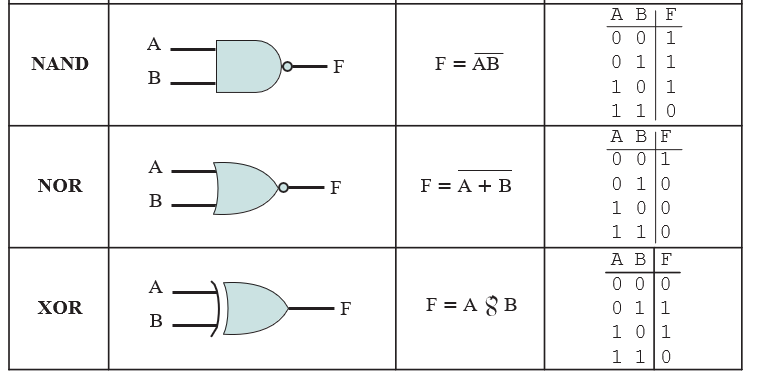
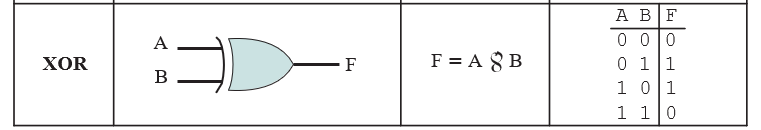
- Là những thiết bị điên tử rất nhỏ bé, có ít nhất 1 ngõ vào nhưng chỉ có duy nhất 1 ngõ ra, các giá trị vào hoặc ra chỉ có thể nhận một trong hai giá trị là 1 hoặc 0.

- Các **transistor** được ghép nối lại để tạo thành các cổng logic có thể thực hiện các phép toán cơ bản của đại số Boolean. Các cổng cơ bản này lại được lắp ghép thành các phần tử chức năng lớn hơn như mạch cộng 1 bit, nhớ 1 bit,… từ đó tạo thành 1 máy tính hoàn chỉnh.

**Các cổng logic cơ bản**

****

**Một số cổng ghép thông dụng như**

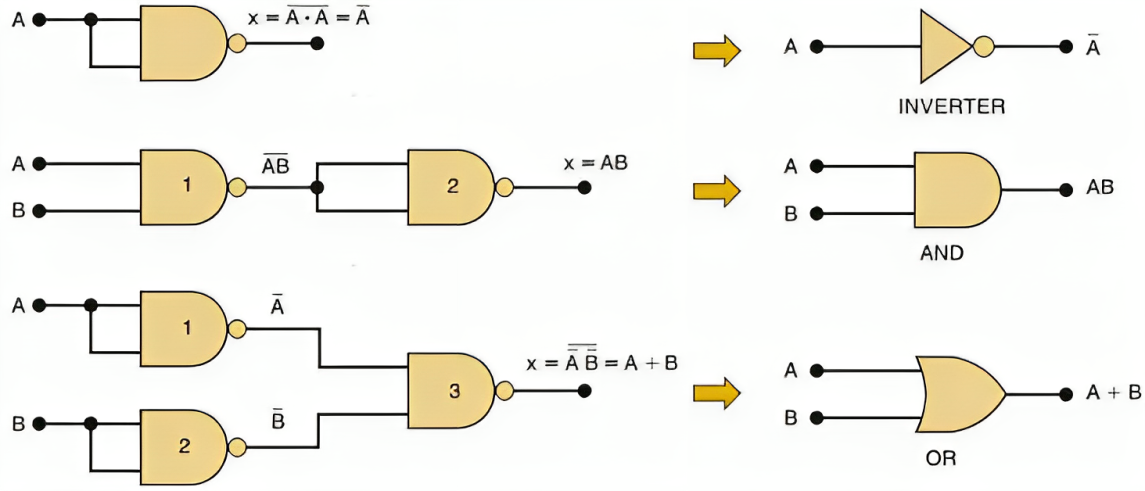
****

****

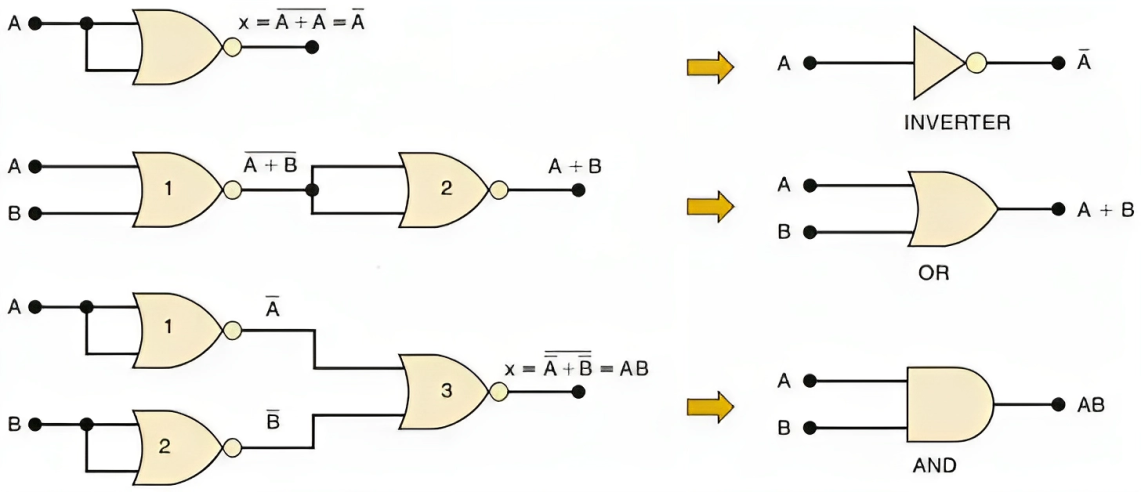


**Sự đa năng của cổng NAND và cổng NOR**

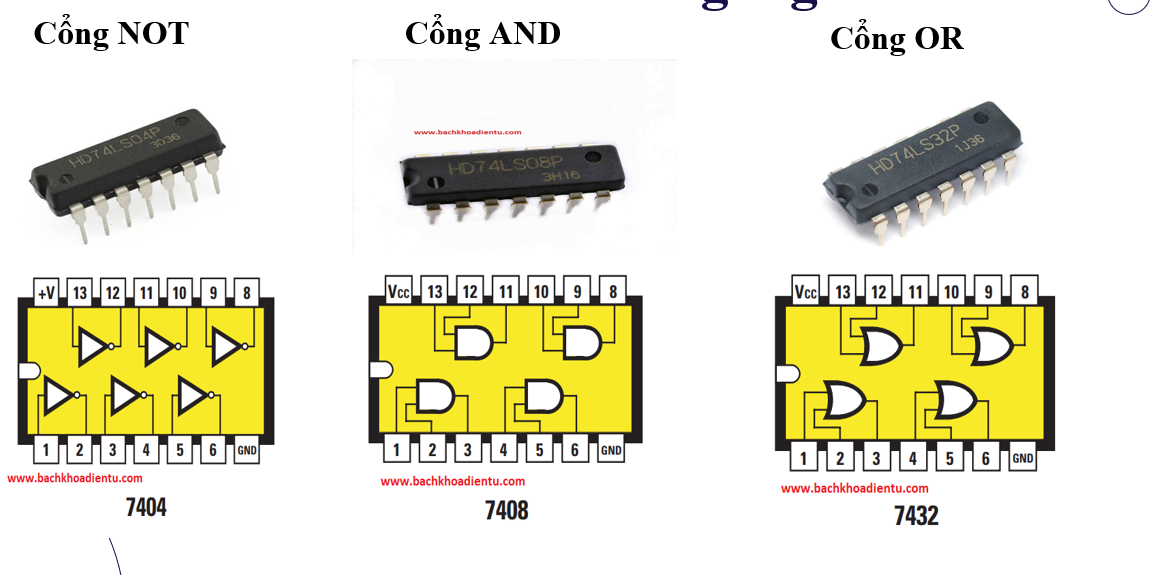
Từ cổng NAND có thể tạo ra các cổng NOT,AND,OR

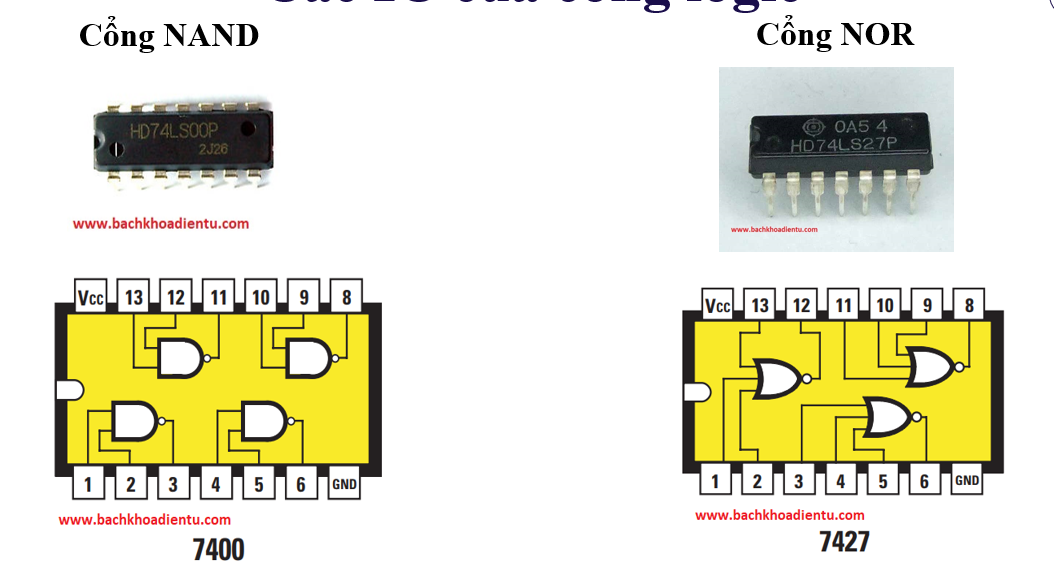


Từ cổng **NOR** có thể tạo ra các cổng NOT,AND,OR



**Các IC cổng logic**

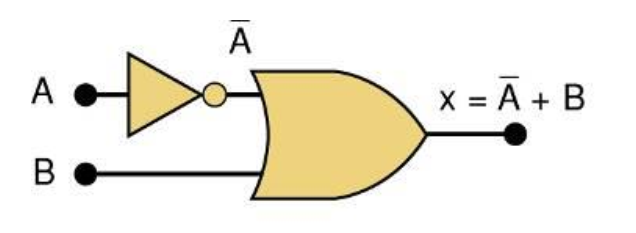




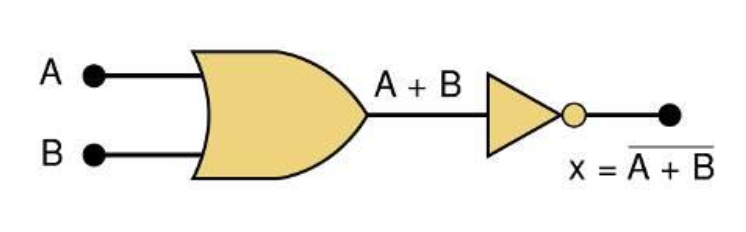
**Bài tập:**

Xây dựng 1 mạch logic từ biểu thức

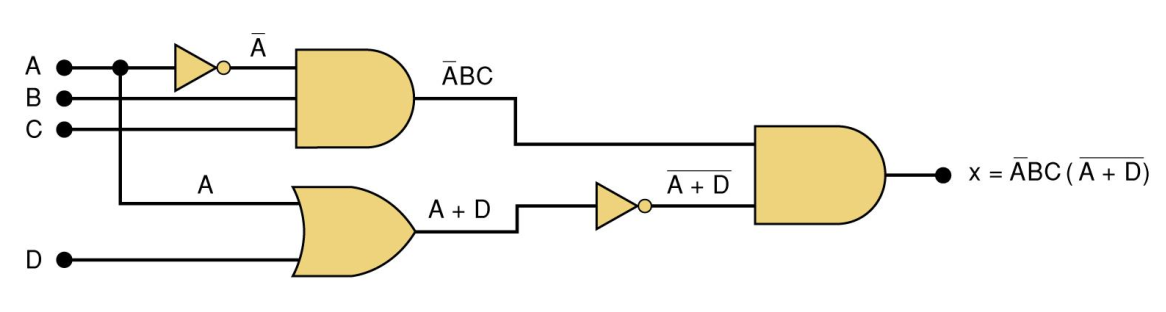
F = + B

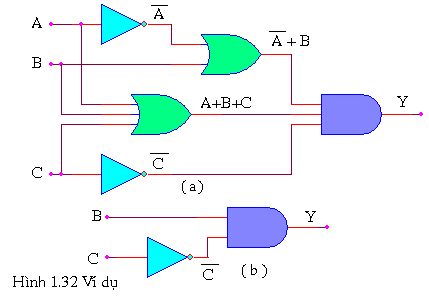


F =

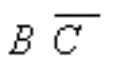


**Tìm biểu thức logic của mạch logic**



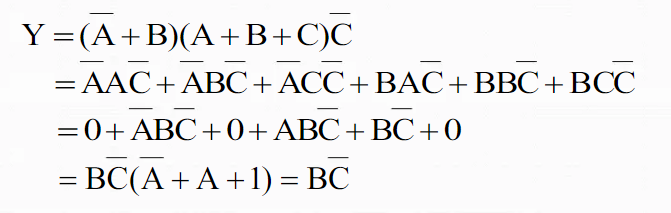
**Đơn giản mạch**







**+ Rút gọn biểu thức**



Sau khi rút gọn được biểu thức ta sẽ thu được mạch logic mới ở hình b