

# 关于组织申报2019年大学生创新创业训练计划项目工作的通知

<http://cx.hfut.edu.cn/2019/0304/c2633a206272/page.htm>

[http://cx.hfut.edu.cn/2019/0228/c2633a206108/page.htm?tdsourcetag=s\\_pcqq\\_aiomsg](http://cx.hfut.edu.cn/2019/0228/c2633a206108/page.htm?tdsourcetag=s_pcqq_aiomsg)

---

## 1、国家级大学生创新创业训练项目

2019年立国家级大学生创新创业训练项目100项（其中宣城校区15项），项目分为创新训练项目、创业训练项目和创业实践项目三类，创新、创业训练项目每项资助经费8000元至20000元，创业实践项目每项资助经费5万元至10万元，研究周期为两年（2019年5月至2021年4月）。

## 2、省级大学生创新创业训练项目

2019年立省级大学生创新创业训练项目360项（其中宣城校区60项），项目分为创新训练项目、创业训练项目两类，每项资助经费3000元至8000元，研究周期为一年（2019年5月至2020年4月）。

## 3、校级大学生创新创业训练项目

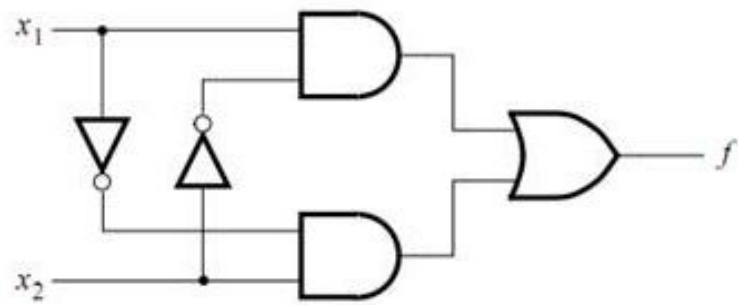
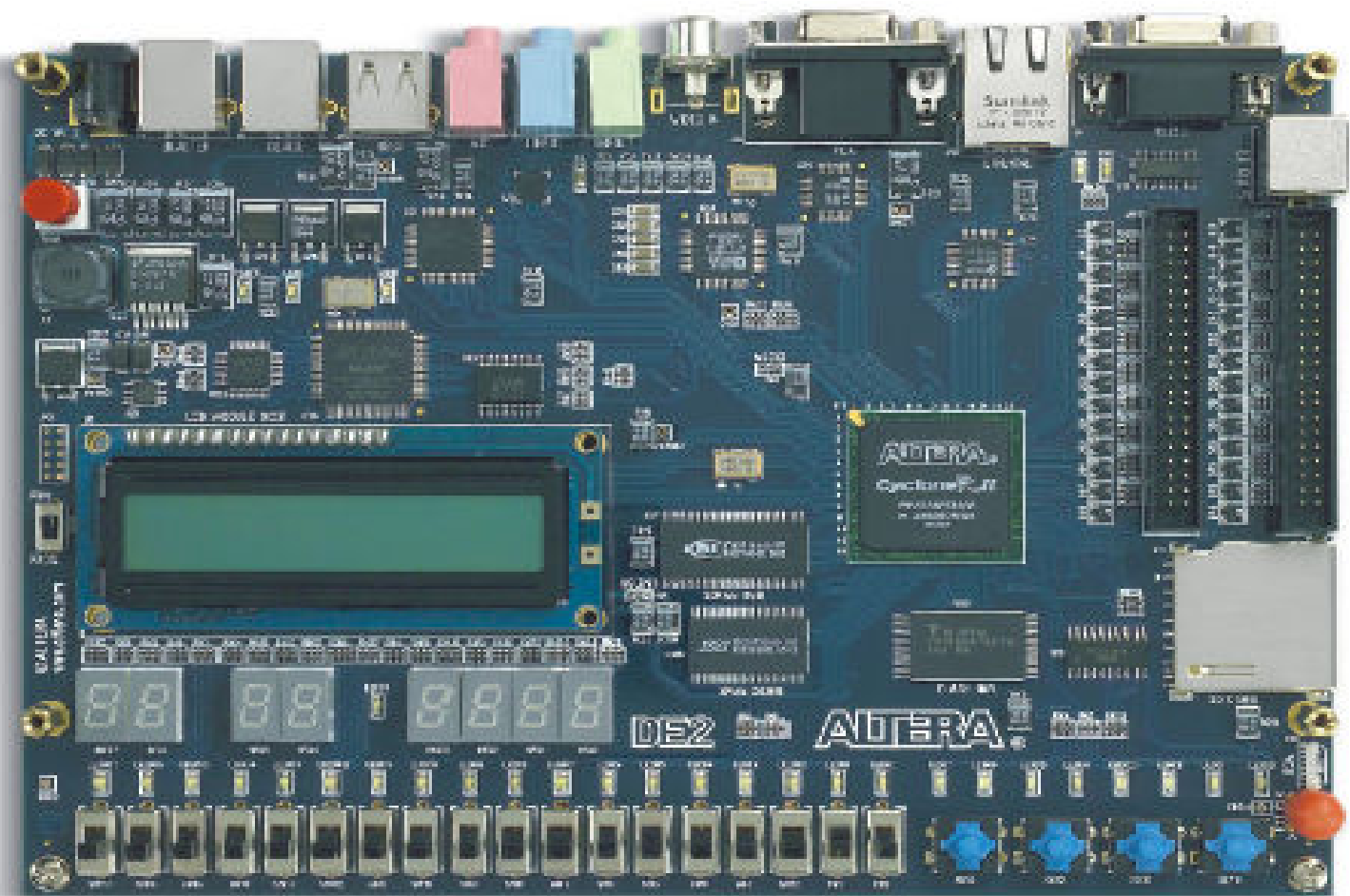
2019年立校级大学生创新创业训练项目700项（其中宣城校区150项），项目分为创新训练项目和创业训练项目两类，每项资助经费1000元至2000元，研究周期为一年（2019年5月至2020年4月）。

# 一. FPGA介绍

---

- **FPGA** (Field Programmable Gate Array)即**现场可编程门阵列**，它是在PAL、GAL、EPLD等可编程器件的基础上进一步发展的产物。它是作为专用集成电路(ASIC)领域中的一种半定制电路而出现的，既解决了定制电路的不足，又克服了原有可编程器件门电路数有限的缺点。FPGA的使用非常灵活，**同一片FPGA通过不同的编程数据可以产生不同的电路功能**。FPGA在通信、数据处理、网络、仪器、工业控制、军事和航空航天等众多领域得到了广泛应用。随着功耗和成本的进一步降低，FPGA还将进入更多的应用领域。目前市场上最大的**两个FPGA厂商分别为 Xilinx 和 Altera**。

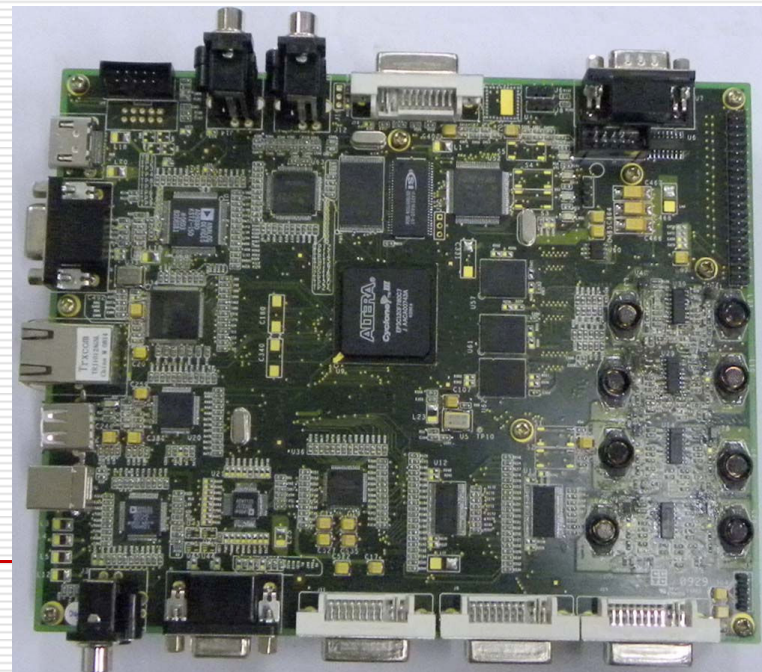
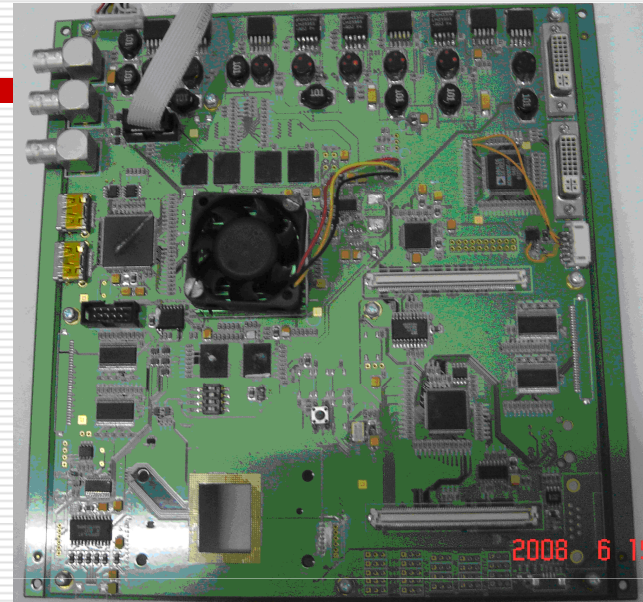
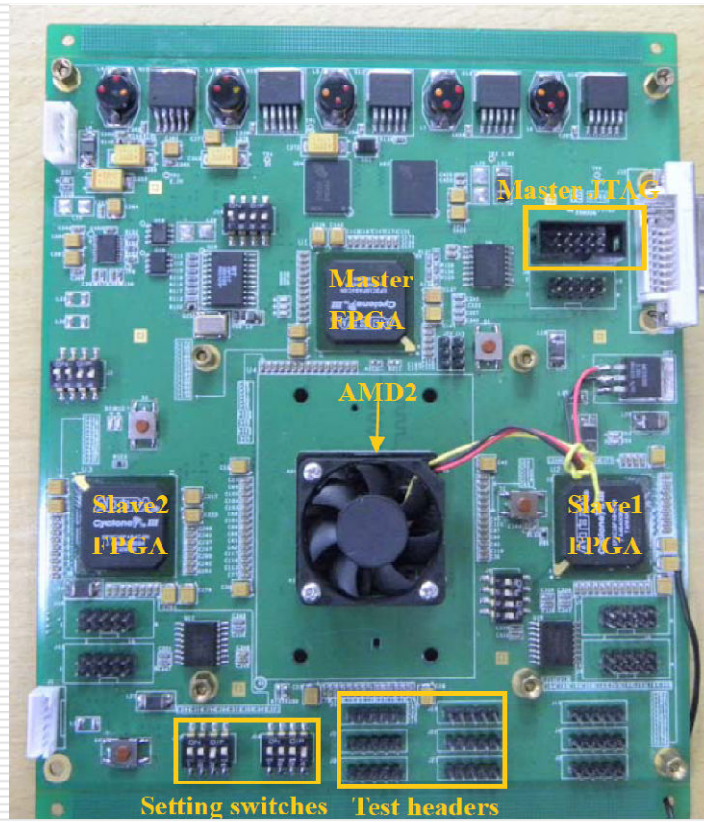




$x_1$	$x_2$	$f$
0	0	0
0	1	1
1	0	1
1	1	0



# FPGA开发板

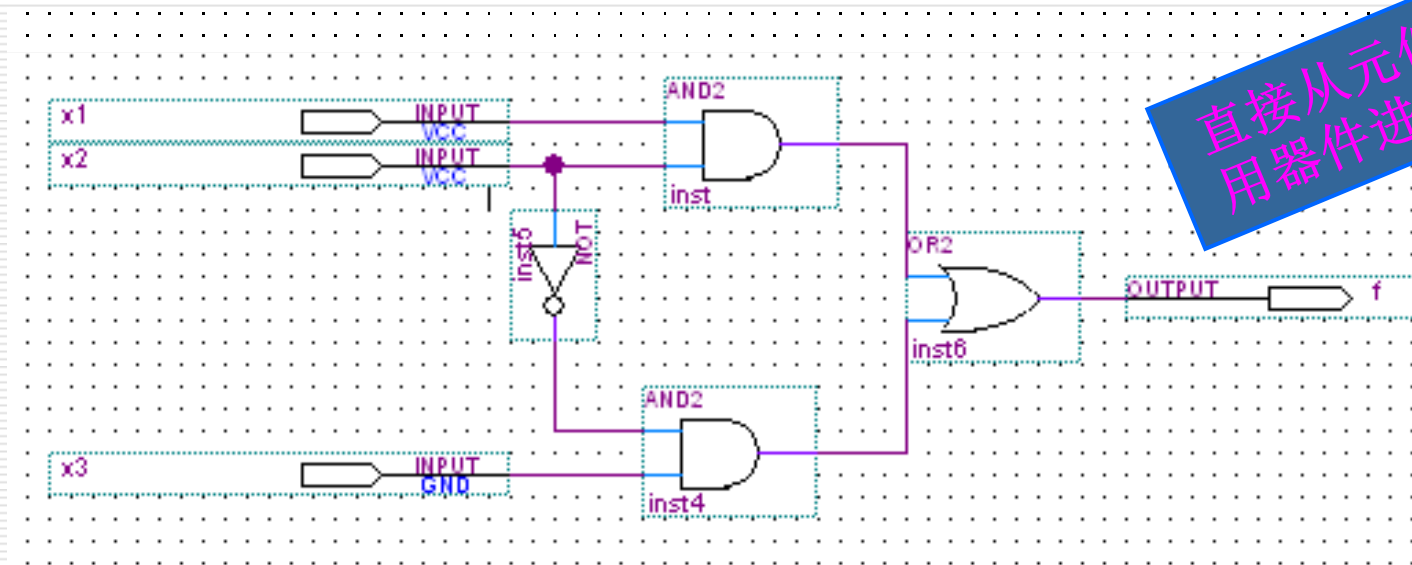
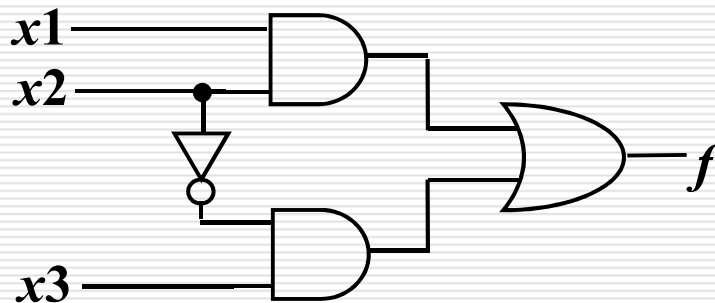


## ● 1、原理图输入

## 可编程逻辑设计

元件库

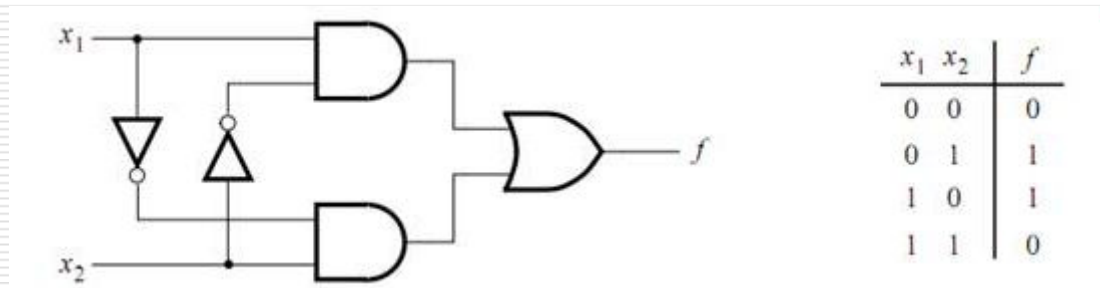
CAD工具提供一系列表示不同输入端数的各种类型门的图形符号。



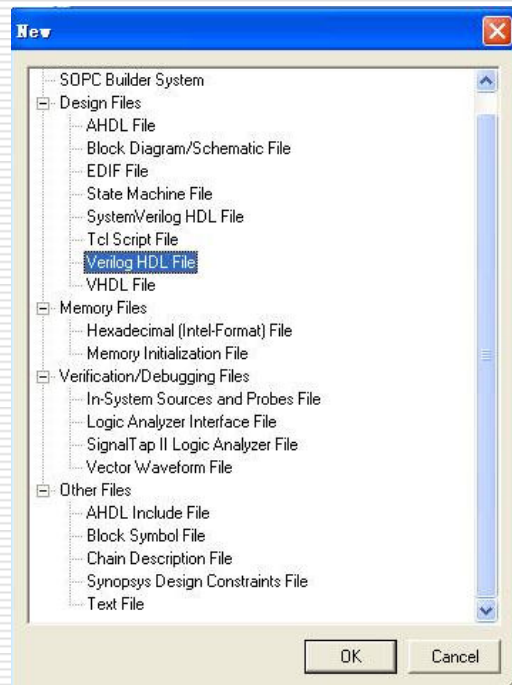
直接从元件库中调用器件进行设计。

## 2、用Verilog代码设计(与原理图类似)

- 实现一个2路输入控制灯开关的电路，如下图， $x_1$ ， $x_2$ 为2个开关， $f$ 为电路输出。



- 使用Quartus II的编程环境，在它的的文本编辑器输入设计。  
点击菜单栏File > New出现下图，选择Verilog HDL File，ok确认。然后在编辑器里编辑代码如下。最后后点击菜单栏File > Save as, 保存文件名为light.v



```
1 module light(x1,x2,f);  
2     input x1,x2;  
3     output f;  
4     assign f=(x1 & ~x2) | (~x1 & x2);  
5 endmodule
```

# 两种常用的硬件描述语言VHDL和Verilog HDL

---

## VHDL

VHDL主要用于描述数字系统的结构、行为、功能和接口。

## Verilog HDL

Verilog HDL是在C语言的基础上发展而来的硬件描述语言，具有简洁、高效、易用的特点。

---

## 二. Verilog HDL介绍

---

### 1、什么是Verilog HDL

- Verilog HDL是一种用于**数字**逻辑电路设计的硬件描述语言 ( Hardware Description Language ) , 可以用来进行数字电路的仿真验证、时序分析、逻辑综合。
  - 用Verilog HDL描述的电路设计就是该电路的**Verilog HDL模型**。
  - Verilog HDL 既是一种**行为**描述语言也是一种**结构**描述语言。
- 既可以用电路的功能描述 , 也可以用元器件及其之间的连接来建立Verilog HDL模型。



---

## 2、Verilog HDL的发展历史

- 1983 年 , 由 GDA ( GateWay Design Automation ) 公司的Phil Moorby首创 ;
- 1989年 , Cadence公司收购了GDA公司 ;
- 1990 年 , Cadence 公司公开发表 Verilog HDL ;
- 1995 年 , IEEE 制定并公开发表 Verilog HDL1364-1995标准 ;
- 1999年 , 模拟和数字电路都适用的Verilog标准公开发表

### 3、不同层次的Verilog HDL抽象

---

□ Verilog HDL模型可以是实际电路的不同级别的抽象。抽象级别可分为**五级**：

- **系统级** (system level)：用高级语言结构（如case语句）实现的设计模块外部性能的模式；
- **算法级** (algorithmic level)：用高级语言结构实现的设计算法模型（写出逻辑表达式）；
- **RTL级** (register transfer level)：描述数据在寄存器之间流动和如何处理这些数据的模型；
- **门级** (gate level)：描述逻辑门（如与门、非门、或门、与非门、三态门等）以及逻辑门之间连接的模型；
- **开关级** (switch level)：描述器件中三极管和储存节点及其之间连接的模型。

## 4、Verilog HDL的特点

### □ 语法结构上的主要特点：

- 形式化地表示电路的**行为**和**结构**；
- 借用**C语言**的结构和语句；
- 可在多个层次上对所设计的系统加以描述，语言对设计规模不加任何限制；
- 具有混合建模能力：一个设计中的各子模块可用不同级别的抽象模型来描述；
- 基本逻辑门、开关级结构模型均内置于语言中，可直接调用；
- 易创建用户定义原语（UDP，User Designed Primitive）。



与C语言  
非常相  
似！

### □ 易学易用，功能强

# 三、Verilog HDL基本结构

## 特点

- Verilog HDL程序是由**模块**构成的。每个模块嵌套在**module**和**endmodule**声明语句中。模块是可以进行层次嵌套的。
- 每个Verilog HDL源文件中只准有一个**顶层模块**，其他为子模块。
- 每个模块要进行端口定义，并说明输入输出端口，然后对模块的功能进行行为逻辑描述。
- 程序书写格式自由，一行可以写几个语句，一个语句也可以分多行写。
- 除了**endmodule**语句、**begin\_end**语句和**fork\_join**语句外，**每个语句和数据定义的最后必须有分号**。
- 可用**/\*.....\*/**和**//...**对程序的任何部分作注释。加上必要的注释，以增强程序的可读性和可维护性。

## 1、Verilog HDL模块的结构

- Verilog的基本设计单元是“**模块 (block)**”。
- Verilog 模块的结构由在**module**和**endmodule**关键词之间的**4**个主要部分组成：

- 1 端口定义
- 2 I/O说明
- 3 信号类型声明
- 4 功能描述

```
module block1(a, b, c, d );  
    input a, b, c;  
    output d;  
    wire x;  
    assign d = a | x;  
    assign x = ( b & ~c );  
endmodule
```



## 2、逻辑功能定义

□ 在Verilog 模块中有2种方法可以描述电路的逻辑功能：

① 用assign 语句

```
assign x = ( b & ~c );
```

连续赋值语句

常用于描述  
组合逻辑

② 用元件例化 (instantiate)

```
and myand3( f,a,b,c);
```

门元件例化

门元件关键字

例化元件名

- ❖ 注1：元件例化即是调用Verilog HDL提供的元件；
- ❖ 注2：元件例化包括门元件例化和模块元件例化；
- ❖ 注3：每个实例元件的名字必须唯一！以避免与其它调用元件的实例相混淆。
- ❖ 注4：例化元件名也可以省略！

模块元件例化

### 3、数据类型及常量、变量

#### □ wire型变量

- 最常用的nets型变量，常用来表示以assign语句赋值的组合逻辑信号。
- 模块中的输入/输出信号类型缺省为wire型。
- 可用做任何方程式的输入，或“assign”语句和实例元件的输出

格式

wire 数据名1, 数据名2, ....., 数据名n;

wire型向量（总线）

wire[n-1:0] 数据名1, 数据名2, ....., 数据名m;  
或 wire[n:1] 数据名1, 数据名2, ....., 数据名m;

每条总线  
位宽为n

共有m条  
总线

# 数据类型及常量、变量

## register型变量

- **定义**——对应具有状态保持作用的电路元件（如触发器、寄存器等），常用来表示过程块语句（如initial, always, task, function）内的指定信号。
- **常用register型变量：**
  - **reg**：常代表触发器
  - **integer**：32位带符号整数型变量
  - **real**：64位带符号实数型变量
  - **time**：无符号时间变量

纯数学的  
抽象描述

# 数据类型及常量、变量

---

- ❖ register型变量与nets型变量的根本区别是： register型变量需要被明确地赋值，并且在被重新赋值前一直保持原值。
- ❖ register型变量必须通过过程赋值语句赋值！不能通过assign语句赋值！
- ❖ 在过程块内被赋值的每个信号必须定义成register型！

# 数据类型及常量、变量

## □ reg型变量

- **定义**——在过程块中被赋值的信号，**往往**代表触发器，但**不一定**就是触发器（也可以是组合逻辑信号）！

**格式**

**reg** 数据名1, 数据名2, ....., 数据名n;

reg型向量（总线）

**reg[n-1:0]** 数据名1, 数据名2, ....., 数据名m;  
或 **reg[n:1]** 数据名1, 数据名2, ....., 数据名m;

每个向量  
位宽为**n**

共有**m**个reg  
型向量

➤ **[例]** `reg[4:1] regc, regd;` //regc,regd为4位宽的reg型向量



# 数据类型及常量、变量

## Verilog中reg与wire的区别

- 用reg型变量生成组合逻辑举例：

```
module rw1( a, b, out1, out2 ) ;
```

```
    input a, b ;
```

```
    output out1, out2 ;
```

```
    reg out1 ;
```

```
    wire out2 ;
```

连续赋值语句

```
    assign out2 = a ;
```

```
    always @(b)
```

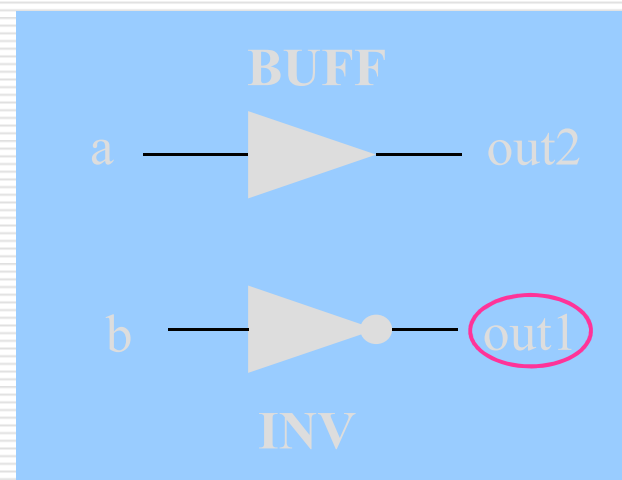
电平触发

```
        out1 <= ~b ;
```

```
endmodule
```

过程赋值语句

reg型变量既可生成触发器，也可生成组合逻辑； wire型变量只能生成组合逻辑。



## 2.5 硬件描述语言Verilog HDL基础

---

### 2.5.1 Verilog语言的基本语法规则

### 2.5.2 变量的数据类型

### 2.5.3 运算符及其优先级

### 2.5.4 Verilog内部的基本门级元件

### 2.5.5 Verilog程序的基本结构

### 2.5.6 逻辑功能的仿真与测试

## **2.5 硬件描述语言Verilog HDL基础**

---

**硬件描述语言HDL(Hardware Description Language )**

**类似于高级程序设计语言.它是一种以文本形式来描述数字系统硬件的结构和行为的语言,用它可以表示逻辑电路图、逻辑表达式，复杂数字逻辑系统完成的逻辑功能。HDL是高层次自动化设计的起点和基础。**

---

## 计算机对HDL的处理:

---

**逻辑仿真** 是指用计算机仿真软件对数字逻辑电路的结构和行为进行预测.仿真器对HDL描述进行解释,以文本形式或时序波形图形式给出电路的输出。在仿真期间如发现设计中存在错误,就再要对HDL描述进行及时的修改。

**逻辑综合** 是指从HDL描述的数字逻辑电路模型中导出电路基本元件列表以及元件之间的连接关系(常称为门级网表)的过程。类似对高级程序设计进行编译产生目标代码的过程.产生门级元件及其连接关系的数据库,根据这个数据库可以制作出集成电路或印刷电路板PCB。

---

## 2.5.1 Verilog语言的基本语法规则

---

为对数字电路进行描述（常称为建模），Verilog语言规定了一套完整的语法结构。

1．间隔符：Verilog 的间隔符主要起分隔文本的作用，可以使文本错落有致，便于阅读与修改。

间隔符包括空格符（\b）、TAB 键（\t）、换行符（\n）及换页符。

2．注释符：注释只是为了改善程序的可读性，在编译时不起作用。

**多行注释符**（用于写多行注释）：**`/* --- */`**；

**单行注释符**：以**`//`**开始到行尾结束为注释文字。

---



### 3 . 标识符和**关键词**

**标识符:**给对象（如模块名、电路的输入与输出端口、变量等）

**取名所用的字符串。以英文字母或下划线开始**

**如，clk、counter8、\_net、bus\_A 。**

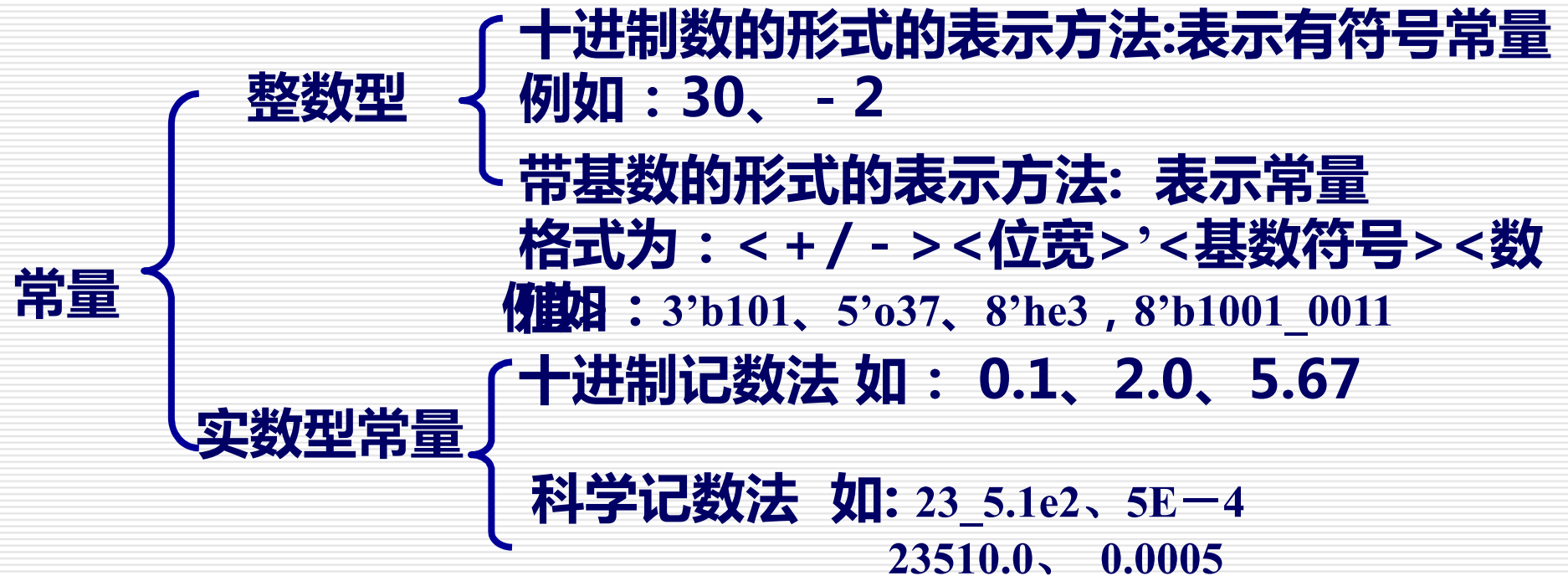
**关键词:**是Verilog语言本身规定的特殊字符串，用来定义语言的结构。例如，module、endmodule、input、output、wire、reg、and等都是关键词。关键词都是小写，关键词不能作为标识符使用。

### 4 . 逻辑值集合

**为了表示数字逻辑电路的逻辑状态，Verilog语言规定了4种基本的逻辑值。**

0	逻辑0、逻辑假
1	逻辑1、逻辑真
x或X	不确定的值（未知状态）
z或Z	高阻态

## 5 . 常量及其表示



Verilog允许用参数定义语句定义一个标识符来代表一个常量，称为符号常量。定义的格式为：

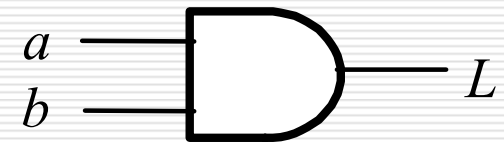
parameter 参数名1 = 常量表达式1 , 参数名2 = 常量表达式2 , ..... ; 如 parameter BIT=1, BYTE=8, PI=3.14;

## 6 . 字符串:字符串是双撇号内的字符序列

## 2.5.2 变量的数据类型

**1线网类型:**是指输出始终根据输入的变化而更新其值的变量,它一般指的是硬件电路中的各种物理连接.

例:网络型变量L的值由与门的驱动信号a和b所决定,即 $L = a \& b$ 。a、b的值发生变化,线网L的值会立即跟着变化。



常用的网络类型由关键词wire定义  
wire型变量的定义格式如下:

wire [n-1:0] 变量名1, 变量名2, ..., 变量名n;

变量宽度

例:wire L; //将上述电路的输出信号L声明为网络型变量  
wire [7:0] data bus; //声明一个8-bit宽的网络型总线变量

## 2、寄存器类型

寄存器型变量对应的是具有状态保持作用的电等路元件,如触发器寄存器。寄存器型变量只能在initial或always内部被赋值。

### 4种寄存器类型的变量

抽象描述,  
不对应具  
体硬件

寄存器类型	功能说明
reg	常用的寄存器型变量
integer	32位带符号的整数型变量
real	64位带符号的实数型变量 ,
time	64位无符号的时间变量

例 : reg clock ; //定义一个1位寄存器变量  
reg [3:0] counter; //定义一个4位寄存器变量

## 2.5.3 运算符及其优先级

### 1. 运算符

运算符分为**算术运算符、逻辑运算符、关系运算符、移位运算符等**

类型	符号	功能说明	类型	符号	功能说明
算术运算符	+ - - * /	二进制加 二进制减 2的补码 二进制乘 二进制除	关系运算符 (双目运算符)	> < >= <= == !=	大于 小于 大于或等于 小于或等于 相等 不相等
位运算符 (双目运算符)	~ &   ^ ^~ 或 ~^	按位取反 按位与 按位或 按位异或 按位同或	缩位运算符 (单目运算符)	& ~&   ~  ^ ^~ 或 ~^	缩位与 缩位与非 缩位或 缩位或非 缩位异或 缩位同或
逻辑运算符 (双目运算符)	! && 	逻辑非 逻辑与 逻辑或	移位运算符 (双目运算符)	>> <<	右移 左移
位拼接运算符	{} {}	将多个操作数 拼接成为一个 操作数	条件运算符 (三目运算符)	?:	根据条件表达式是否成立, 选择表达式



---

## 位拼接运算符

作用是将两个或多个信号的某些位拼接起来成为一个新的操作数，进行运算操作。

设  $A=1'b1$  ,  $B=2'b10$  ,  $C=2'b00$

则  $\{B,C\} = 4'b1000$

$\{A,B[1],C[0]\} = 3'b110$

$\{A,B,C,3'b101\}=8'b11000101$ 。

对同一个操作数的重复拼接还可以双重大括号构成的运算符 $\{\{\}\}$   
例如  $\{4\{A\}\}=4'b1111$  ,  $\{2\{A\},2\{B\},C\}=8'b11101000$ 。

---

## 位运算符与缩位运算的比较

A : 4'b1010、  
B : 4'b1111 ,

位运算	$\sim A = 0101$ $\sim B = 0000$	$A \& B = 1010$	$A   B = 1111$	$A \wedge B = 0101$	$A \sim \wedge B = 1010$
缩位运算	$\& A = 1 \& 0 \& 1 \& 0 = 0$	$\sim \& A = 1$ $\& B = 1$	$  A = 1$ $\sim   B = 0$	$\wedge A = 0$ $\wedge B = 0$	$\sim \wedge A = 1$ $\sim \wedge B = 1$

## 2. 运算符的**优先级**

优先级的顺序从下向上依次增加。

类型	符号	优先级别	
取反	! ~ -(求2的补码)	最高优先级	
算术	* / + -		
移位	>> <<		
关系	< <= > >=		
等于	== !=		
缩位	& ~& ^ ^~   ~		
逻辑	&& 		
条件	?:	最低优先级	

---

## 条件运算符

是三目运算符，运算时根据条件表达式的值选择表达式。

一般用法：

`condition_expr?expr1:expr2;`

首先计算第一个操作数`condition_expr`的值，如果结果为逻辑1，则选择第二个操作数`expr1`的值作为结果返回，结果为逻辑0，选择第三个操作数`expr2`的值作为结果返回。

---

## 2.5.4 Verilog内部的基本门级元件

门级建模:将逻辑电路图用HDL规定的文本语言表示出来。

### 基本门级元件模型

三态门

多输出门

多输入门

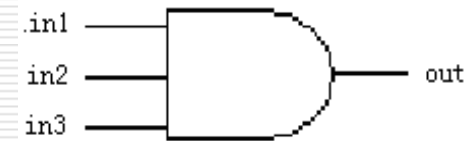
元件符号	功能说明	元件符号	功能说明
and	多输入端的与门	nand	多输入端的与非门
or	多输入端的或门	nor	多输入端的或非门
xor	多输入端的异或门	xnor	多输入端的异或非门
buf	多输出端的缓冲器	not	多输出端的反相器
bufif1	控制信号高电平有效的三态缓冲器	notif1	控制信号高电平有效的三态反相器
bufif0	控制信号低电平有效的三态缓冲器	notif0	控制信号低电平有效的三态反相器

## 1、多输入门

只允许有一个输出，但可以有多多个输入。

调用名

and A1 ( out , in1 , in2 , in3 ) ;



and真值表

and		输入1			
		0	1	X	Z
输入2	0	0	0	0	0
	1	0	1	x	x
	x	0	x	x	x
	Z	0	x	x	x

X- 不确定状态

Z- 高阻态

or真值表

or		输入1			
		0	1	X	Z
输入2	0	0	1	X	X
	1	1	1	1	1
	X	X	1	X	X
	Z	X	1	X	X

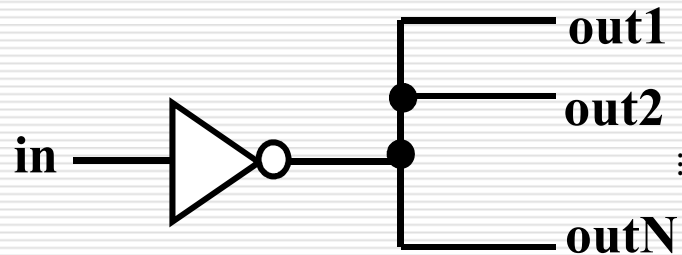
xor真值表

xor		输入1			
		0	1	X	Z
输入2	0	0	1	X	X
	1	1	0	X	X
	X	X	X	X	X
	Z	X	X	X	X

## 2、多输出门

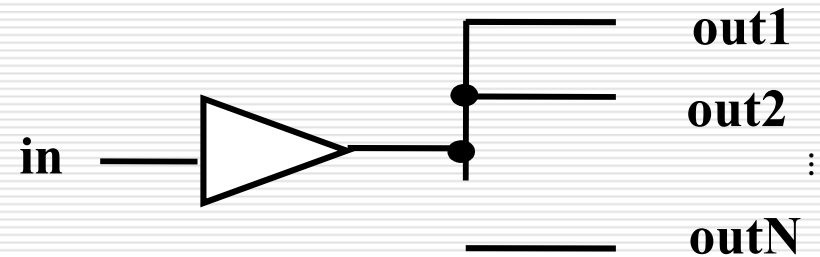
允许有多个输出，但只有一个输入。

`not N1 ( out1 , out2 , ... , in ) ;`  
`buf B1 ( out1 , out2 , ... , in ) ;`



not真值表

not	输入			
	0	1	x	z
输出	1	0	x	x



buf真值表

buf	输入			
	0	1	x	z
输出	0	1	x	x



### 3、三态门

有一个输出、一个数据输入和一个输入控制。  
如果输入控制信号无效，则三态门的输出为高阻态z。

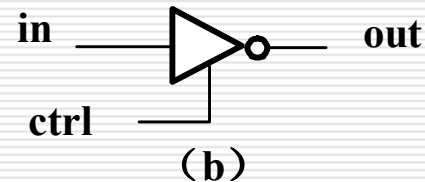
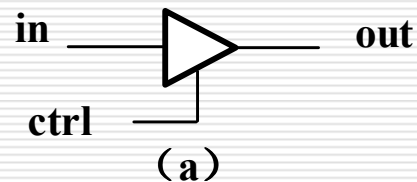


图 4.6.3 三态门元件模型

(a) bufif1 (b) notif1

bufif1真值表

bufif1		控制输入			
		0	1	x	z
数据输入	0	z	0	0/z	0/z
	1	z	1	1/z	1/z
	x	z	x	x	x
	z	z	x	x	x

notif1真值表

notif1		控制输入			
		0	1	x	z
数据输入	0	z	1	1/z	1/z
	1	z	0	0/z	0/z
	x	z	x	x	x
	z	z	x	x	x

## 2.5.5 Verilog程序的基本结构

---

模块是Verilog描述电路的基本单元。对数字电路建模时，用一个或多个模块。不同模块之间通过端口进行连接。

- 1、每个模块以关键词module开始，以endmodule结束。
  - 2、每个模块先要进行端口的定义，并说明输入 (input)和输出 (output),然后对模块功能进行描述。
  - 3、除了endmodule语句外，每个语句后必须有分号。
  - 4、可以用/\* --- \*/和//.....对程序的任何部分做注释。
  - 5、逻辑功能的描述方式有三种不同风格：结构描述方式（门级描述方式）数据流描述方式，行为描述方式。
-

## 模块定义的一般语法结构如下：

**module** 模块名（端口名1，端口名2，端口名3，…）；

    端口类型说明(input, outout, inout)；

    参数定义(可选)；

    数据类型定义(wire, reg等)；

} 说明部分

    实例化低层模块和基本门级元件；

    连续赋值语句(assign)；

    过程块结构(initial和always)

        行为描述语句；

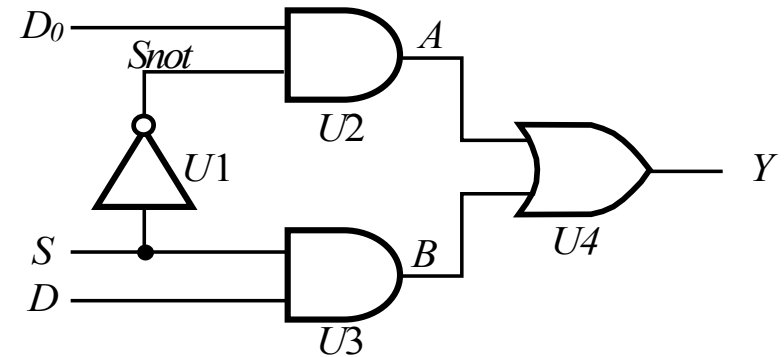
} 逻辑功能描述部分，其顺序是任意的

**endmodule**

## 例 用结构描述方式建立门电路Verilog模型

模块名

```
module mux2to1(D0, D1, S, Y);  
  input D0, D1, S; //定义输入信号  
  output Y; //定义输出信号  
  wire Snot, A, B ; //定义内部节点信号数据类型  
  //下面对电路的逻辑功能进行描述  
  not U1(Snot, S);  
  and U2(A, D0, Snot);  
  and U3(B, D1, S);  
  or U4(Y, A, B);  
endmodule
```



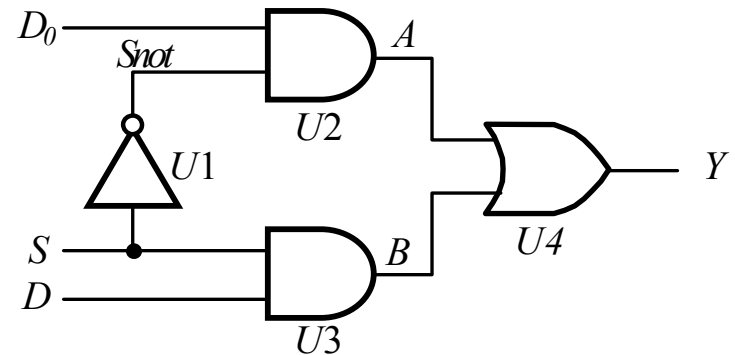
端口类型说明

数据类型说明

电路结构描述

## 例 用数据流描述方式建立模型

$$Y = D_0 \cdot \bar{S} + D_1 \cdot S$$



```
module mux2to1_dataflow(D0, D1, S, Y);
```

```
  input D0, D1, S;
```

```
  output Y;
```

```
  wire Y;
```

```
//下面是逻辑功能描述
```

```
  assign Y = (~S & D0) | (S & D1); //表达式左边Y必须是wire型
```

```
endmodule
```

端口类型说明

数据类型说明

电路结构描述

注意，在assign语句中，左边变量的数据类型必须是wire型。

## 例 用行为描述方式建立模型

$$Y = D_0 \cdot \bar{S} + D_1 \cdot S$$

```
module mux2to1_bh(D0, D1, S, Y);
```

```
  input D0, D1, S;
```

```
  output Y;
```

```
  reg Y;
```

数据类型  
说明

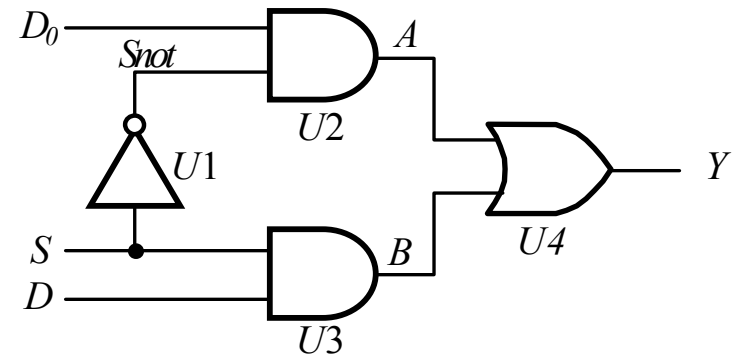
```
//逻辑功能描述
```

```
  always @(S or D0 or D1)
```

```
    if (S == 1) Y = D1; //也可以写成 if (S) Y = D1;
```

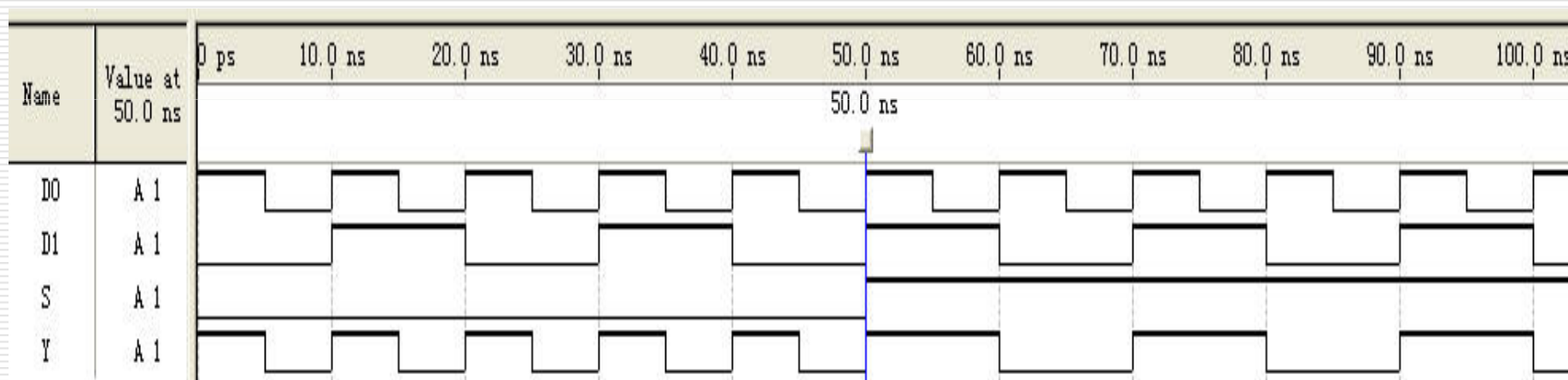
```
    else Y = D0; //注意表达式左边的Y必须是reg型
```

```
endmodule
```



## 2.5.6 逻辑功能的仿真与测试

逻辑电路的设计块完成后，就要测试这个设计块描述的逻辑功能是否正确。为此必须在输入端口加入测试信号，而从其输出端口检测其结果是否正确，这一过程常称为搭建测试平台。根据仿真软件的不同，搭建测试平台的方法也不同。



# 第三章

---

## 逻辑门电路



## 3.1 逻辑门电路简介

---

### 3.1.1 各种逻辑门电路系列简介

### 3.1.2 开关电路

---

### 3.1.1 各种逻辑门电路系列简介

---

1、逻辑门:实现基本逻辑运算和常用逻辑运算的单元电路。

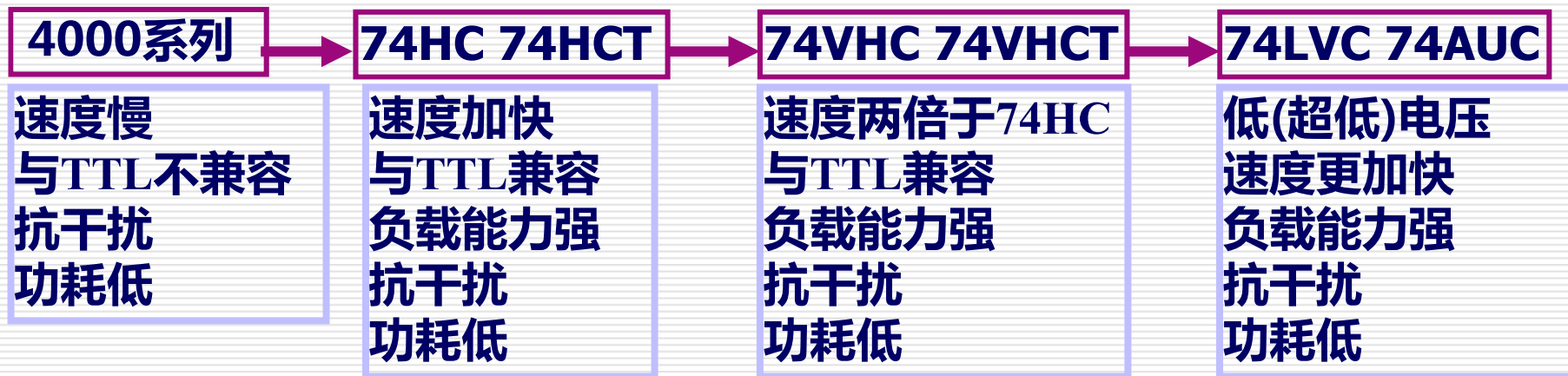
2、逻辑门电路的分类



## 3.1.1 数字集成电路简介

### 1.CMOS集成电路:

广泛应用于超大规模、甚大规模集成电路



### 2.TTL 集成电路:

广泛应用于中大规模集成电路



## ● 1、TTL门电路的基本结构

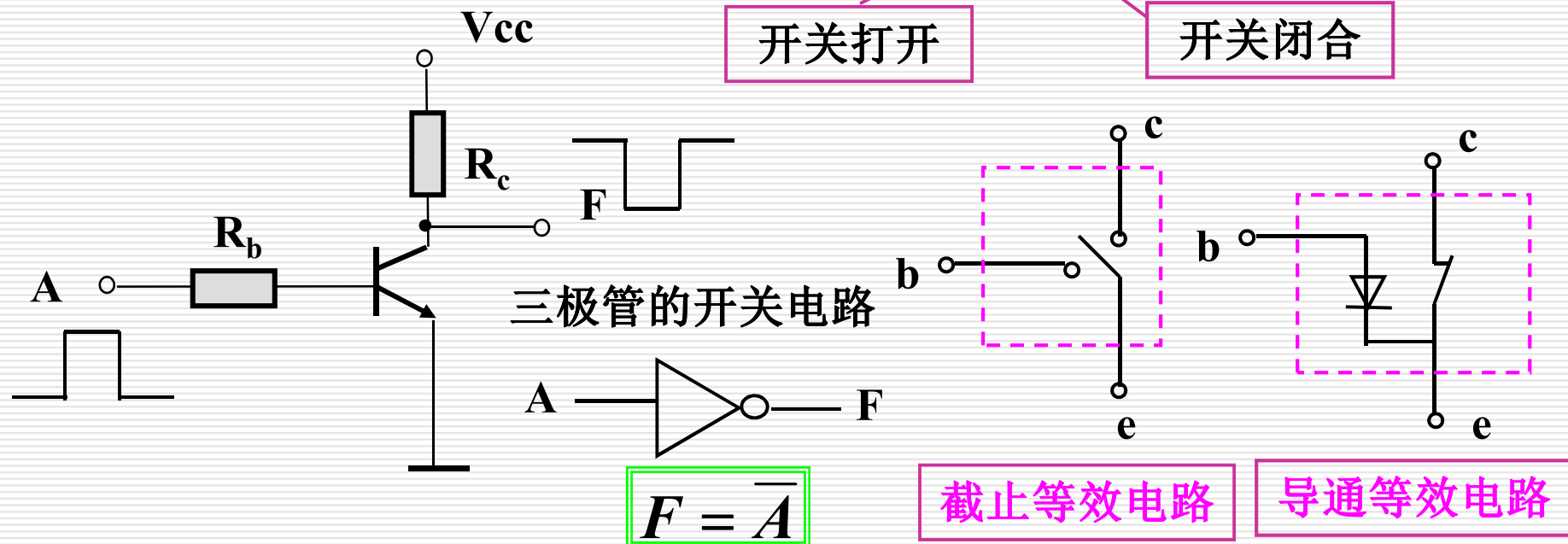
晶体管-晶体管逻辑（电路）

Transistor-Transistor Logic

TTL门

### ● （1）三极管非门电路

数字电路利用三极管的开关特性——截止及饱和导通状态。

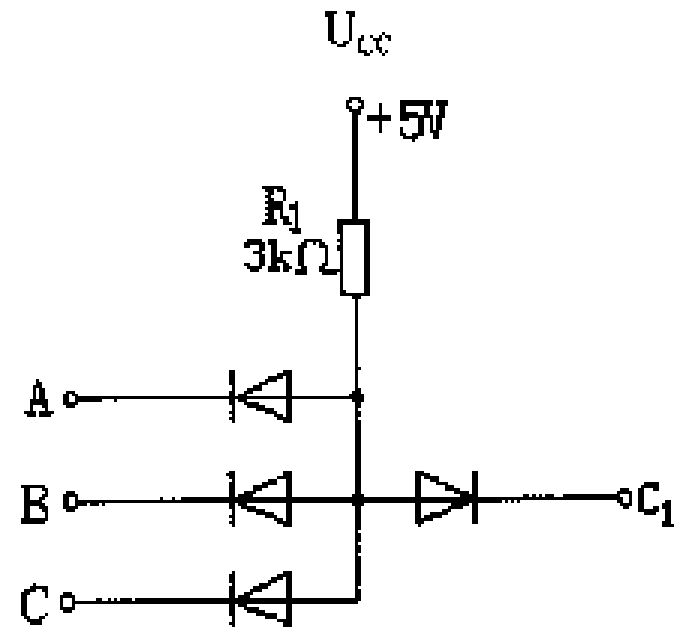
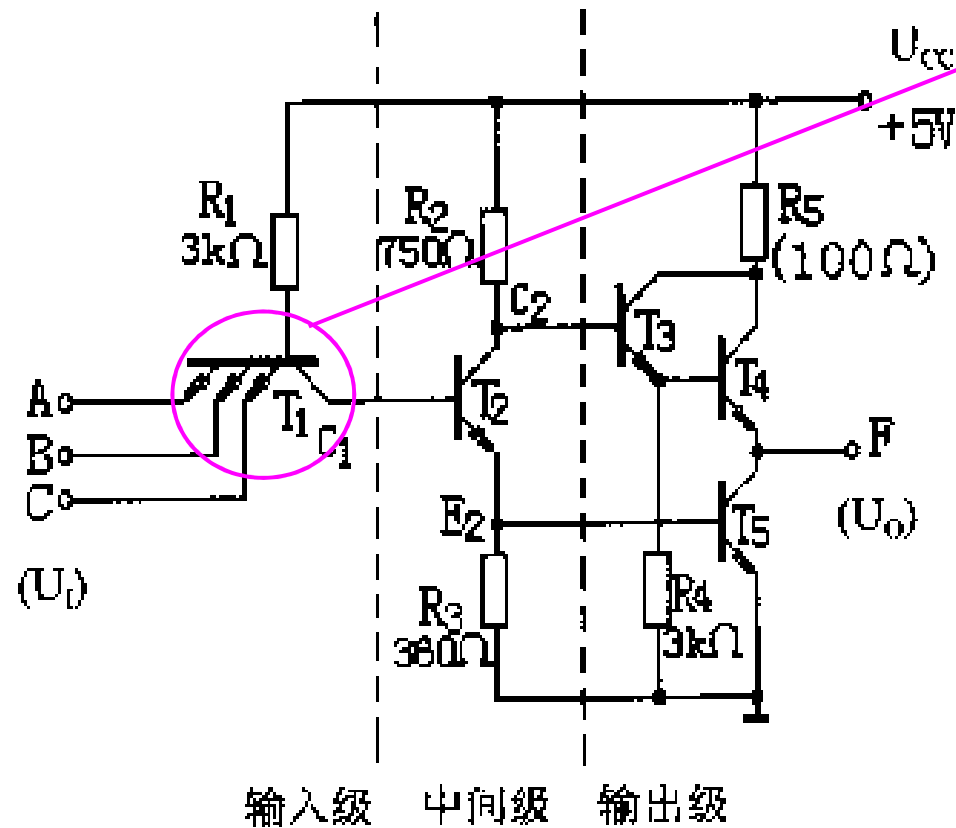


当输入信号为高电平时，三极管饱和导通，输出为低电平。

当输入信号为低电平时，三极管截止，输出为高电平。

反相器

## ● (2)、三极管与非门电路

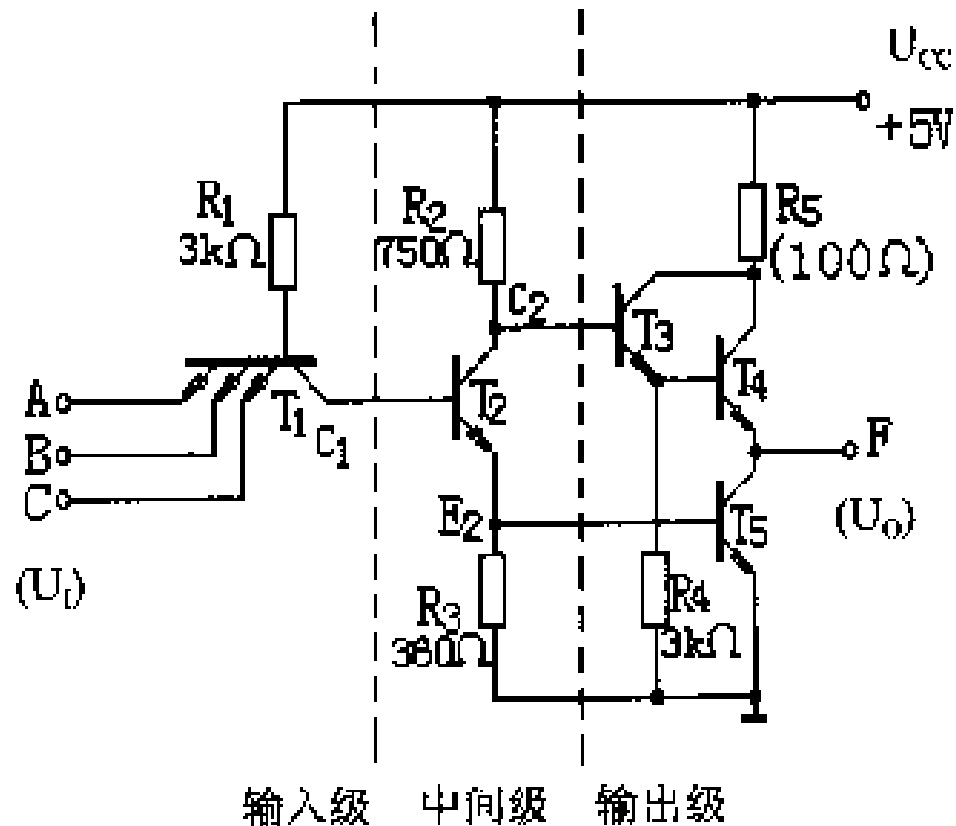


(1) 有一个(或几个)输入端接低电平的情况

当有一个(或几个)输入端接低电平 $U_i(+0.3V)$ 时,  $T_1$ 导通,  $C_1$ 为低电位  
 $T_2$ 截止、 $T_5$ 截止,  $C_2$ 为高电位,  $T_3 T_4$ 导通,  $F$ 高电位,  $F=1$ 。

$$F = \overline{ABC} = 1$$

$$F = \overline{ABC}$$



## (2) 所有输入端都接高电平的情况

当输入端A、B、C全部接高电平 $U_{IH}(+3.6V)$ 时， $T_1$ 管处于截止状态， $C_1$ 为高电位 $T_2$ 导通、 $T_5$ 导通， $C_2$ 为低电位， $T_3$   $T_4$ 截止，F低电位，**F=0**。

$$Y = \overline{ABC} = 0$$

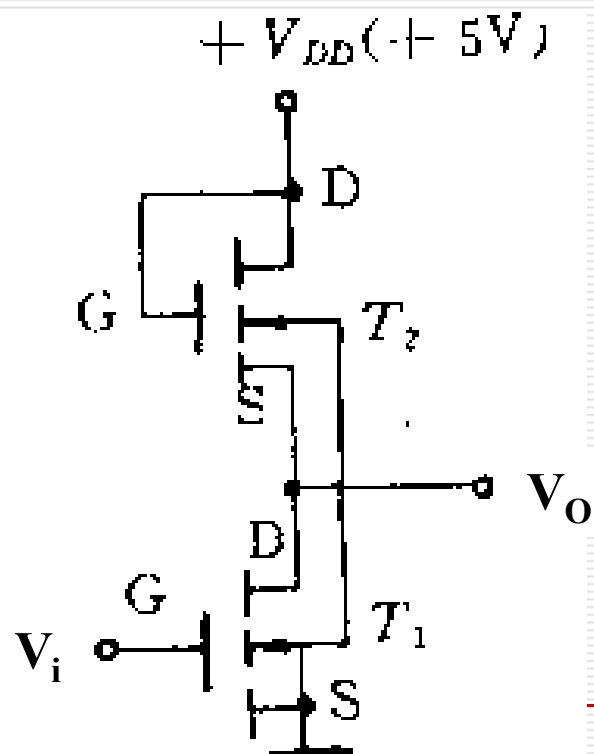
## ● 2、MOS管电路

# Metal\_Oxide\_Semiconductor type Field Effect Transistor 金属-氧化物-半导体场效应管

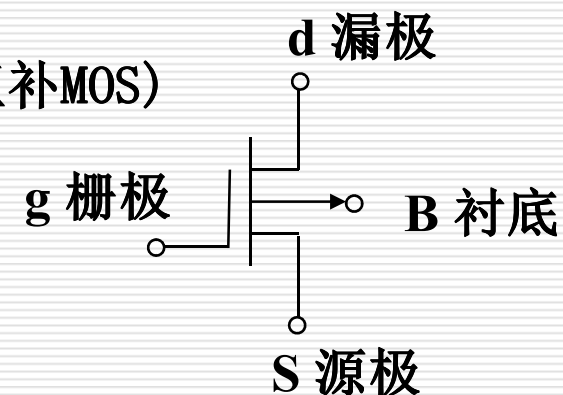
MOS

MOS管电路的类型 {  
PMOS P沟道管  
NMOS N沟道管  
CMOS 同时使用PMOS和NMOS (互补MOS)

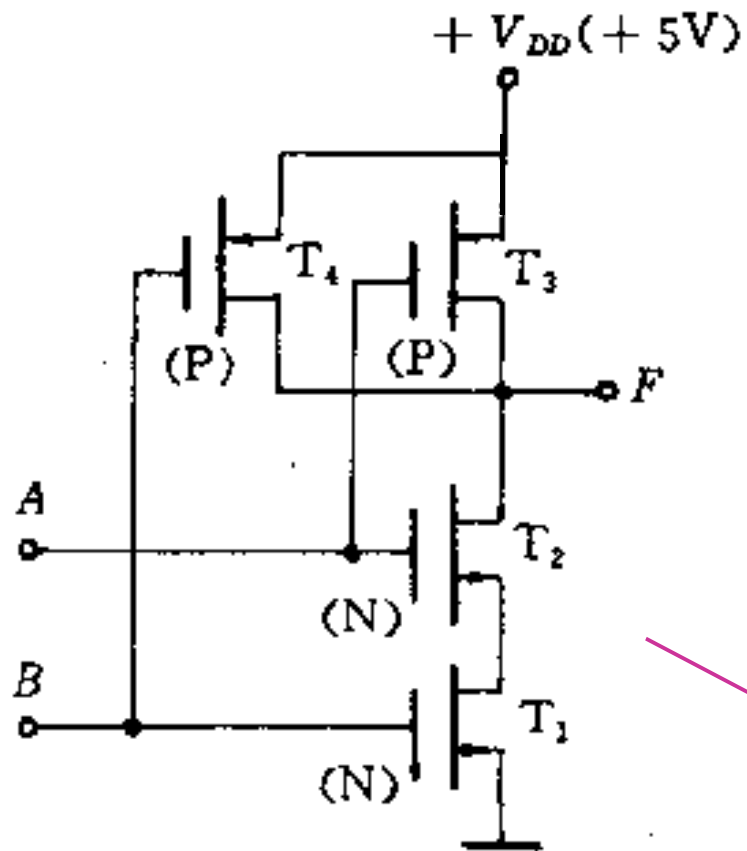
### ● (1)、MOS管非门电路



$T_1$ 对信号起反相作用 {  
 $V_i=1, V_o=0$   
 $V_i=0, V_o=1$



## ● (2)、CMOS管与非门电路



AB输入均为高电平时， $T_1T_2$ 导通。

输出为低电平。

AB输入均为低电平时， $T_1T_2$ 截止。

输出为高电平。

AB输入有一个为低电平时， $T_1$ 或 $T_2$ 截止。

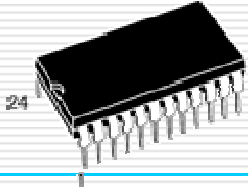
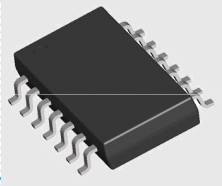
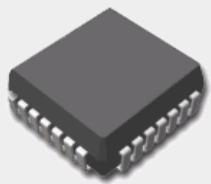
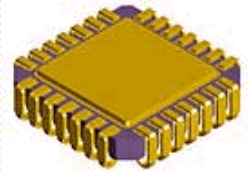
输出为高电平。

$$F = \overline{AB}$$

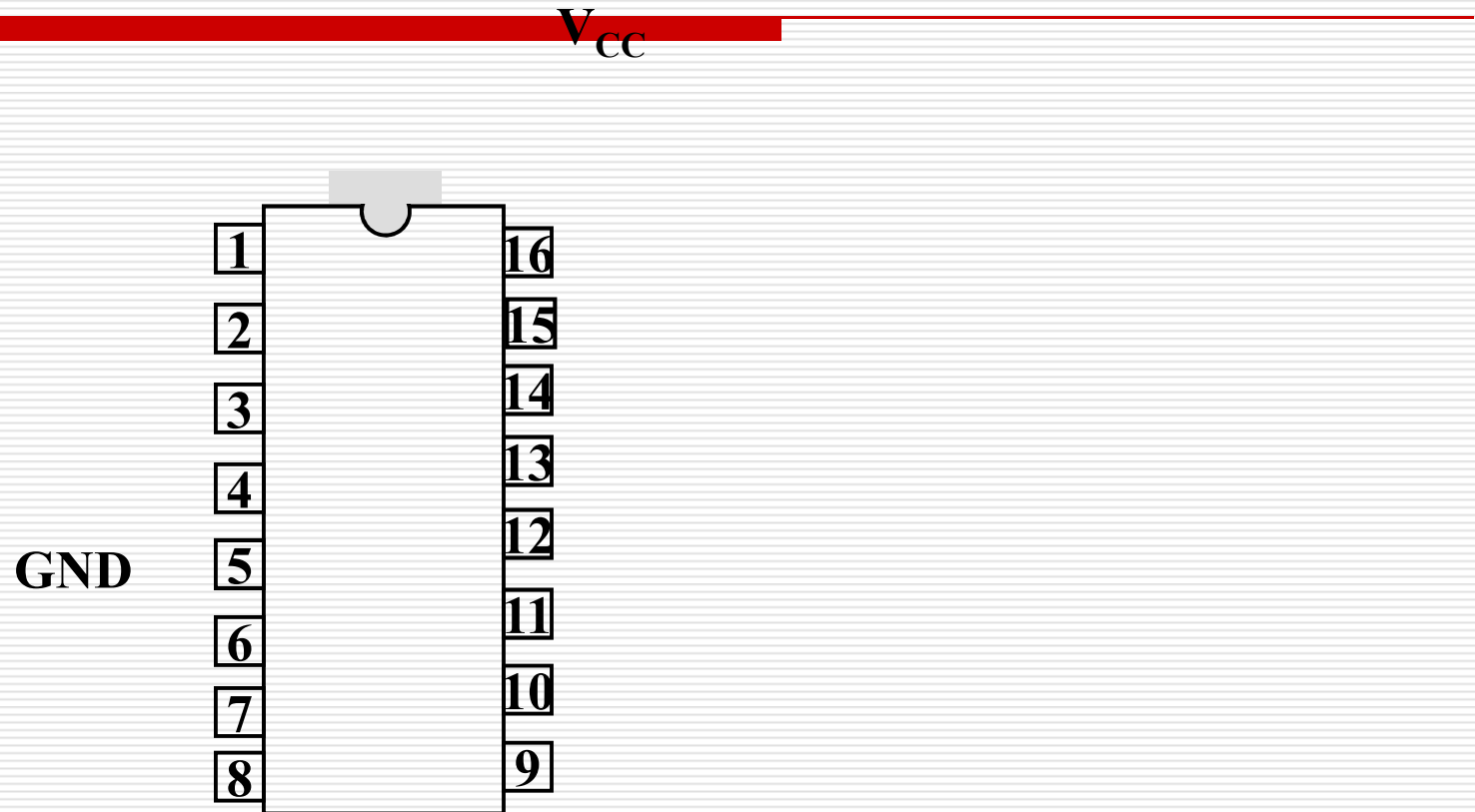
实现与非逻辑



## ● 二、集成电路的封装类型

封装形式		外观
DIP	双列直插	
SOIC	表面贴	
PLCC	嵌入式	
LCCC		

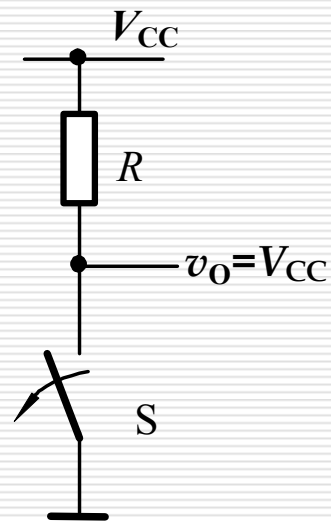
## ● 标准封装与管脚



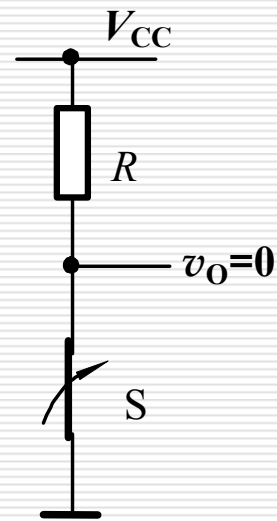
## 3.1.2 开关电路

逻辑变量取值0或1，对应电路中电子器件的“闭合”与“断开”。

MOS管或BJT管可以作为开关。



(a) 输出逻辑1



(b) 输出逻辑0

## **3.2 基本CMOS逻辑门电路**

---

### **3.2.1 MOS管及其开关特性**

### **3.2.2 CMOS反相器**

### **3.2.3 其他基本CMOS逻辑门电路**

### **3.2.4 CMOS传输门**

---

## 3.2.1 MOS管及其开关特性

---

CMOS门电路是以MOS管为开关器件。

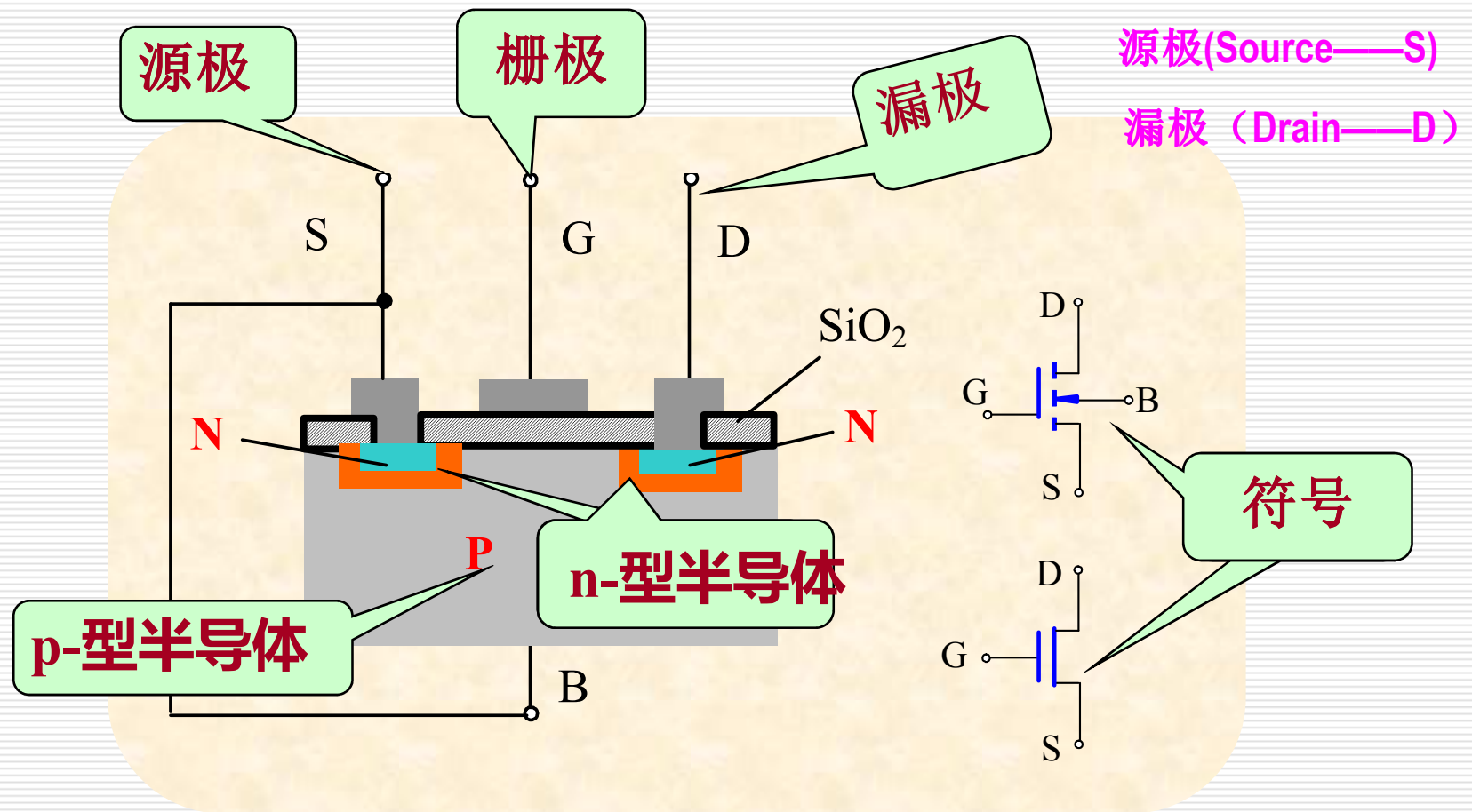
MOS管的分类：



# 1. N沟道增强型MOS管的结构和工作原理

## MOS管的分类：

栅极（Gate——G，也叫做门极）



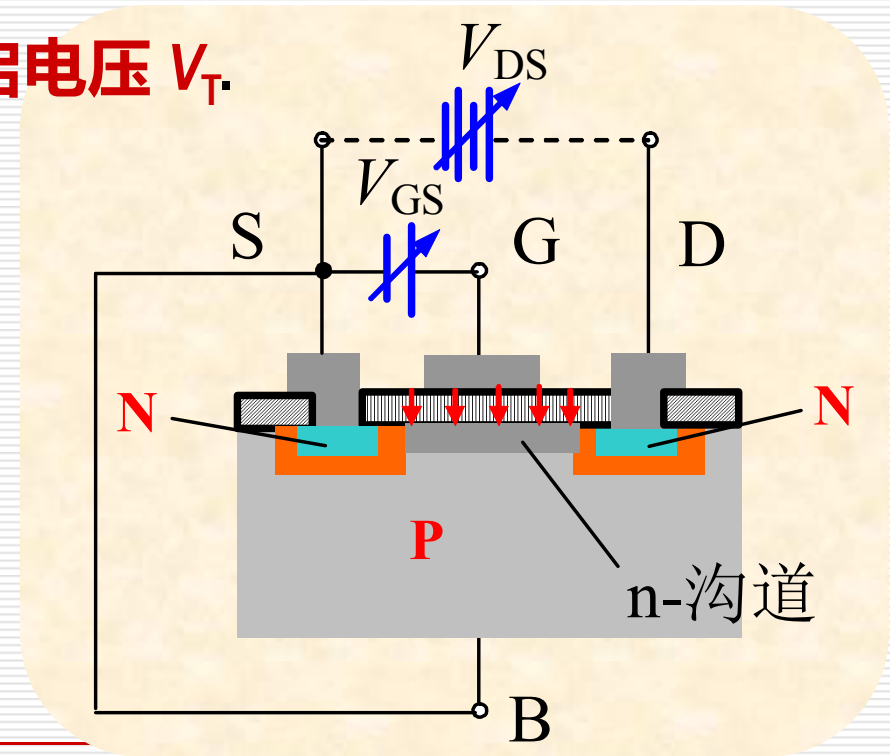
# 1. N沟道增强型MOS管的结构和工作原理

## (1) $V_{GS}$ 控制沟道的导电性

□  $v_{GS}=0, v_{DS} \neq 0$ , 等效背靠背连接的两个二极管,  $i_D \approx 0$ 。

□  $v_{GS} > 0$ , 建立电场  $\rightarrow$  反型层  $\rightarrow v_{DS} > 0, i_D \neq 0$ 。

□ 沟道建立的最小  $v_{GS}$  值称为开启电压  $V_T$ 。



# 1. N沟道增强型MOS管的结构和工作原理

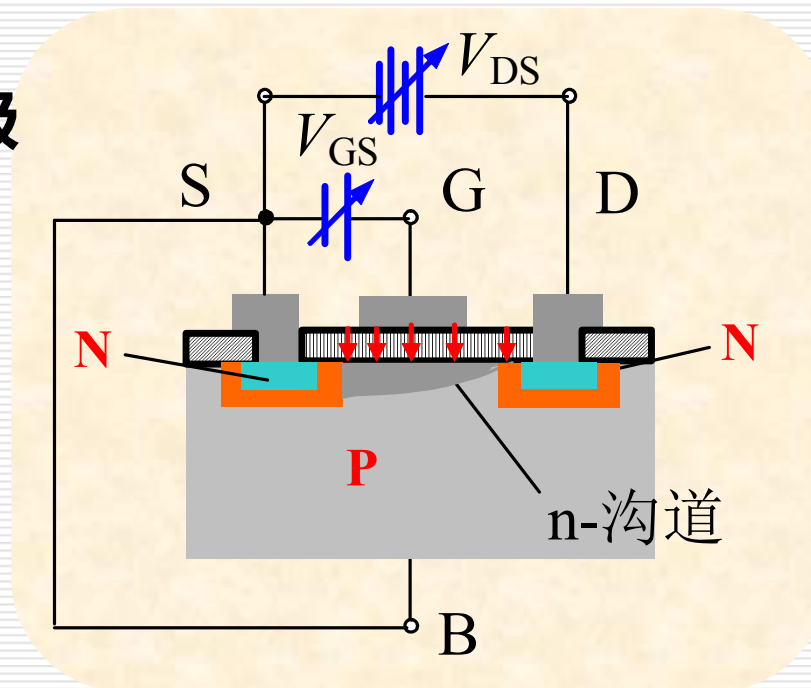
## (2) $V_{GS}$ 和 $V_{DS}$ 共同作用

□  $v_{GS} > V_T$ ,  $v_{DS} > 0$ , 靠近漏极的电压减小。

□ 当  $V_{GS} > V_T$ ,  $i_D$  随  $V_{DS}$  增加几乎成线性增加。

□ 当  $v_{DS} \uparrow \rightarrow v_{GD} = (v_{GS} - v_{DS}) \leq V_T$ , 漏极处出现**夹断**。

□ 继续增加  $V_{DS} \uparrow \rightarrow$  夹断区域变大,  $i_D$  饱和。





## 2. N沟道增强型MOS管的输出特性和转移特性

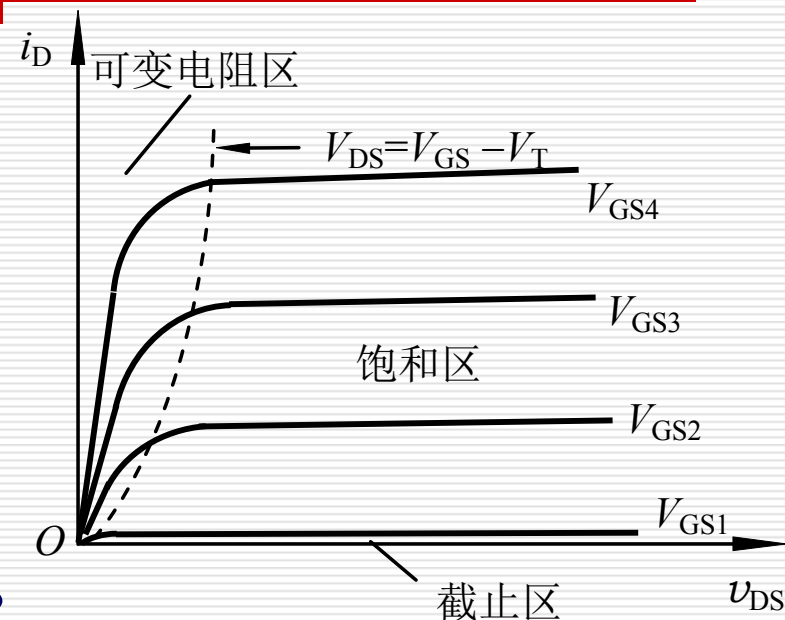
### 输出特性分为

□ **截止区** :  $v_{GS} < V_T$ ,  $i_D = 0$

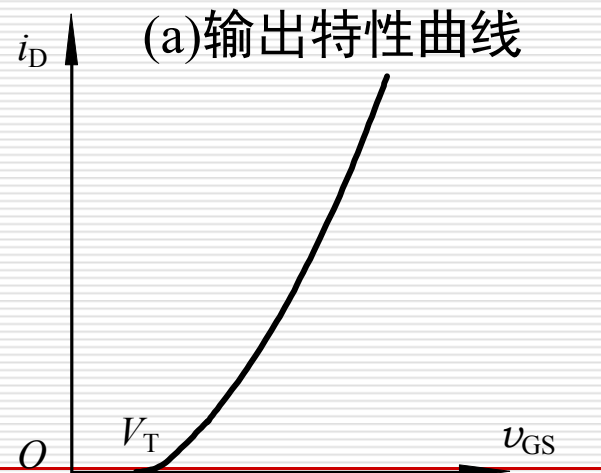
□ **可变电阻区** : 沟道产生,  $i_D$  随  $v_{DS}$  线性增加,  $r_{ds}$  为受  $v_{GS}$  控制可变电阻。

$$r_{ds} = \left. \frac{dv_{DS}}{di_D} \right|_{v_{GS}=\text{const}} = \frac{1}{2K_n (v_{GS} - V_T)}$$

□ **饱和区** :  $v_{GS} > V_T$ ,  $v_{DS} > v_{GS} - V_T$



(a) 输出特性曲线

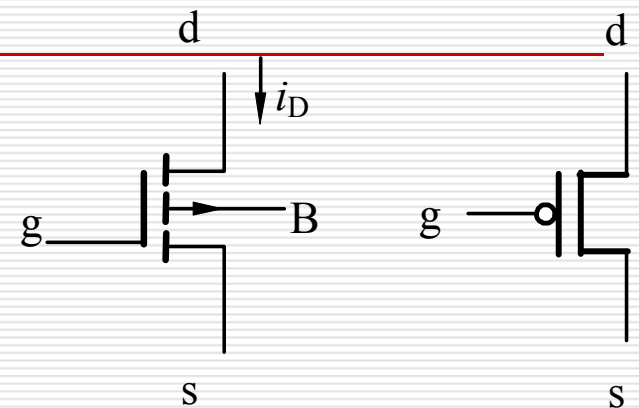


(b) 转移特性曲线

### 3. 其他类型的MOS管

#### (1) P沟道增强型MOS管

- 结构与NMOS管相反。
- $V_{GS}$ 、 $V_{DS}$  电压极性与NMOS管相反。
- 开启电压 $V_T$ 为负值

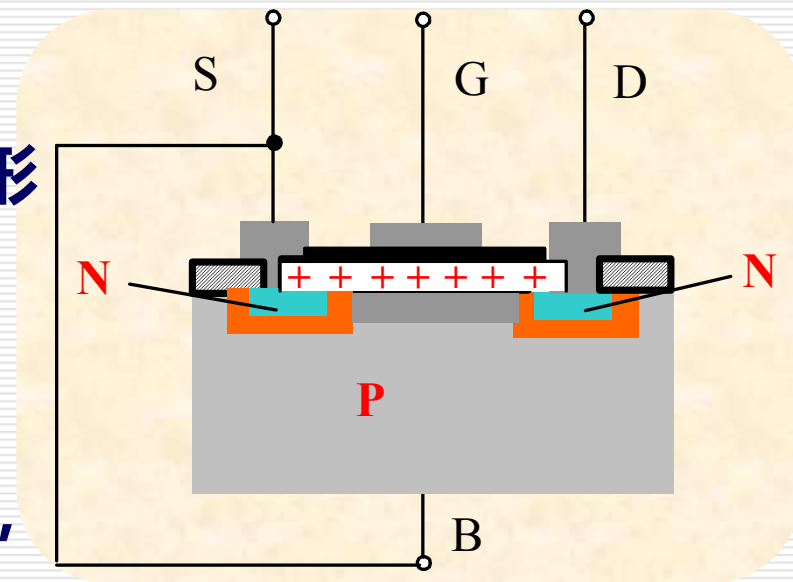


(a) 标准符号

(b) 简化符号

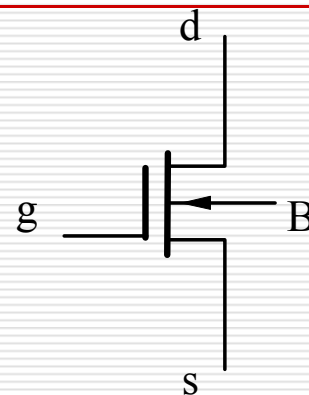
#### (2) N沟道耗尽型MOS管

- 绝缘层掺入正离子，使衬底表面形成N沟道。
- $V_{GS}$  电压可以是正值、零或负值。
- $V_{GS}$  达到某一负值 $V_P$ ，沟道被夹断， $i_D = 0$ 。

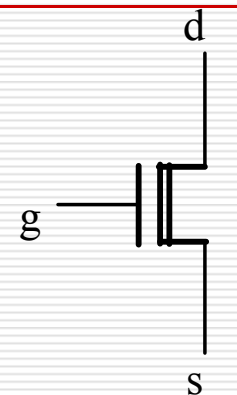


## (2) N沟道耗尽型MOS管

N沟道耗尽型MOS管符号如图。



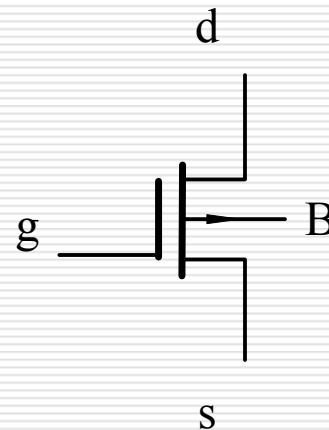
(a) 标准符号



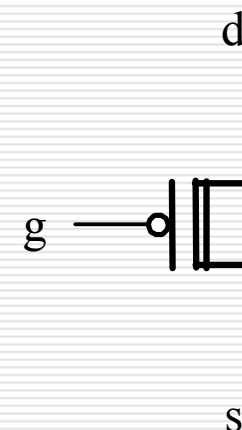
(b) 简化符号

## (3) P沟道耗尽型MOS管

结构与N沟道耗尽型MOS管相反。  
符号如图所示。

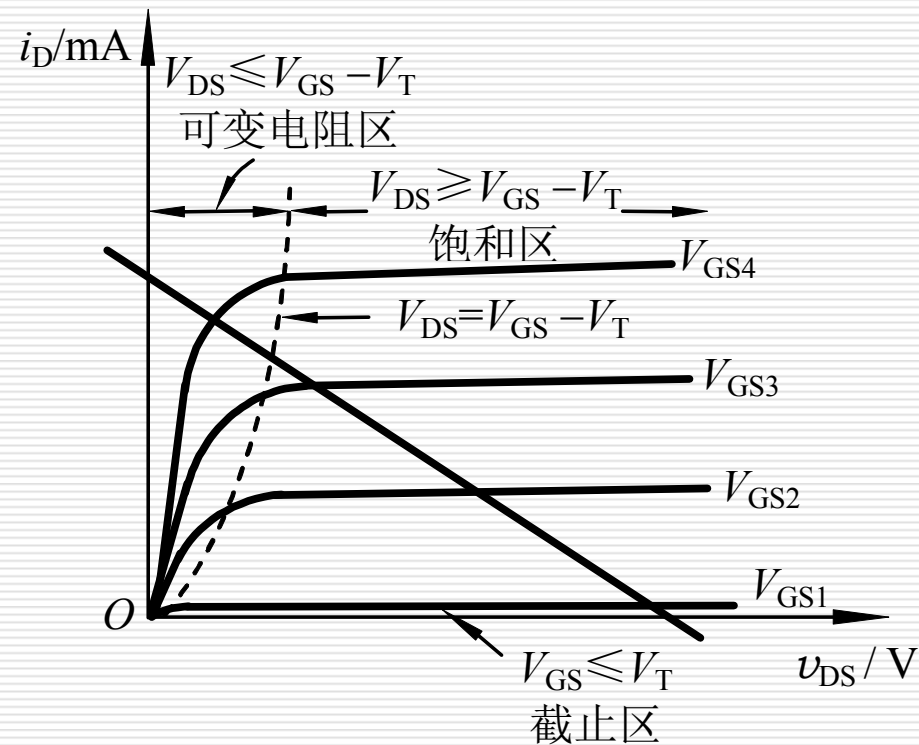
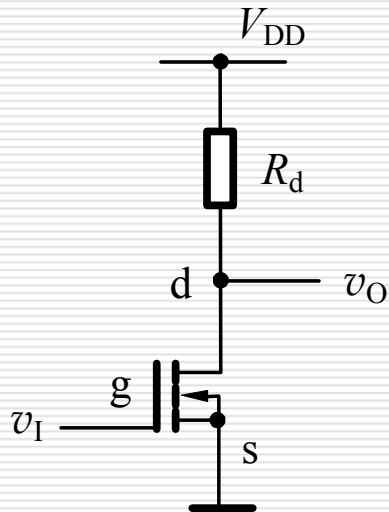


(a) 标准符号



(b) 简化符号

## 4. MOS管开关电路



当  $v_I < V_T$  : MOS管截止，输出高电平

当  $v_I > V_T$  : MOS管工作在可变电阻区，输出低电平

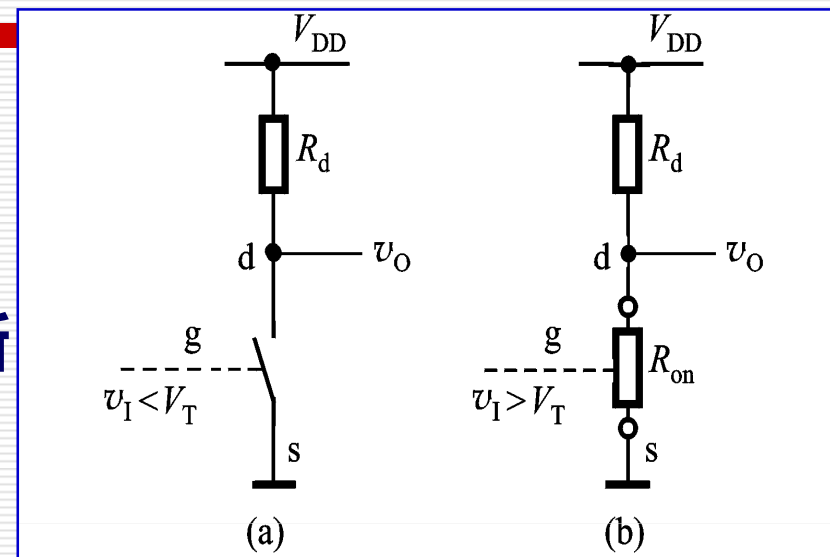
□当 $v_i$ 为低电平时：

MOS管截止，相当于开关“断开”，输出为高电平。

□当 $v_i$ 为高电平时：

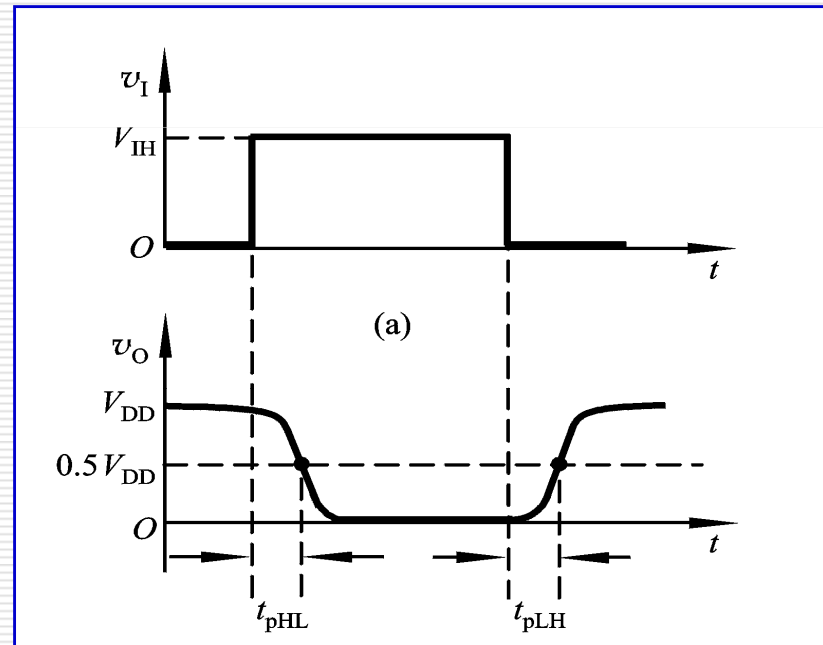
MOS管工作在可变电阻区，相当于开关“闭合”，输出为低电平。

□MOS管相当于一个由 $v_{GS}$ 控制的无触点开关。



## 5. MOS管开关电路的动态特性

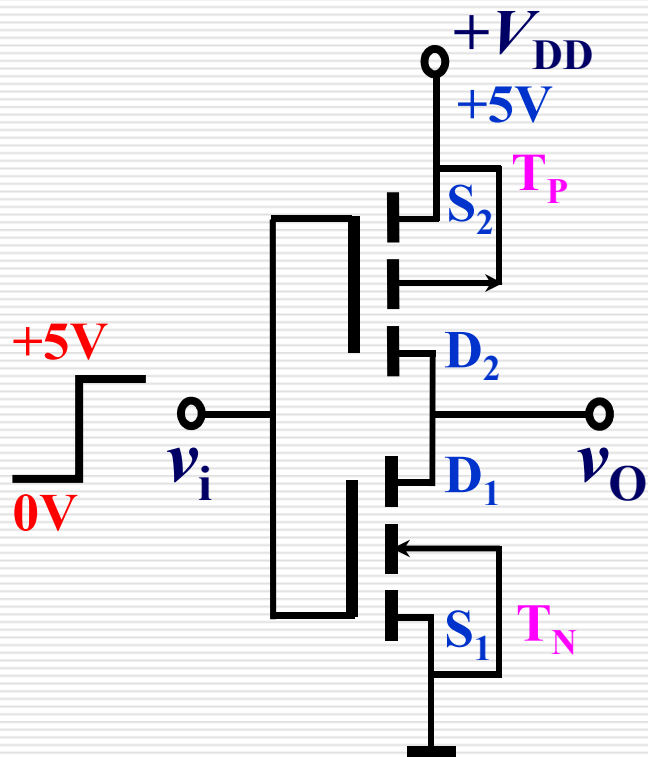
- 由于MOS管栅极、漏极与衬底间电容，栅极与漏极之间的电容存在，电路在状态转换之间有电容充、放电过程。
- 输出波形上升沿、下降沿变得缓慢。



## 3.2.2 CMOS 反相器

### 1.工作原理

$$V_{TN} = 2\text{ V} \quad V_{TP} = -2\text{ V} \quad V_{DD} > (V_{TN} + |V_{TP}|)$$



$v_i$	$v_{GSN}$	$v_{GSP}$	$T_N$	$T_P$	$v_O$
0 V	0V	-5V	截止	导通	5V
5 V	5V	0V	导通	截止	0 V

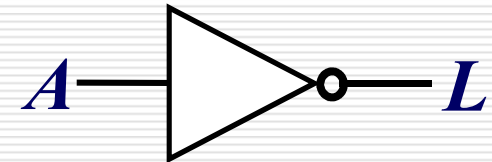
逻辑真值表

$v_i (A)$	$v_O (L)$
0	1
1	0

逻辑图

逻辑表达式

$$L = \overline{A}$$



## CMOS反相器的重要特点：

---

第一， $v_I$ 是高电平还是低电平， $T_N$ 和 $T_P$ 中总是一个导通而另一个截止。CMOS反相器的静态功耗几乎为零。

第二，MOS管导通电阻低，截止电阻高。使充、放电时间常数小，开关速度更快，具有更强的带负载能力。

第三，MOS管的， $I_G \approx 0$ ，输入电阻高。理论上可以带任意同类门，但负载门输入杂散电容会影响开关速度。

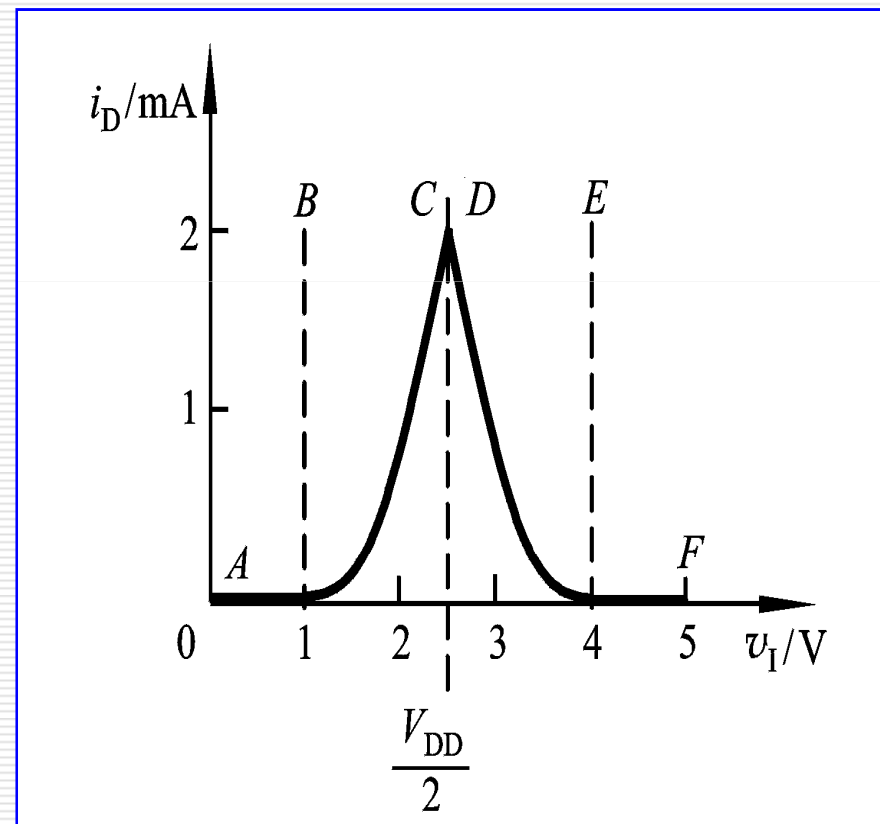
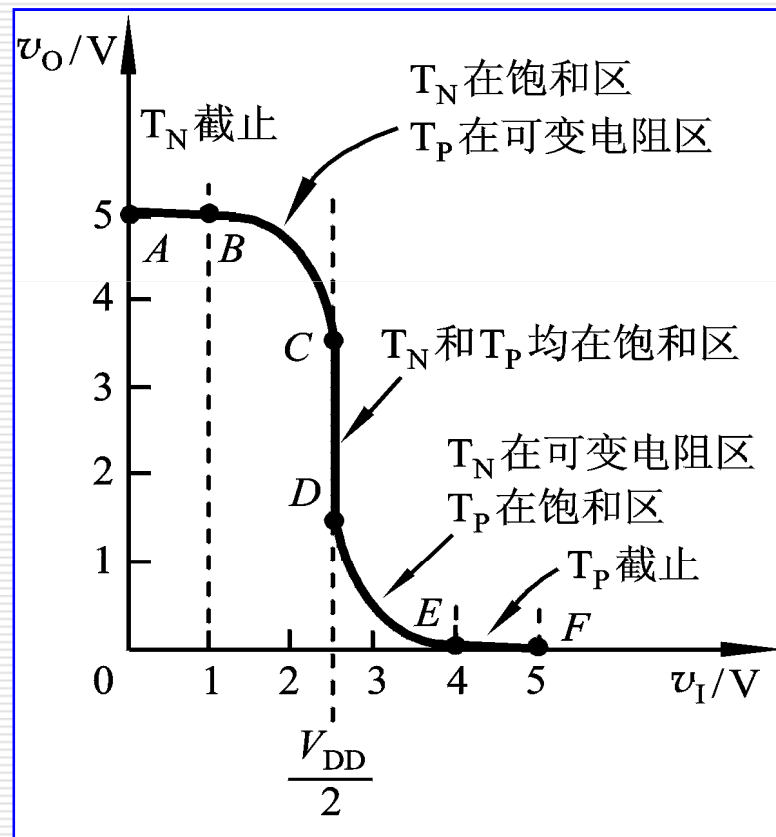
---



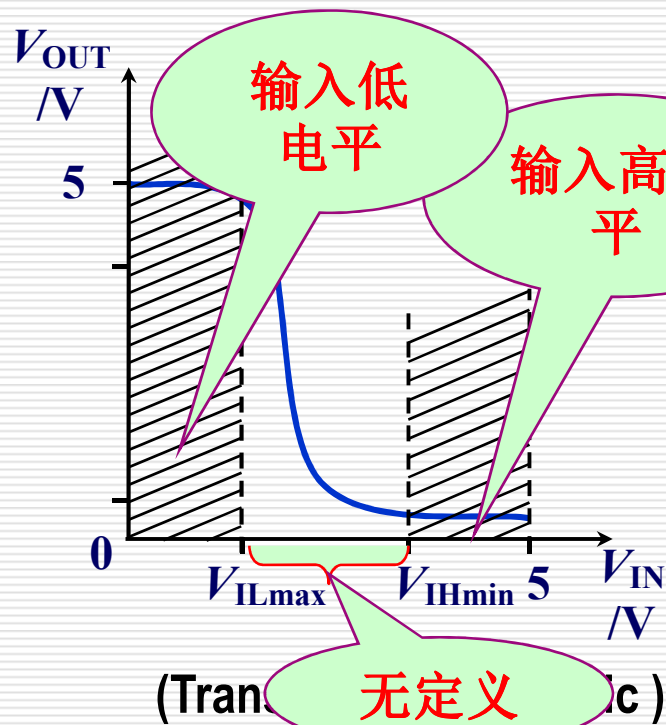
## 2. 电压传输特性和电流传输特性

电压传输特性  $v_O = f(v_I)$

电流传输特性  $i_D = f(v_I)$



### 3. 输入逻辑电平和输出逻辑电平

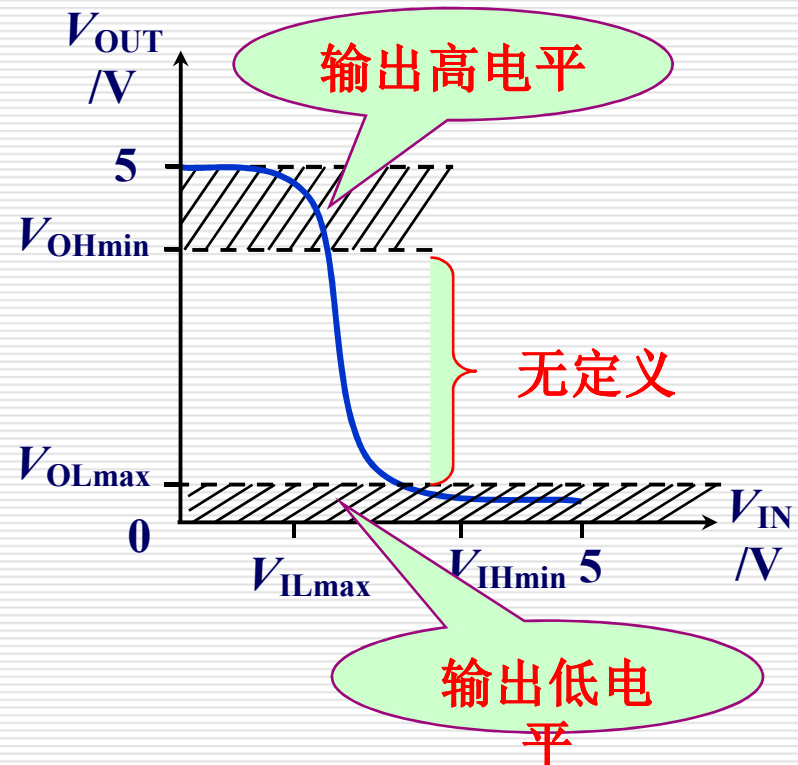


输入低电平的上限值

$$V_{IL(max)}$$

输出高电平的下限值

$$V_{OH(min)}$$



输入高电平的下限值

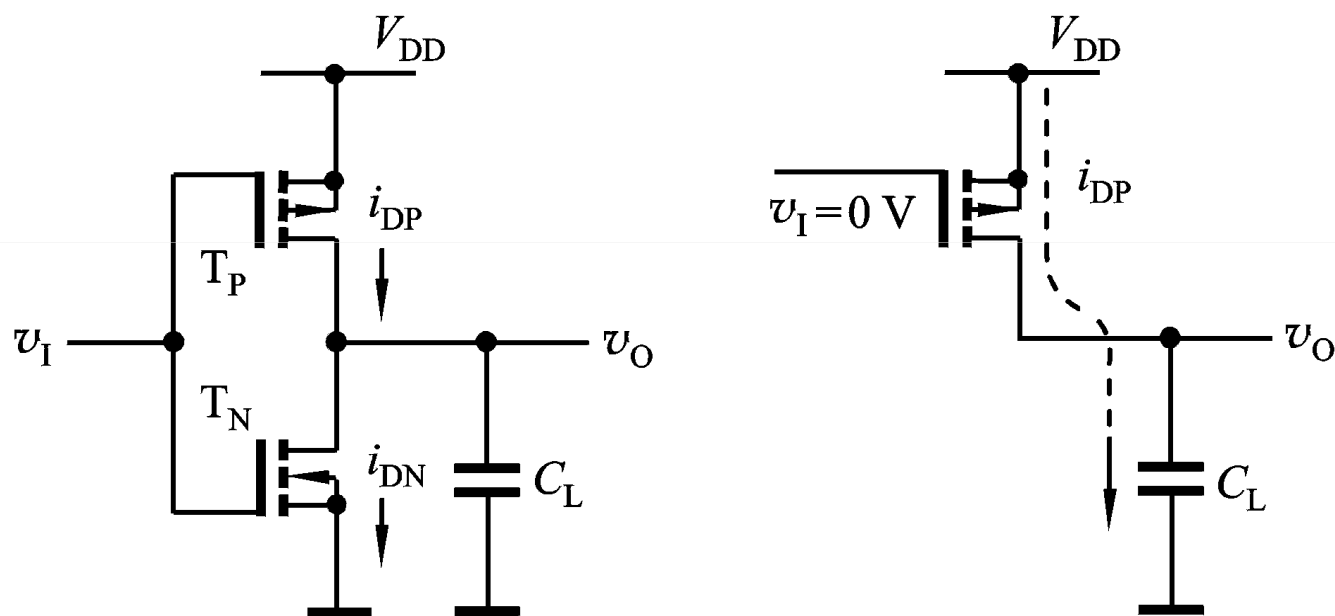
$$V_{IH(min)}$$

输出低电平的上限值

$$V_{OL(max)}$$

## 4.CMOS反相器的工作速度

### 带电容负载



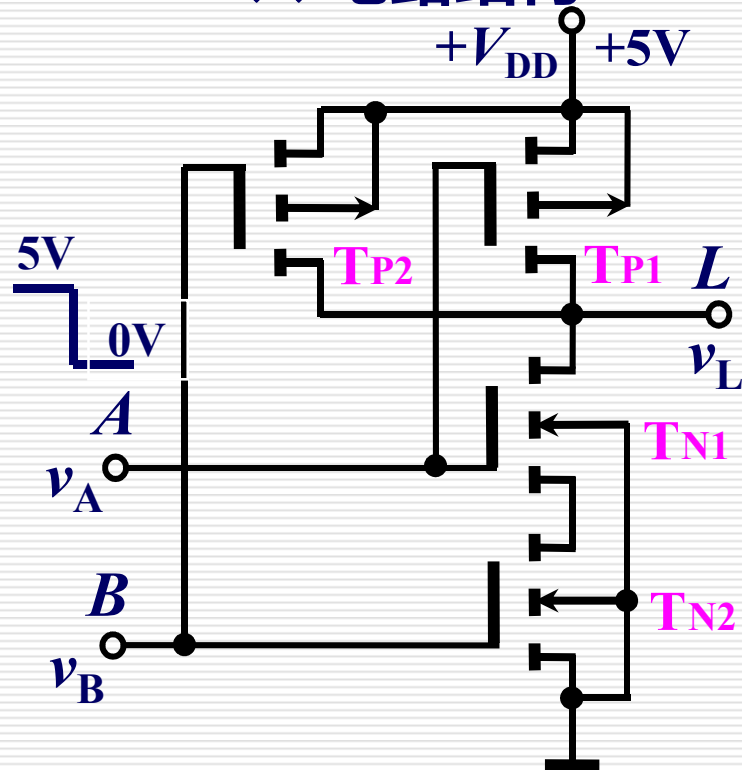
在由于电路具有互补对称的性质，它的开通时间与关闭时间是相等的。平均延迟时间小于10 ns。

### 3.2.3 其他基本CMOS 逻辑门电路

#### 1. CMOS 与非门

$$V_{TN} = 2\text{ V} \quad V_{TP} = -2\text{ V}$$

(a) 电路结构

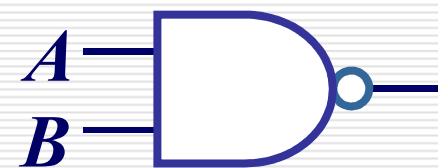


(b) 工作原理

$A$	$B$	$T_{N1}$	$T_{P1}$	$T_{N2}$	$T_{P2}$	$L$
0	0	截止	导通	截止	导通	1
0	1	截止	导通	导通	截止	1
1	0	导通	截止	截止	导通	1
1	1	导通	截止	导通	截止	0

与非门

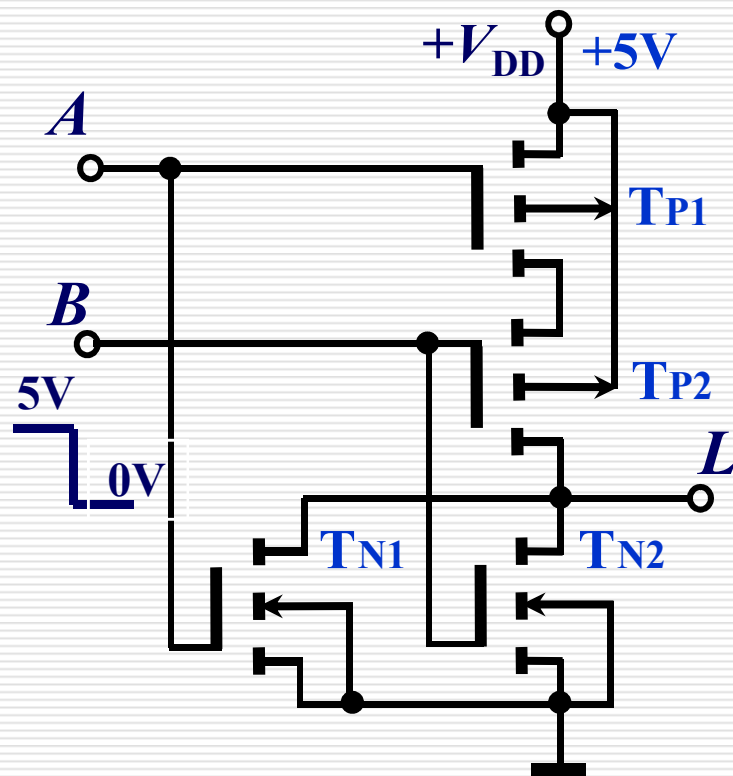
$$L = \overline{AB}$$



N输入的与非门的电路?

输入端增加有什么问题?

## 2. CMOS 或非门



$$V_{TN} = 2 \text{ V} \quad V_{TP} = -2 \text{ V}$$

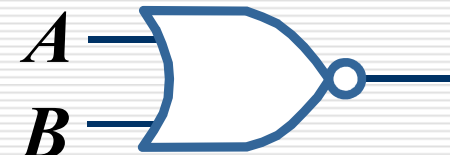
$A$	$B$	$T_{N1}$	$T_{P1}$	$T_{N2}$	$T_{P2}$	$L$
0	0	截止	导通	截止	导通	1
0	1	截止	导通	导通	截止	0
1	0	导通	截止	截止	导通	0
1	1	导通	截止	导通	截止	0

N输入的或非门的电路的结构?

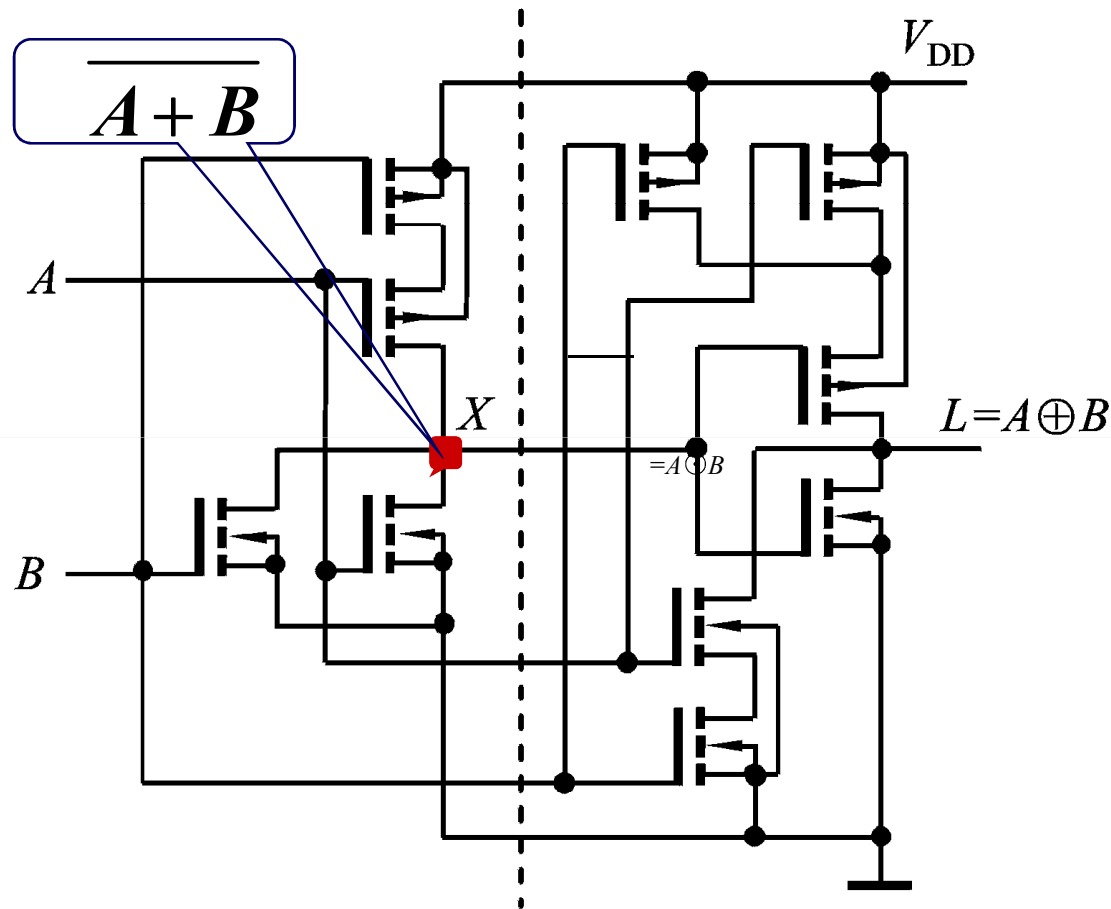
输入端增加有什么问题?

或非门

$$L = \overline{A + B}$$



## 例：分析CMOS电路，说明其逻辑功能。

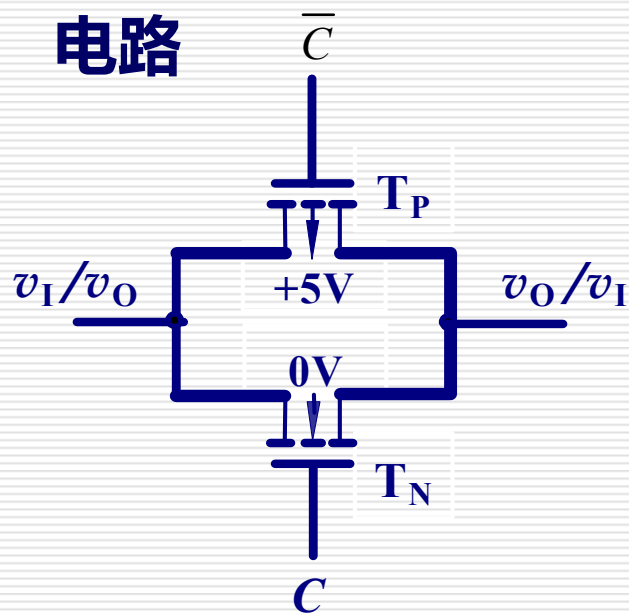


$$\begin{aligned} L &= \overline{A \cdot B + X} \\ &= \overline{A \cdot B + \overline{A + B}} \\ &= \overline{A \cdot B + \overline{A} \cdot \overline{B}} \\ &= A \oplus B \end{aligned}$$

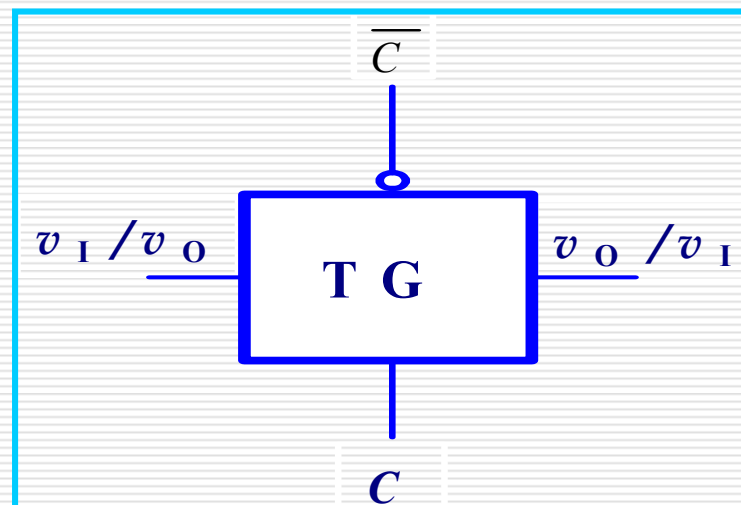
异或门电路

### 3.2.4 CMOS传输门(双向模拟开关)

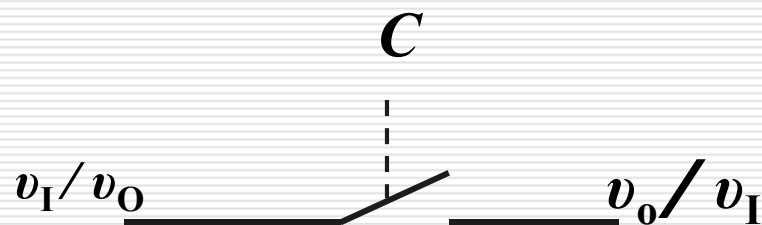
#### 1. 传输门的结构及工作原理



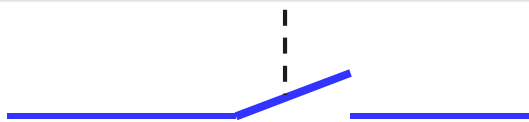
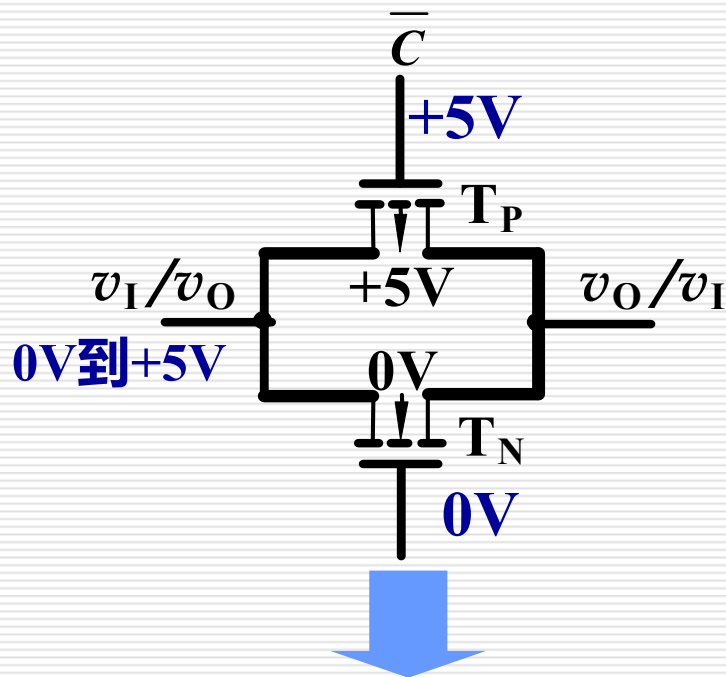
逻辑符号



等效电路



## 1、传输门的结构及工作原理



设 $T_P: |V_{TP}|=2V$ ,  $T_N: V_{TN}=2V$ ,  
 $v_I$ 的变化范围为0到+5V。

$c=0=0V$ ,  $\bar{c}=1=+5V$

1) 当 $c=0$ ,  $\bar{c}=1$ 时

$$v_{GSN} = 0V - (0V \text{到} +5V) = (0 \text{到} -5)V$$

$$v_{GSN} < V_{TN}, T_N \text{截止}$$

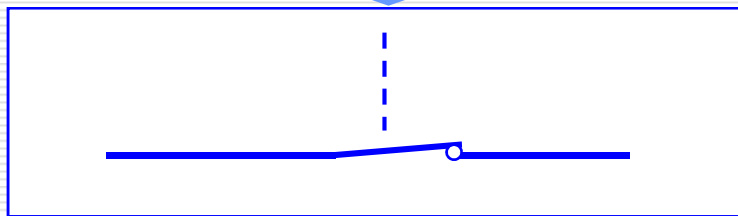
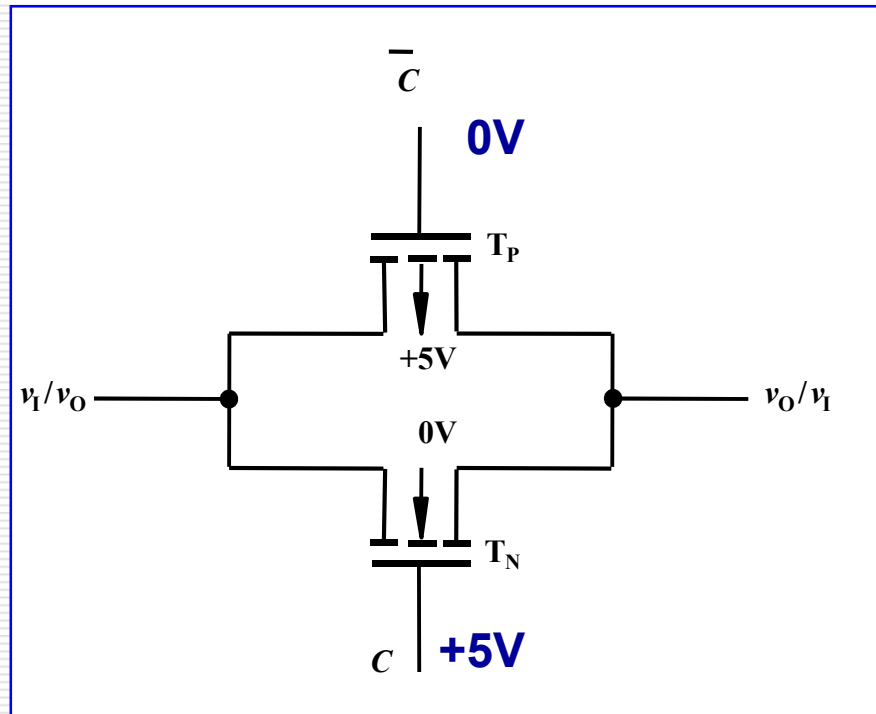
$$v_{GSP} = +5V - (0V \text{到} +5V) = (5 \text{到} 0)V$$

$$v_{GSP} > 0, T_P \text{截止}$$

开关断开，不能转送信号



## 2) 当 $c=1$ , $\bar{c}=0$ 时



**a**、 $v_I = 0V \sim 3V$

$$v_{GSN} = 5V - (0V \sim +3V) = (5 \sim 2)V$$

$v_{GSN} > V_{TN}$ ,  $T_N$  导通

**b**、 $v_I = 2V \sim 5V$

$$v_{GSP} = 0V - (2V \sim +5V) = -2V \sim -5V$$

$|v_{GSP}| > |V_{TP}|$ ,  $T_P$  导通

**c**、 $v_I = 2V \sim 3V$

$T_N$  导通,  $T_P$  导通

$$v_O = v_I$$

## 2. 传输门的应用

### (1) 传输门组成的异或门

**B=0**

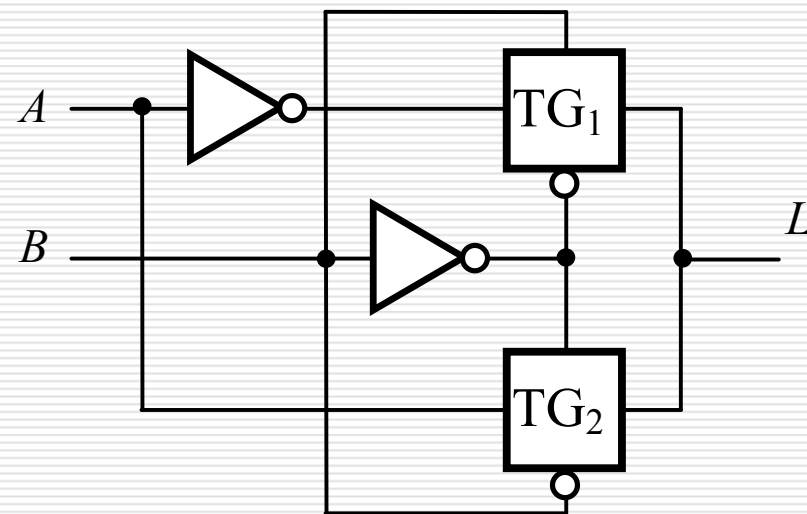
**TG1断开, TG2导通**

**L=A**

**B=1**

**TG1导通, TG2断开**

**$L=\bar{A}$**



## 2. 传输门的应用

### (2) 传输门组成的数据选择器

**C=0**

**TG1导通, TG2断开**

**L=X**

**C=1**

**TG2导通, TG1断开**

**L=Y**

