

第5章 可编程逻辑

[5.1 PLD的基本概念](#)

[5.2 现场可编程门阵列FPGA](#)

[5.3 在系统可编程ISP](#)

[5.4 可编程逻辑的原理图方式设计](#)

[5.5 可编程逻辑的VHDL文本方式设计](#)



5.1 PLD的基本概念

5.1.1 可编程阵列

5.1.2 PLD的类型

导入

数字电路经历了分离元件 --> 中小规模标准化集成电路 --> 可编程逻辑器件 (PLD) 这样的发展历程。PLD是用户根据需要自行设计芯片中特定逻辑电路的器件。

积的和 和的积



5.1.1 可编程阵列

所有的PLD是用可编程阵列组成的。

可编程阵列本质上是行、列导线组成的导电网格。

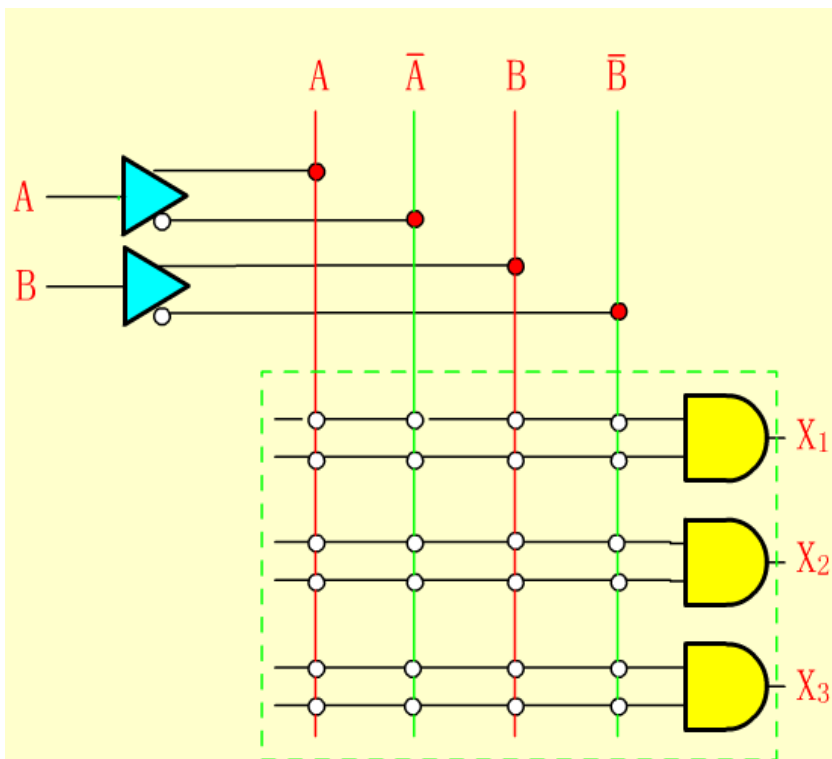
在网格的交叉点上，通过熔断金属丝或E²COMS管等连接技术来实现逻辑 **1** 或逻辑 **0** 。

可编程阵列分类：与阵列和或阵列

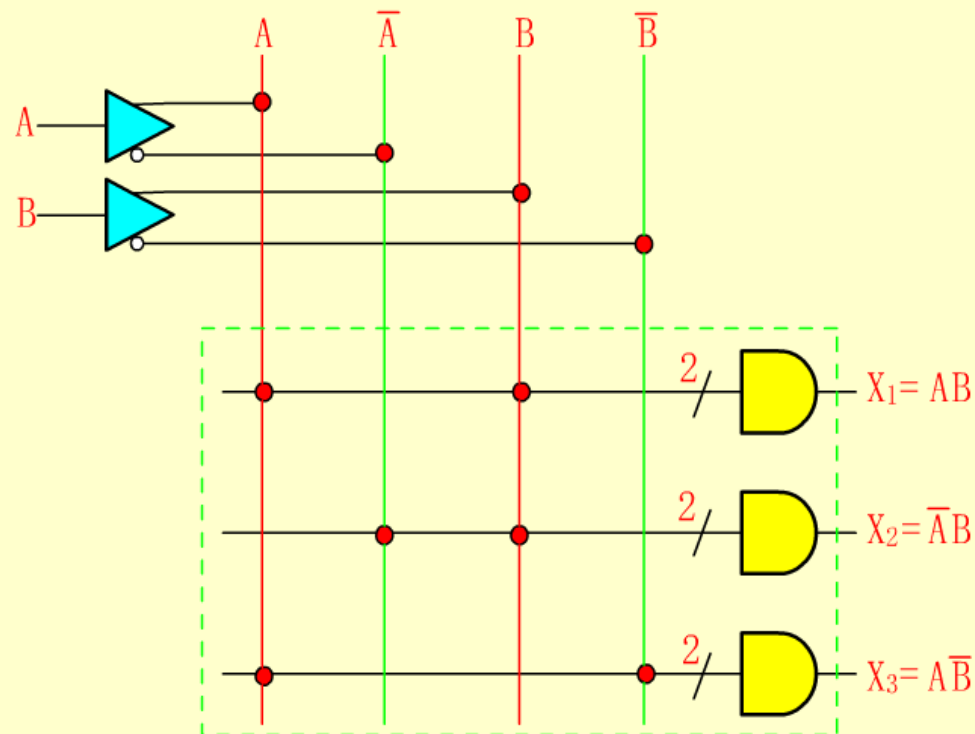
PLD: Programmable Logic Device

5.1.1 可编程阵列

1. 与阵列



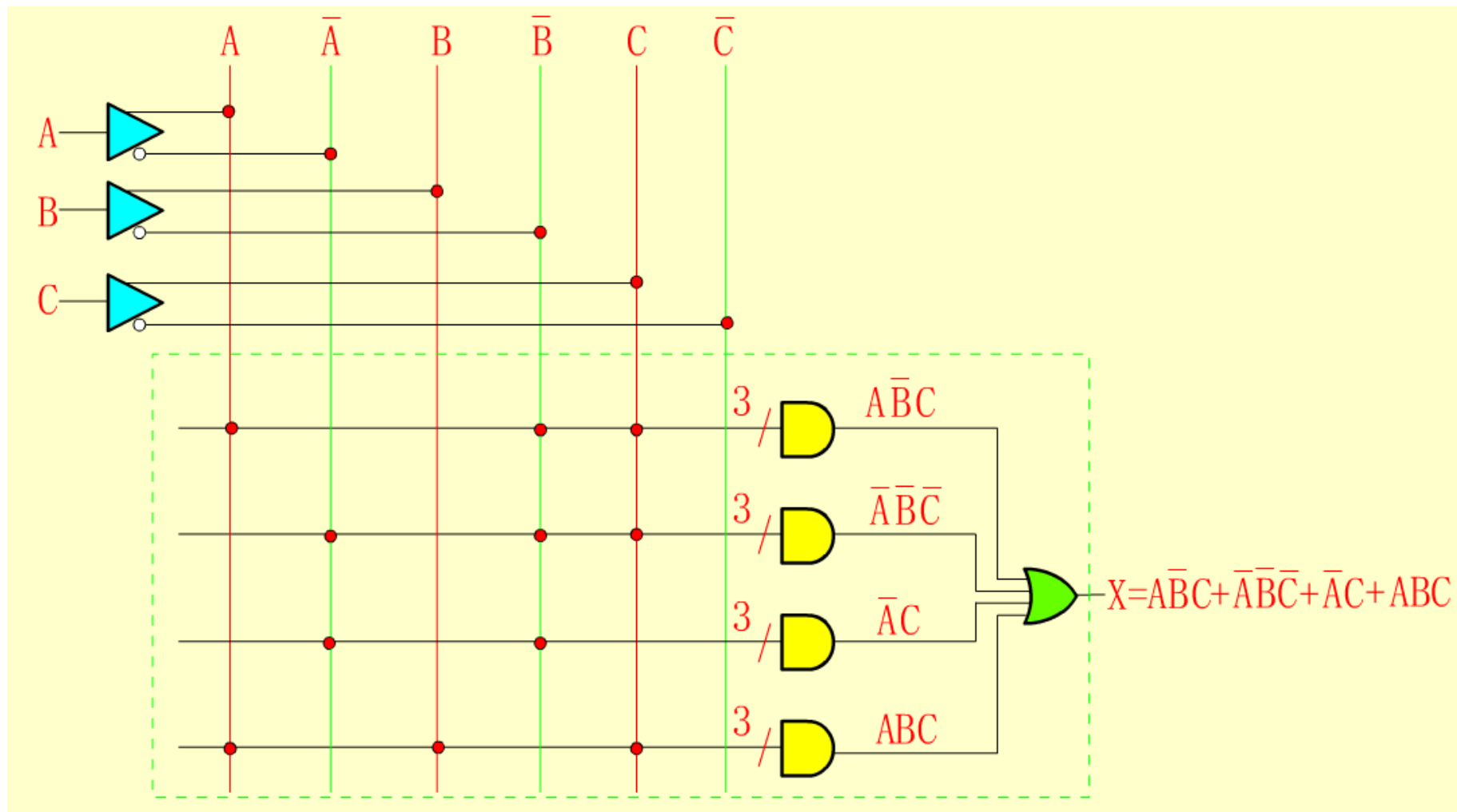
(a) 未编程



(b) 已编程

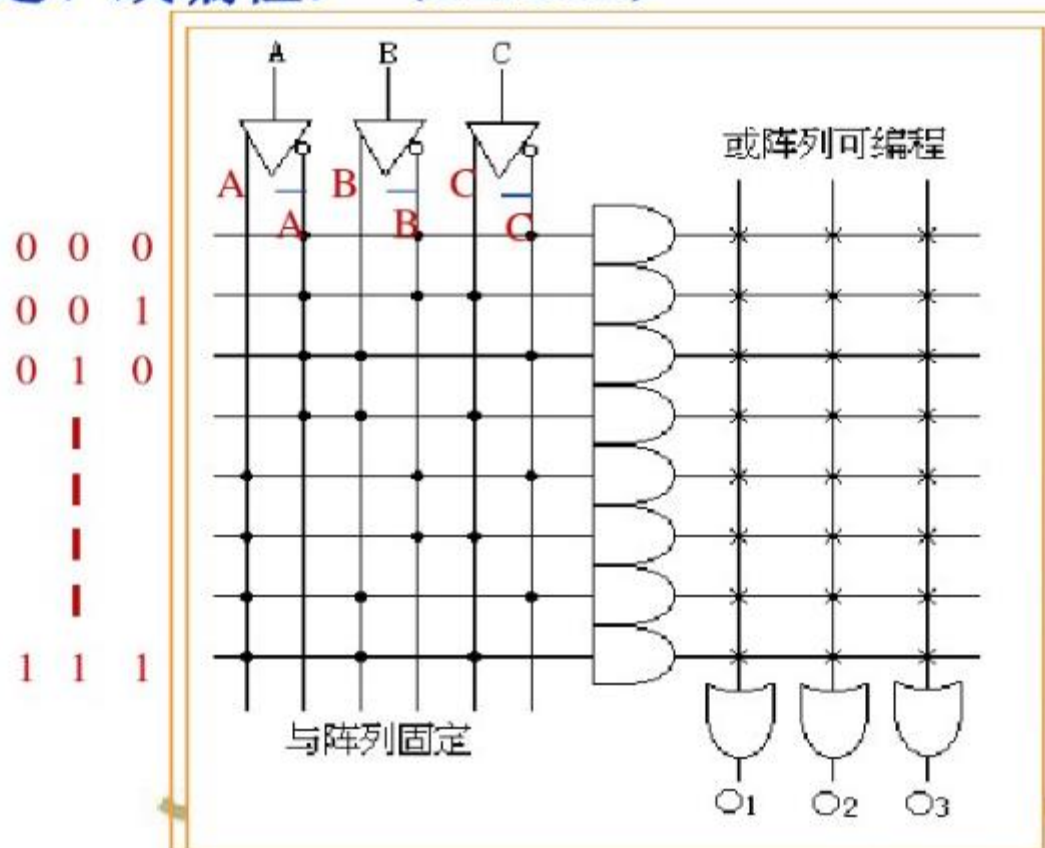
【例1】 用与阵列实现三变量A、B、C的与或表达式

$$X = A\bar{B}C + \bar{A}\bar{B}\bar{C} + \bar{A}C + ABC$$

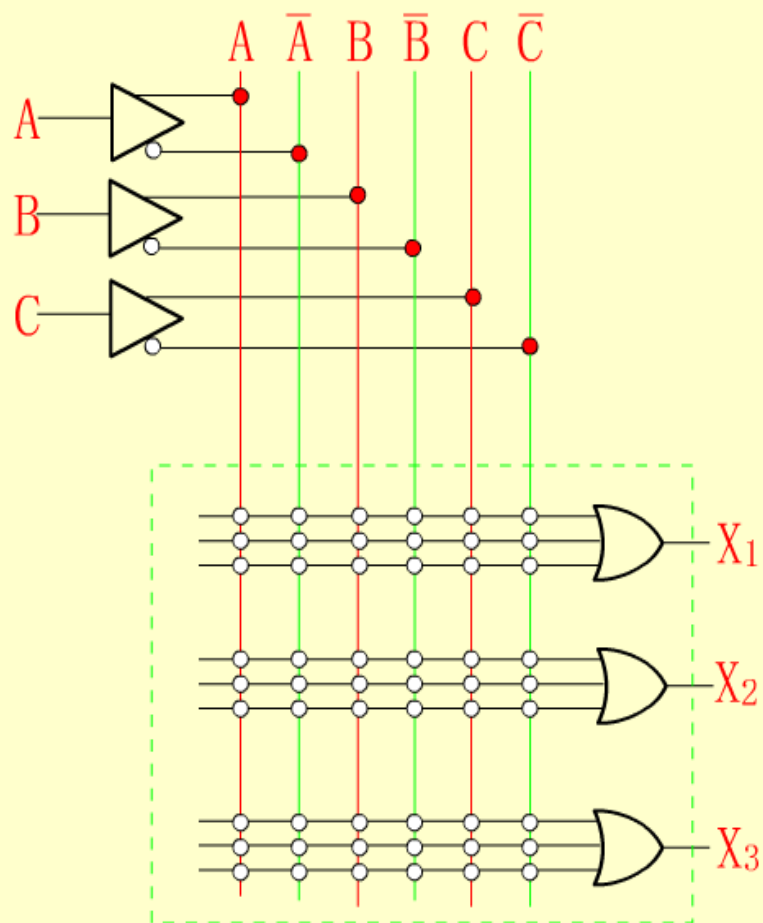


与固定、或编程

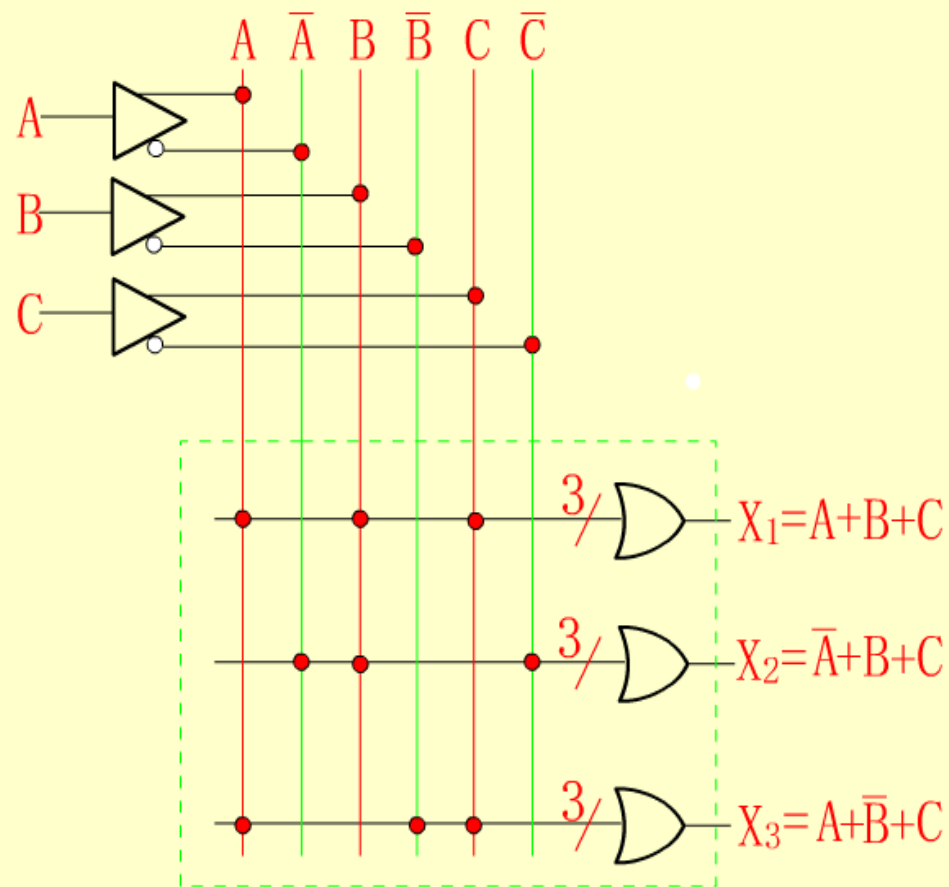
与固定、或编程：（PROM）



2. 或阵列



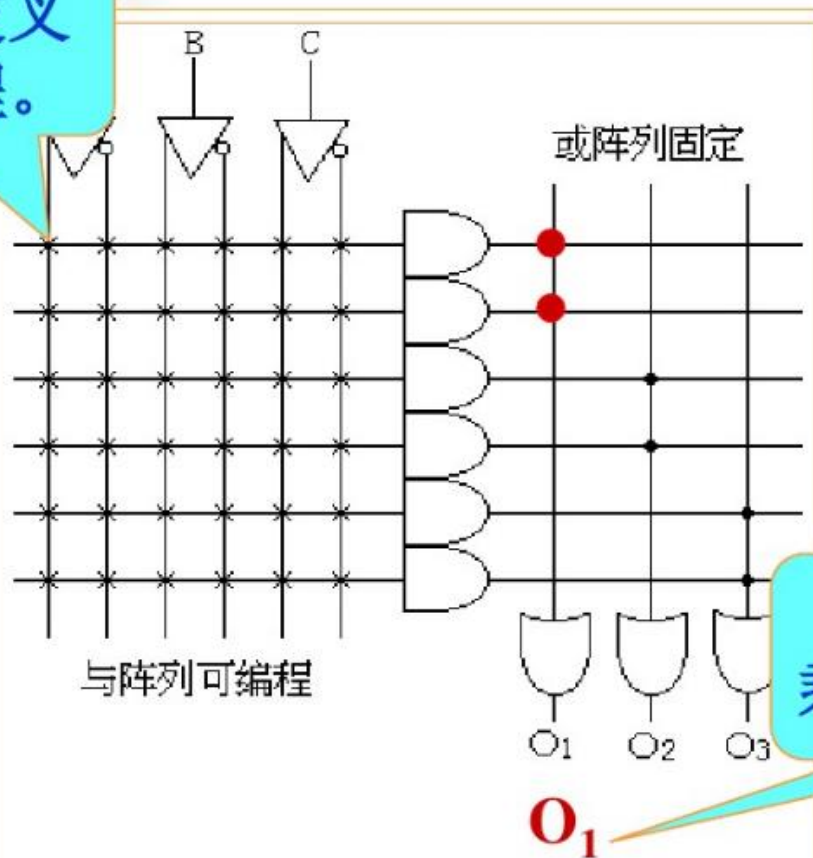
(a) 未编程



(b) 已编程

或固定、与编程

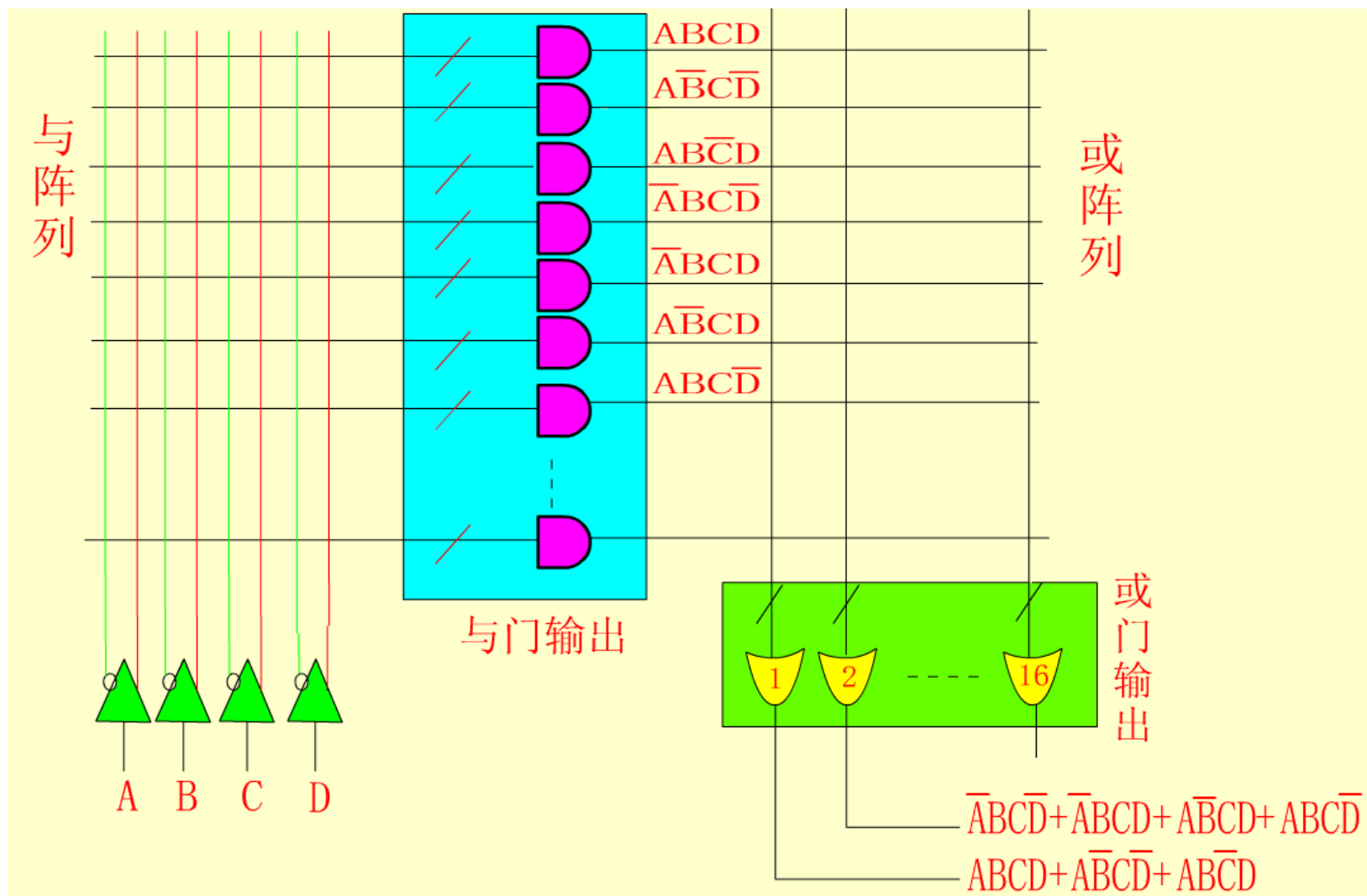
每个交叉点都可编程。



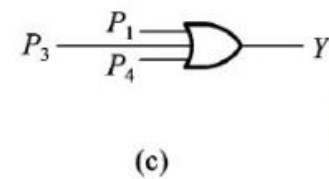
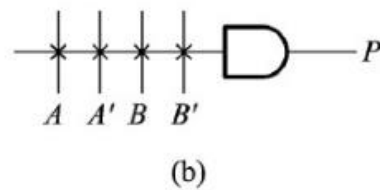
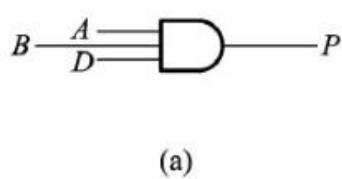
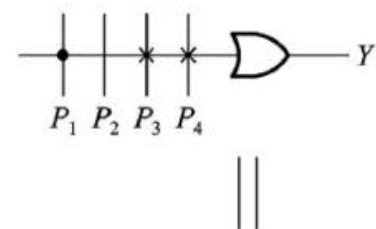
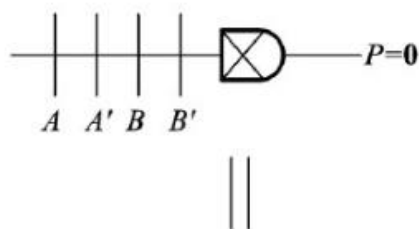
O₁为两个乘积项之和。

O₁

【例3】图5.4是一个与阵列和一个或阵列组成的SOP形式复合阵列，或阵列制定的逻辑表达式由或门1、或门2输出。请在图中画出编程点，并写出与门输出的表达式。



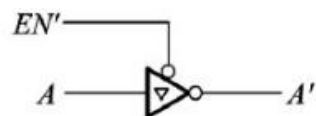
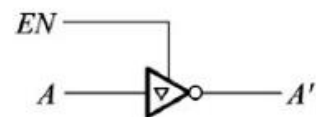
PLD中用的逻辑图符号

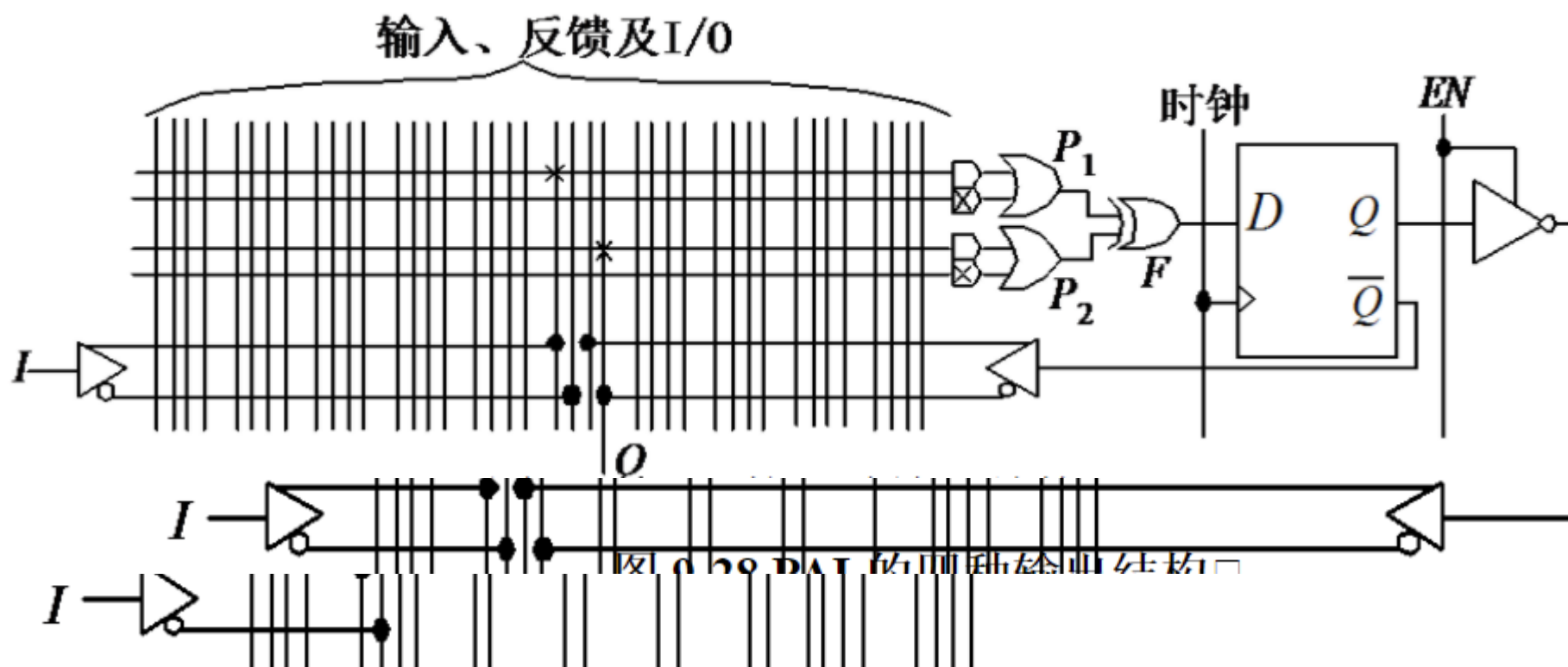
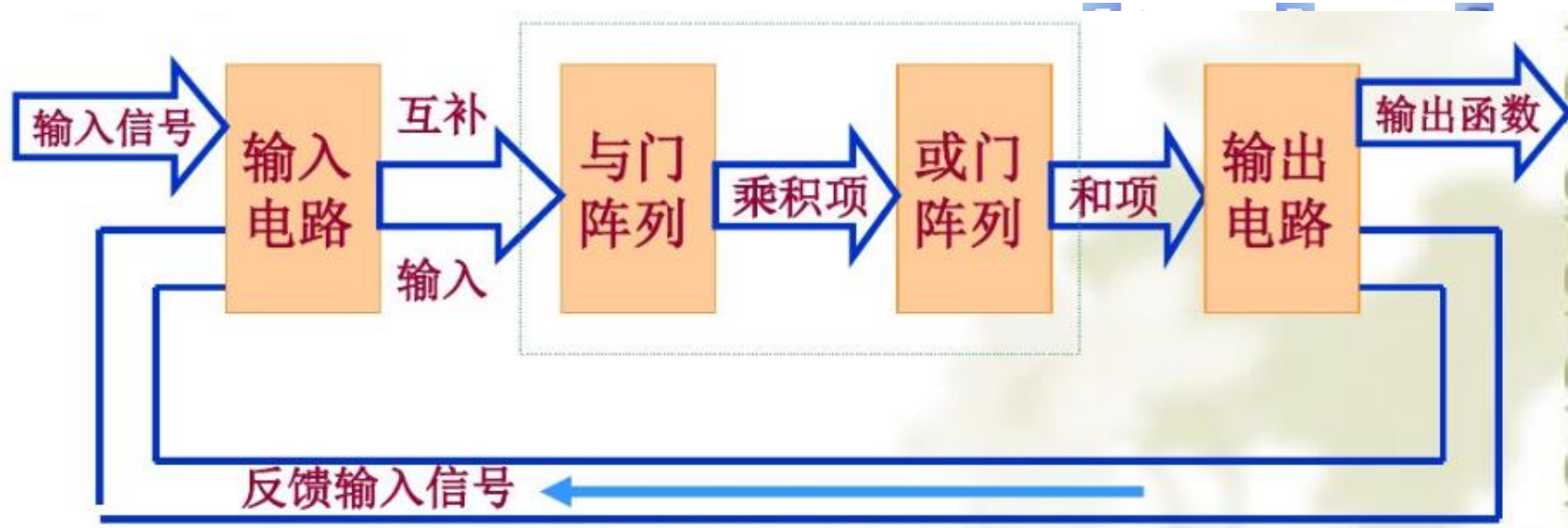


(a)

(b)

(c)





3.可编程连接技术

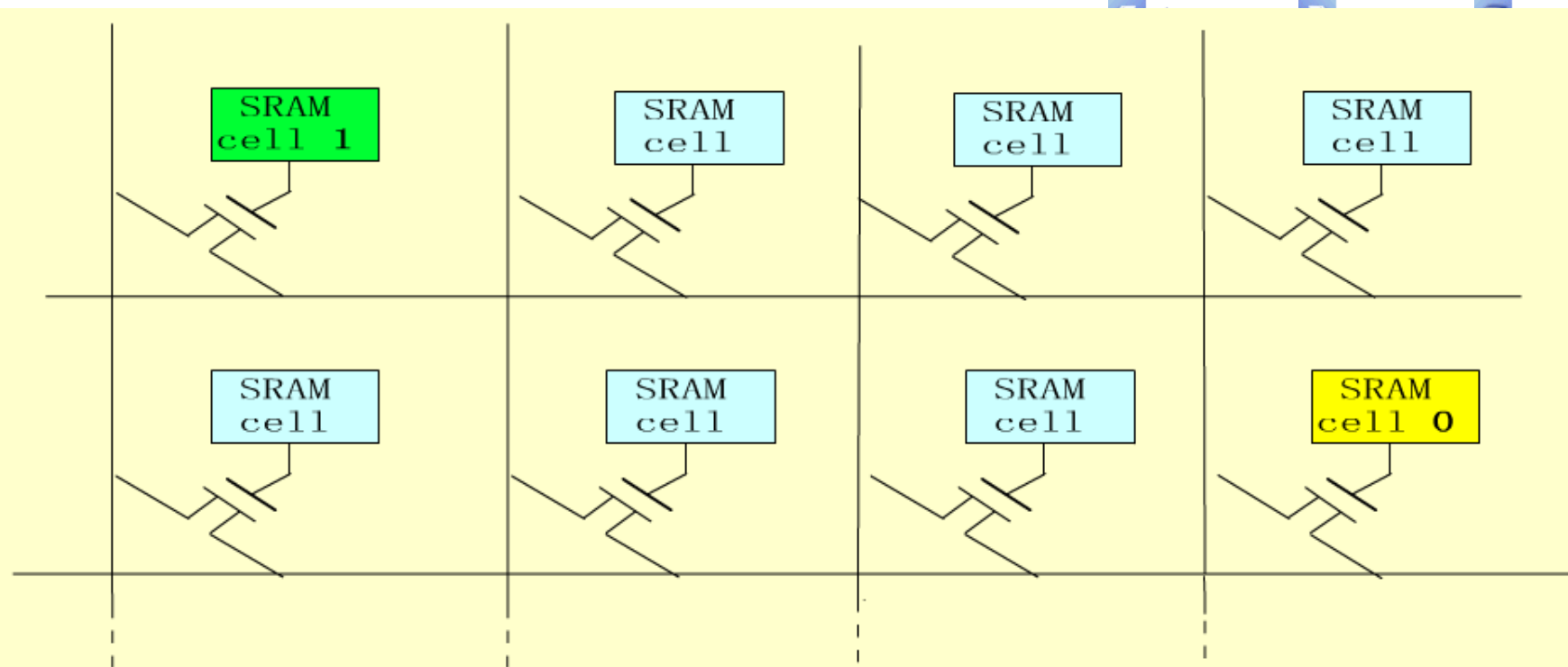
连接技术有以下4种：

熔丝技术, 编程之前, 熔丝通连接（原始），状态称为逻辑 1；对选定熔丝**熔断**后的状态称为逻辑 0。**非易失**。

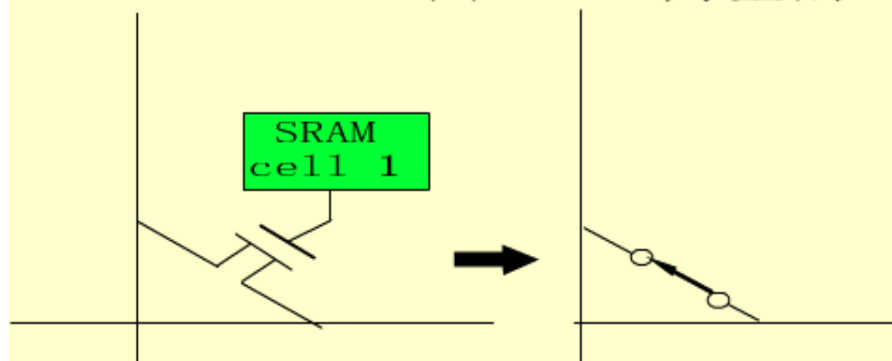
反熔丝技术, 这种连接与熔丝连接正好相反，不是破坏连接，而是建立连接。**非易失**。

E²PROM技术, 加电的方式可以擦除或重写，ISP在系统编程

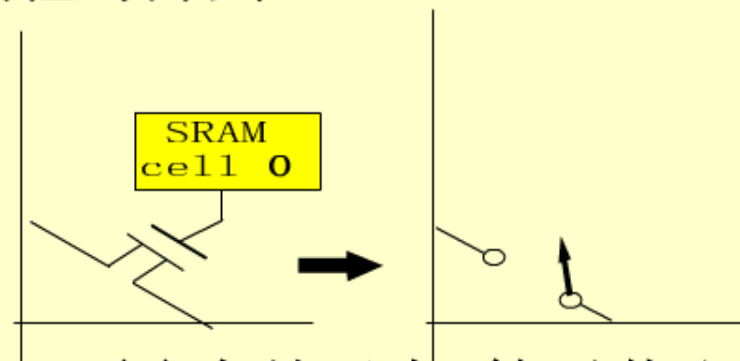
SRAM技术, SRAM存储元通过将触发器实现连接、断开。**易失**，每次加电，PLD编程数据必须**重新写入**。



(a) SRAM为基的可编程与阵列



(b) 存储元存1管子导通



(c) 存储元存0管子截止

SRAM 为基的与阵列

5.1.2 PLD的类型

1.简单可编程逻辑器件SPLD

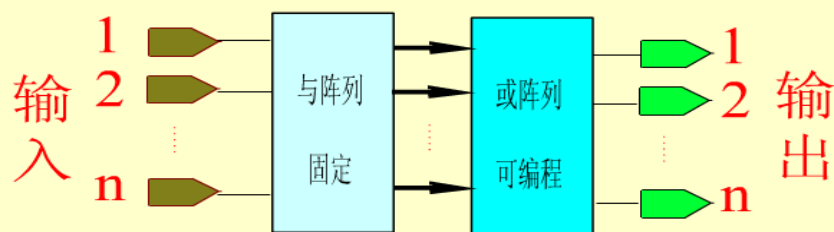
SPLD是指简单可编程逻辑器件。根据与阵列、或阵列是否可以编程，分类：

PROM（一次可编程只读存储器）；

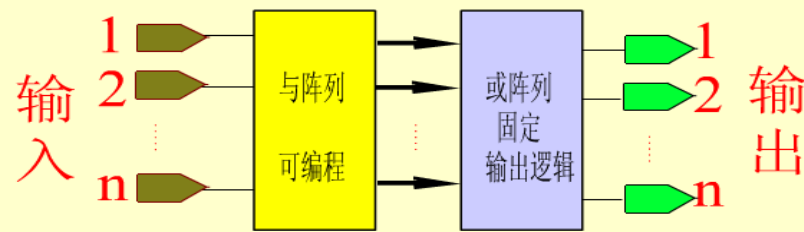
PLA（可编程逻辑阵列）；

PAL（可编程阵列逻辑）；

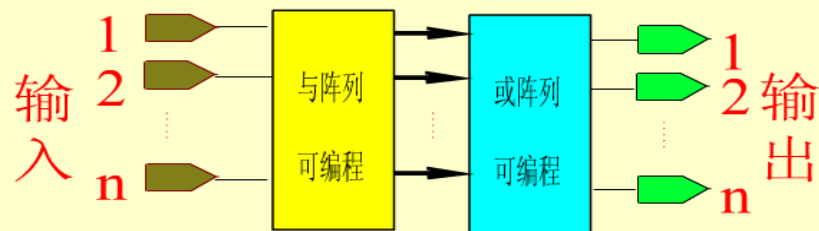
GAL（通用阵列逻辑）。



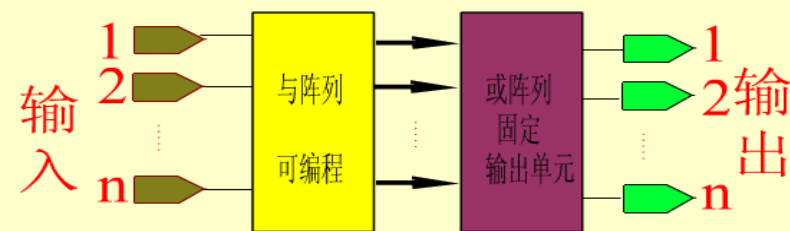
(a)PROM



(b)PAL



(c)PLA

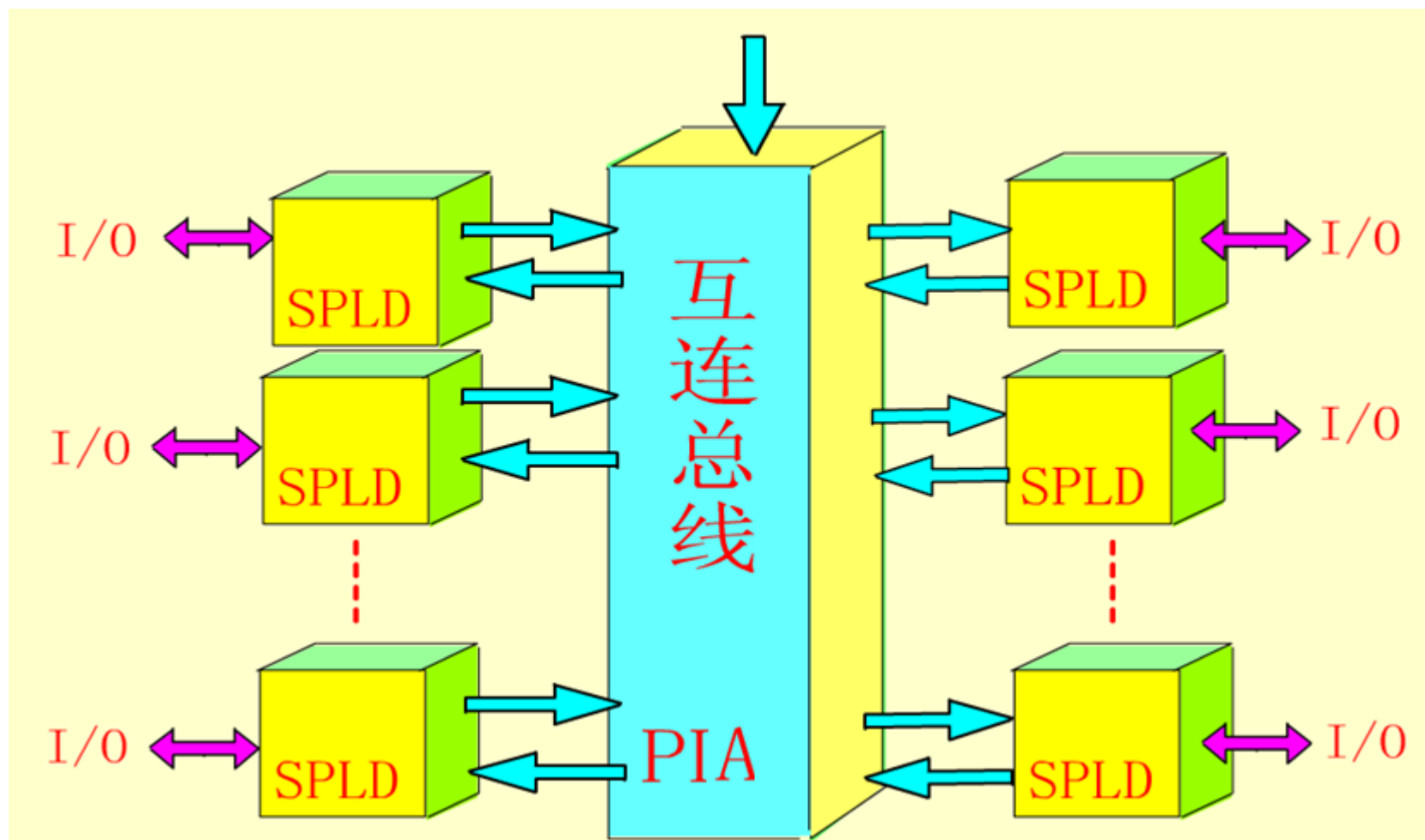


(d)GAL

SPLD内部结构框图

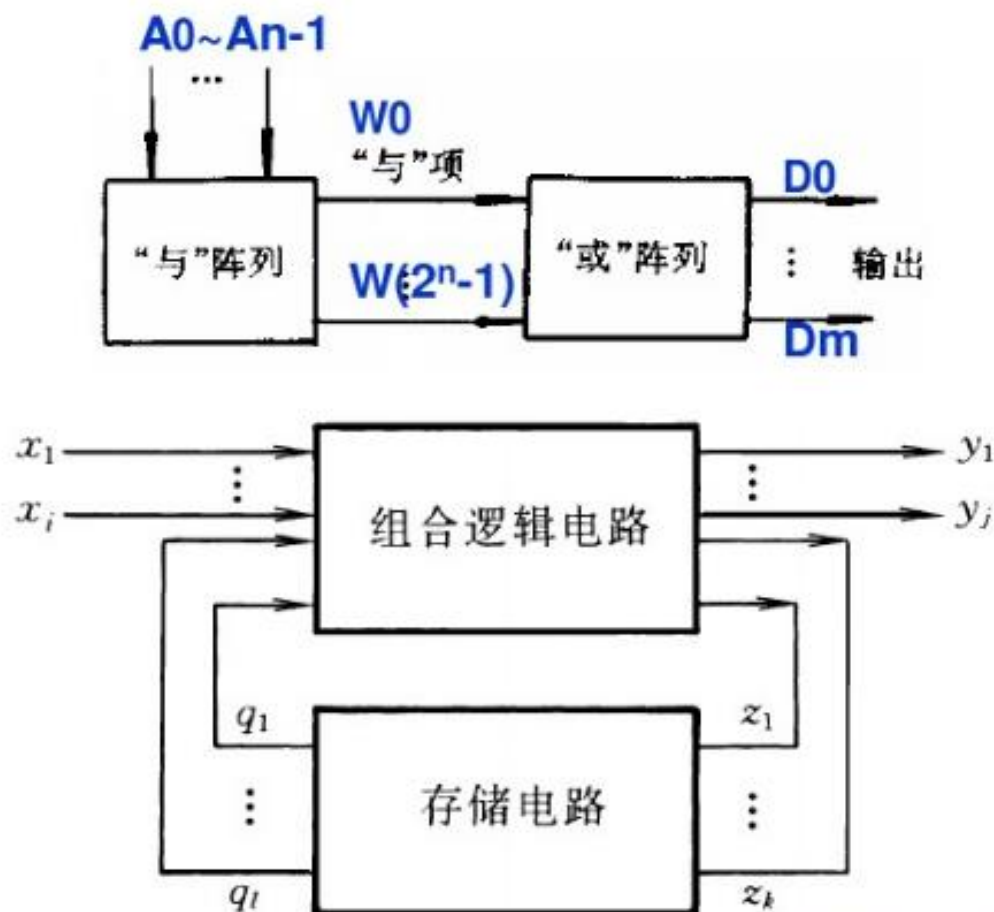
2.复杂可编程逻辑器件CPLD

CPLD本质上是：利用可编程的互连总线连接起来的多路SPLD。

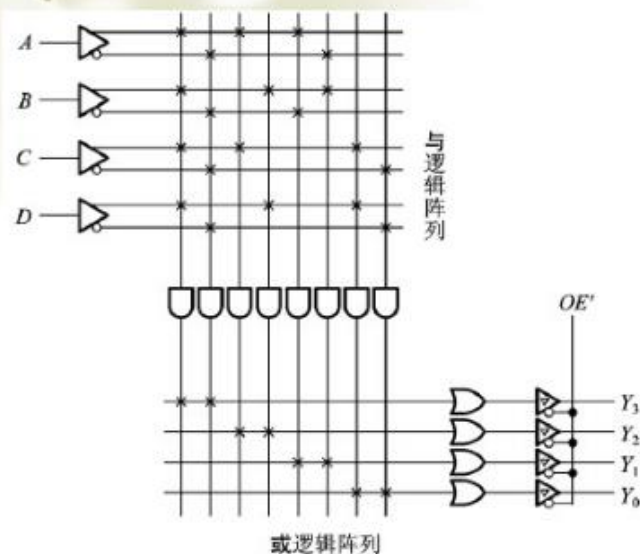


CPLD结构框图

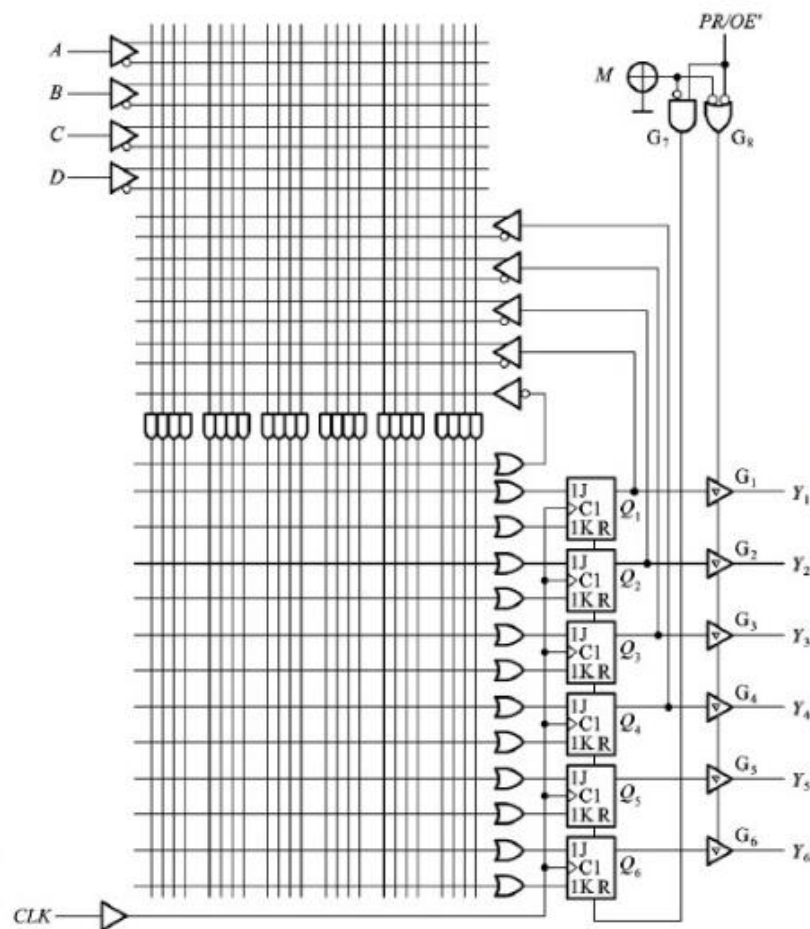
现场可编程逻辑阵列 (FPLA)

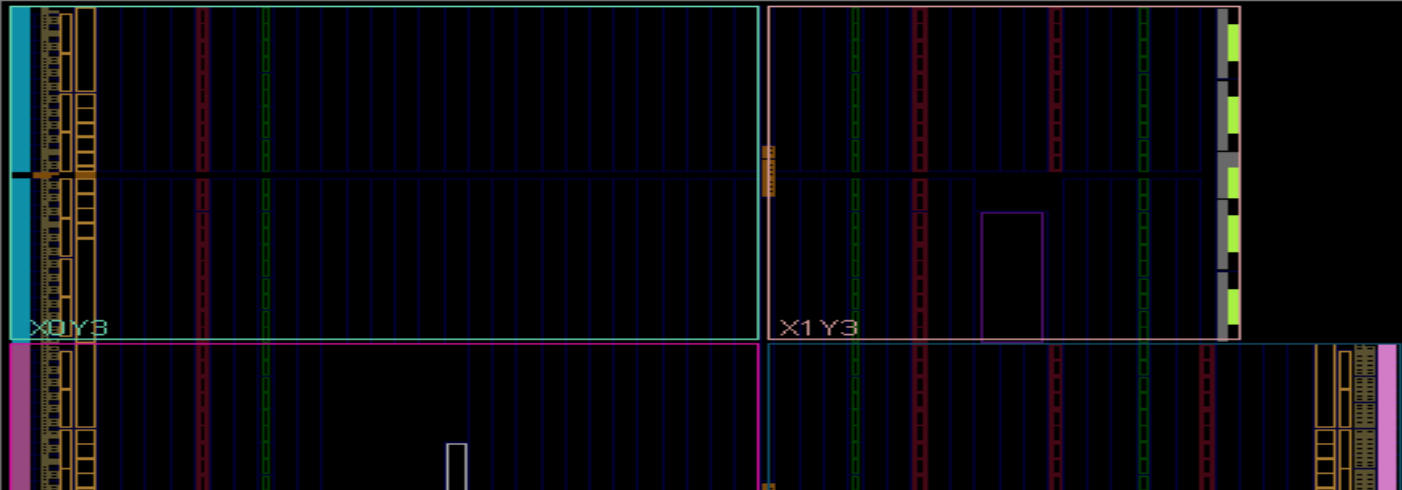


组合电路和时序电路结构的通用形式

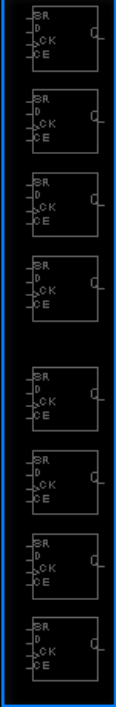
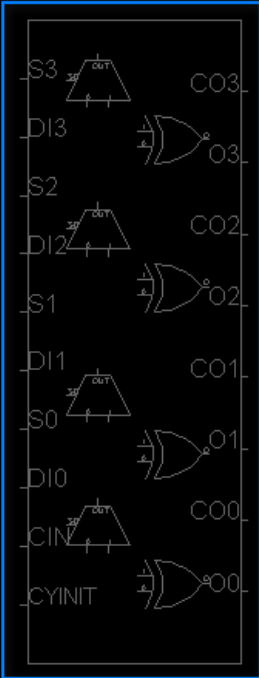
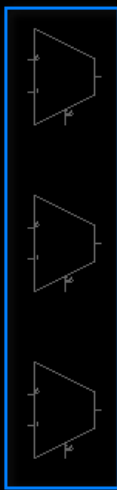
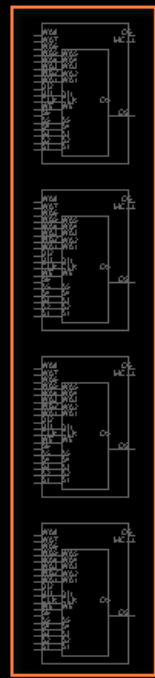


可编程的“与”阵列
+ 可编程的“或”阵列





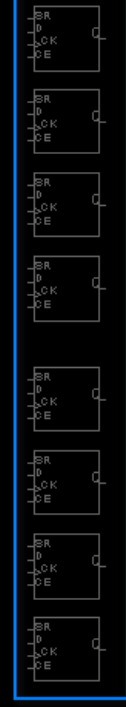
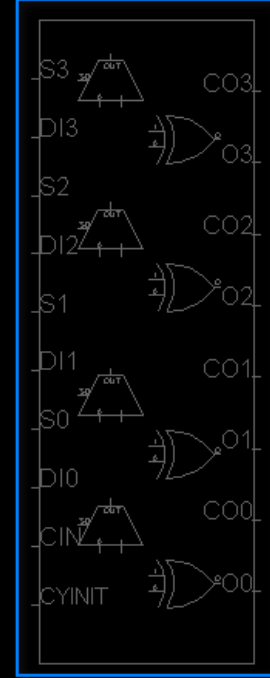
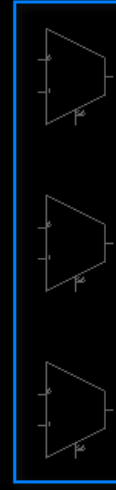
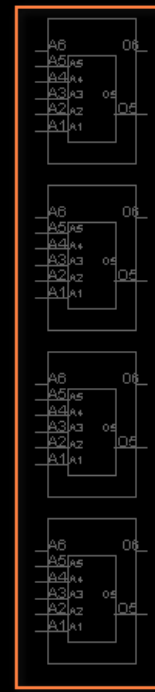
CLBLM_L_X20Y172



LUT
SLICEM 72 (SLICEM)

CARRY4

FF



LUT
SLICEL (SLICEL)

CARRY4

FF

CLB

CLBLM_L_X20Y171

5.2 现场可编程门阵列FPGA

5.2.1 FPGA的基本结构

5.2.2 可组态逻辑块CLB

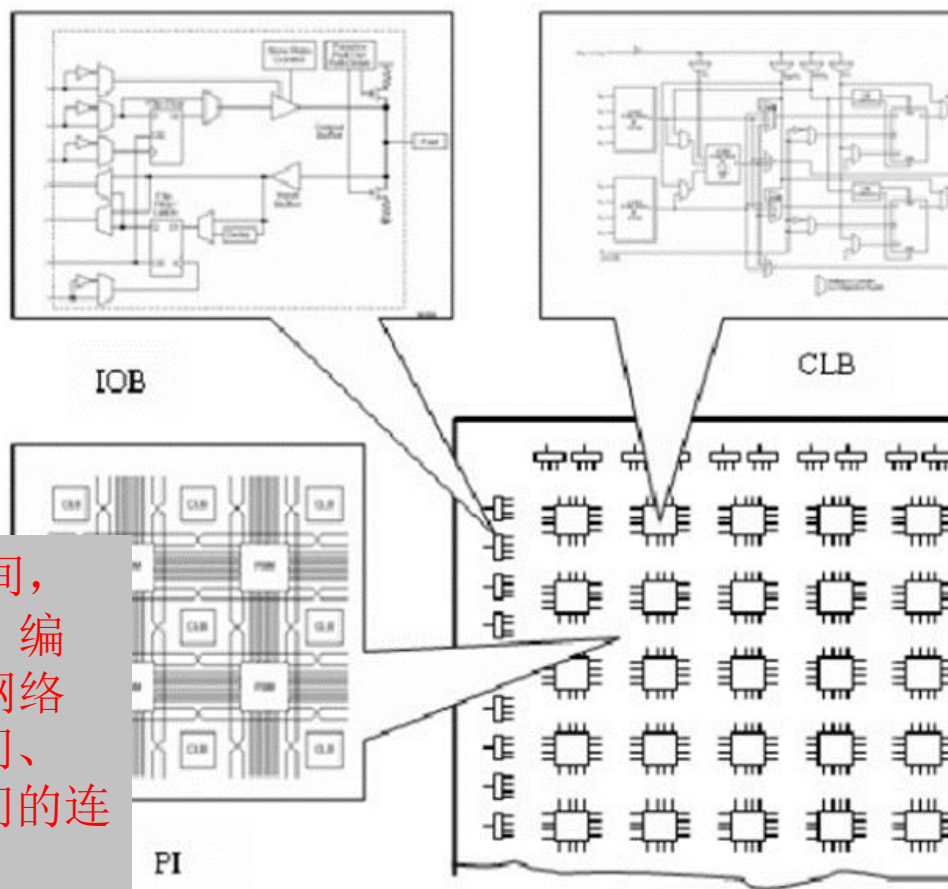
5.2.3 SRAM为基础的FPGA

5.2.1 FPGA的基本结构

FPGA是现场可编程门阵列的英文缩写，比CPLD还要复杂，逻辑块数量大，体系结构也不同，有细粒度、粗粒度之分

三个基本元素（逻辑块CLB、可编程互连总线、I/O输入输出块）

I/OB:位于芯片内部四周，主要由逻辑门、触发器和控制单元组成。在内部逻辑阵列与外部芯片封装引脚之间提供一个可编程接口。

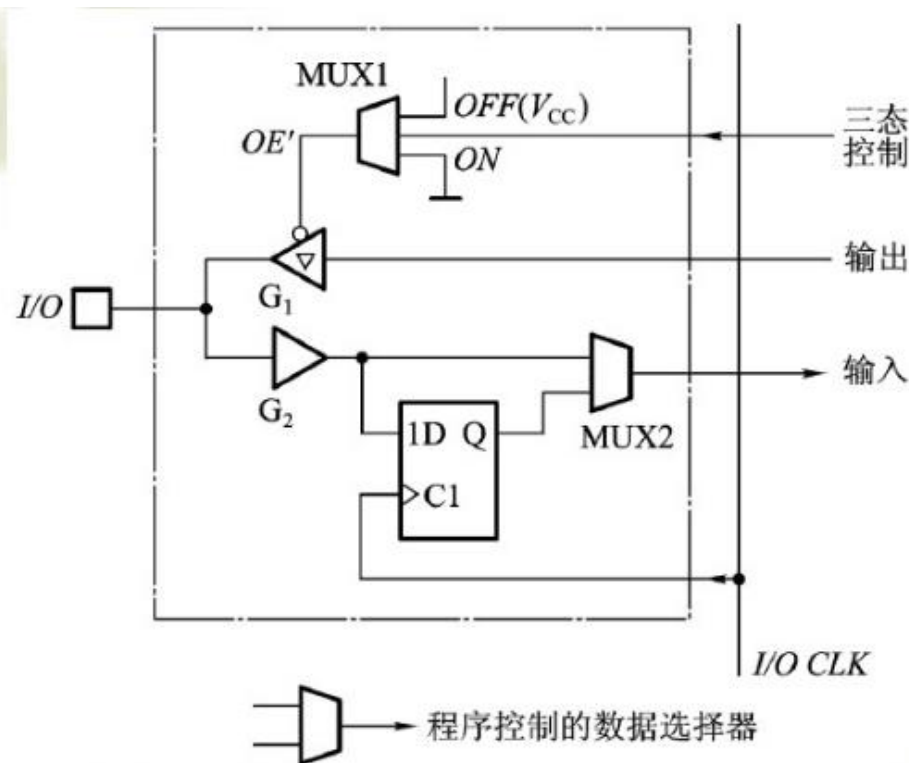


CLB:是FPGA的核心阵列，用于构造用户指定的逻辑功能，不同生产厂商的FPGA器件其不同之处主要在核心阵列。每个CLB主要由查找表LUT(Look Up Table)、触发器、数据选择器和控制单元组成。

PI:位于CLB之间，用于传递信息。编程后形成连线网络，提供CLB之间、CLB与I/OB之间的连线



I/OB

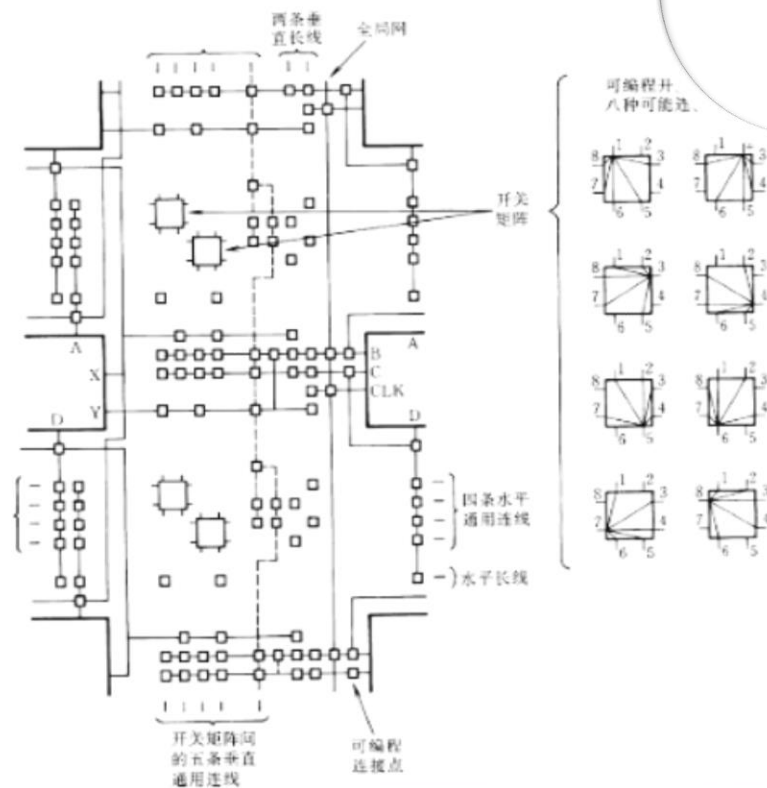
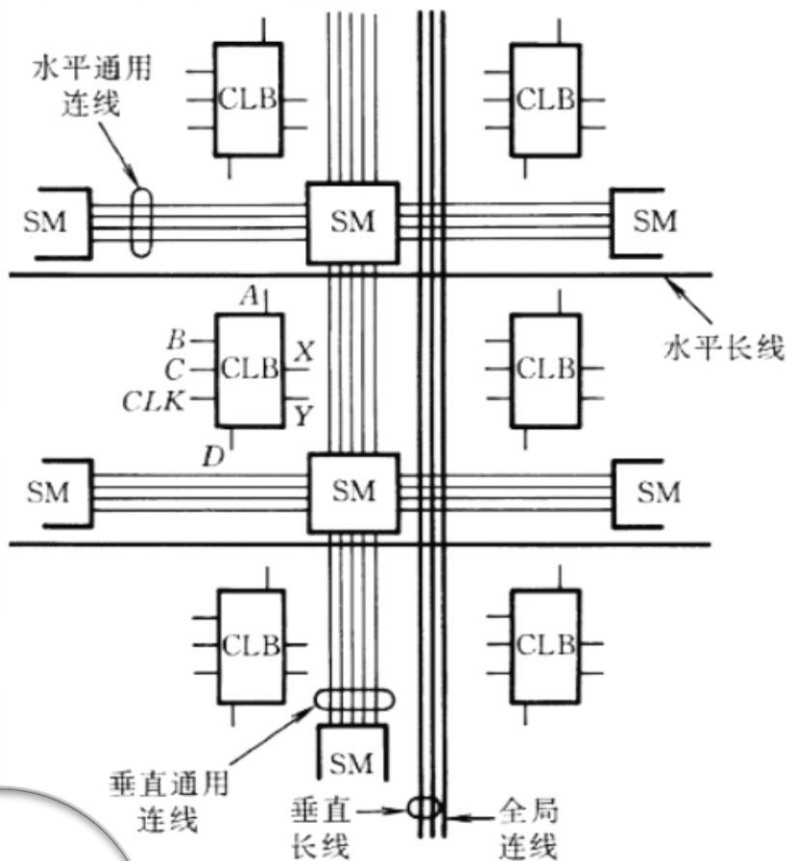


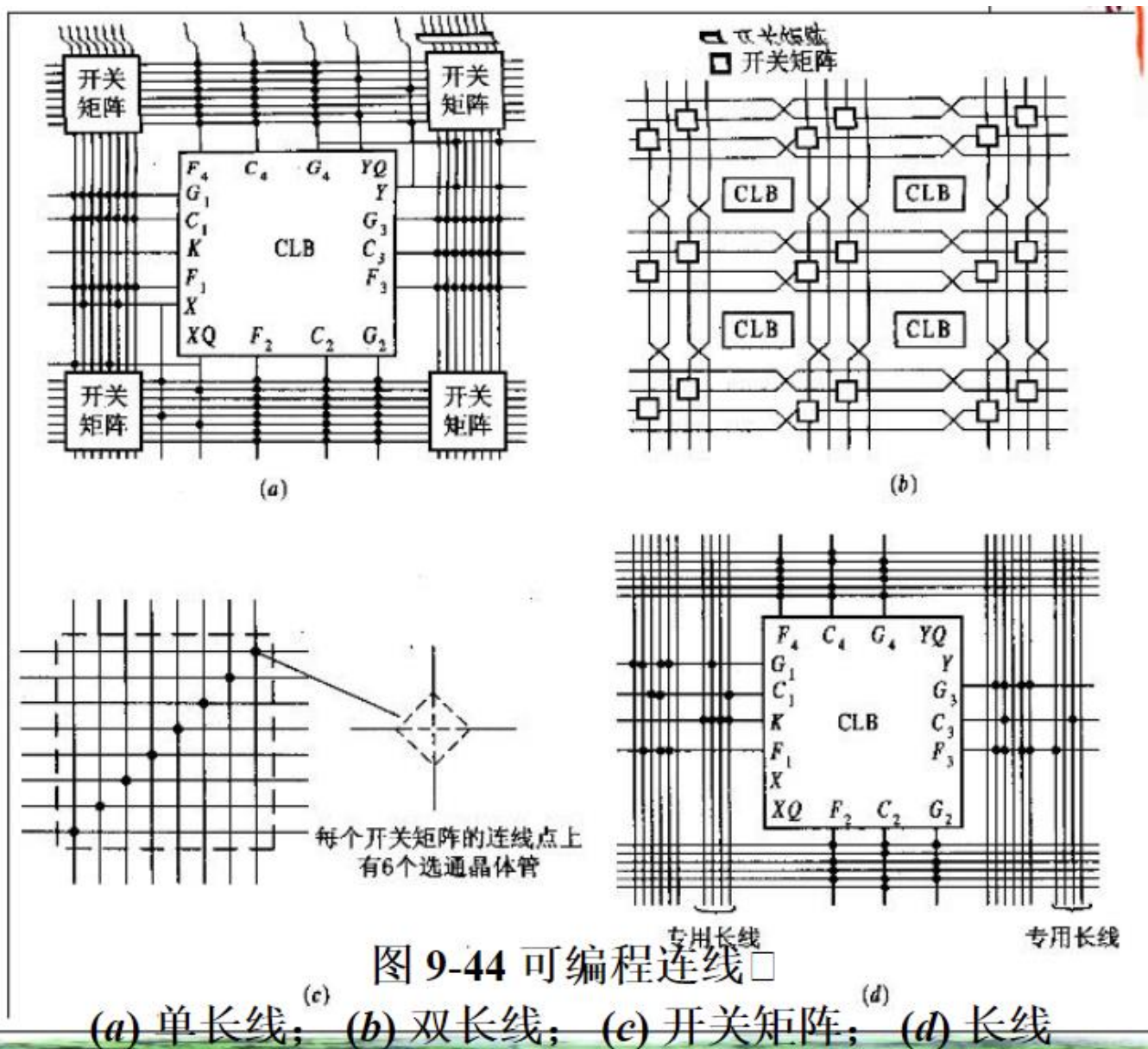
可以设置为输入/输出；

输入时可设置为：同步（经触发器）

异步（不经触发器）

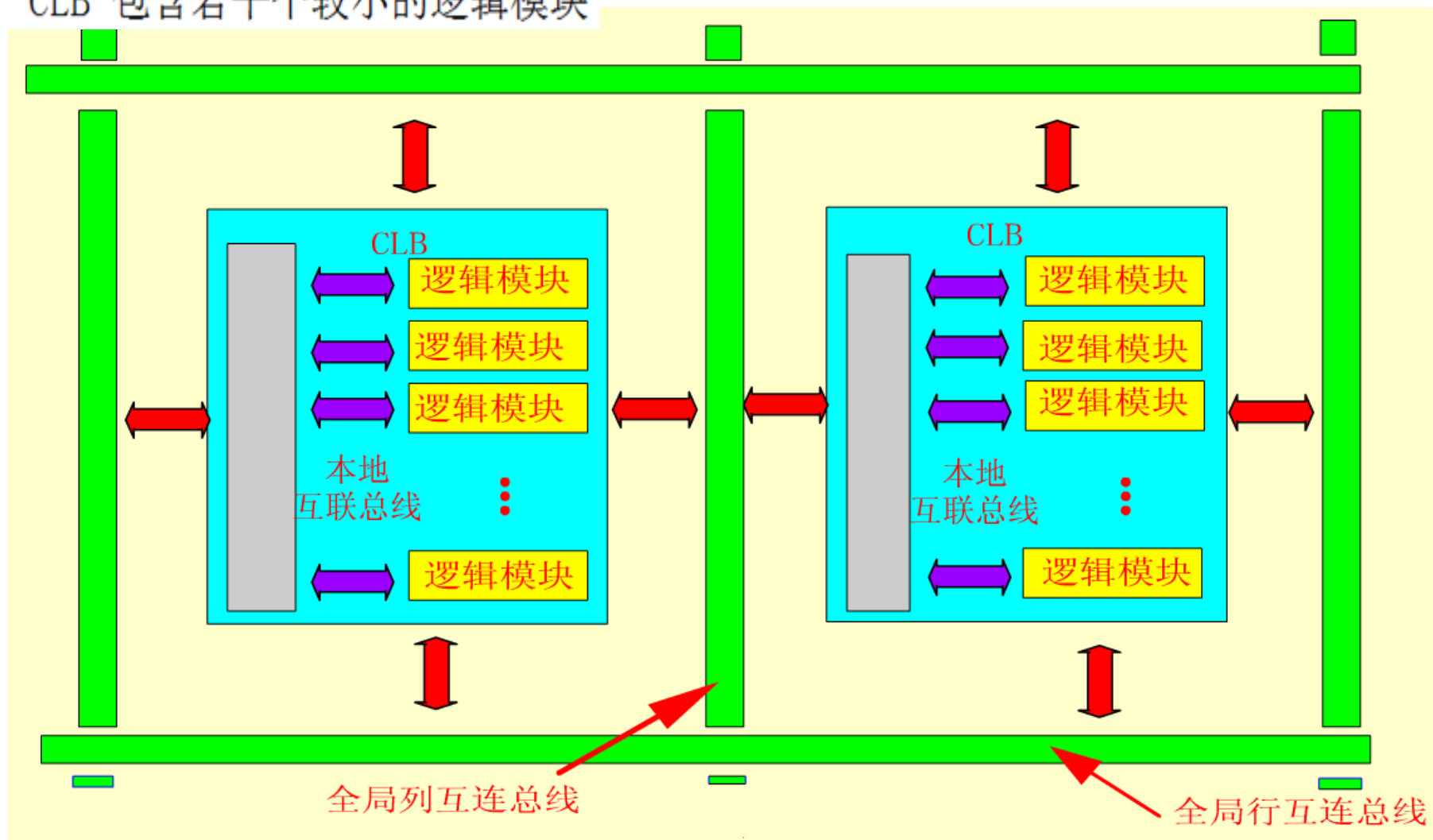
互连资源





CLB

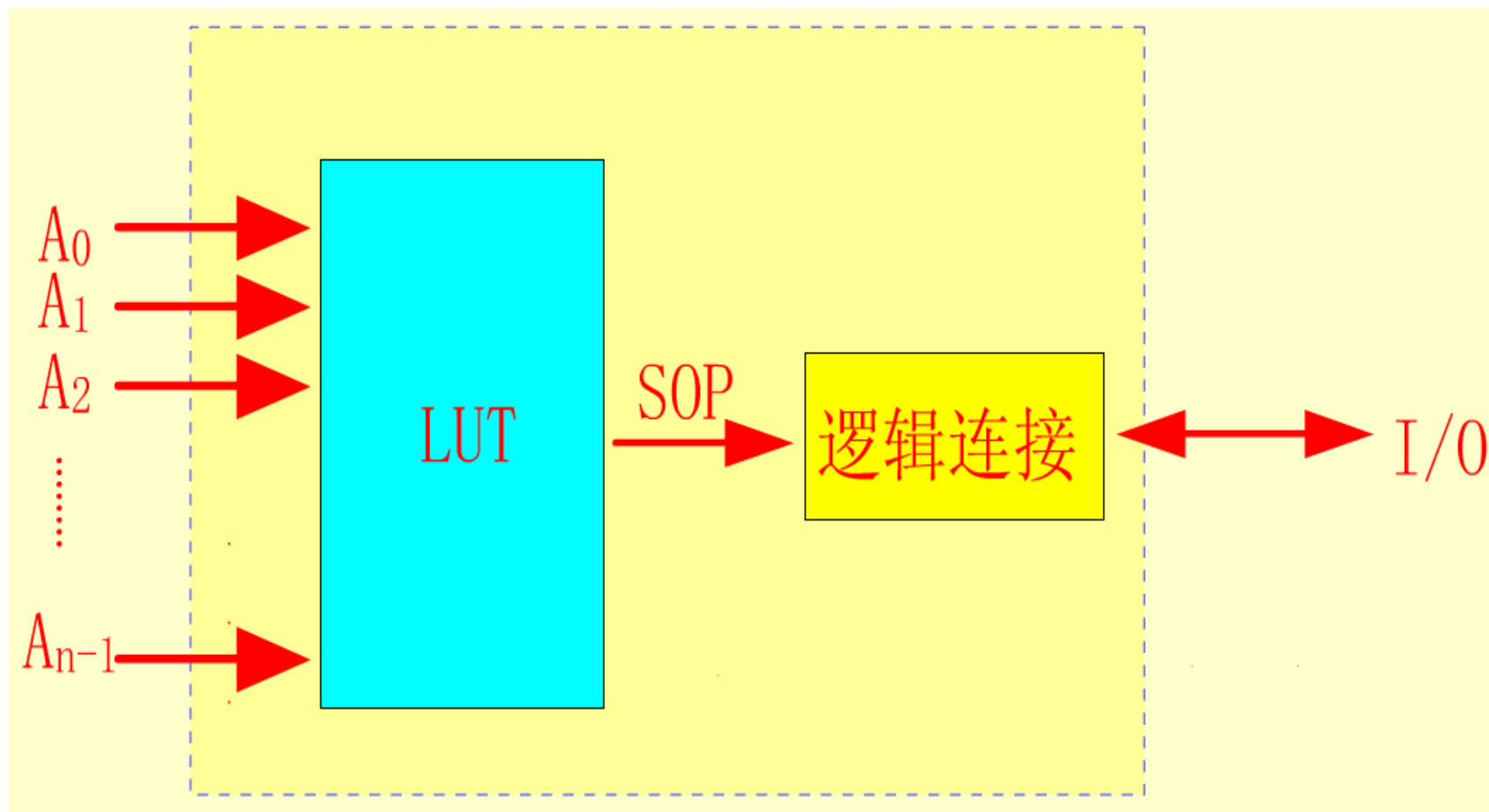
CLB 包含若干个较小的逻辑模块



基本可组态CLB

CLB—查找表

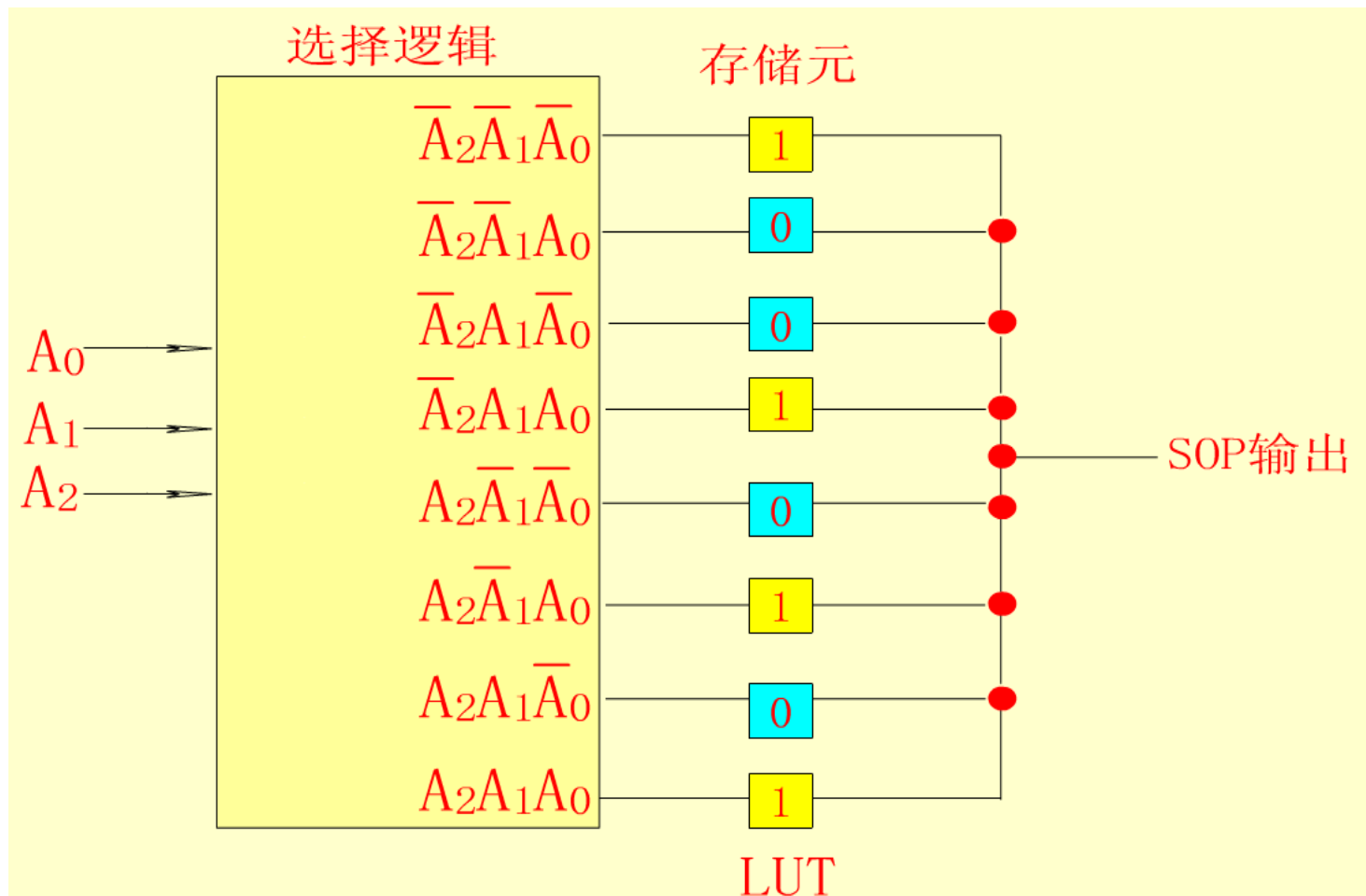
一个用逻辑模块实现的典型LUT（查找表类型的存储器）的框图



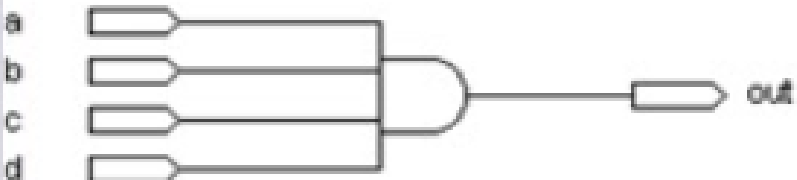
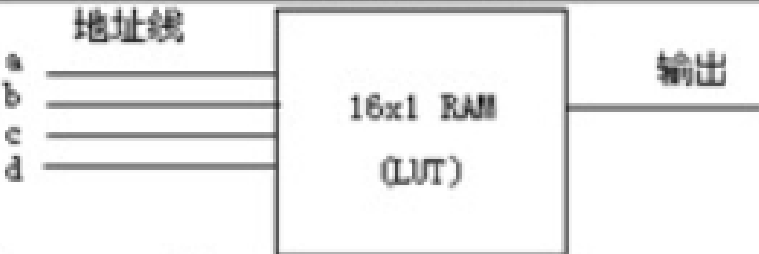
FPGA中一个逻辑模块的框图

CLB

LUT由数目等于 2^n 的存储元组成



LUT编程后用作SOP表达式输出

实际逻辑电路		LUT的实现方式	
			
a, b, c, d 输入	逻辑输出	地址	RAM中存储的内容
0000	0	0000	0
0001	0	0001	0
...	0	...	0
1111	1	1111	CSDN @芝士高达

CLB—查找表

例：以查找表型复杂可编程逻辑器件实现全加器。

A_n 为加数、 B_n 为被加数、 C_n 为低位进位、 S_n 为和、 C_{n+1} 为高位进位，则 $S_n = A_n \oplus B_n \oplus C_n$ ； $C_{n+1} = A_n B_n + A_n C_n + B_n C_n$ 。

把对应函数的真值表存放在SRAM中即可实现相应的函数运算：

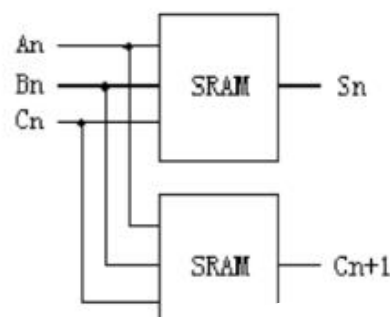
A_n	B_n	C_n	S_n	C_{n+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S_n = \bar{A}_n \bar{B}_n C_n + \bar{A}_n B_n \bar{C}_n + A_n \bar{B}_n \bar{C}_n + A_n B_n C_n$$

$$= A_n \oplus B_n \oplus C_n$$

$$C_{n+1} = \bar{A}_n B_n C_n + A_n \bar{B}_n C_n + A_n B_n \bar{C}_n + A_n B_n C_n$$

$$= A_n B_n + B_n C_n + A_n C_n$$



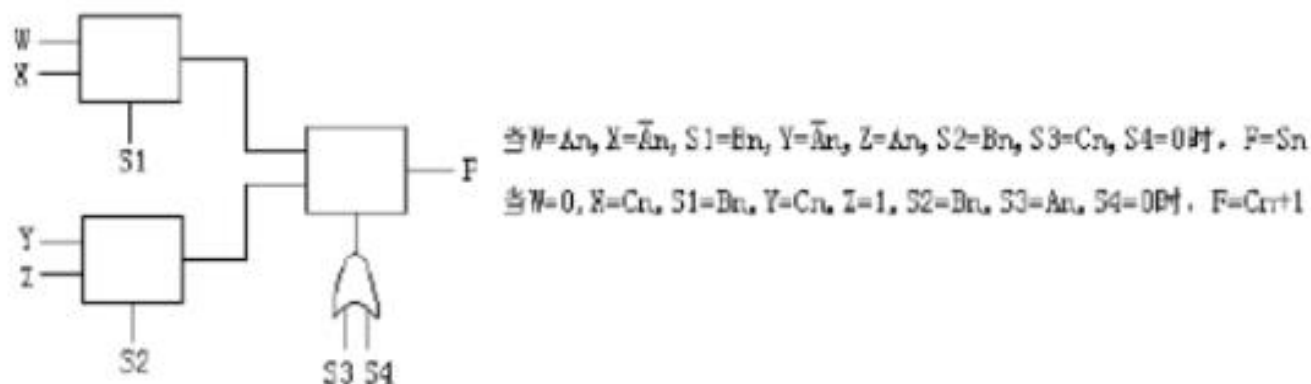
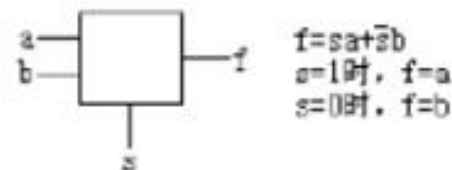
SRAM的地址线起输入线的作用，地址即输入变量值；SRAM的输出为逻辑函数值。

CLB—多路选择器

一个用逻辑模块实现的典型LUT（查找表类型的存储器）的框图

例：以多路开关型复杂可编程逻辑器件实现全加器。

多路开关的基本工作原理：

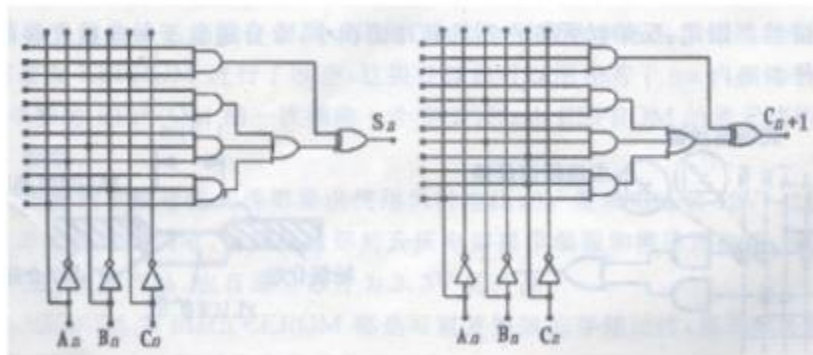


CLB—乘积项

●乘积项型

乘积项型复杂可编程逻辑器件其CLB由“与—或—异或”构成。

例：以乘积项型复杂可编程逻辑器件实现全加器。



$$S_n = (A_n \bar{B}_n + \bar{A}_n B_n + 0) \oplus C_n \\ = A_n \oplus B_n \oplus C_n$$

$$C_{n+1} = A_n C_n + B_n C_n + A_n B_n$$

乘积项型适合于多输入组合逻辑的实现；

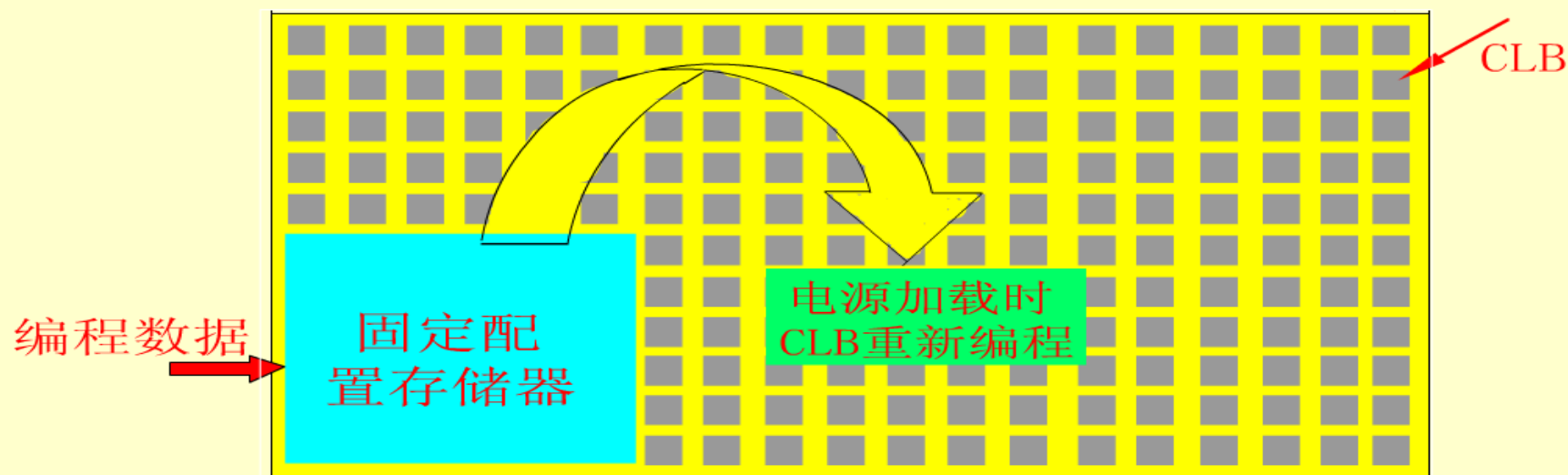
查找表型和多路开关型更适用于实现触发器较多的时序电路。

5.2.3 SRAM为基础的FPGA

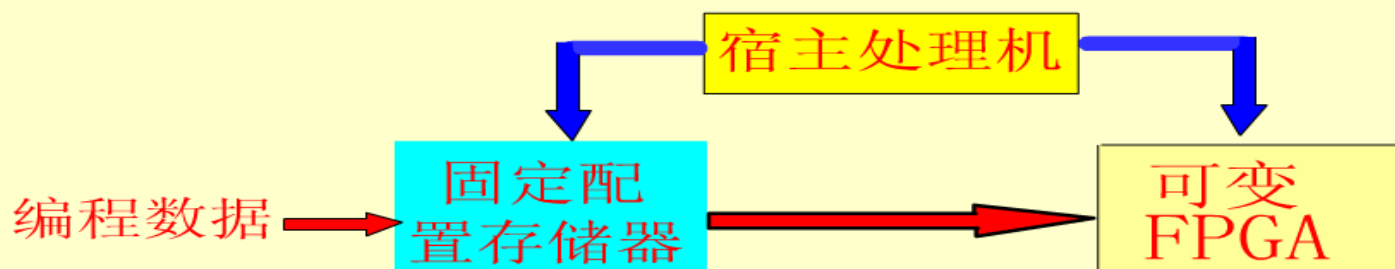
根据制造技术不同，FPGA在体系结构上分可变或不变两种。

“可变”意味着：当电源关闭时，被编程到组态逻辑块的所有数据将丢失。

可变FPGA配置的基本概念



(a) 可变FPGA利用芯片上的固定配置存储器



(b) 可变FPGA利用板上存储器和宿主处理机

5.4 可编程逻辑的原理图方式设计

5.4.1 编程环境和设计流程图

5.4.2 设计输入

5.4.3 功能模拟

5.4.4 综合和实现（软件）

5.4.5 时序模拟

5.4.6 器件下载

5.4.1 编程环境 and 设计流程图

1. 编程环境

EDA叫电子设计自动化

目标器件的放置采用两种方法：

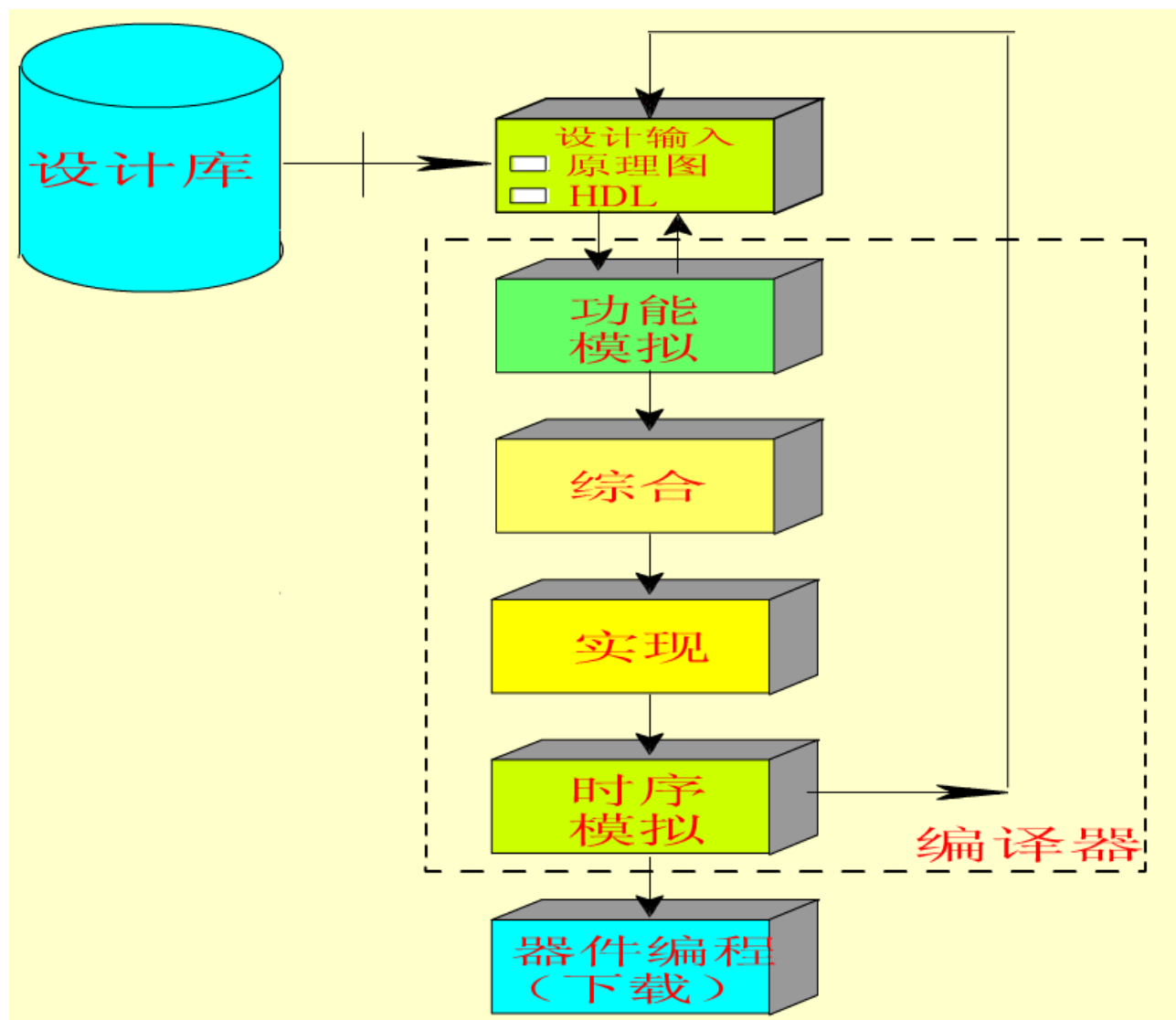
1. 用编程器
2. 用开发板




编程环境

5.4.1 编程环境 and 设计流程图

2. 设计流程图





设计输入：将所设计的电路输入到计算机。

1. 文本方式, 如: 用硬件描述语言完成  逻辑设计
2. 原理图方式

编译状态：源代码 目标代码（二进制代码）

功能模拟：确认逻辑设计实现预期的功能，利用波形编辑器

综合：设计被翻译成 网表Netlist

实现：网表描述的逻辑结构 被编程的指定器件

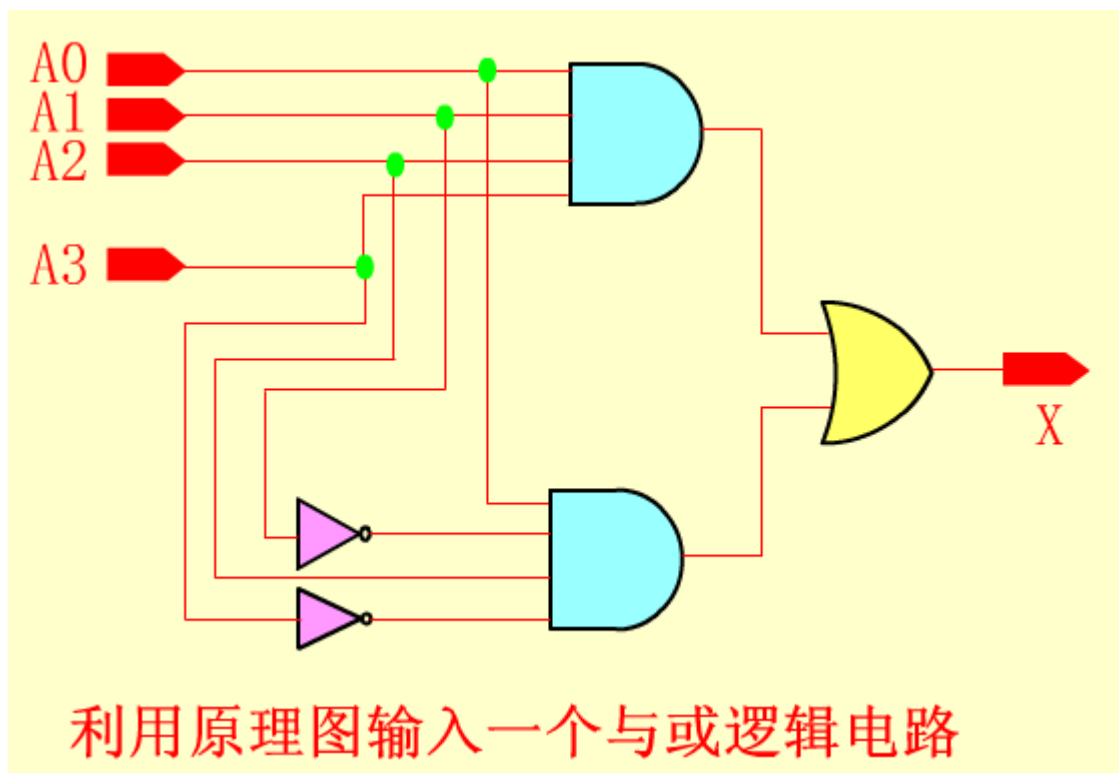
时序模拟：确保没有导致传播延迟的设计错误或时序问题

下载：位流下载到器件上

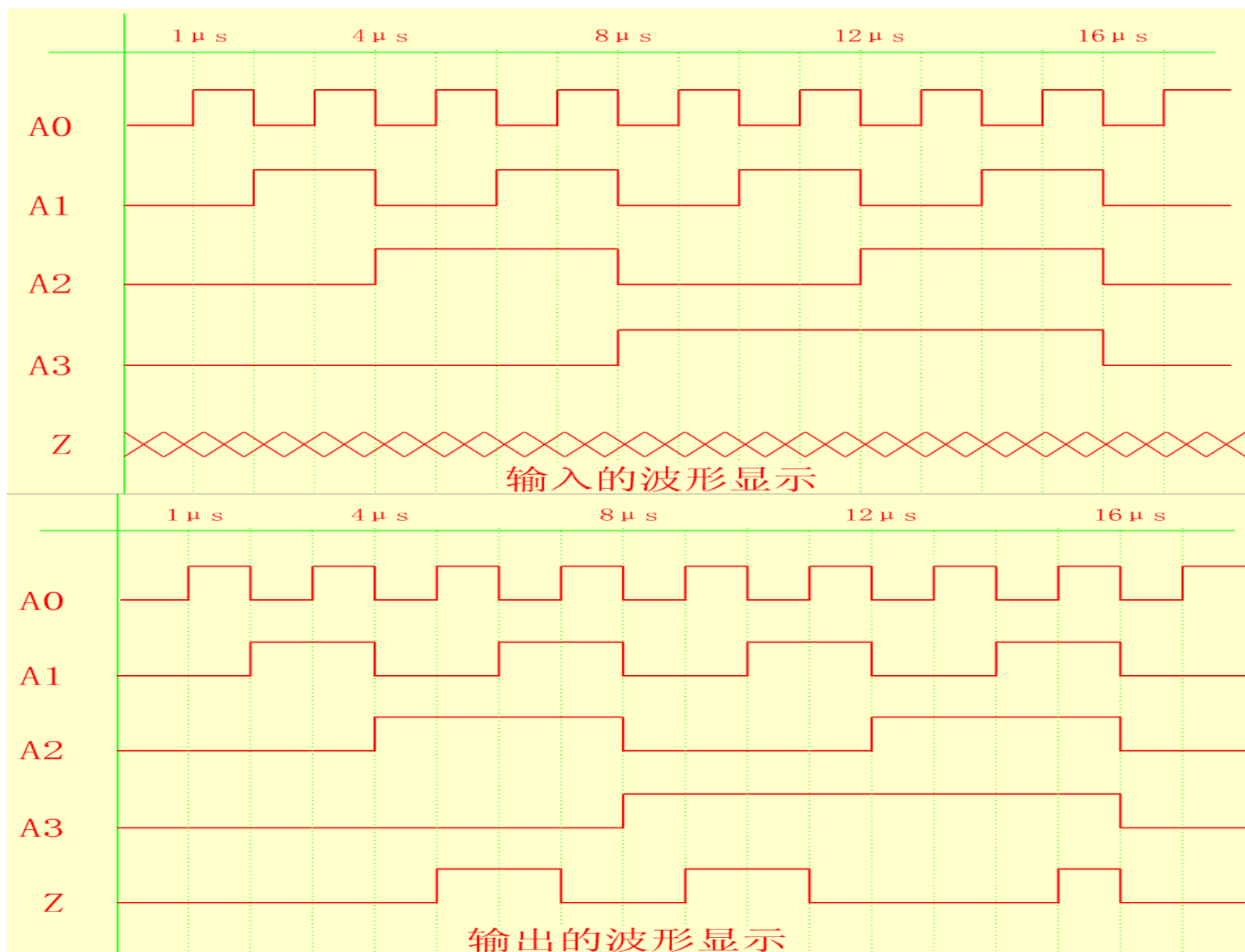
5.4.2 设计输入

可编程逻辑器件中实现的逻辑电路设计，可采用两种基本方式之一进行输入：

1. **文本输入**：用**硬件描述语言**完成，通用性好
2. **原理图输入**：可视化逻辑元件拖放、连接，式直观简单，难做大。



5.4.3 功能模拟



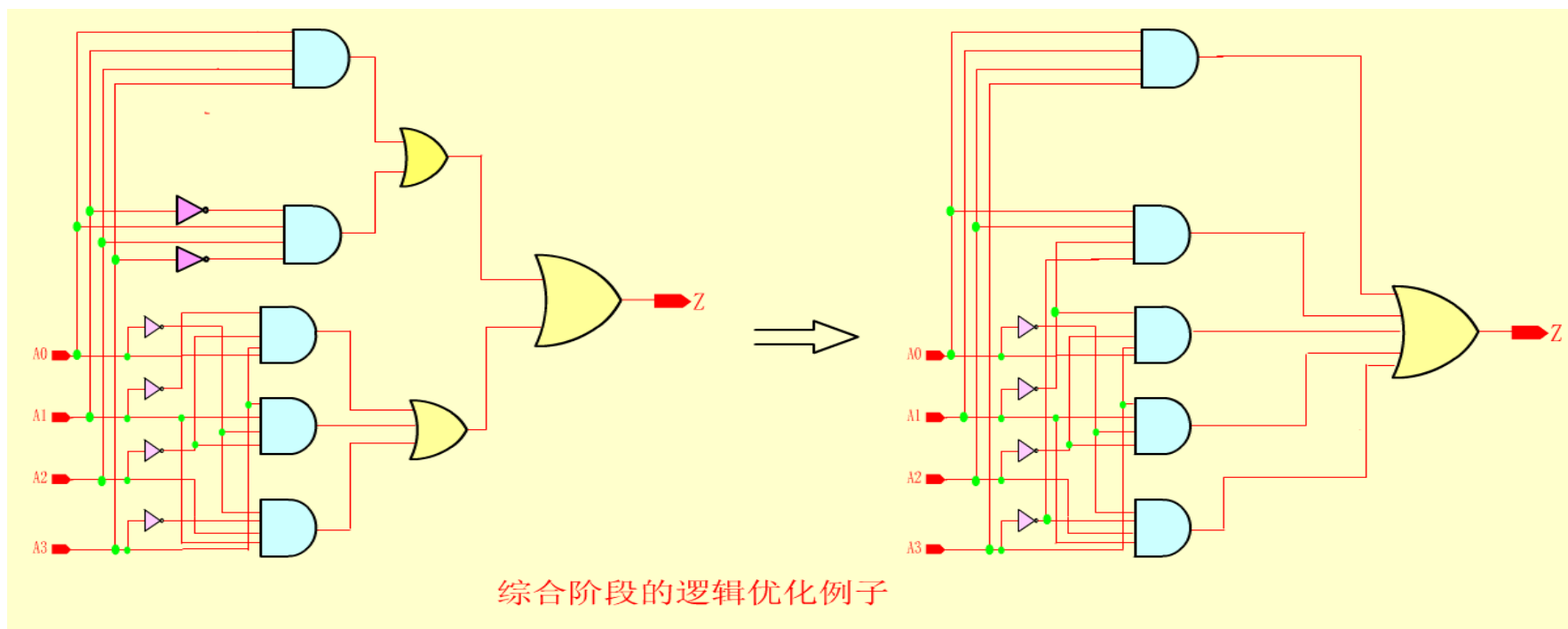
5.4.4 综合和实现（软件）

1. 综合阶段

优化：门的数量最小化

综合阶段输出：网表。

网表：由综合软件生成，它基本上是一个描述元件和它们怎样相互连接在一起的连接表。



5.4.4 综合和实现（软件）

2. 实现阶段

完成网表描述的逻辑结构 被编程的指定器件的映射：

使设计和器件自身体系结构、引脚配置的特定目标器件相适应。（设置和选径，简称适配）。

所有可能用到的目标器件的完整数据，通常保存在软件库中。

5.4.5 时序模拟

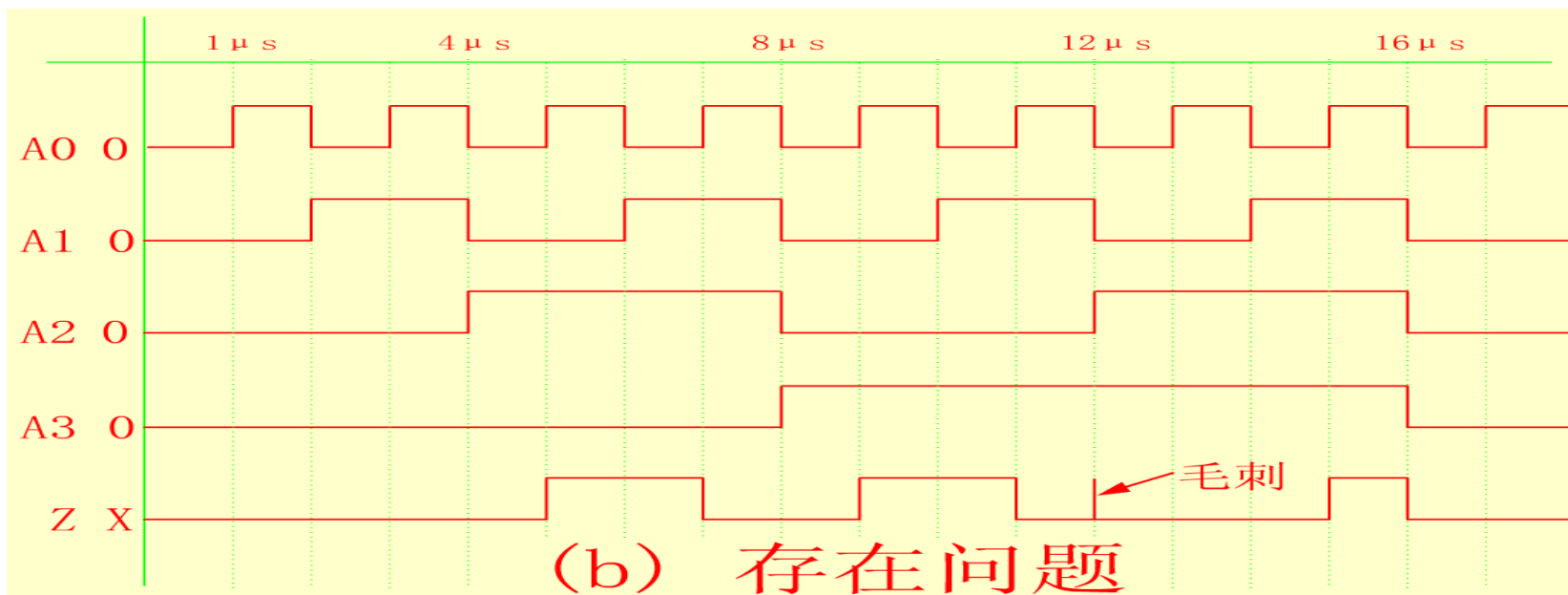
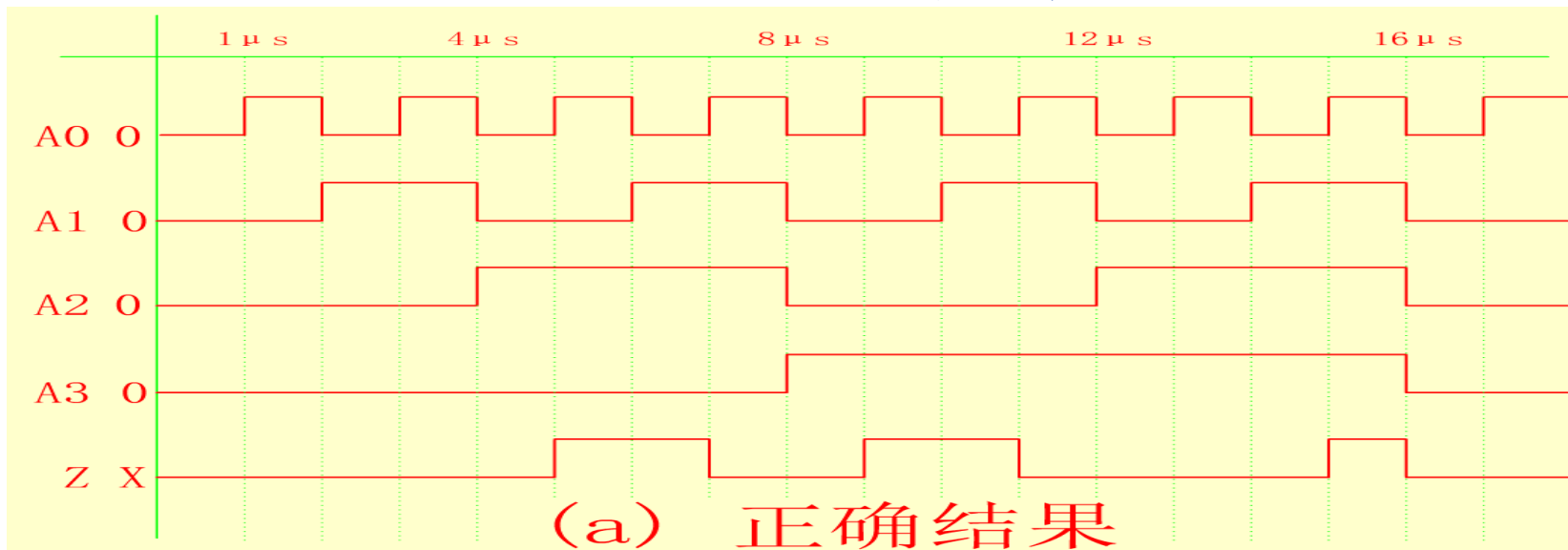
功能模拟已经通过了，但考虑特定目标器件的信息（如门的传输延迟），可能存在时序问题。

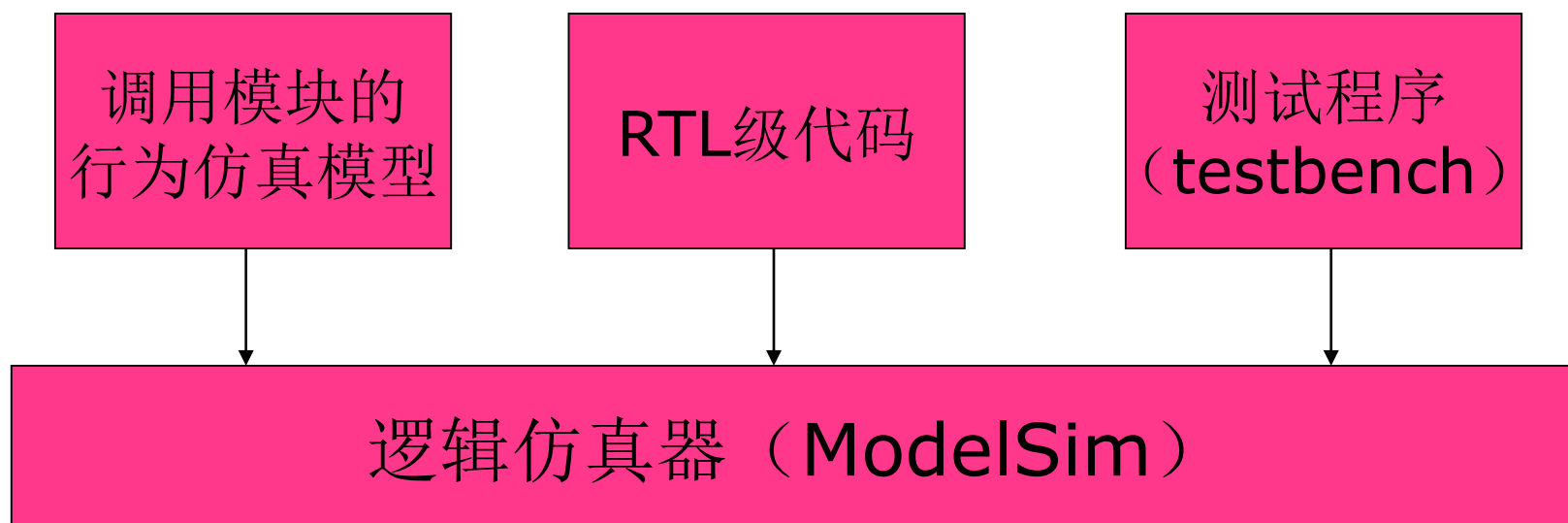
注：功能模拟中不需要选定目标器件。

时序模拟的目的：

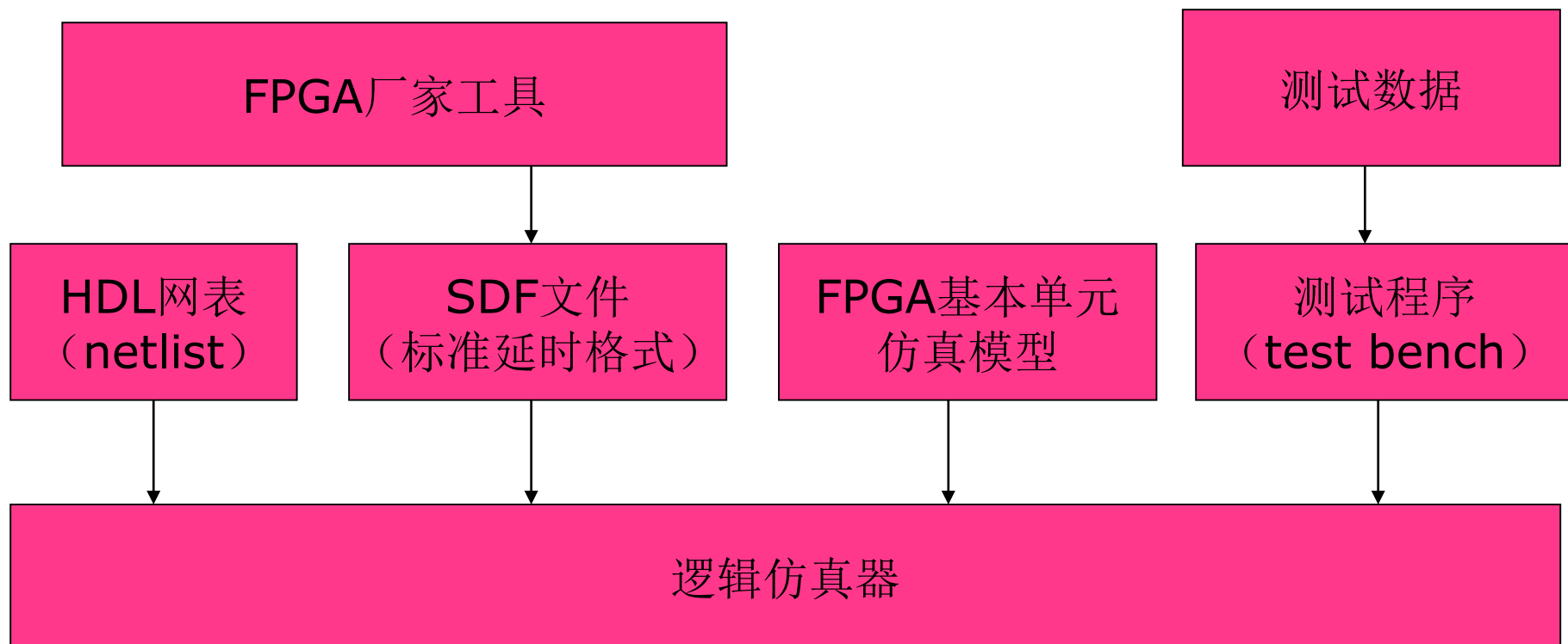
确保电路以设计频率工作，且没有传输延迟或其他影响全局操作的时序问题。

5.4.5 时序模拟





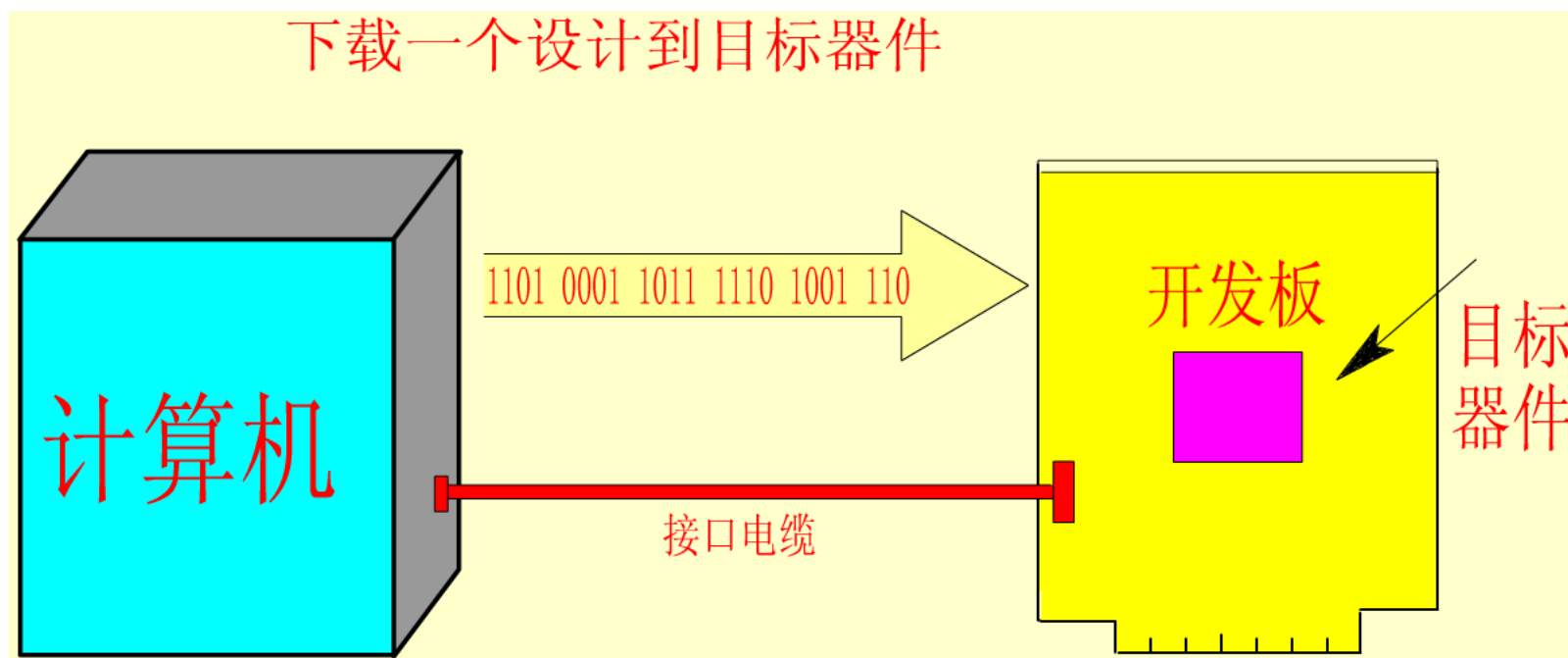
功能仿真示意图



后仿真示意图

5.4.6 器件下载

产生二进制码表示的位流后，发送到目标器件进行自动配置
设计已放置在目标器件上，并且可以在电路上进行测试。



- begin
- B=A;
- C=B+1;
- end

```
begin
    B<=A;
    C<=B+1;
end
```

代码先将A的值赋值给B，C的值是A+1

将A赋值给了B，但是C的值是B原来的值+1

= 立即生效

=> 要到下一个时钟沿赋值才会生效

- module blocking(clk, a, b, c);

- output [3:0] b,c;

- input [3:0] a;

- input clk;

- reg [3:0] b,c;

- always@(posedg

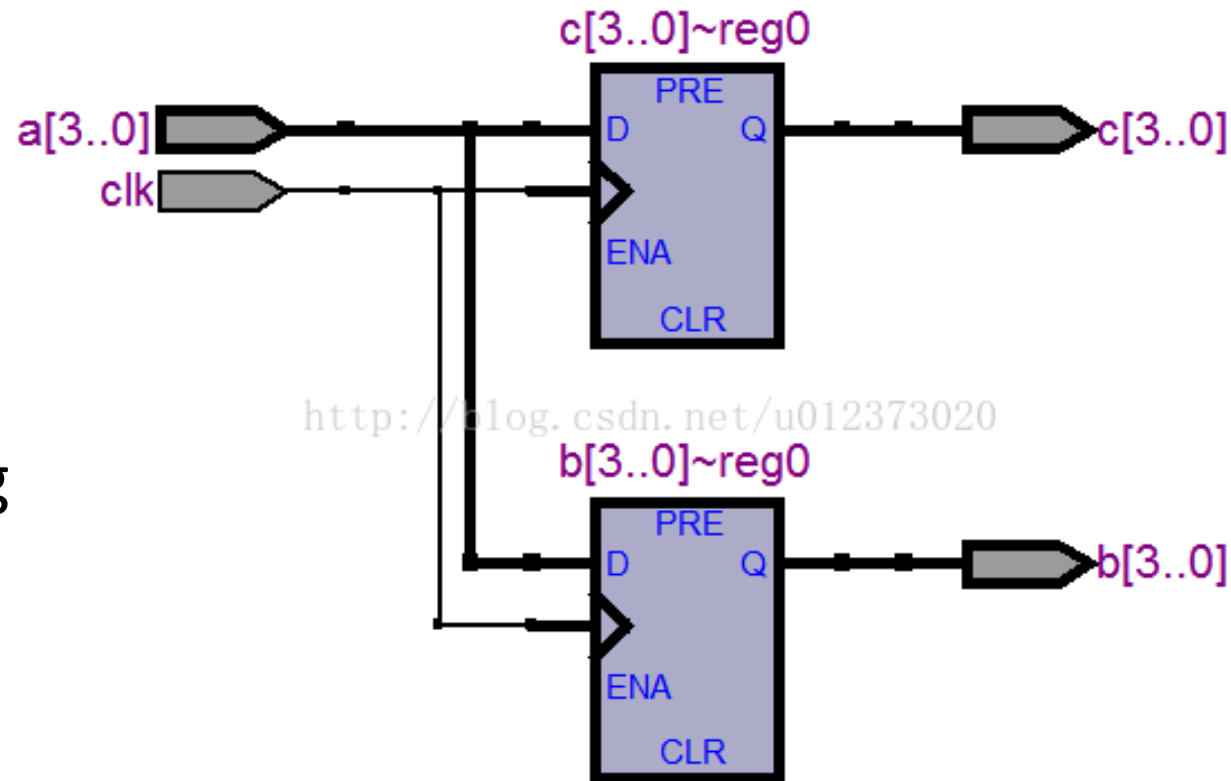
- begin

- b = a;

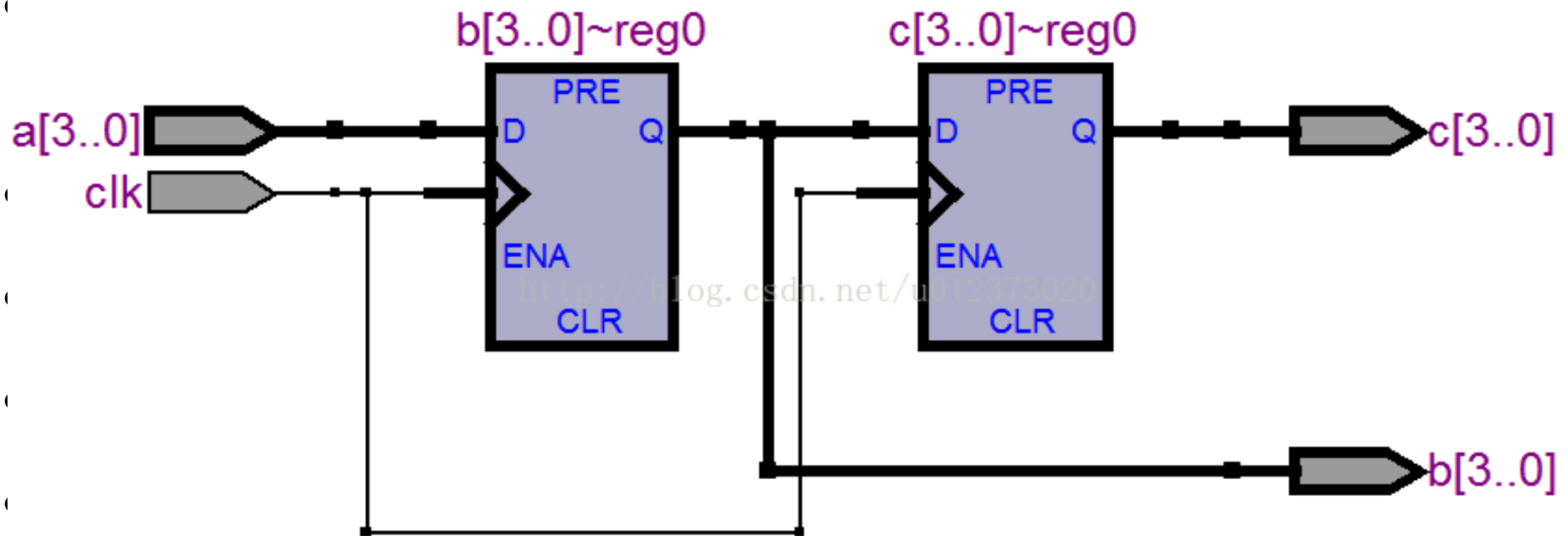
- c = b;

- end

- endendmodule



<http://blog.csdn.net/u012373020>

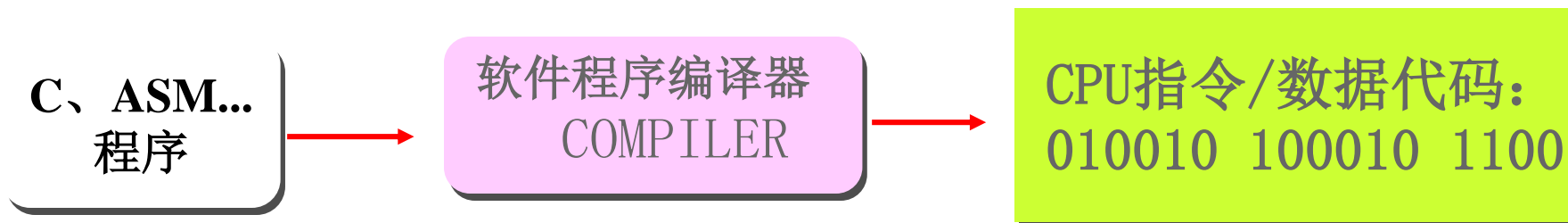


- `always@(posedge clk)`
`begin`
 - `b <= a;`
 - `c <= b;`
- `end`
- `endmodule`

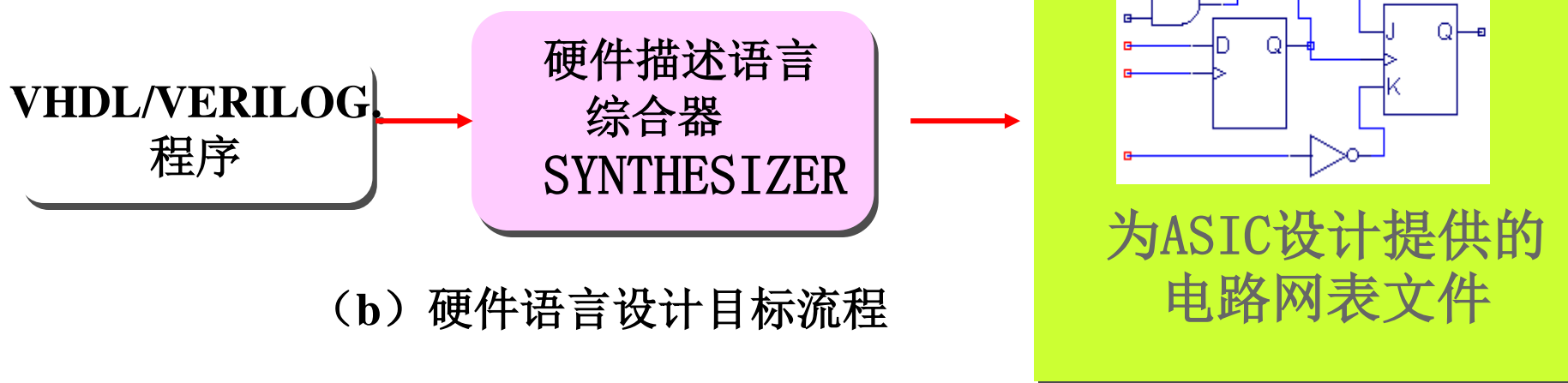
- (1) 时序电路建模时，用非阻塞赋值。
- (2) 锁存器电路建模时，用非阻塞赋值。
- (3) 用**always**块建立组合逻辑模型时，用阻塞赋值。
- (4) 在同一个**always**块中建立时序和组合逻辑电路时，用非阻塞赋值。
- (5) 在同一个**always**块中不要既用非阻塞赋值又用阻塞赋值。
- (6) 不要在一个以上的**always**块中为同一个变量赋值。
- (7) 用**\$strobe**系统任务来显示用非阻塞赋值的变量值。
- (8) 在赋值时不要使用**#0**延时。

如果在一个过程块中阻塞赋值的RHS变量正好是另一个过程块中阻塞赋值的LHS变量，这两个过程块又用同一个时钟沿触发，这时阻塞赋值操作会出现问题，即如果阻塞赋值的次序安排不好，就会出现竞争。若这两个阻塞赋值操作用同一个时钟沿触发，则执行的次序是无法确定的。

软件编译器和硬件综合器区别



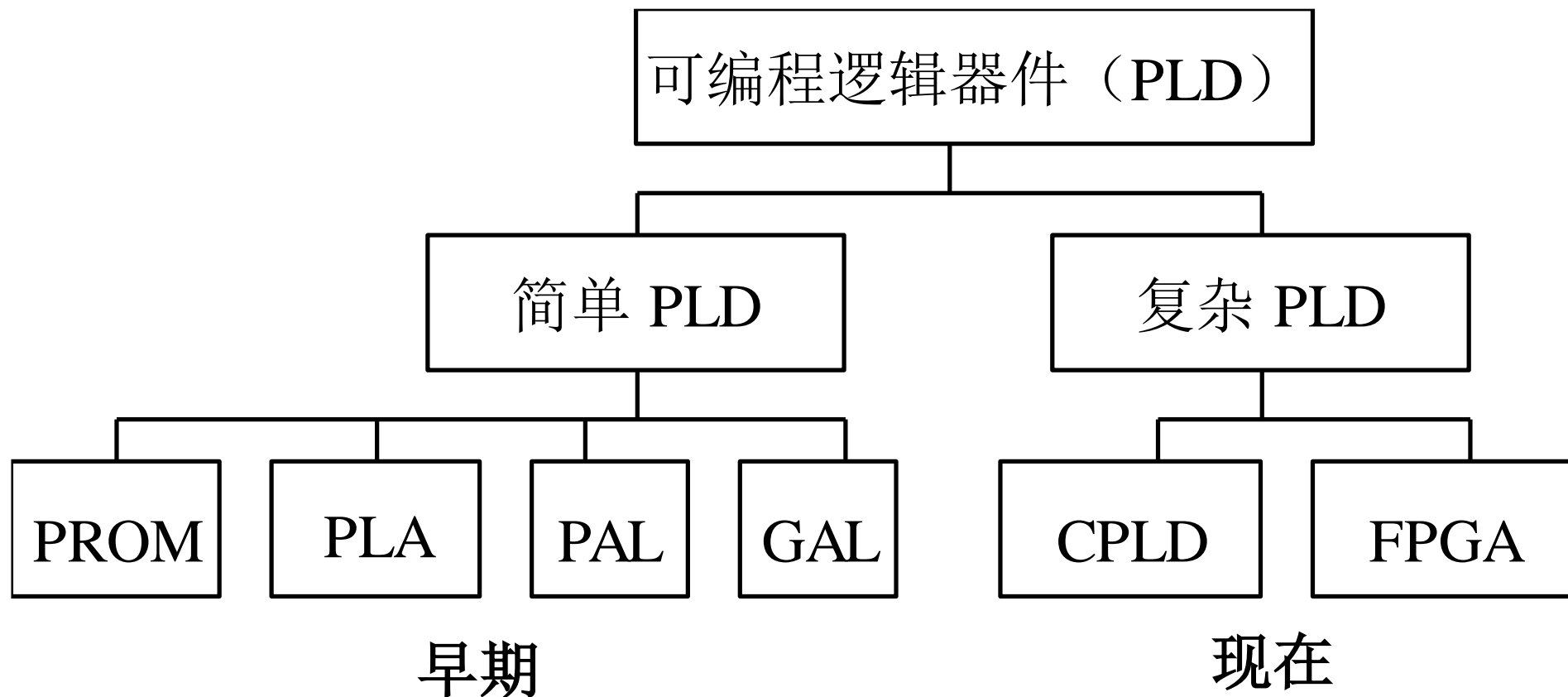
(a) 软件语言设计目标流程



(b) 硬件语言设计目标流程

总结

1.按集成度分类



一般将**GAL22V10**（500门~750门）作为简单PLD和高密度PLD的分水岭。

四种SPLD器件的区别

器件	与阵列	或阵列	输出电路
PROM	固定	可编程	固定
PLA	可编程	可编程	固定
PAL	可编程	固定	固定
GAL	可编程	固定	可组态

2.按编程特点分类

1) **PLD**器件按照可以编程的次数可以分为两类:

(1) 一次性编程器件 (**OTP, One Time Programmable**)

(2) 可多次编程器件

OTP类器件的特点是: 只允许对器件编程一次, 不能修改, 而可多次编程器件则允许对器件多次编程, 适合于科研开发中使用。

2) 按编程元件和编程工艺分类

(1) 熔丝 (**Fuse**)

(2) 反熔丝 (**Antifuse**) 编程元件

非易失性

(3) 紫外线擦除、电可编程，如**EPROM**。器件

(4) 电擦除、电可编程方式，(**EEPROM**、快闪存储器 (**Flash Memory**))，如多数**CPLD**

(5) 静态存储器 (**SRAM**) 结构，如多数**FPGA**



易失性器件

3.按结构特点分类

- 1) 基于乘积项 (Product-Term) 结构的PLD器件
- 2) 基于查找表 (Look Up Table,LUT) 结构的PLD器件

可编程逻辑器件的发展趋势

1. 最先进的生产工艺广泛应用于以FPGA为代表的可编程逻辑器件

40nm系列产品大获成功的基础上，Altera 28nm系列产品实现了很多创新。Stratix V系列FPGA是Altera推出的首款28nm产品，今后还会推出更多的产品。

2. 越来越多的高端FPGA产品将包含CPU或DSP等处理器内核
3. 传统ASIC和FPGA进一步相互融合
4. 低成本的FPGA的密度越来越高，价格越来越合理

关于FPAG的一些基本概念

1.FPGA与ASIC

2.FPGA与CPLD

3.Altera与Xilinx

全球总部在同一城市——美国的圣何塞

4.Verilog与VHDL

FPGA与ASIC

FPGA 设计

优势	优势
可加速上市进程	更快的上市时间 - 无需布局、掩模和其它制造步骤。
非提前支付的一次性开支 (NRE)	这些成本通常与 ASIC 设计相关
更简化的设计周期	由于软件可以处理很多布线、布局和时序问题
更具预测性的项目周期	由于消除了潜在的重新设计和晶圆容量等
现场可重编程功能	可以远程上传的新比特流

ASIC 设计

优势	优势
完整的定制功能	由于器件是根据设计规格来生产的
降低器件成本	可实现大批量设计
更小巧的尺寸	由于器件是根据设计规格来生产的

都是可编程ASIC器件，CPLD和FPGA结构上的差异，具有各自的特点：

①CPLD更适合算法和组合逻辑，FPGA更适合时序逻辑。即FPGA更适合于[触发器](#)丰富的结构，而CPLD更适合于触发器有限而乘积项丰富的结构。

②CPLD的连续式[布线](#)结构决定了它的时序延迟是均匀的和可预测的，而FPGA的分段式布线结构决定了其延迟的不可预测性。

③编程上FPGA更大的灵活性。CPLD修改固定内连电路的逻辑功能来编程，FPGA主要通过改变内部连线的布线来编程；FPGA可在逻辑门下编程，而CPLD是在逻辑块下编程。

④FPGA的集成度比CPLD高，具有更复杂的布线结构和逻辑实现。

⑤CPLD比FPGA使用更方便。CPLD的编程采用E2PROM或FASTFLASH技术，无需外部[存储器](#)芯片，使用简单。FPGA的编程信息需存在外部存储器，使用方法复杂。

⑥CPLD的速度比FPGA快，具有较大的时间可预测性。FPGA是门级编程，并且CLB之间采用分布式互联，而CPLD是逻辑块级编程，并且其逻辑块之间的互联是集总式的。

⑦ 编程方式上，CPLD基于E2PROM或FLASH存储器编程，1万次，优点是系统断电时编程信息也不丢失。CPLD又可分为在编程器上编程和在系统编程两类。FPGA大部分是基于[SRAM](#)编程，编程信息在系统断电时丢失，每次上电时，需从器件外部将编程数据重新写入