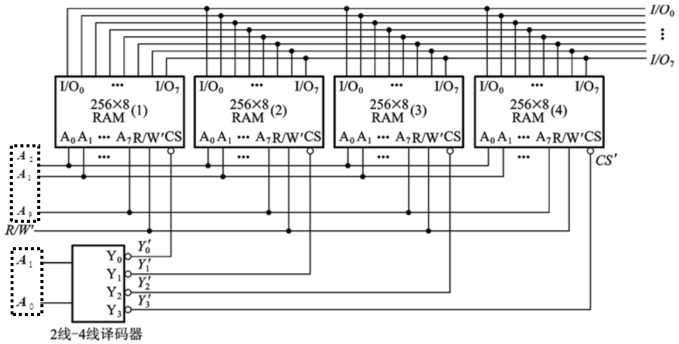
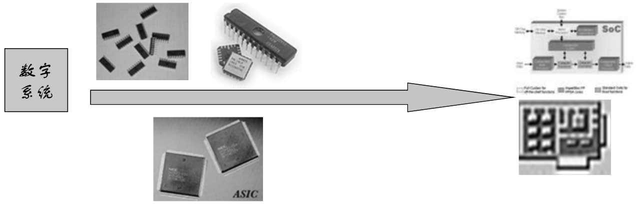


第八章　可编程逻辑器件
（**PLD, Programmable Logic Device**）

8.1 概述

一、**PLD**的基本特点：

1. 数字集成电路从功能上有分为　通用型、专用型两大类



2. **PLD**的特点：是一种按通用器件来生产，但逻辑功能是由用户通过对器件编程来设定的

二、**PLD**的发展和分类

PROM是最早的**PLD**

1. **FPLA**　现场可编程逻辑阵列

2. **PAL**可编程逻辑阵列

3. **GAL** 通用阵列逻辑

4. **EPLD** 可擦除的可编程逻辑器件

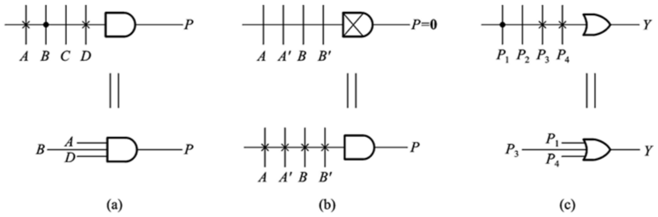
5. **CPLD** 复杂的可编程逻辑器件

6. **FPGA** 现场可编程门阵列

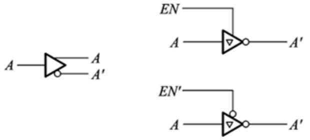
Field　Programmable　Logic

Array　General　Erase　Complex　Device

三、**LSI**中用的逻辑图符号

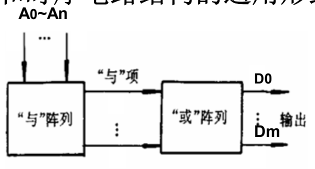


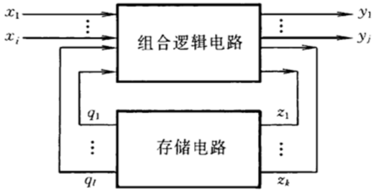
(a)　(b)　(c)



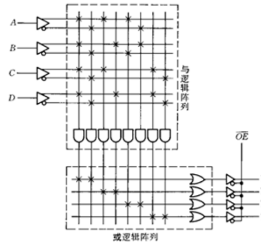
8.2 FPLA

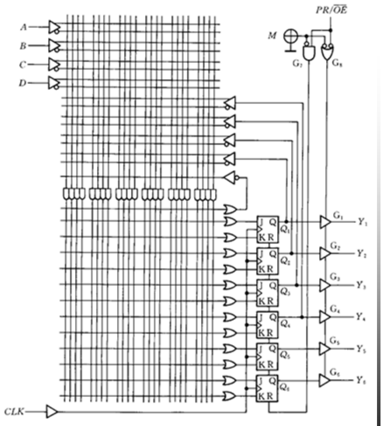
组合电路和时序电路结构的通用形式





8.2 FPLA





可编程的“与”阵列

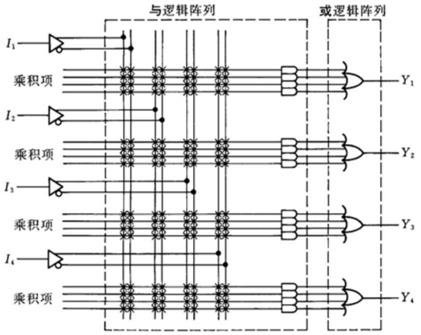
＋可编程的“或”阵列

8.3 PAL（Programmable Array Logic）

一、基本结构形式

可编程“与”阵列 + 固定“或”阵列 + 输出电路

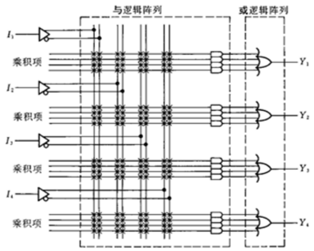
最简单的形式为：

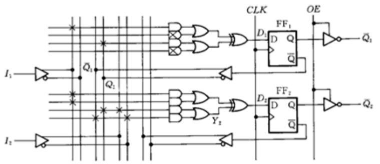


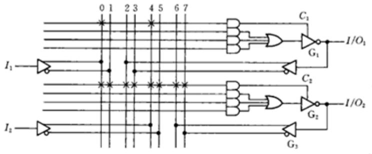
二、编程单元

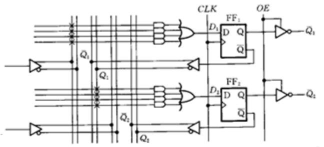
出厂时，

所有的交叉点均有熔丝







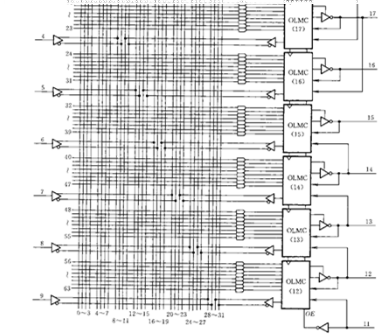


8.4 GAL（Generic Array Logic）

一、电路结构形式

可编程“与”阵列 + 固定“或”阵列 + 可编程输出电路

OLMC

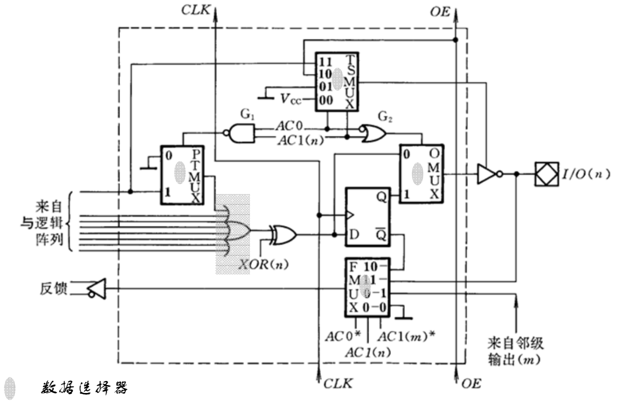


二、编程单元

采用E^2CMOS 可改写

GAL16V8

三、OLMC



8.5 EPLD

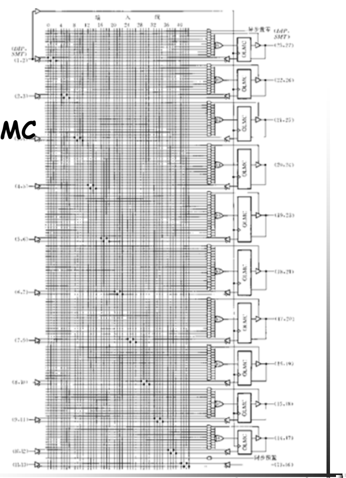
一、结构特点

相当于

“与-或”阵列（PAL） + OLMC

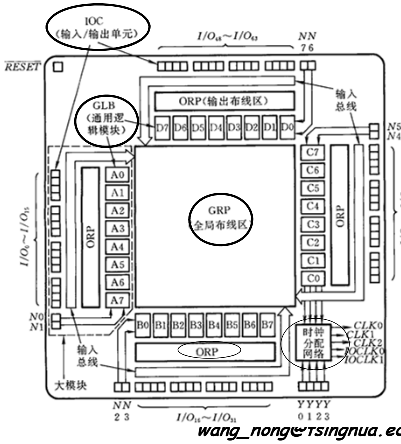
二、采用EPROM工艺

集成度提高



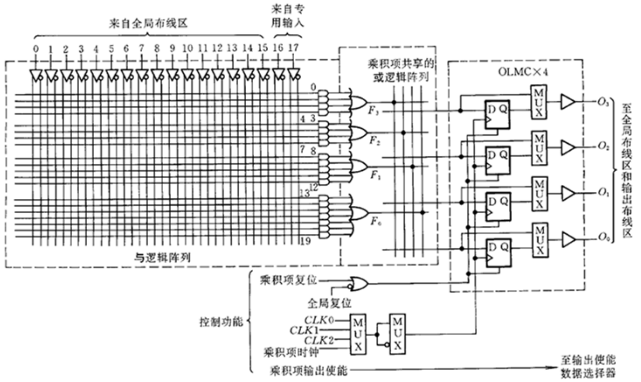
8.6 CPLD

结构

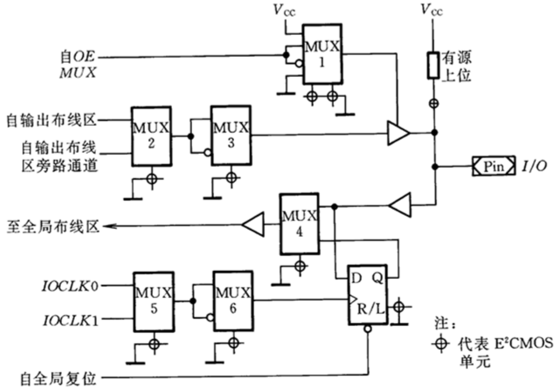


ispLSI1032

GLB

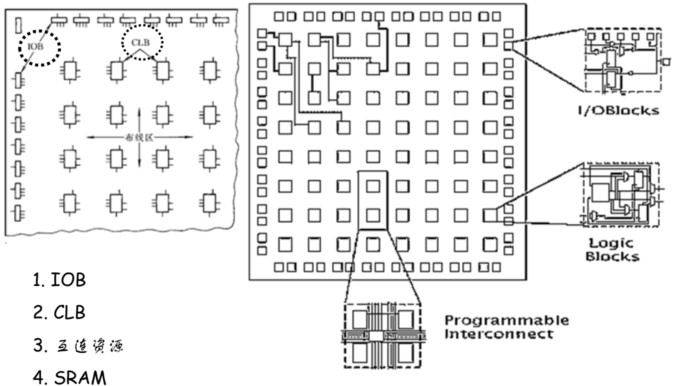


IOC



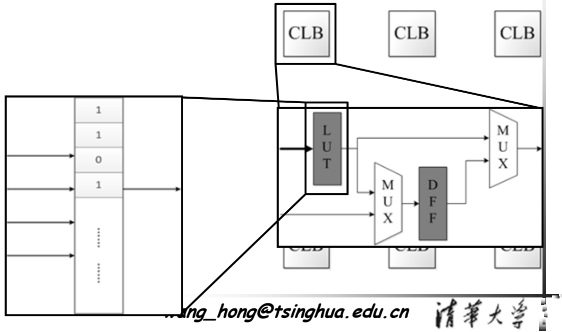
8.7 FPGA（Field Programmable Gate Array）

一、基本结构

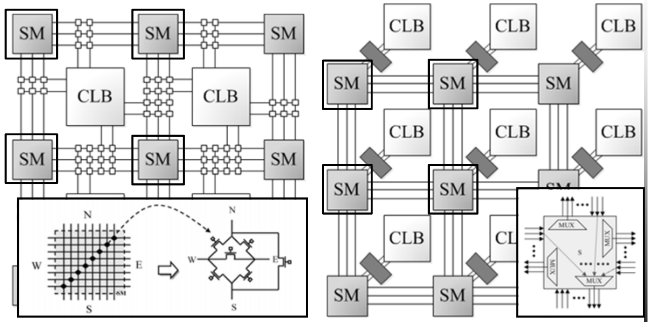


1. IOB
2. CLB
3. 互连资源
4. SRAM

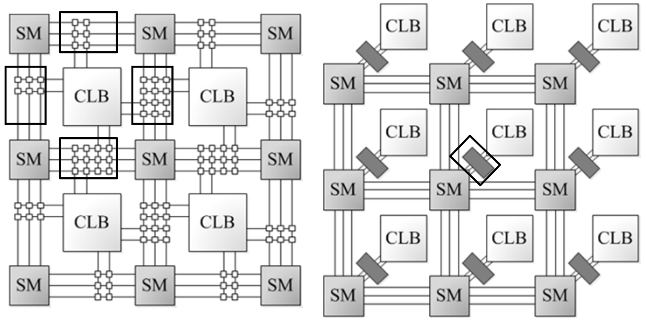
- **SRAM型FPGA芯片的结构[Xilinx][Altera]**
  - **CLB**
    - **LUT**（Look Up Table，查找表）
    - **DFF**（D Flip-Flop，D触发器）
    - **MUX**（Multiplexer，数据选择器）



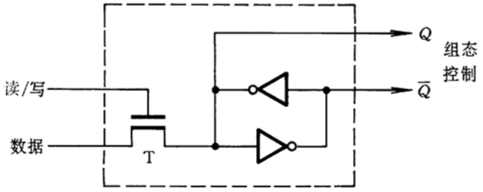
- SRAM型FPGA芯片的结构[Xilinx][Altera]
  - **Interconnect**
    - 全局布线资源：**SM**（**Switch Matrix**，开关矩阵）阵列



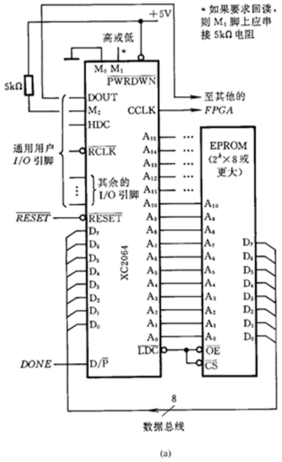
- SRAM型FPGA芯片的组成部分[Xilinx][Altera]
  - **Interconnect**
    - 局部布线资源

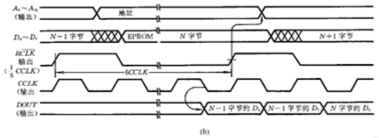


- SRAM型FPGA芯片的组成部分[Xilinx][Altera]
  - **SRAM**
    分布式
    每一位触发器控制一个编程点



二、编程数据的装载



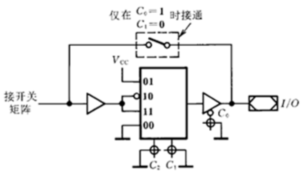


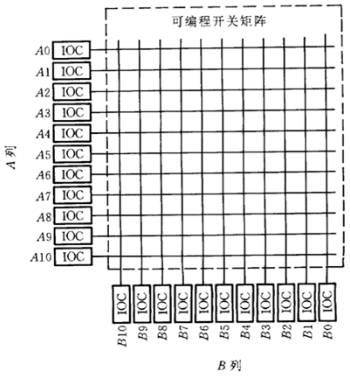
1. 数据可先放在EPROM或PC机中
2. 通电后，自行启动FPGA内部的一个时序控制逻辑电路，将在EPROM中存放的数据读入FPGA的SRAM中
3. “装载”结束后，进入编程设定的工作状态

！！每次停电后，SRAM中数据消失
下次工作仍需重新装载

8.8 可编程的数字开关

ispGDS22的
结构框图





8.9 PLD的使用

以上各种PLD均需编程操作，使用相应的开发系统

一、开发系统

1. 硬件：计算机 + 编程器

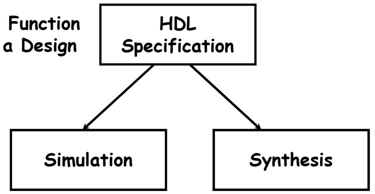
2. 软件：开发环境（软件平台）------EDA

二、步骤

- 抽象

- 选定PLD
- 选定开发系统
- 编写源程序（或输入文件）
- 调试，运行仿真，产生下载文件
- 下载
- 测试

Design Methodology


Structure and Function (Behavior) of a Design

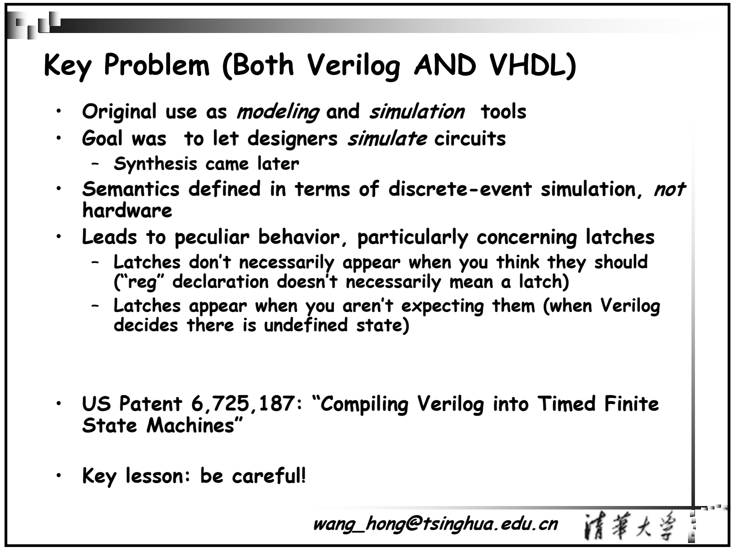
HDL Specification

Simulation

Synthesis

Verification: Design Behave as Required?
Functional: I/O Behavior
 Register-Level (Architectural)
 Logic-Level (Gates)
 Transistor-Level (Electrical)
 Timing: Waveform Behavior

Generation: Map Specification to Implementation



Key Problem (Both Verilog AND VHDL)

- Original use as *modeling* and *simulation* tools
- Goal was to let designers *simulate* circuits
  - Synthesis came later
- Semantics defined in terms of discrete-event simulation, *not* hardware
- Leads to peculiar behavior, particularly concerning latches
  - Latches don't necessarily appear when you think they should ("reg" declaration doesn't necessarily mean a latch)
  - Latches appear when you aren't expecting them (when Verilog decides there is undefined state)
- US Patent 6,725,187: "Compiling Verilog into Timed Finite State Machines"
- Key lesson: be careful!

wang_hong@tsinghua.edu.cn