

# Some

作者：Pannenets.F

时间：November 20, 2020

Je reviendrai et je serai des millions. ——«Spartacus»

# 第一部分

# 课件

# 绪论

# 数字电路高层次综合设计

北京航空航天大学  
微电子学院  
贾小涛

# 基本信息

- ❖ 上课时间：周二/周四，第一、二节
- ❖ 上课地点：新主楼F207
- ❖ 总学时：64学时
  - 授课：32学时
  - 实验：32学时
- ❖ 学分：3
- ❖ 类型：专业基础课
- ❖ 先修课程：电子电路、数字集成电路设计、C语言
- ❖ 后续课程：集成电路设计实训、智能芯片设计

# 课程内容

## ❖ 课堂授课

- 数字系统概述 (4学时)
- Verilog电路设计 (22学时)
- 逻辑综合 (4学时)
- 物理综合 (2学时)

数字电路高层次综合设计



数字系统设计

## ❖ 课程实验 (8-10个)

- 全加器设计
- 有限状态机
- 数码管显示
- 交通信号灯设计

触发器、移位寄存器

异步FIFO

数字钟设计

等

# 课程目的

## ❖ 能说

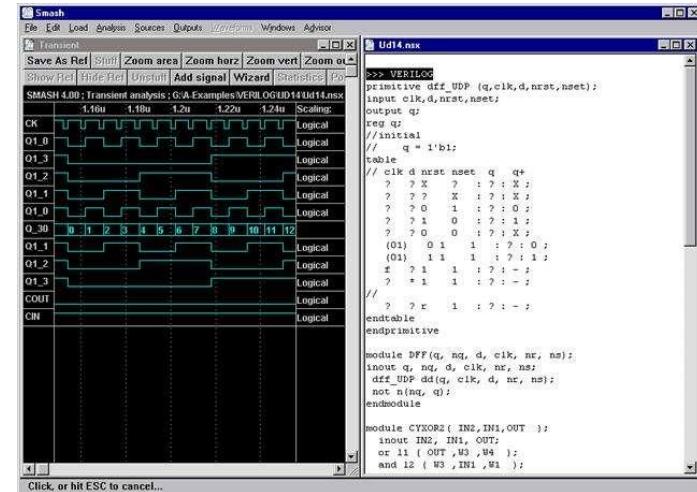
- 了解数字系统基本概念
- 了解数字系统设计流程
- 了解FPGA原理

## ❖ 会写

- 熟练编写Verilog代码
- 数量掌握仿真工具的使用

## ❖ 玩起来

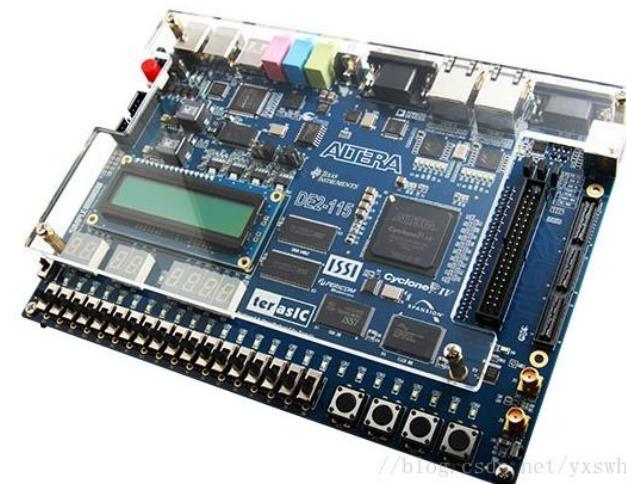
- 在FPGA上设计简易数字系统
- Altera DE2-115开发板



The screenshot shows the Smash simulation software interface. On the left, there's a waveform viewer with multiple channels: CK, Q1\_0, Q1\_1, Q1\_2, Q1\_3, Q1\_4, Q1\_5, Q1\_6, Q1\_7, Q1\_8, Q1\_9, Q1\_10, Q1\_11, Q1\_12, COUT, and CIN. The CK signal is a square wave, while the Q outputs show the state transitions over time. On the right, the code editor displays the Verilog code for a D flip-flop:

```
module DFF(q, nq, d, clk, nrst, ns);
    inout q, nq, d, clk, nrst, ns;
    dff UDP dff(q, clk, d, nrst, ns);
    not n(nq, q);
endmodule

module CYKOR2(IN2, IN1, OUT);
    input IN2, IN1, OUT;
    or 11 ( OUT, W3, W4 );
    and 12 ( W3, IN1, W1 );
endmodule
```



//blog.csust.net/yxswhy

# 教材与参考资料

- ❖ 不指定教材
- ❖ 参考资料
  - 数字系统设计与Verilog HDL, 第7版, 王金明著
  - Verilog HDL数字系统设计入门与应用实例, 王秀琴等编
  - EDA技术与Verilog设计, 第二版, 王金明著
  - Verilog编程艺术, 魏家明 著
  - Digital Design Principles and Practices,  
JohnF. Wakerly, Pearson Education

# 课程评价

## ❖ 成绩评定

- 平时考勤: 30%
- 实验成绩: 30%
- 期末考试: 40%

## ❖ 平时成绩

- 考勤: **不得迟到、旷课!**
- 作业: 随堂小测、课后作业

## ❖ 实验成绩

- 初定8次上机实验

## ❖ 期末成绩

- 闭卷考试



# 授课教师

- ❖ 贾小涛
- ❖ 办公地址：新主楼A916/第一馆203
- ❖ 邮箱：[jiaxt@buaa.edu.cn](mailto:jiaxt@buaa.edu.cn)
- ❖ 2011—2016，清华大学 计算机系
  - EDA算法：超大规模集成电路布局、布线算法
- ❖ 2016—， 北京航空航天大学 微电子学院
  - 贝叶斯深度学习加速器
  - 贝叶斯深度学习系统
  - 基于STT-MRAM的存内计算架构
  - EDA算法设计

# 助教

## ❖ 成镇

- [chengzhen@buaa.edu.cn](mailto:chengzhen@buaa.edu.cn)
- 具有完整的芯片设计与流片经验
- 第一馆
- 课程二维码



# 数字系统设计



课程号: B3I493310    开课学期: 2020-2021学年秋季

# 数字系统设计

北京航空航天大学  
微电子学院  
贾小涛



# 第一章 概述

- 一、数字系统概述
- 二、数字系统设计方法学
- 三、数字系统设计自动化



# 第一章 概述

- 一、数字系统概述
- 二、数字系统设计方法学
- 三、数字系统设计自动化

# 数字系统概述

微  
处  
理  
器

## ■ 数字系统涉及生活的方方面面



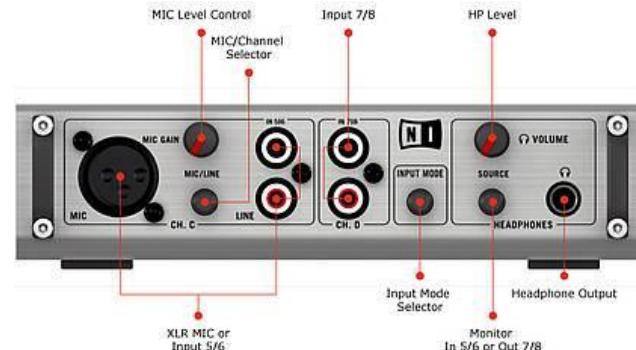
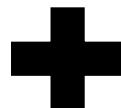
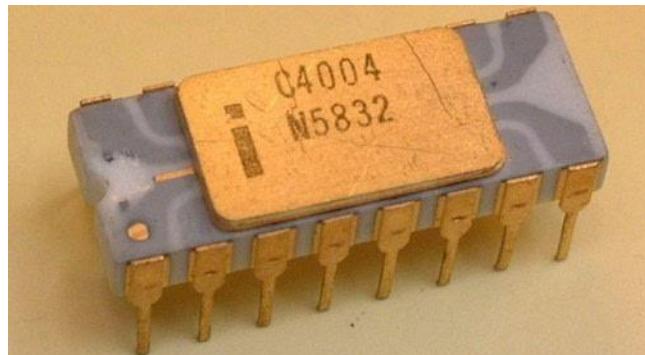
- 处理器
  - CPU, DSP, Controllers
- 存储芯片
  - RAM, ROM, EEPROM
- 模拟芯片
  - Mobile communication, audio/video processing
- 可编程器件
  - PLA, FPGA
- 嵌入式系统
  - Used in cars, factories
  - Network cards
- System-on-chip (SoC)



# 数字系统概述

## ■ 数字系统的基本组成

数字  
系统



数字系统可以看做是一个微处理器外加交互接口



算术逻辑单元  
存储单元  
控制单元

硅集成  
→

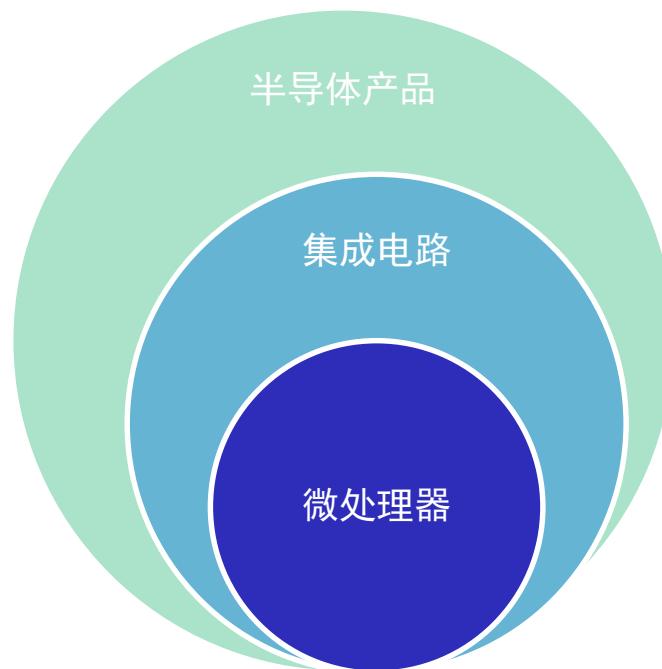
集成电路  
Integrated Circuit



# 数字系统概述

## ■ 微处理器

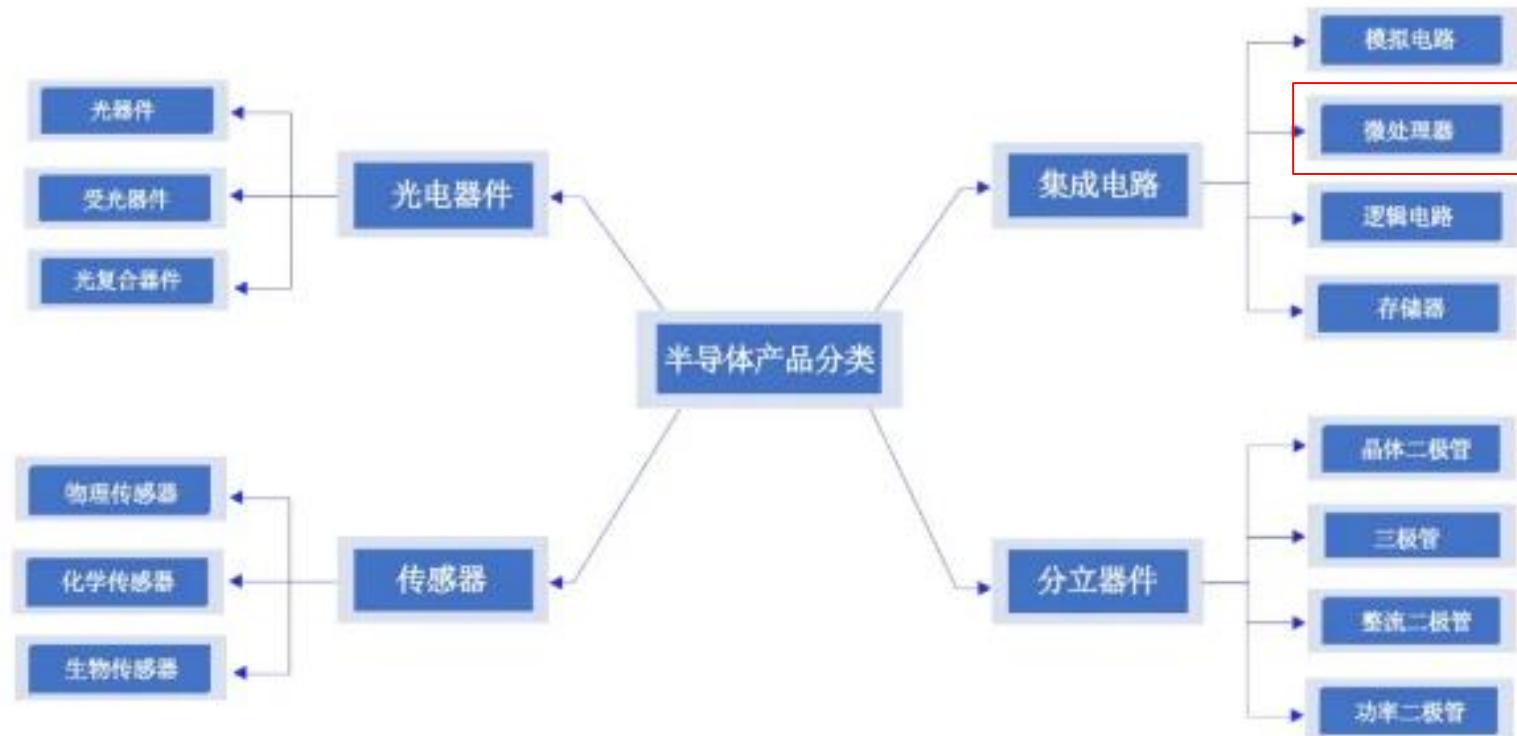
微  
处  
理  
器



# 数字系统概述

## ■ 微处理器

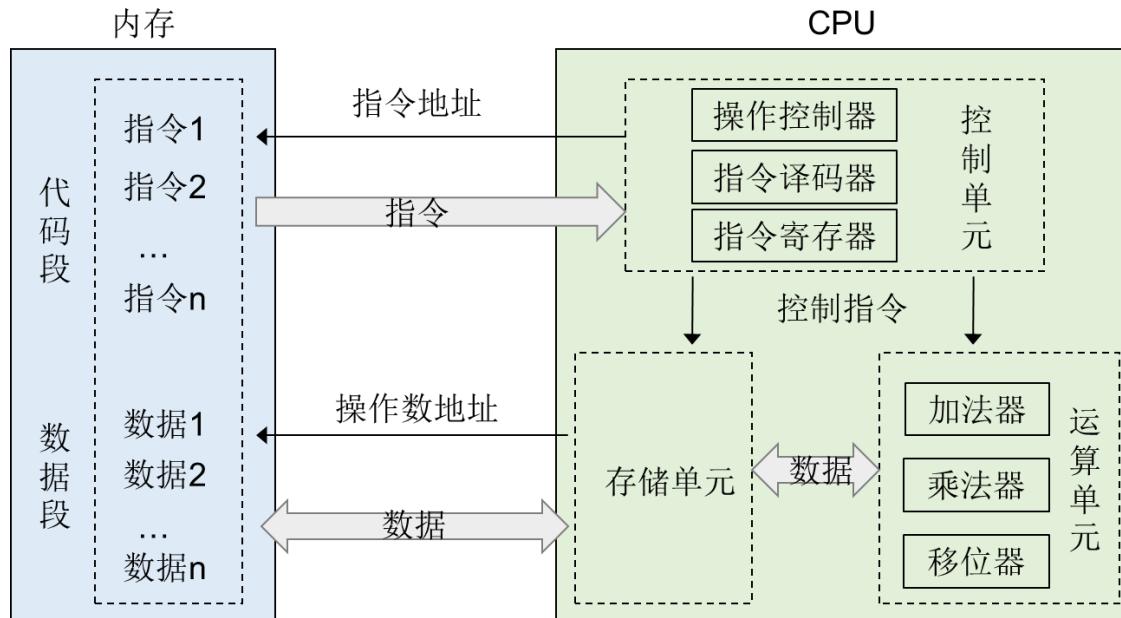
微处  
理器



# 数字系统概述

## ■ 常见微处理器

### — 中央处理器 (Central Processing Unit, CPU)



CPU架构与工作原理

intel®

AMD Smarter Choice

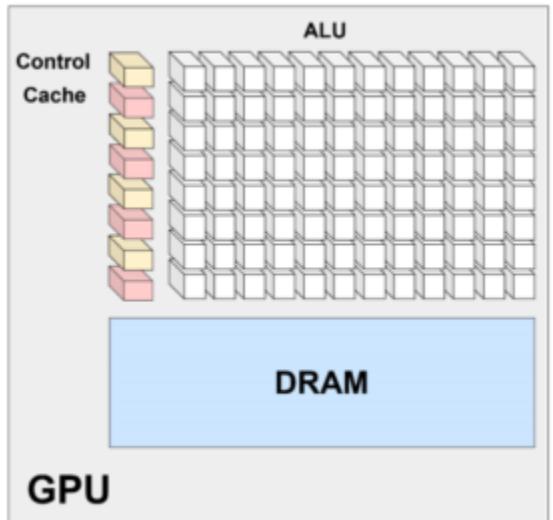
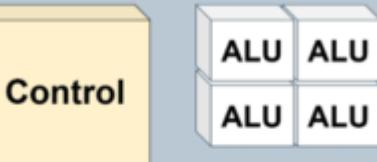
LOONGSON 龙芯

中科曙光  
Sugon

# 数字系统概述

## ■ 常见微处理器

### — 图形处理器（Graphics Processing Unit, GPU）



CPU

CPU和GPU的构成区别

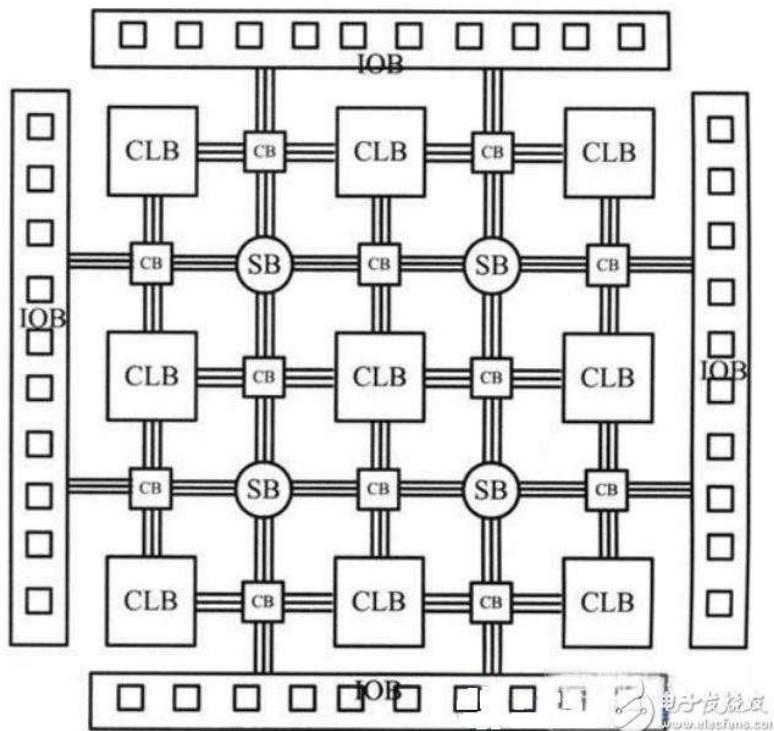




# 数字系统概述

## ■ 常见微处理器

- 现场可编程逻辑门阵列（Field Programmable Gate Array，FPGA）



XILINX®

ALTERA®

now part of Intel

京微齐力

LOGIC

FUDAN  
MICRO

复旦微电子集团



# 数字系统概述

微  
处  
理  
器

## ■ 常见微处理器

- 专用处理器（Application Specific Integrated Circuit, ASIC）

➤ 应特定用户要求和特定电子系统的需要而设计、制造的集成电路

射频芯片



矿机芯片



Canaan

BITMAIN

人工智能芯片

Cambricon  
寒 武 纪

HISILICON

Enflame  
燧 原 科 技

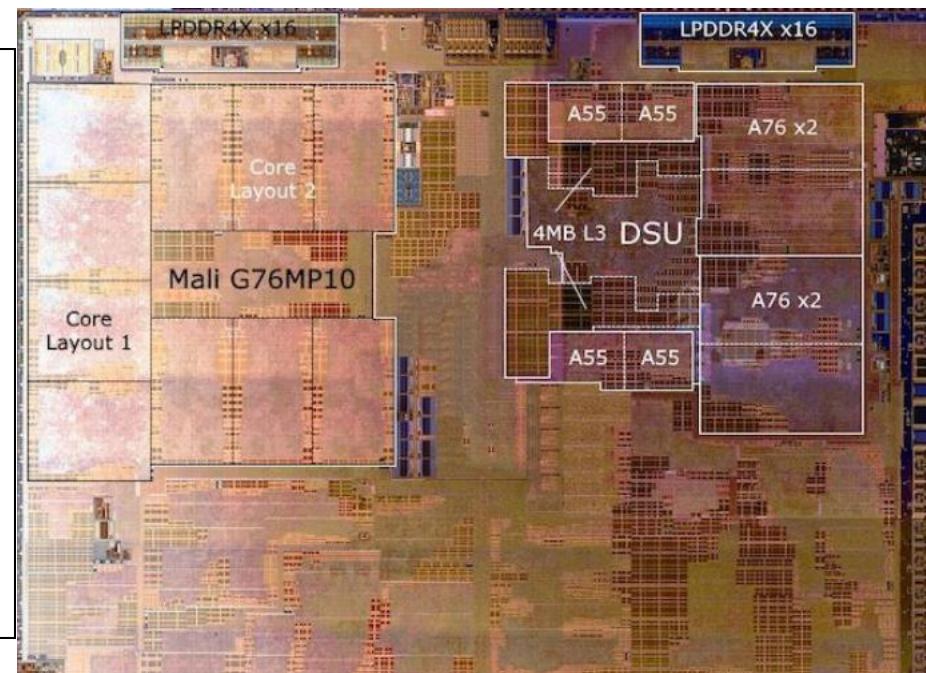
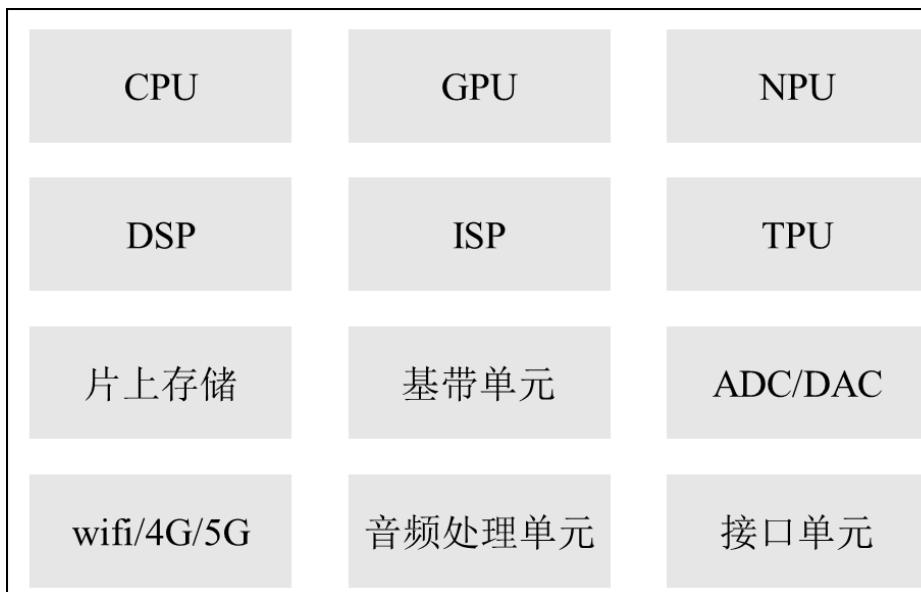


# 数字系统概述

微处理器

## ■ 其他微处理器

- DSP 数字信号处理器
- ISP 图像信号处理器
- MCU 微控制器
- SoC 系统级芯片/片上系统





# 数字系统概述

逻辑  
电路

## ■ 数字逻辑电路

- 是微处理器实现数字信号逻辑运算的电路

与  
AND

A	B	$Q$
0	0	0
0	1	0
1	0	0
1	1	1

或  
OR

A	B	$Q$
0	0	0
0	1	1
1	0	1
1	1	1

非  
NOT

A	$Q$
0	1
1	0

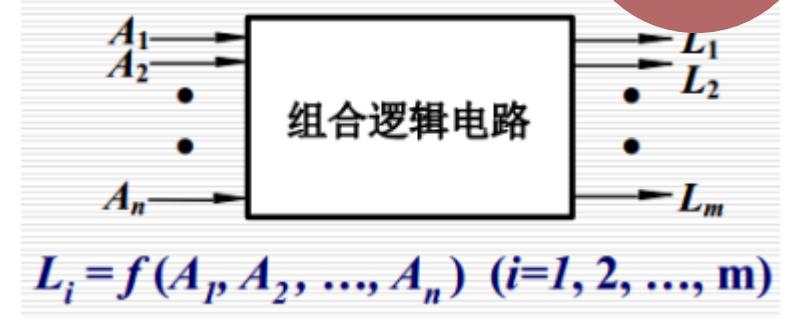
- 处理与传递离散信号（数字信号）
- 组合逻辑电路
- 时序逻辑电路

# 数字系统概述

## ■ 组合逻辑电路

结构特征：

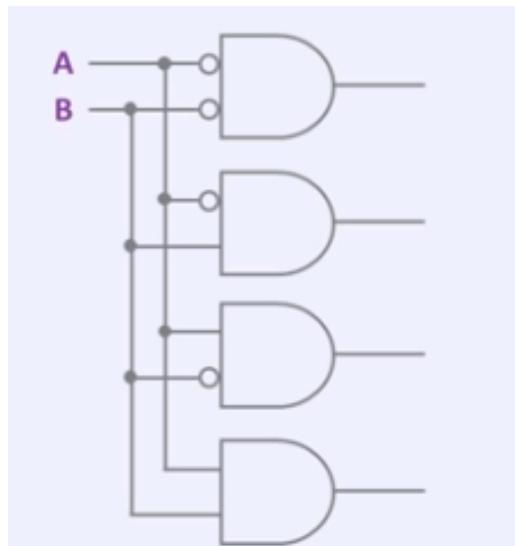
1. 输入输出之间没有反馈延迟通路
2. 不含存储单元



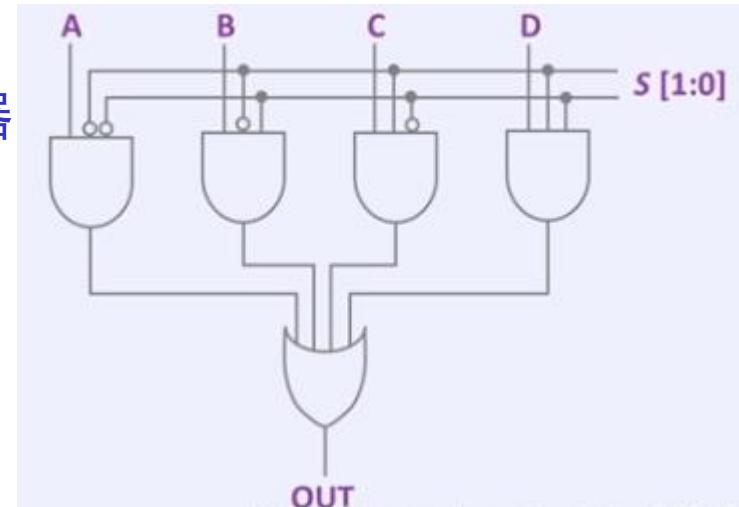
工作特征：

任意时刻的输出只取决于该时刻的输入，与电路原有的状态无关

2-4译码器



多路  
选择器



# 数字系统概述

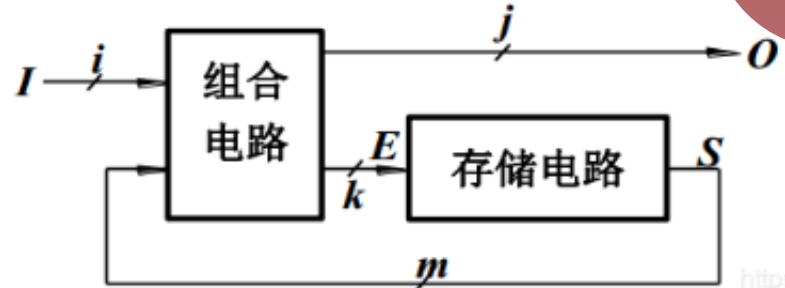
## ■ 时序逻辑电路

### 结构特征：

1. 电路由组合电路和存储电路组成
2. 存在反馈电路

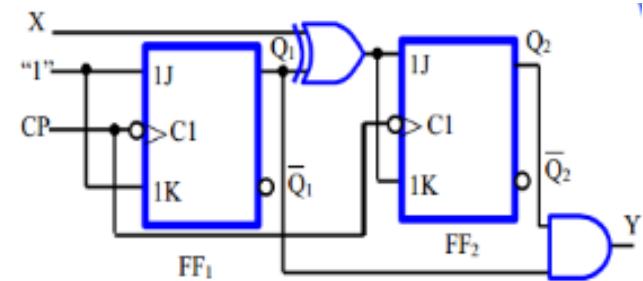
### 工作特征：

电路输出即与该时刻的输入有关，也与电路原有的状态有关



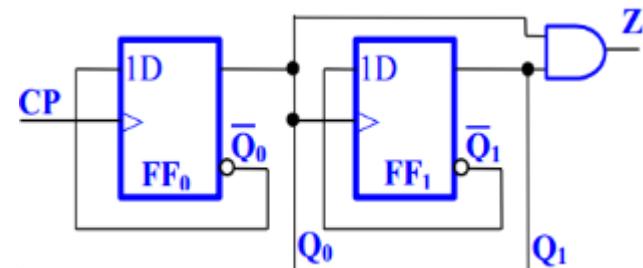
### 同步时序电路：

存储电路里所有触发器有一个统一的时钟源，它们的状态在同一时刻更新



### 异步时序电路：

没有统一的时钟脉冲，电路的状态更新不在同一时刻发生





# 数字系统概述

逻辑  
门

## ■ 逻辑门电路

- 数字逻辑电路的基本单元。执行“或”、“与”、“非”、“或非”、“与非”等逻辑运算的电路
- 任何复杂的逻辑电路都可由基本逻辑门组成

与  
AND

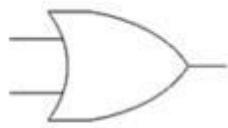
A	B	$Q$
0	0	0
0	1	0
1	0	0
1	1	1



AND

或  
OR

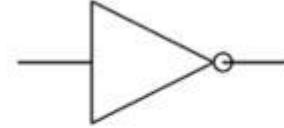
A	B	$Q$
0	0	0
0	1	1
1	0	1
1	1	1



OR

非  
NOT

A	$Q$
0	1
1	0



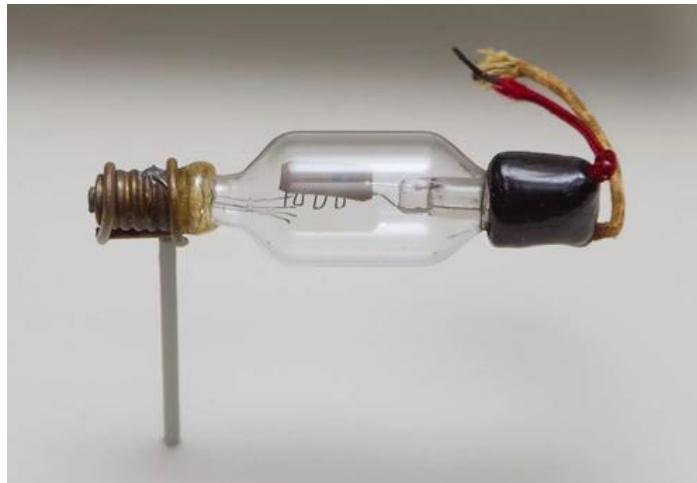
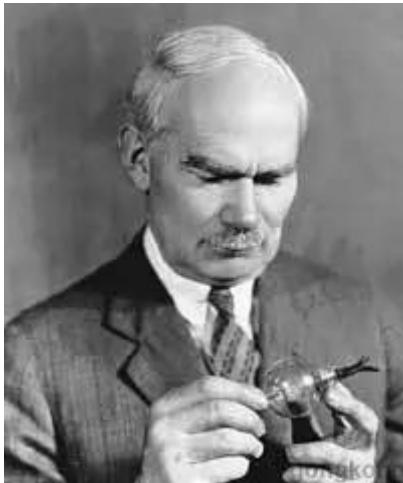
NOT

# 数字系统概述

## ■ 逻辑器件

逻辑  
器件

真空电子管



德福雷斯特在对二极管的研究基础上，发明了真空三极管。  
**耗电量巨大  
寿命短**

晶体管



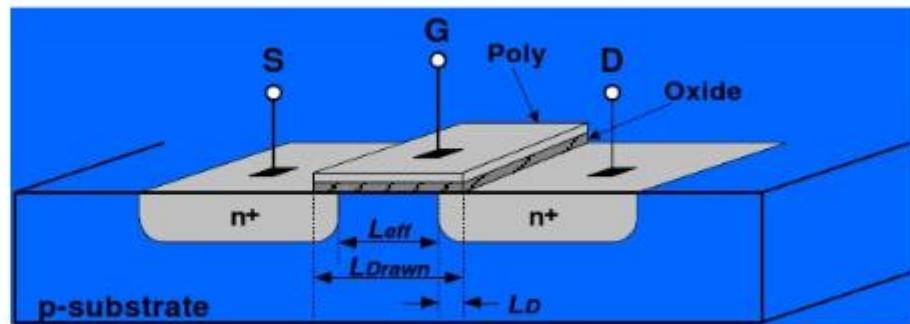
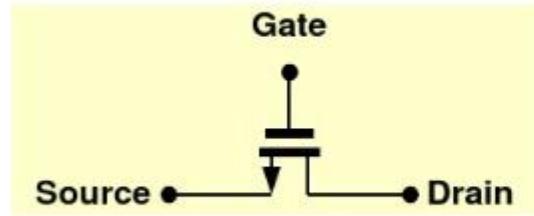
相较于真空三极管，晶体管在体积、寿命、制作工艺上都有极大的优势



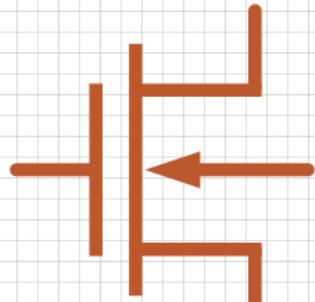
# 数字电路概述

逻辑  
器件

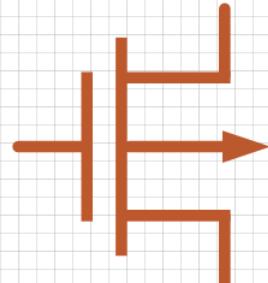
## ■ 金属氧化物半导体场效应管（Metal Oxide Semiconductor Field Effect Transistor, MOSFET）



PMOS



NMOS



通过调节栅极电压，控制晶体管的开关

晶体管

数字  
开关

开

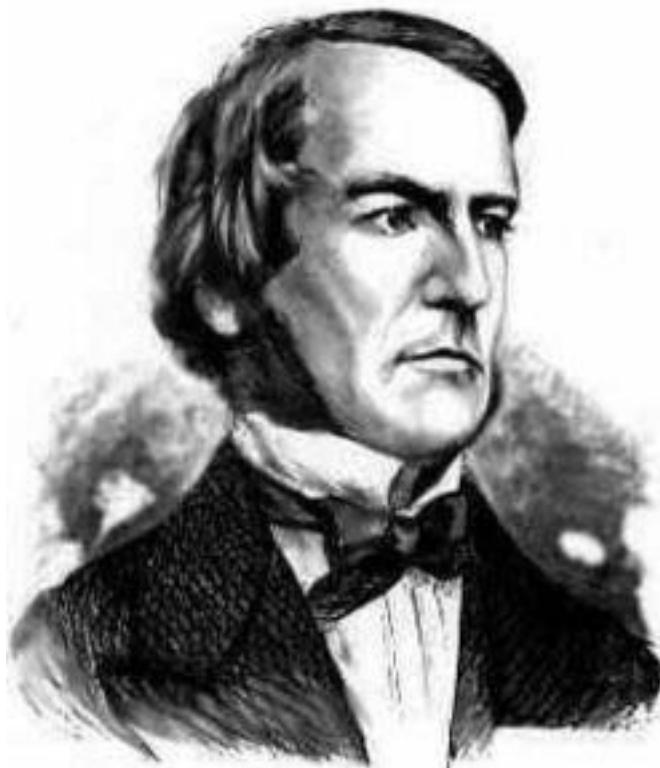
关



# 数字电路概述

## ■ 逻辑

逻辑



乔治·布尔

逻辑不仅是哲学，还是数学

任何事情都能用数学形式表达其处理过程

可以用0和1表达逻辑

可以用0和1表征各种不同任务



# 数字电路概述

## ■ 逻辑

逻辑



香农

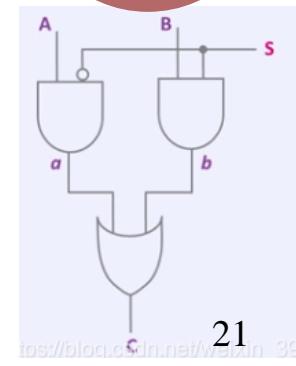
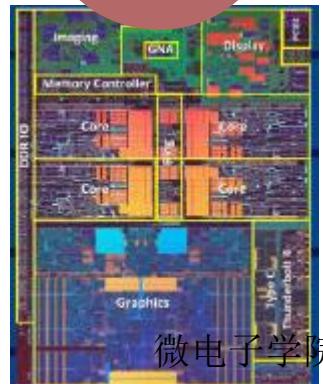
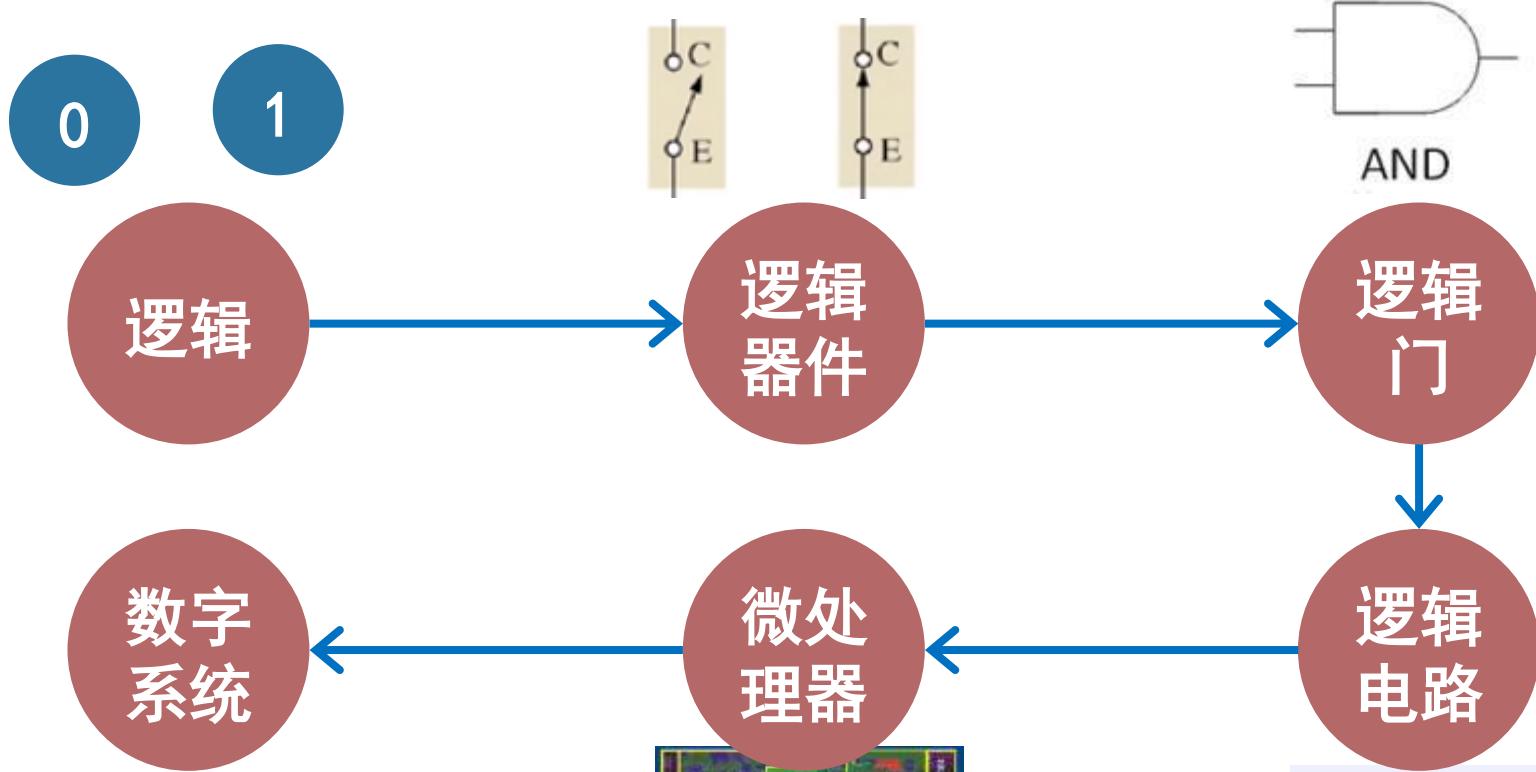
逻辑可以用逻辑电路来实现，  
也就是用电子管来实现

用电子管的开关实现0和1



# 数字系统概述

## ■ 从晶体管到数字系统





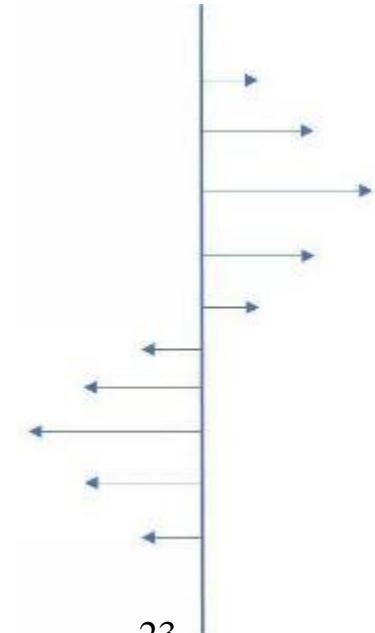
# 数字系统概述

## ■ 数字信号

- 什么是数字信号
- 为什么使用用数字信号
- 如何获得数字信号



# 数字系统概述





# 数字系统概述

## ■ 数字信号与模拟信号的区别 — 图像

胶片相机



感光、冲洗



数码相机



像素点GRB采样





# 数字系统概述

## ■ 数字信号与模拟信号的区别 — 时间



时间不确定、可取连续时间



时间确定、时间精度有限



# 数字系统概述

## ■ 数字信号与模拟信号的区别 — 电视



# 数字系统概述

## ■ 数字信号与模拟信号的区别





# 数字系统概述

## ■ 数字信号与模拟信号的区别

模拟信号具有连续性：

1. 时域连续
2. 取值连续

实数

数字信号具有离散型：

1. 时域为离散的
2. 值域为一个有限集合

整数

**自然界中绝大部分信号都是模拟信号**

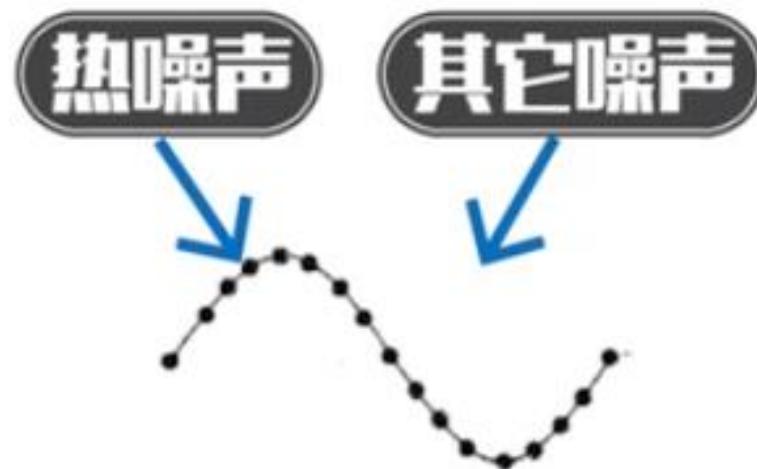
# 数字系统概述

## ■ 为什么要采用数字信号

噪声多

噪声难消除

噪声积累





# 数字系统概述

## ■ 为什么要采用数字信号

噪声多

噪声难消除

噪声积累





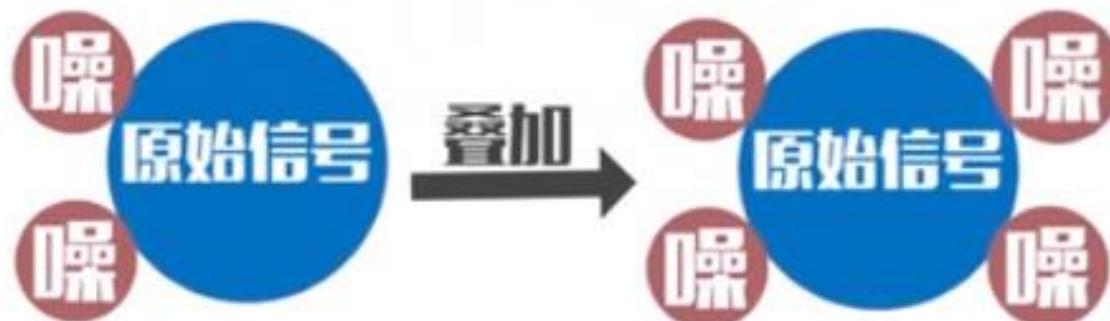
# 数字系统概述

## ■ 为什么要采用数字信号

噪声多

噪声难消除

噪声积累





# 数字系统概述

## 模拟信号

- 来自于物理世界
- 热波动噪声
- 噪声易累积

## 数字信号

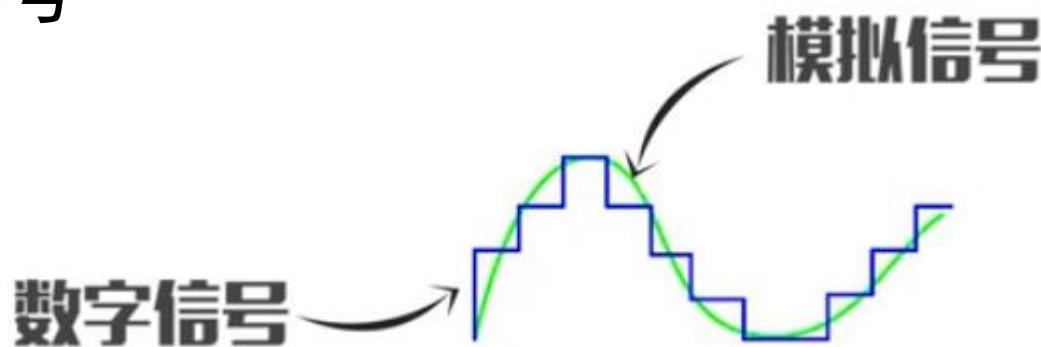
- 基于布尔方程
- 采样误差噪声
- 噪声不会累积



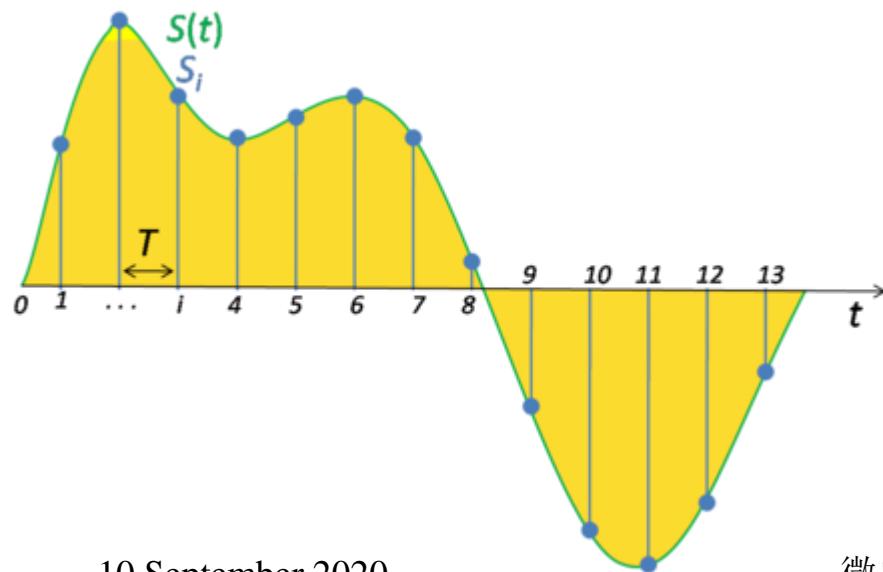
# 数字系统概述

## ■ 如何获取数字信号

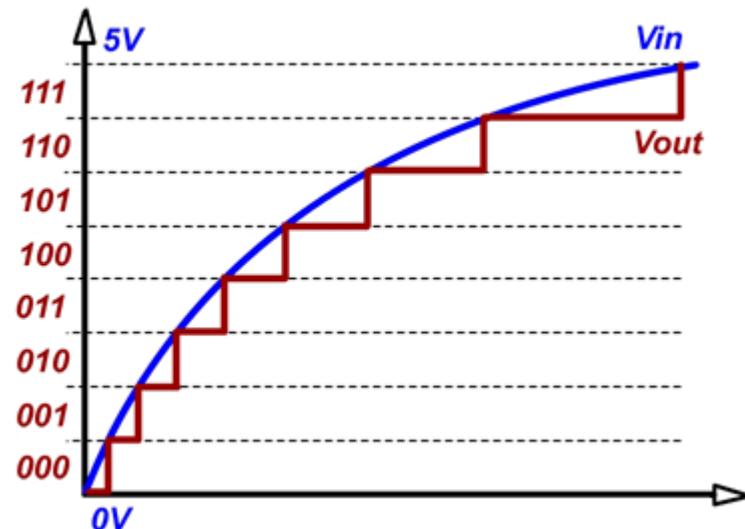
采样



采样频率



采样精度





# 第一章 概述

- 一、数字系统概述
- 二、数字系统设计方法学
- 三、数字系统设计自动化

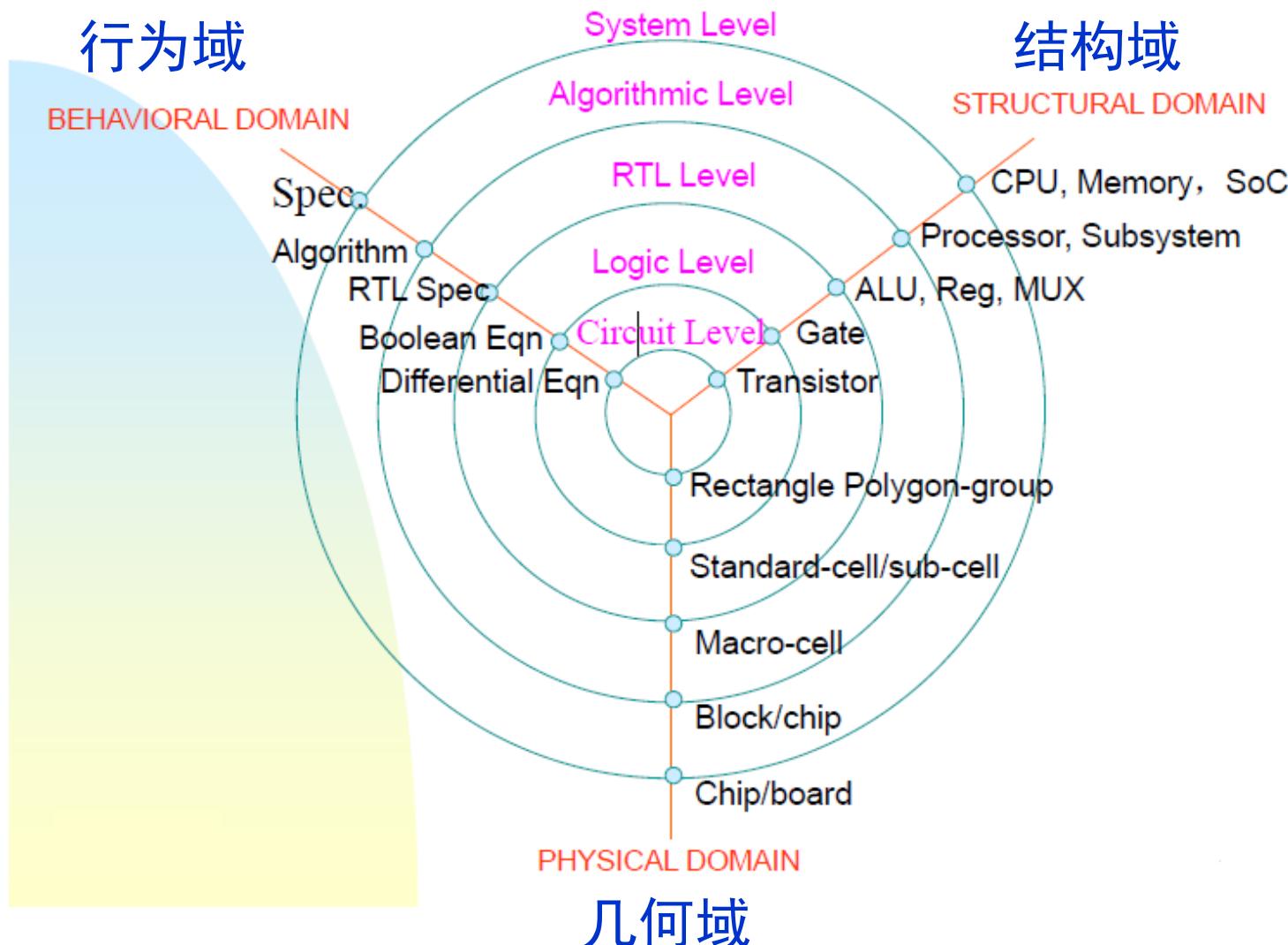


# 数字电路设计方法学

- 数字电路设计的三个域
  - 行为域、结构域、几何域
- 数字电路的层次化设计
  - 自顶向下
  - 自底向上
- 数字电路的实现技术
  - 全定制
  - 半定制
  - 可编程器件



# 数字电路设计方法学

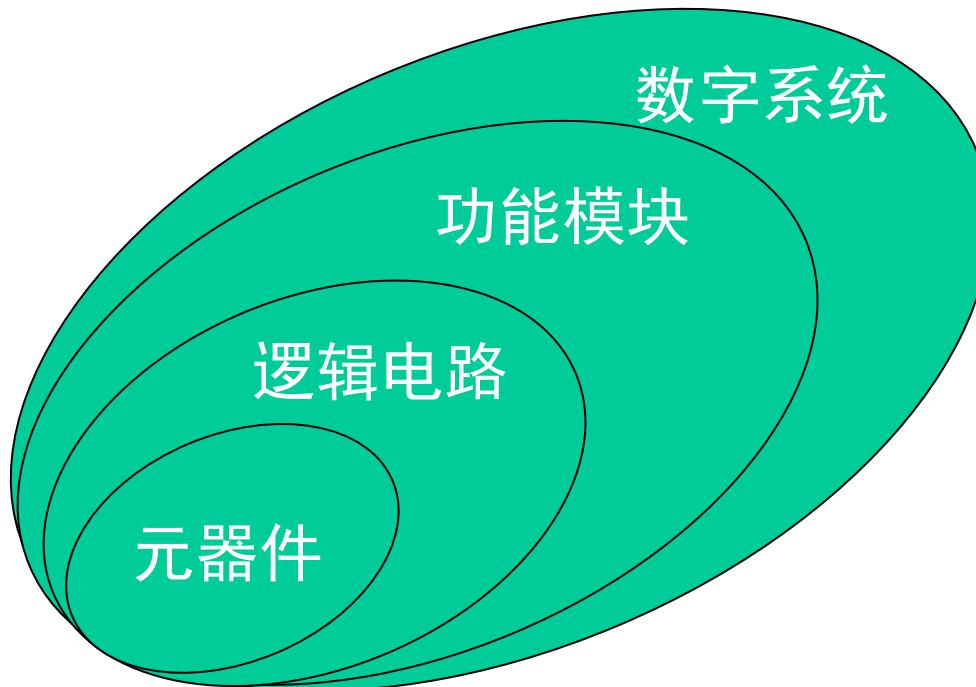




# 数字电路设计方法学

## ■ 层次化设计

- 自底向上（Bottom-Up设计）



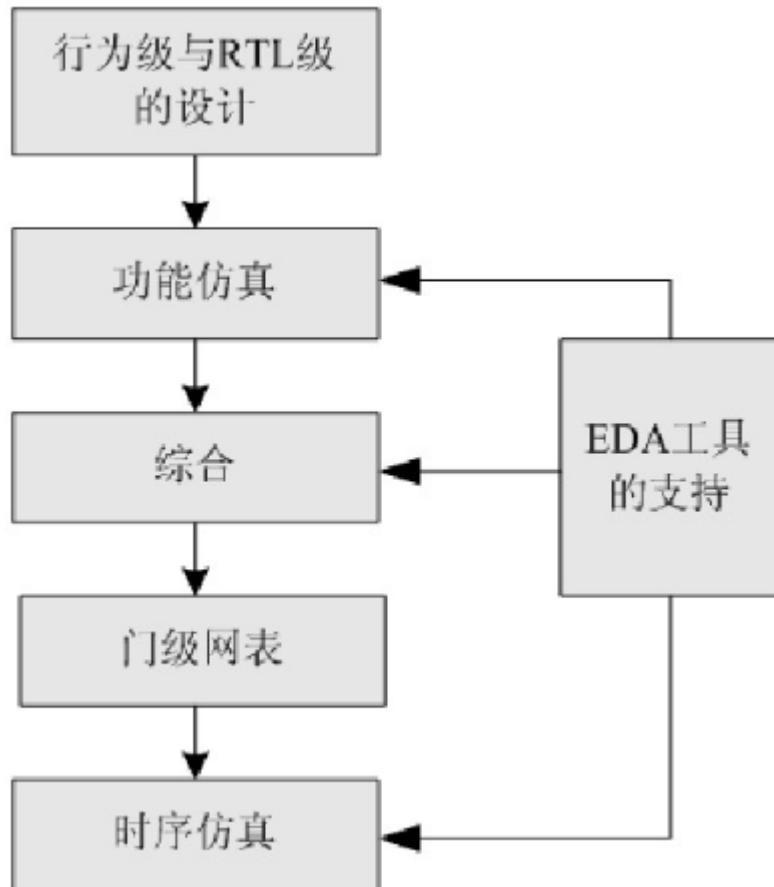
- 缺点：缺乏全局规划、迭代优化难度大



# 数字电路设计方法学

## ■ 层次式设计

### — 自顶向下 (Top-Down设计)

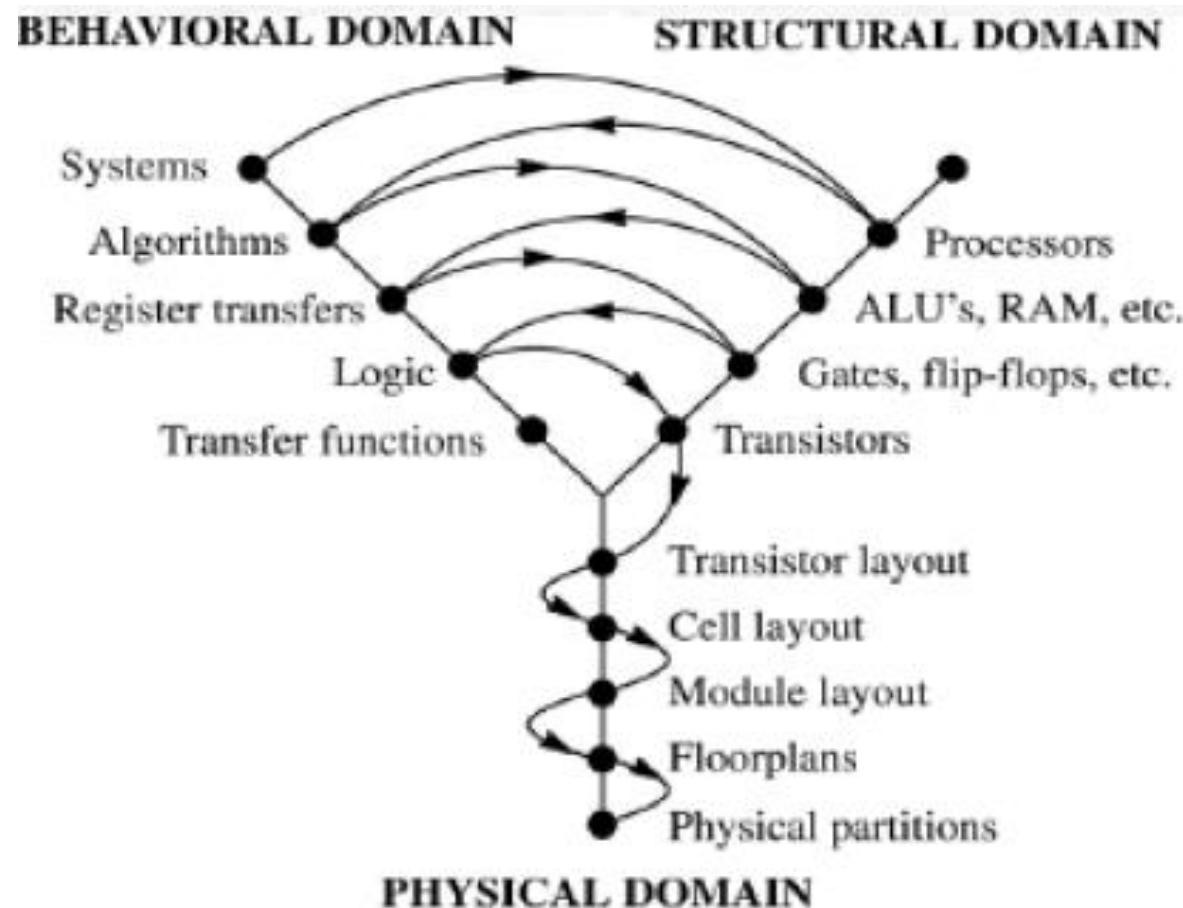


自顶向下方法的设计通常需要经过“设计—验证—修改设计—再验证”的过程，通过不断的迭代优化与验证，获得满足性能与功能要求的结果



# 数字系统设计方法学

## ■ 混合设计模式



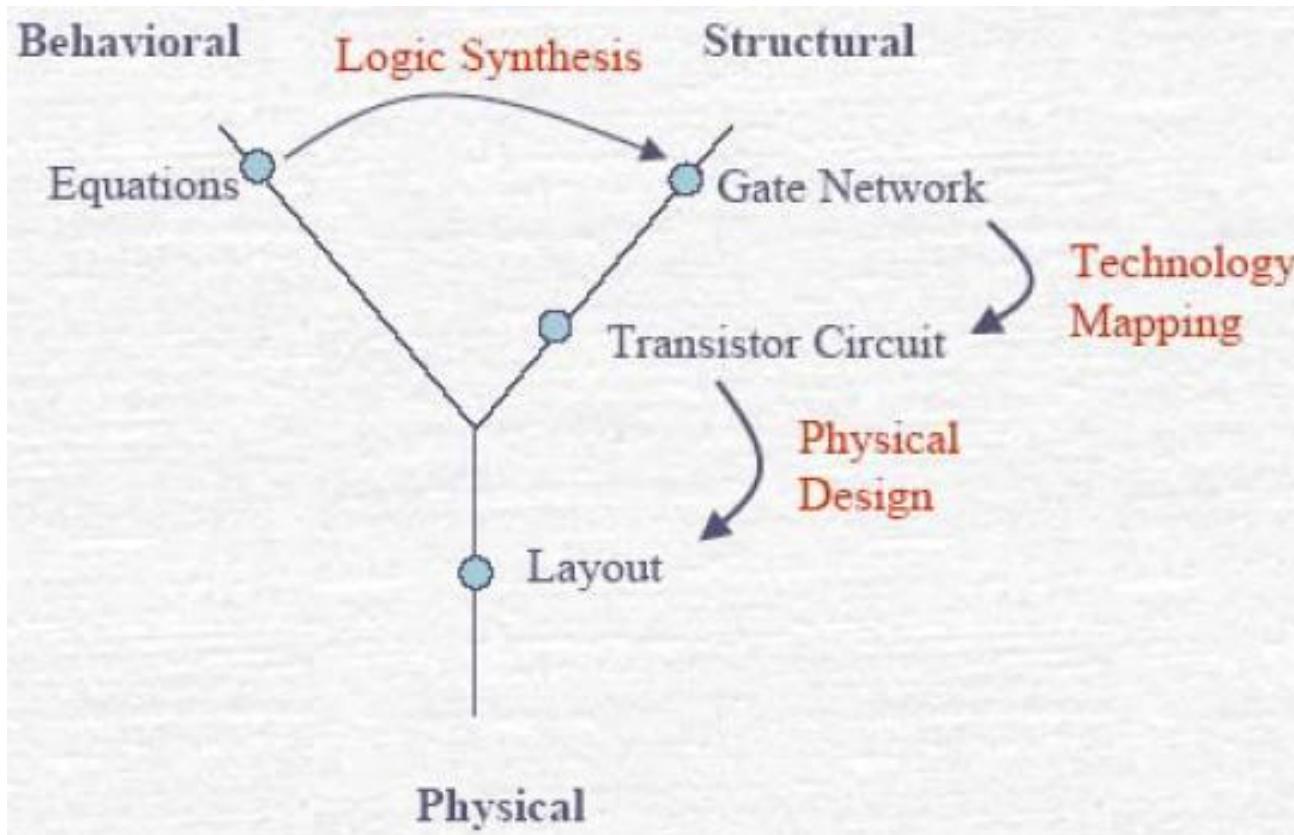
Top-Down  
Structural design

Bottom-Up  
realization



# 数字系统设计方法学

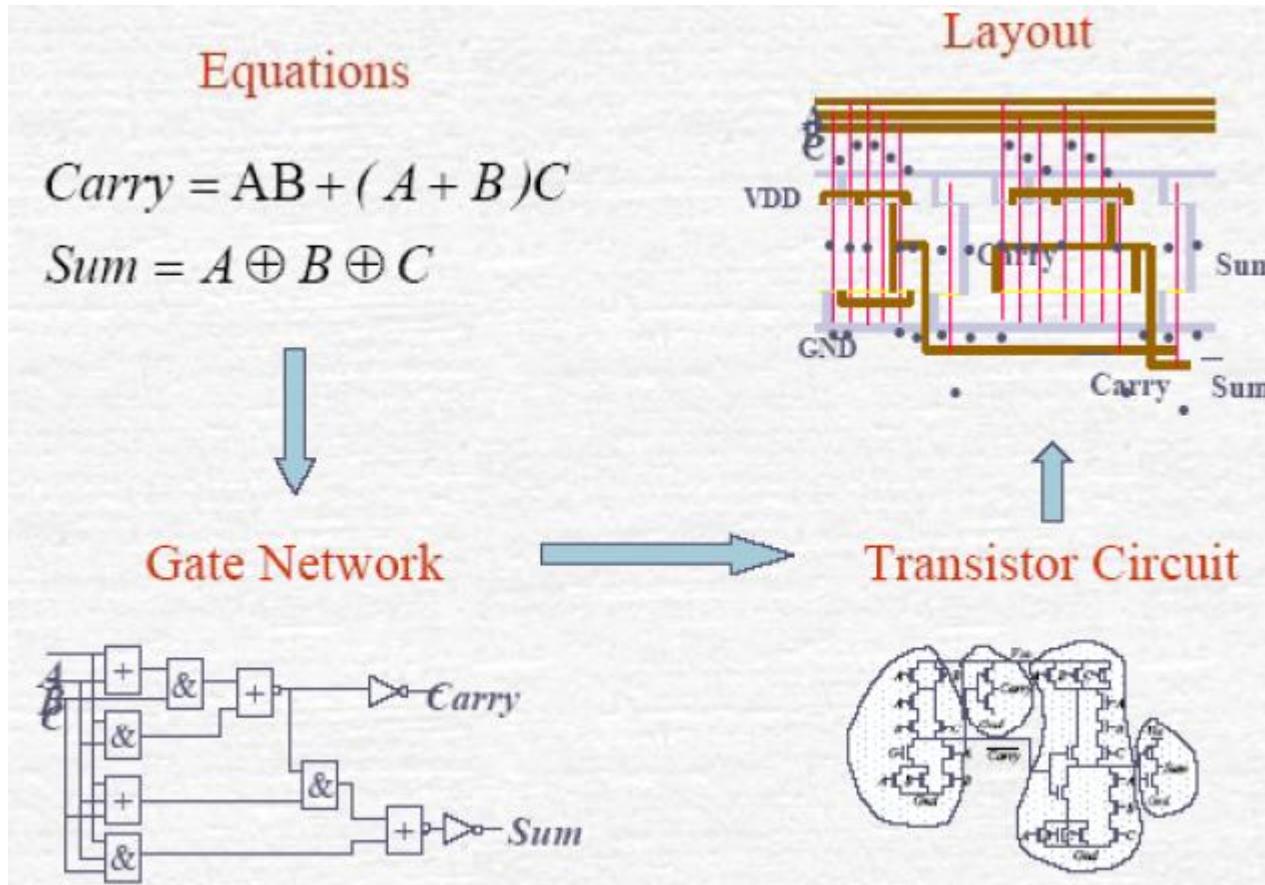
## ■ 以全加器为例





# 数字系统设计方法学

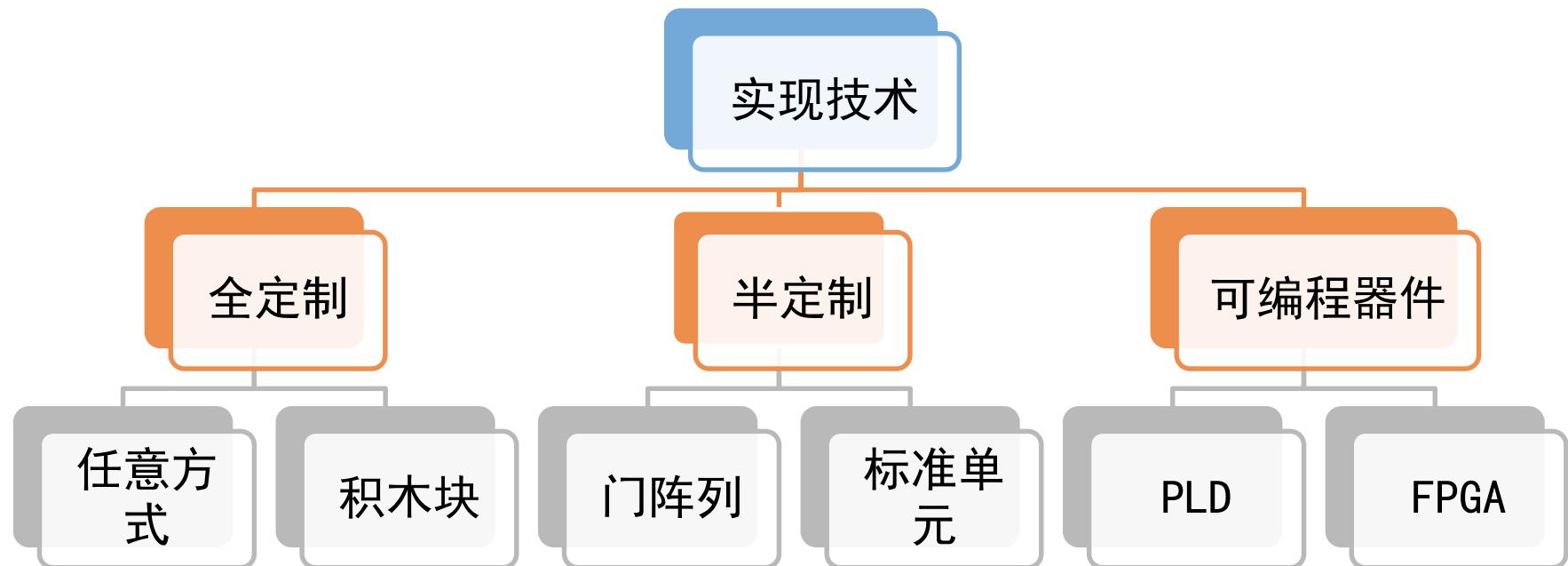
## ■ 以全加器为例





# 数字系统设计方法学

## ■ 数字系统的实现技术





# 数字系统设计方法

## ■ 全定制

### — “搭积木”式

- 标准逻辑器件+外围电路
- 性能高
- 设计成本高
- 几乎没有灵活性
- 设计复杂度高

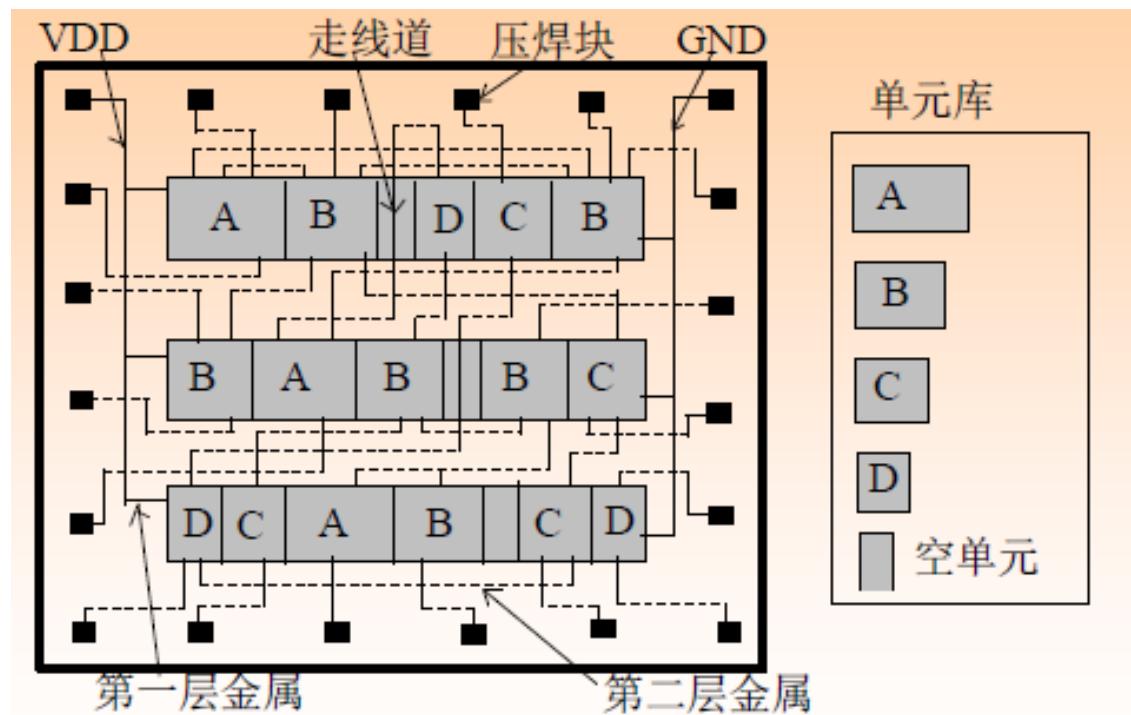


# 数字系统设计方法

## ■ 半定制

### — 标准单元模式

- 设计灵活
- 标准单元库有助于提高布图效率
- 自动化程度高
- 设计周期短
- ASIC设计广泛采用



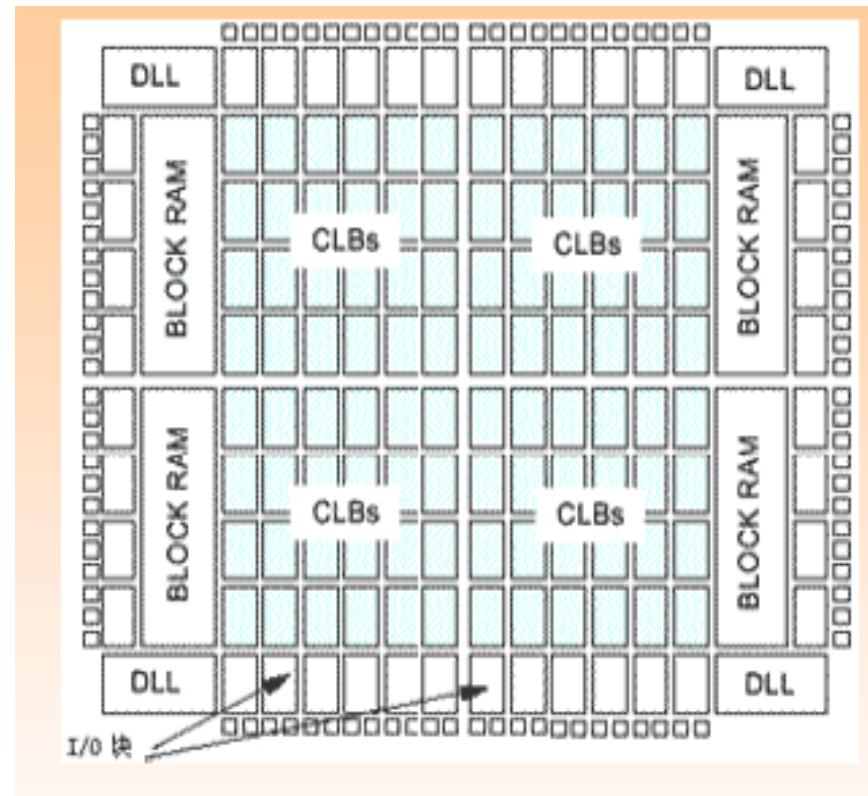


# 数字系统设计方法学

## ■ 可编程器件

### — FPGA

- 设计灵活度高，为用户提供了参与系统与电路设计的可能性
- 设计周期短、上市快
  
- 功耗大
- 成本高
- 速度慢



Xilinx Spartan-II 芯片内部结构



# 第一章 概述

- 一、数字系统概述
- 二、数字系统设计方法学
- 三、数字系统设计自动化



# 数字系统设计自动化

## ■ 如何设计芯片？

手工设计→计算机辅助→电子设计自动化

**EDA**

Electronic Design Automation



# 数字系统设计自动化

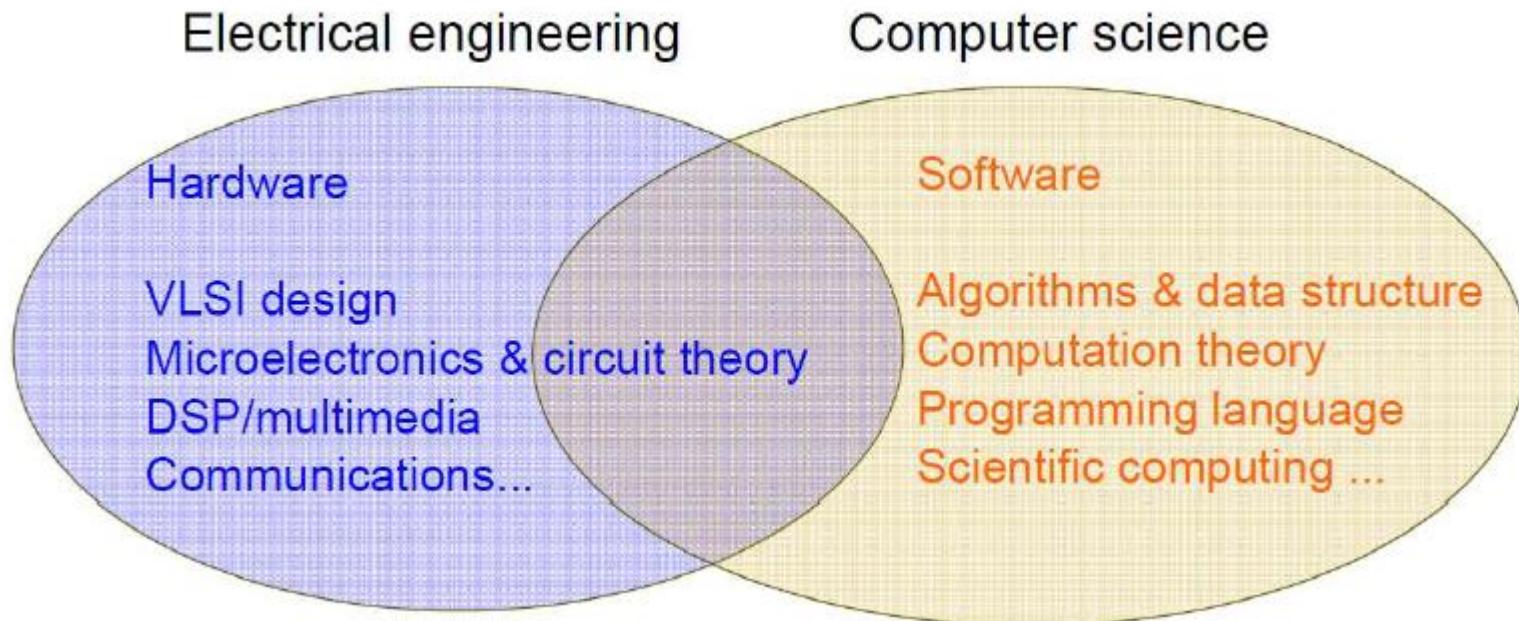
## ■ EDA: Electronic Design Automation 电子设计自动化

- 利用计算机完成集成电路的设计工作
- 以计算机和微电子技术为先导，汇集计算机图形学、拓扑学、逻辑学、计算数学等多种计算机应用学科最新成果的先进技术
- 代替设计者完成逻辑综合、布局布线、仿真验证等工作



# 数字系统设计自动化

## ■ EDA: Where HW and SW meet each other





# 数字系统设计自动化

## ■ EDA技术发展的三个阶段

### — CAD阶段

- 20世纪60年代中期~20世纪80年代初期
- 计算机辅助版图编辑、PCB设计、电路模拟
- 常用工具：Tango、SPICE

### — CAE（Computer Aided Engineering）阶段

- 20世纪80年代初期~20世纪90年代初期
- 原理图输入、逻辑仿真、自动布局、功能模拟、分析验证
- 常用工具：Mentor Graphics、Valid Daisy



# 数字系统设计自动化

## ■ EDA技术发展的三个阶段

### — EDA阶段

- 20世纪90年代以来
- 硬件描述语言、系统级仿真与综合技术
- 自顶向下的设计理念
- 精力主要集中于方案创新与架构创新
- 常用软件：Cadence、Design Compiler、IC Compiler、Quartus



# 数字系统设计自动化



- EDA被定位为集成电路产业链龙头
  - 2000年国务院出台了8号文件
- 推动了封装测试行业从二维转向三维
- PCB板级系统的硅上互联
- 维持“摩尔定律”的语言
- 时间摩尔到空间摩尔的转变



# 数字系统设计自动化

## ■ EDA工具分类

### — 电子电路设计

- HSPICE、SPECTRE

### — PCB设计

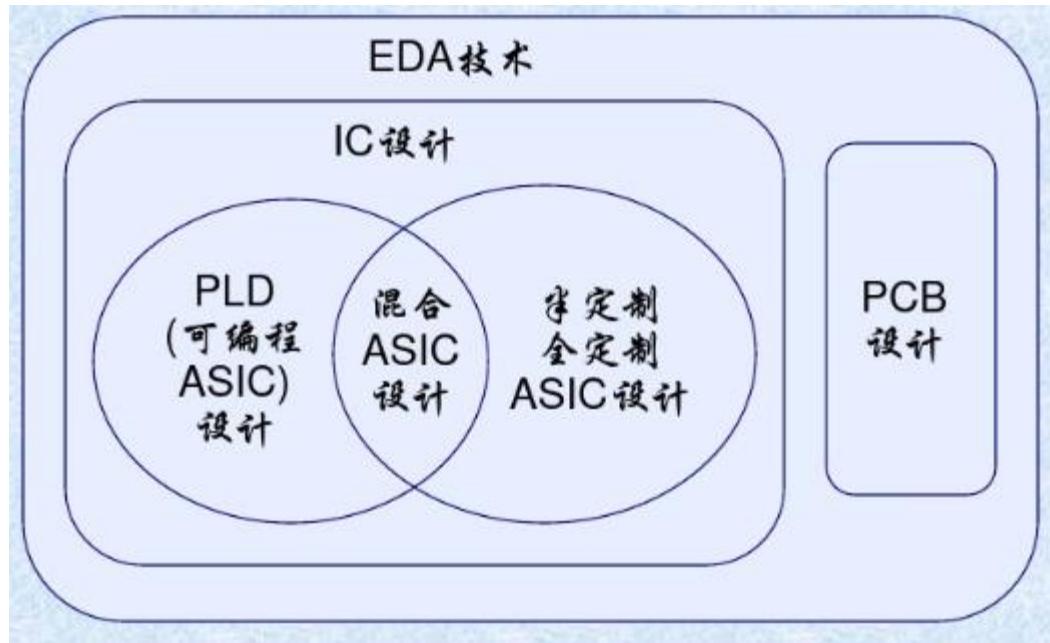
- Protel

### — PLD设计

- Quartus II、ISE

### — IC设计

- Modelsim、Design Compiler, Encounter、IC compiler

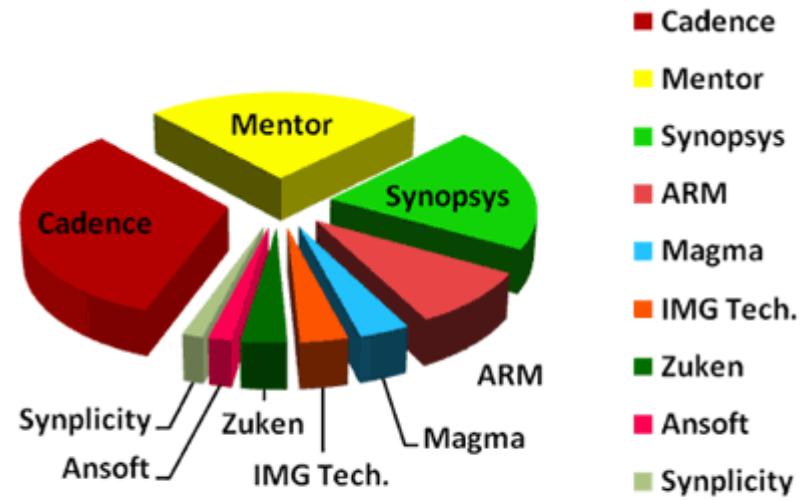




# 数字系统设计自动化

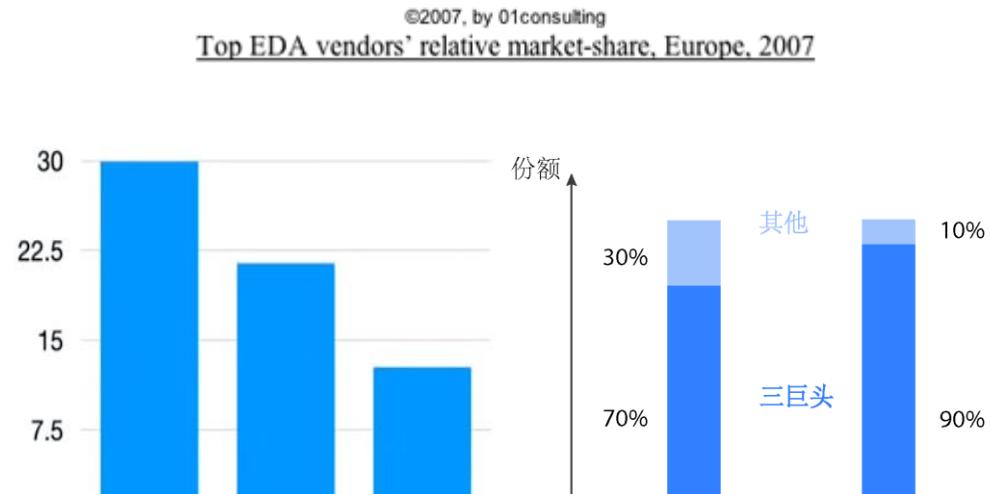
## ■ IC EDA 三巨头

- Cadence
- Synopsys
- Mentor



## ■ 2019年全球行业产值

- 90亿美元
- 三巨头垄断全球70%
- 三巨头垄断国内90%



90亿元的产值控制全球5000亿美元的半导体市场



# 数字系统设计自动化

## ■ 国内芯片设计厂商严重依赖EDA三巨头

华为	DxDesigner (前端)	Mentor Graphics
	HAPS (验证)	Synopsys
	Allegro (PCB)	Cadence
中兴	Allegro (前端到后端)	Cadence
	ExpeditionPCB (布线)	Mentor Graphics
联想	Allegro	Cadence
联想	PADS	Mentor Graphics
朗科	PADS	Mentor Graphics
	Orcad	Cadence
神达电脑	Allegro	Cadence
英业达	Allegro	Cadence
威盛	Allegro	Cadence
天弘电子	ExpeditionPCB (WG)	Mentor Graphics
	Allegro	Cadence
宏碁	Allegro	Cadence
	BoardstationPCB(EN)	Mentor Graphics

华硕	Allegro	Cadence
创维	PADS	Mentor Graphics
TCL	PADS	Mentor Graphics
迈瑞医疗	PADS	Mentor Graphics
清华同方	PADS	Mentor Graphics
长城	Allegro	Cadence
海尔	CR5000	ZUKEN
海信	CR5000	ZUKEN
新北洋	CR5000	ZUKEN
中海油服	AD	Altium
博士力士乐	AD	Altium
中芯国际	AD	Altium
南京南瑞	AD	Altium
南京三宝	AD	Altium
纬创资通	Allegro	Cadence
	BoardstationPCB(EN)	Mentor Graphics



# 数字系统设计自动化

## ■ EDA对半导体产业的重要性

- 谷歌断供→鸿蒙系统
- ARM断供→备胎转正、V8架构永久授权
- EDA断供→?



美国断供EDA工具,是为最后一招做准备,华为不可不防 设计

全球三大EDA工具欲断供华为,华为芯片设计该何去何从?

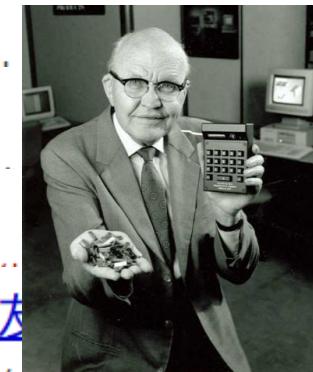
美国三大EDA巨头断供华为,华为芯片还能做下去吗?

三大EDA软件公司断供华为!本土EDA公司能否抗旗?-电子发烧友网

2019年6月9日 - 华为在缺少EDA公司支持的情况下,台积电遭遇美国更严格的审查。... 瞩科

Cadence、Mentor、Synopsys三大EDA软件公司,都已相继断供华为。华为在缺少EDA公...

电子发烧友网 - 百度快照



杰克·基尔比



# 数字系统设计自动化

## ■ 国产EDA产业发展

### — EDA熊猫系统

- 1986年开始研发，1993年问世
- 但未取得实质性成功



### — 华大九天

- 承载熊猫系统的技术
- 数模混合IC设计、后端优化、平板（FPD）全流程设计

### — 芯禾科技、广立微





# 数字系统设计自动化

## ■ 与国外差距显著

- 产品不够全，性能不够优，尤其是在数字电路方面
- 政策与研发的影响
- 与先进工艺结合的缺失
  - 与先进工艺接触的机会少，限制了我们的提高

- **人才** 我在该公司的10年中（1999年到2009年），当时几个产品在商业上还不太成功。不太成功的标志是：无法与主流EDA工具进行正面竞争，在功能和性能上有差距。当时国内EDA产业的环境与目前差别很大，当时的现状是：做出的EDA工具必须比主流EDA工具要好，并且要明显地好，才可能有市场空间。而目前国内EDA产业的现状是：首先解决EDA工具的有无问题，其次再解决好与更好的问题。相对来讲，目前国内EDA产业的需求对EDA公司的商业化成功更加友好。有时候，我甚至猜想：如果把今天国内EDA产业环境提前10年，也许我们当年开发EDA就不会那么痛苦了。
- **研发** 影响
- **市场**



# 数字系统设计自动化

## ■ “巴统”

- 对共产主义国家出口管制统筹委员会
- 总部在巴黎的美大使馆——巴黎统筹委员会



20世纪80年代之前，禁运EDA

1991年，国产熊猫EDA系统发布

1994年，取消对华EDA禁运

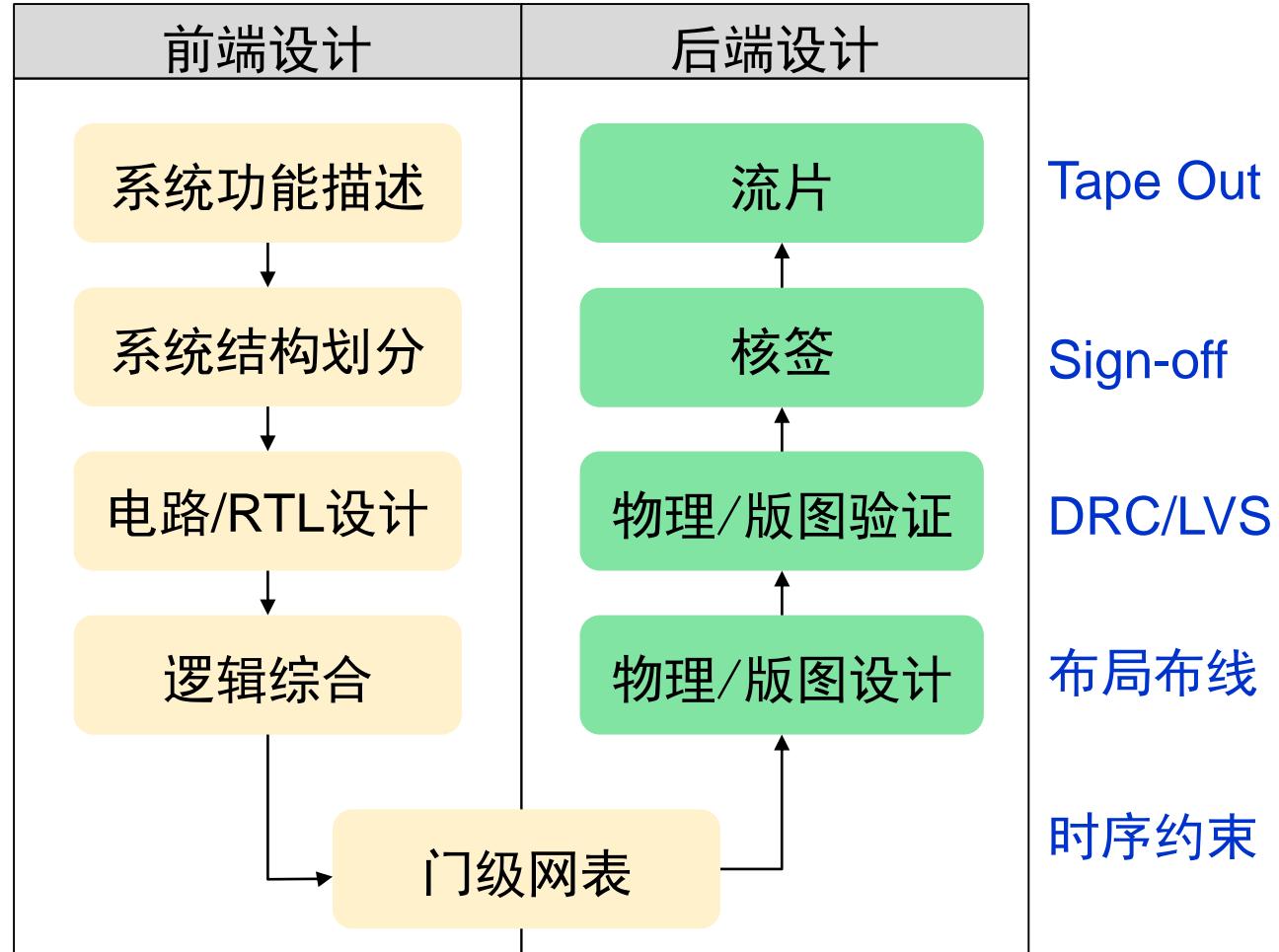
2008年，EDA软件再度起航



# 数字系统设计自动化

## ■ 数字系统（集成电路）设计流程

编写系统Spec文件  
ALU、Controller  
Verilog SystemC  
约束文件、库  
与或非门





# 数字系统设计自动化

- 系统功能描述
- 功能结构设计
  - 基于芯片功能及性能要求设计规划
- 电路设计
  - 基于硬件描述语言（Hardware Description Language, HDL）: VHDL, Verilog; RTL代码
  - 不再是手工画图
- 电路仿真
  - 功能验证、行为仿真



# 数字系统设计自动化

## ■ 逻辑综合

- 将RTL电路转换为门级网表（Netlist）
- 翻译： RTL → 逻辑库      映射： RTL → 器件库
- 包含门延迟信息                  网表是一种记录有逻辑门之间连接关系及门延时信息的文件

## ■ 门级仿真

- 比电路仿真更真实
- 考虑连接关系及门延时



# 数字系统设计自动化

## ■ 版图设计（物理设计）

- 包含多个阶段且各阶段联系紧密
- 决定芯片的最终设计方案

## ■ 静态时序分析（STA）

- 套用时序模型（Timing Model）
- 门延时：Gate Delay——查表法
- 线延时：Wire Delay——Elmore模型

## ■ 后仿真验证



# 数字系统设计自动化

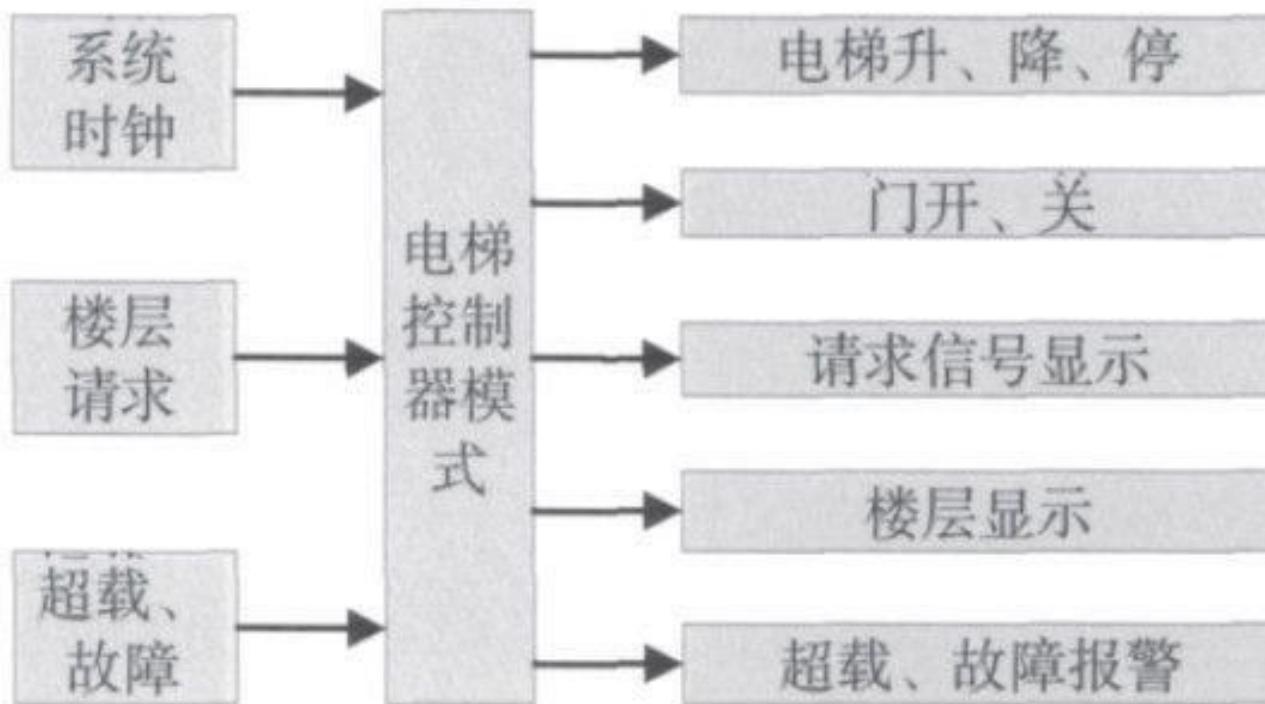
## ■ 数字系统设计层次

抽象层次	时序单位	基本单元	电路的功能描述
系统级	数据处理	进程及通信	自然语言描述系统功能或相互 通信的进程
前端			
后端			



# 数字系统设计自动化

## ■ 电梯的系统级描述





# 数字系统设计自动化

## ■ 数字系统设计层次

	抽象层次	时序单位	基本单元	电路的功能描述
前端	系统级	数据处理	进程及通信	自然语言描述系统功能或相互通信的进程
	行为级	运算步	运算的控制	行为有限状态机、数据流图、控制流图
后端				



# 数字系统设计自动化

## 电梯的行为级描述

现态	次态	转换条件
S0	S1	下一个时钟
S1	S1	up[pos]=1 or stop[pos]=1
S1	S2	S1下一个状态不为S1时
S2	S1	up[pos]=1 or stop[pos]=1
S2	S3	S2下一状态不为S1, 且stop[pos+i]=1 or up[pos+i]=1 or down[pos+i]=1 (i>0且pos+i<=N)
S2	S4	S2下一状态不为S1和S3, 且down[pos]=1
S2	S5	S2下一状态不为S1/S3/S4, 且stop[pos-i]=1 or up[pos-i]=1 or down[pos-i]=1 (i>0且pos-i>=0)
S2	S2	S2下一状态不为S1/S3/S4/S5时
S3	S2	完成pos=pos+1, 下一个时钟
S4	S4	down[pos]=1 or stop[pos]=1
S4	S5	S4下一个状态不为S4时
S5	S4	down[pos]=1 or stop[pos]=1
S5	S6	S5下一状态不为S4, 且stop[pos-i]=1 or up[pos-i]=1 or down[pos-i]=1 (i>0且pos-i>=0)
S5	S1	S5下一状态不为S4和S6, 且up[pos]=1
S5	S2	S5下一状态不为S4/S6/S1, 且stop[pos+i]=1 or up[pos+i]=1 or down[pos+i]=1 (i>0且pos+i<=N)
S5	S5	S2下一状态不为S4/S6/S1/S2时
10 September 2020	S6	完成pos=pos-1, 下一个时钟



# 数字系统设计自动化

## ■ 数字系统设计层次

	抽象层次	时序单位	基本单元	电路的功能描述
前端	系统级	数据处理	进程及通信	自然语言描述系统功能或相互通信的进程
	行为级	运算步	运算的控制	行为有限状态机、数据流图、控制流图
	RTL级	时钟周期	寄存器、运算	布尔方程、卡诺图、有限状态机
后端				



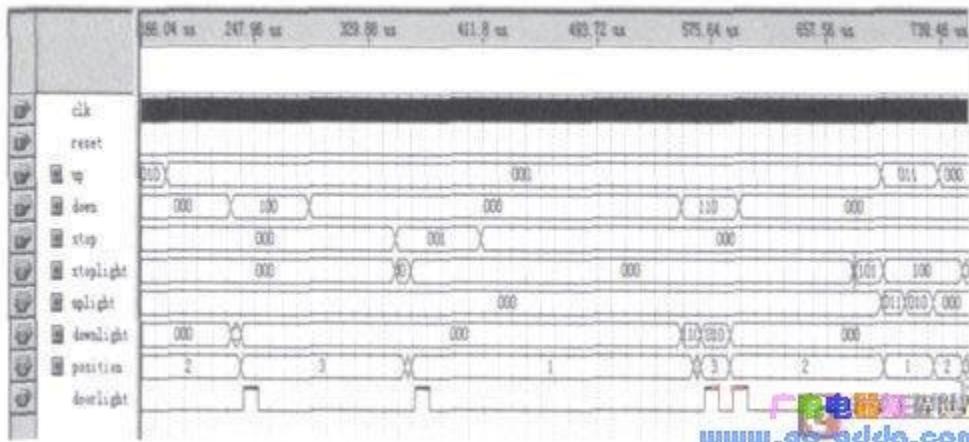
# 数字系统设计自动化

## ■ 电梯的RTL级描述

RTL设计

```
1 module regfile (rna,rnb,d,wn,we,clk,clrn,qa,qb);
2   input  [4:0] rna,rnb,wn; // register #
3   input  [31:0] d;        // data to be written
4   input      we,clk,clrn; // write enable
5   output [31:0] qa,qb;    // two read ports
6
7   reg    [31:0] register [1:31]; // r1 - r31
8   assign qa = (rna == 0) ? 0 : register[rna]; // read port 1
9   assign qb = (rnb == 0) ? 0 : register[rnb]; // read port 2
10  always @(posedge clk or negedge clrn) begin
11    if (clrn == 0) begin // reset
12      integer i;
13      for (i=1; i<32; i=i+1)
14        register[i] <= 0;
15    end else begin
16      if ((wn != 0) && (we == 1))
17        register[wn] <= d;           // write port
18    end
19  end
20 endmodule
```

RTL仿真





# 数字系统设计自动化

## ■ 数字系统设计层次

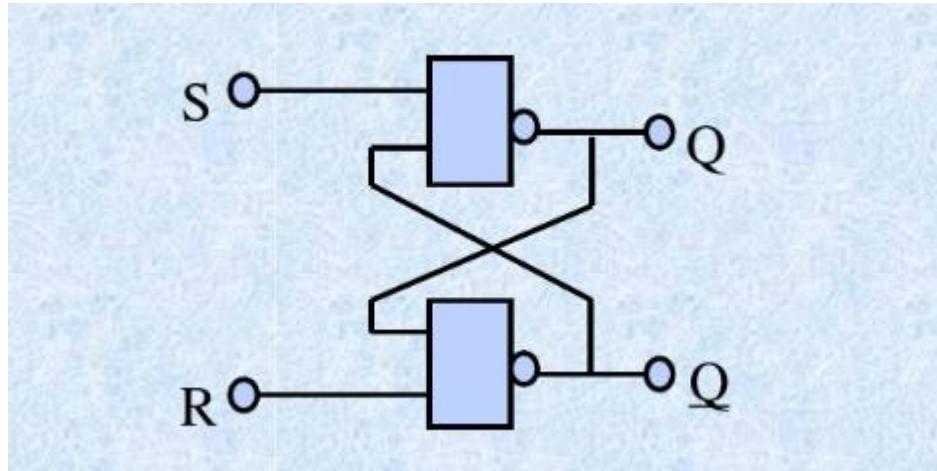
抽象层次		时序单位	基本单元	电路的功能描述
前端	系统级	数据处理	进程及通信	自然语言描述系统功能或相互通信的进程
	行为级	运算步	运算的控制	行为有限状态机、数据流图、控制流图
	RTL级	时钟周期	寄存器、运算	布尔方程、卡诺图、有限状态机
后端	门级	延时模型	逻辑门、器件	原理图



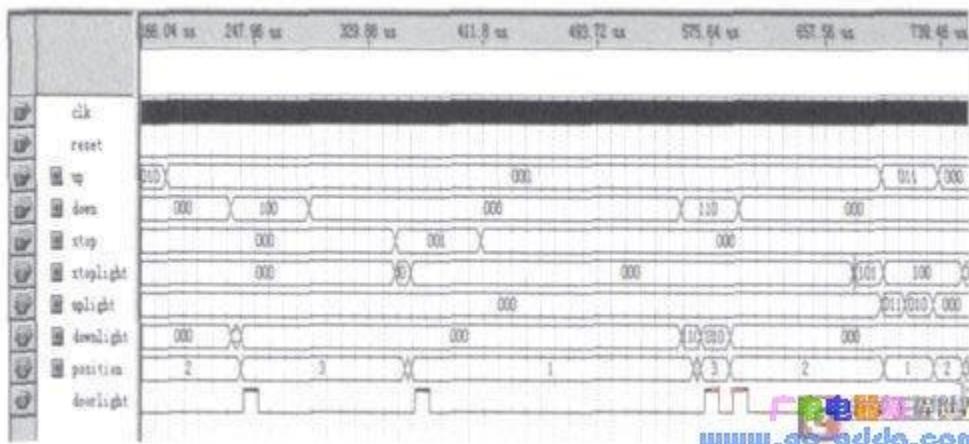
# 数字系统设计自动化

## ■ 电梯的门级描述

门级设计



门级仿真





# 数字系统设计自动化

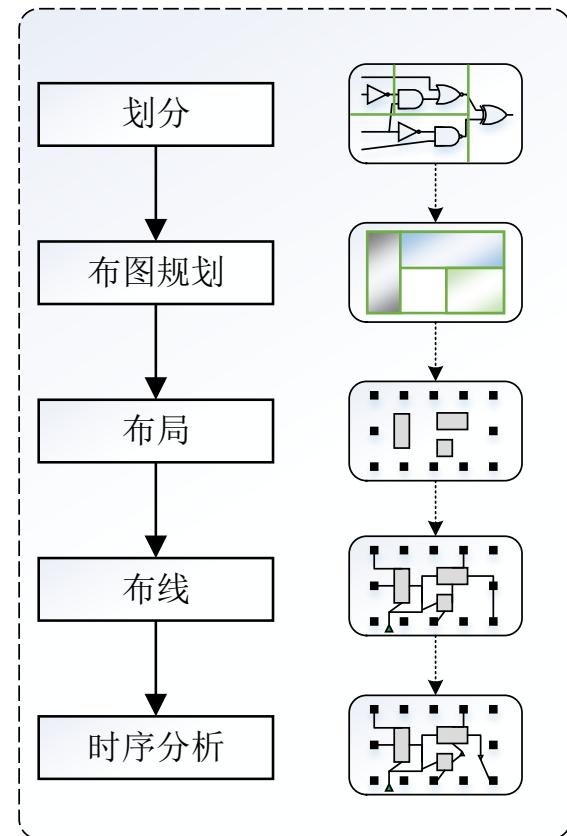
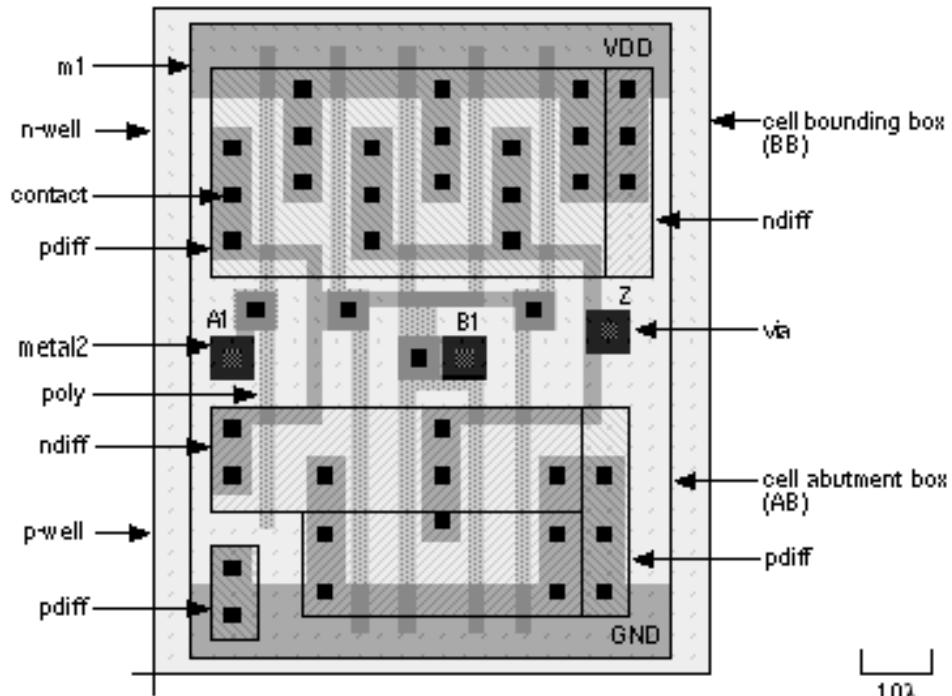
## ■ 数字系统设计层次

抽象层次		时序单位	基本单元	电路的功能描述
前端	系统级	数据处理	进程及通信	自然语言描述系统功能或相互通信的进程
	行为级	运算步	运算的控制	行为有限状态机、数据流图、控制流图
	RTL级	时钟周期	寄存器、运算	布尔方程、卡诺图、有限状态机
后端	门级	延时模型	逻辑门、器件	原理图
	版图级		几何图形	



# 数字系统设计自动化

## ■ 电梯的版图级描述





## ■ 综合 (Synthesis)

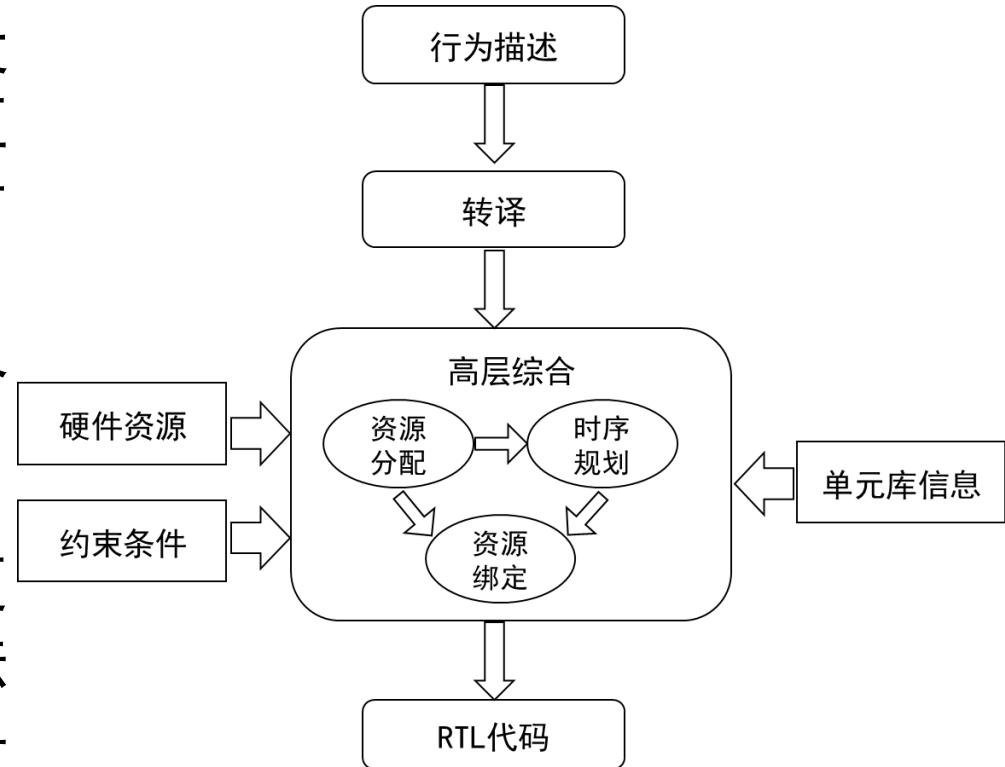
- 是指从较高层次的设计描述到较低层次的设计描述的转换、映射并包含一定的设计优化的过程
- 高层综合(High-level Synthesis)
- 逻辑综合(Logic Synthesis)
- 版图综合(Layout synthesis)



# 数字设计自动化

## ■ 高层综合

- 超大规模集成电路设计是一个多目标优化问题，设计过程必须权衡几个相互冲突的设计目标，如芯片面积、电路延迟和功耗。  
高层综合实现从行为抽象层到RTL的自动化转换，可以大大缩短设计周期，允许在设计过程中尝试更多可替代的电路实现方法，从而为集成电路设计寻找更多的优化设计空间。

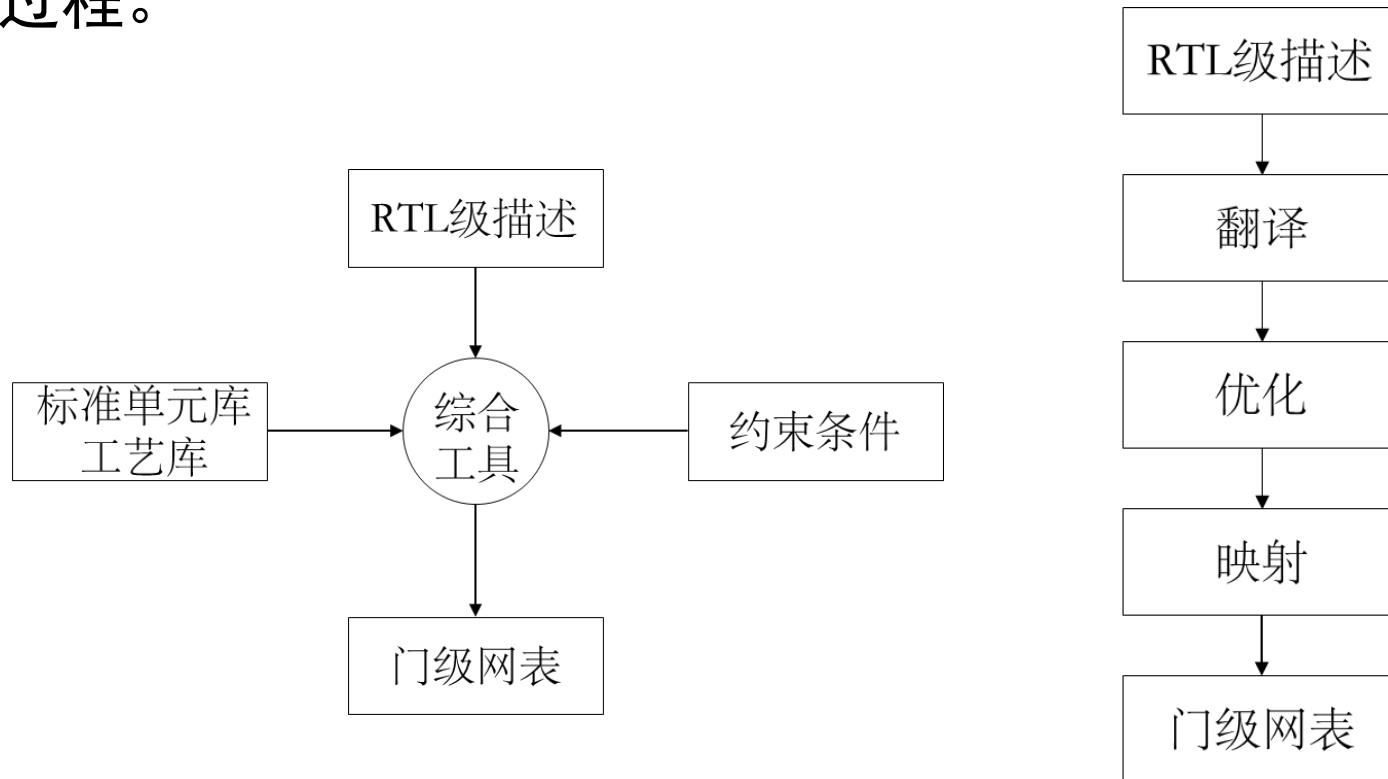




# 数字设计自动化

## ■ 逻辑综合

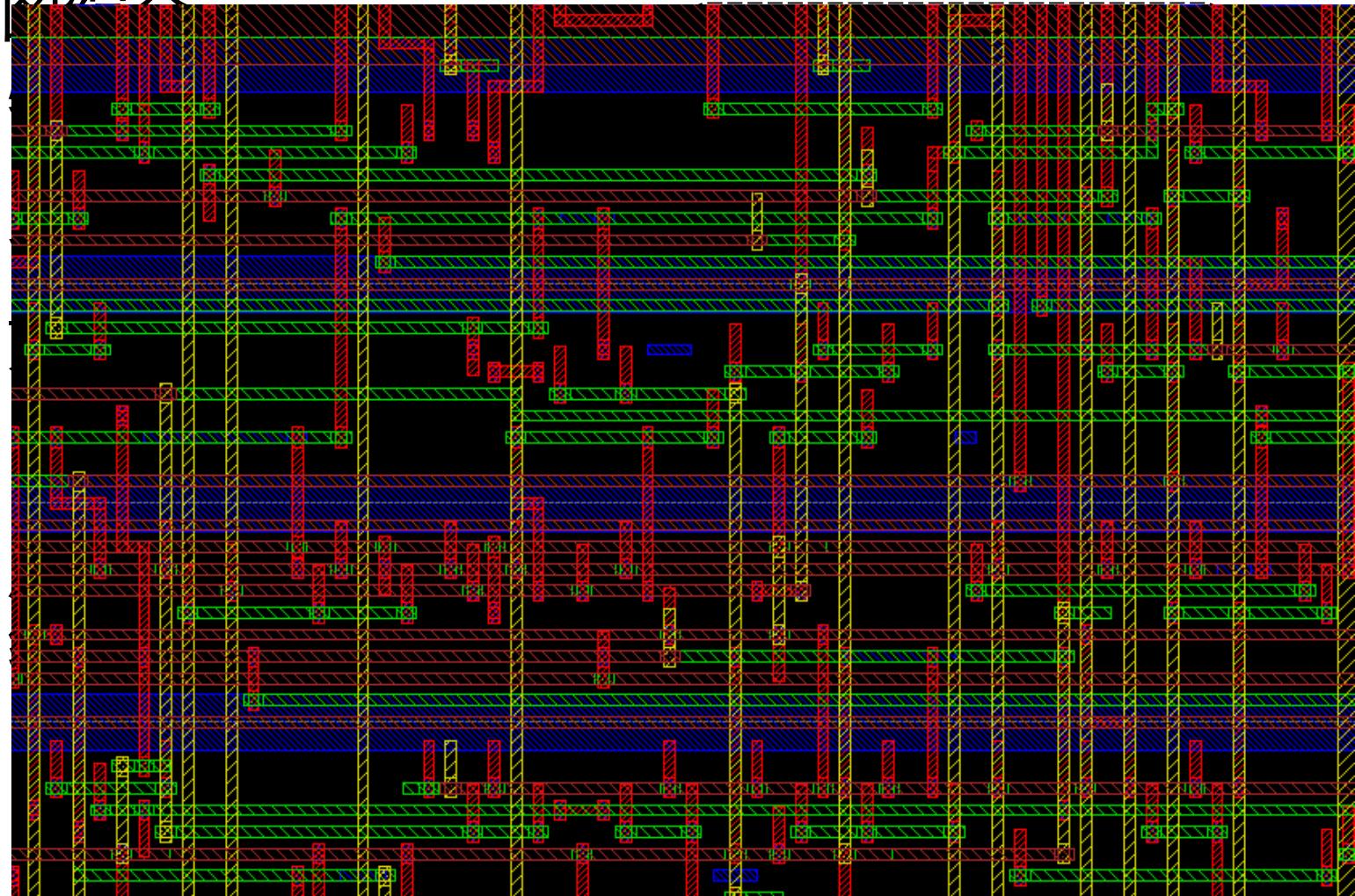
- 是指将把寄存器传输级（Register Transfer Level, RTL）代码所描述的逻辑功能和用户所要求的性能，基于一个完备的逻辑单元库，转换成满足相关约束条件的门级网表的过程。





# 数字设计自动化

## ■ 版图综合





# 数字设计自动化





# 总结

## ■ 一、数字系统概述

- 数字信号、器件、数字逻辑、逻辑电路、微处理器、系统

## ■ 二、数字系统设计方法学

- 三个域、层次式设计、实现技术

## ■ 三、数字系统设计自动化

- EDA的发展历史、国内EDA现状、EDA设计流程



# 课后作业

- 数字系统的实现方式有哪些？各有什么优缺点？
- 简述“片上系统(System on Chip, SOC)”
- 什么是Top-Down设计模式？优缺点是什么？
- 数字系统设计过程中，“设计说明书(Specification, Spec)”的作用是什么？通常包含哪些内容？

**第二部分**

**实验**

# **Linux 使用**



课程号: B3I493310    开课学期: 2020-2021学年秋季

# 数字系统设计

北京航空航天大学  
微电子学院  
贾小涛



# Linux系统的介绍

## ■ 什么是Linux



00:00 / 02:04

音量 1x 标清



# Linux系统的介绍

Linux是一套**免费的、开放源代码的，并可以自由传播**的操作系统，主要用于基于**Intel X86**系列的CPU上





# Linux系统的介绍

- 多用户、多任务
- 良好的兼容性
- 强大的可移植性
- 高度的稳定性
- 操作方便、高效
- 支持多种文件系统





# Linux系统的介绍

## ■ Linux的组成



## ■ Linux内核

- 模块化结构
- CPU和进程管理
- 内存管理
- 文件管理
- 磁盘管理等



# Linux系统的介绍

## ■ Linux的组成



## ■ Shell

- 内核不能直接接受来自用户的命令
- 交互式命令解释程序
  - B Shell bash
  - C Shell csh
  - K Shell



# Linux系统的介绍

## ■ 常见的Linux发行版本



redhat



fedora



CentOS



debian



ubuntu

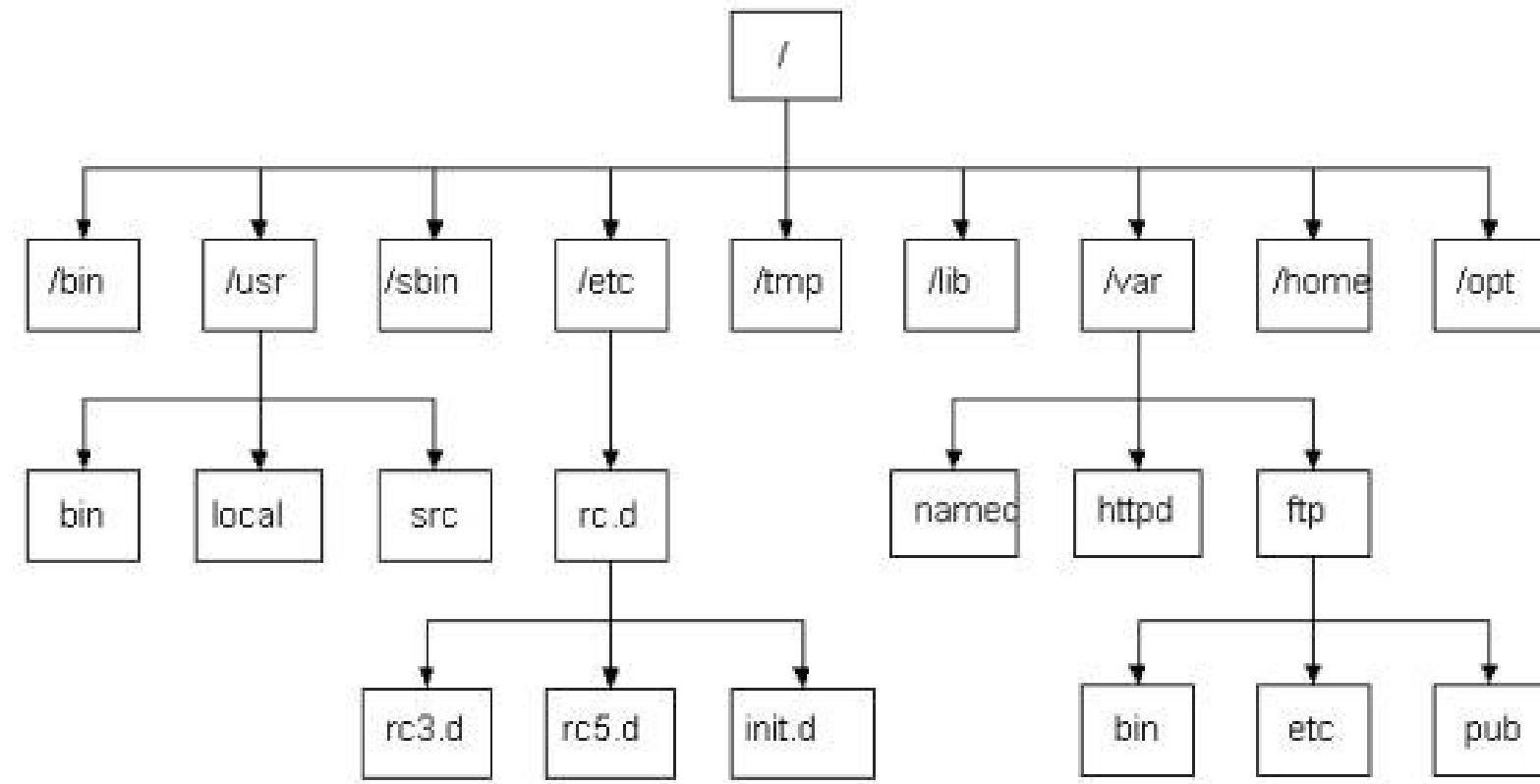




# Linux系统的使用

## ■ 文件系统

— 严格区分大小写，用/做分割（windows是\）





# Linux系统的常用命令

cd /home 进入 '/ home' 目录'

cd .. 返回上一级目录

cd ../../ 返回上两级目录

cd 进入个人的主目录

cd ~user1 进入个人的主目录

cd - 返回上次所在的目录

pwd 显示工作路径

ls 查看目录中的文件

ls -F 查看目录中的文件

ls -l 显示文件和目录的详细资料

ls -a 显示隐藏文件

ls \*[0-9]\* 显示包含数字的文件名和目录名

mkdir dir1 创建一个叫做 'dir1' 的目录'

mkdir dir1 dir2 同时创建两个目录

mkdir -p /tmp/dir1/dir2 创建一个目录树

rm -f file1 删除一个叫做 'file1' 的文件'

rmdir dir1 删除一个叫做 'dir1' 的目录'

rm -rf dir1 删除一个叫做 'dir1' 的目录并同时删除其内容

rm -rf dir1 dir2 同时删除两个目录及它们的内容

mv dir1 new\_dir 重命名/移动 一个目录

cp file1 file2 复制一个文件

cp dir/\* . 复制一个目录下的所有文件到当前工作目录

cp -a /tmp/dir1 . 复制一个目录到当前工作目录

cp -a dir1 dir2 复制一个目录

cp -r dir1 dir2 复制一个目录及子目录

ln -s file1 lnk1 创建一个指向文件或目录的软链接

ln file1 lnk1 创建一个指向文件或目录的物理链接



# VI 文本编辑器

## ■ 编辑模式

- 正常 (normal) 模式，缺省的编辑模式；下面如果不加特殊说明，提到的命令都直接在正常模式下输入；任何其它模式中都可以通过键盘上的 Esc 键回到正常模式。
- 命令 (command) 模式，用于执行较长、较复杂的命令；在正常模式下输入 “.”（一般命令）、“/”（正向搜索）或 “?”（反向搜索）即可进入该模式；命令模式下的命令要输入回车键（Enter）才算完成。
- 插入 (insert) 模式，输入文本时使用；在正常模式下键入 “i” (insert) 或 “a” (append) 即可进入插入模式（也有另外一些命令，如 “c”，也可以进入插入模式，但这些命令有其它的作用）。



# VI编辑器

- i: 在当前光标所在字符的前面，转为输入模式；
- a: 在当前光标所在字符的后面，转为输入模式；
- o: 在当前光标所在行的下方，新建一行，并转为输入模式；
- I: 在当前光标所在行的行首，转换为输入模式
- A: 在当前光标所在行的行尾，转换为输入模式
- O: 在当前光标所在行的上方，新建一行，并转为输入模式；

```
:q 退出  
:wq 保存并退出  
:q! 不保存并退出  
:w 保存  
:w! 强行保存  
:wq --> :x
```



# VI編輯器

## vim命令圖解

The screenshot shows a Java code editor with Vim's highlighting features applied. Several Vim commands are overlaid on the code:

- Character movement:** `h`, `j`, `k`, `l` (left, down, up, right), `Ctrl-B` (beginning of word), `Ctrl-F` (end of word).
- Line movement:** `0` (beginning of line), `$` (end of line), `^` (beginning of line, non-blank), `M` (middle of line), `zz` (bottom of line).
- Block movement:** `{` (beginning of paragraph), `}` (end of paragraph), `[{` (beginning of block), `]} (end of block), % (match括號).`
- Search and replace:** `Ctrl-N` (next search), `Ctrl-F` (previous search), `Ctrl-W p` (previous search), `Ctrl-W j` (next search), `Ctrl-W l` (previous search), `Ctrl-W k` (next search).
- File operations:** `:split`, `:vsplit`, `:diffsplit`.
- Mode switching:** `ESC` (normal mode), `v` (visual mode), `V` (visual line mode), `C-v` (visual block mode), `i` (insert mode), `R` (replace mode), `a` (append mode), `A` (append at end mode), `y` (copy), `d` (delete), `c` (change), `x` (delete character), `D` (delete to end of line), `C` (change to end of line), `p` (paste), `J` (join), `r` (replace), `>` (expand), `<` (contract), `.` (repeat), `u` (undo).
- EX mode:** `:w` (write), `:q` (quit), `:e x` (edit file), `:n` (new file), `:h` (help), `:xx` (jump to line xx).
- Autocompletion:** `C-N` (next completion), `C-P` (previous completion), `C-X C-F` (find file).
- Window splitting:** `:vsp` (vertical split), `:sp` (horizontal split), `:diffs` (diff split).
- Search history:** `/xxx` (search for xxx), `n` (next search result), `N` (previous search result), `C-W p` (jump to previous split), `C-W w` (jump to next split).

游標移動/範圍單位	
字元(character)	
<code>h</code>	左
<code>j</code>	下
<code>k</code>	上
<code>l</code>	右
<code>Ctrl-B</code>	單字(word)
<code>w</code>	前/後個單字
<code>W</code>	前/後個單字(跳過符號)
<code>e</code>	單字尾端
行(line)	
<code>0</code>	行頭
<code>\$</code>	行尾
<code>^</code>	行頭(非空白字元)
段落(paragraph)、區塊(block)	
<code>{</code>	上一段
<code>}</code>	下一段
<code>[{</code>	區塊頭
<code>]} (</code>	區塊尾
<code>%</code>	對應括號
螢幕(screen)、檔案(file)	
<code>H</code>	螢幕頂端
<code>M</code>	螢幕中間
<code>L</code>	螢幕底部
<code>zt</code>	捲至頂端
<code>zz</code>	捲至中間
<code>zb</code>	捲至底部
<code>C-B</code>	上一頁
<code>C-F</code>	下一頁
<code>gg</code>	檔頭
<code>G</code>	檔尾
<code>mx</code>	標記x
<code>'x</code>	跳至標記x
搜尋(search)	
<code>*</code>	向後/向前搜尋目前單字
<code>#</code>	
<code>fx</code>	向後搜尋字元x
<code>gd</code>	跳至目前單字的定義位置
<code>/xxx</code>	搜尋xxx
<code>n</code>	下/上一個搜尋結果
<code>N</code>	

模式切換指令	
<code>ESC</code>	進入normal mode
<code>v</code>	進入visual mode
<code>V</code>	進入visual line mode
<code>C-v</code>	進入visual block mode
<code>i</code>	進入insert mode
<code>R</code>	進入replace mode
<code>a</code>	在游標後附加
<code>A</code>	在行末附加
動作指令	
<code>y</code>	複製(範圍)
<code>d</code>	刪除/剪下(範圍)
<code>c</code>	修改(範圍)
<code>x</code>	刪除/剪下(字元)
<code>D</code>	刪除至行末
<code>C</code>	修改至行末
<code>p</code>	貼上
<code>J</code>	和下一行合併
<code>r</code>	替換(字元)
<code>&gt;</code>	縮排
<code>&lt;</code>	反縮排
<code>.</code>	重複上一命令
<code>u</code>	回復上一命令
EX指令	
<code>:w</code>	儲存(:wq 儲存並退出)
<code>:q</code>	退出(:q!強制退出)
<code>:e x</code>	編輯檔案x
<code>:n</code>	開啟文件
<code>:h</code>	呼叫vim help
<code>:xx</code>	跳至xx行
自動補齊 [insert mode]	
<code>C-N</code>	自動補齊下/上個可能字
<code>C-P</code>	
<code>C-X C-F</code>	自動補齊可能檔名
分割視窗(split window)	
<code>:vsp</code>	垂直/水平分割視窗
<code>:sp</code>	
<code>:diffs</code>	分割視窗並比較(diff)檔案
<code>C-W p</code>	(來回)跳至前一個分割視窗
<code>C-W w</code>	跳至下個分割視窗



# Linux系统登陆方法

## ■ 远程登录

### — IP地址

➤ 10.111.3.128

### — 用户名&密码

➤ 姓名全拼

➤ 及时修改个人密码

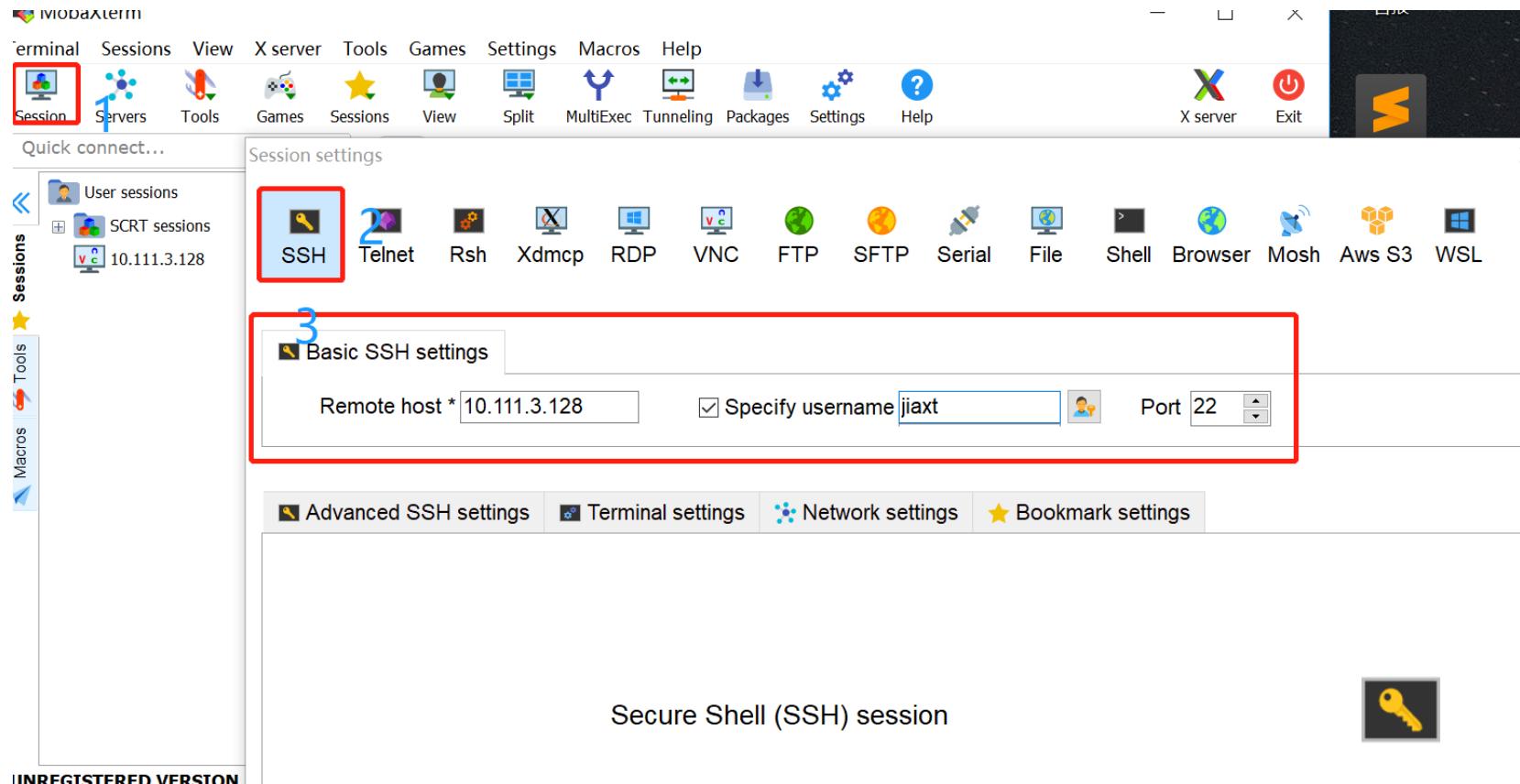
➤ 该服务器仅用作教学

### — mobaxterm

➤ <https://mobaxterm.mobatek.net/download.html>

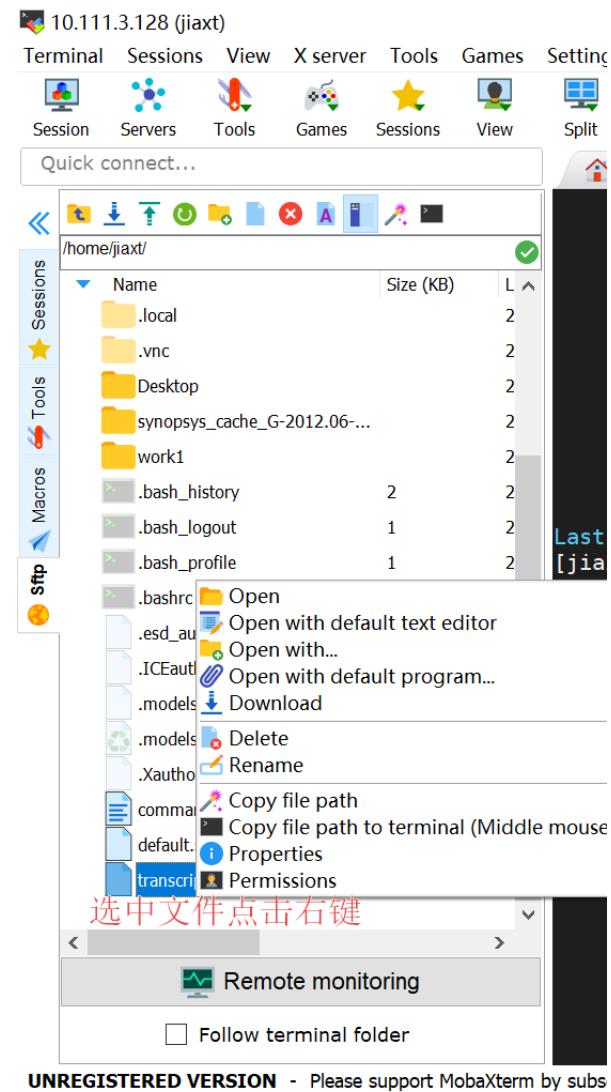
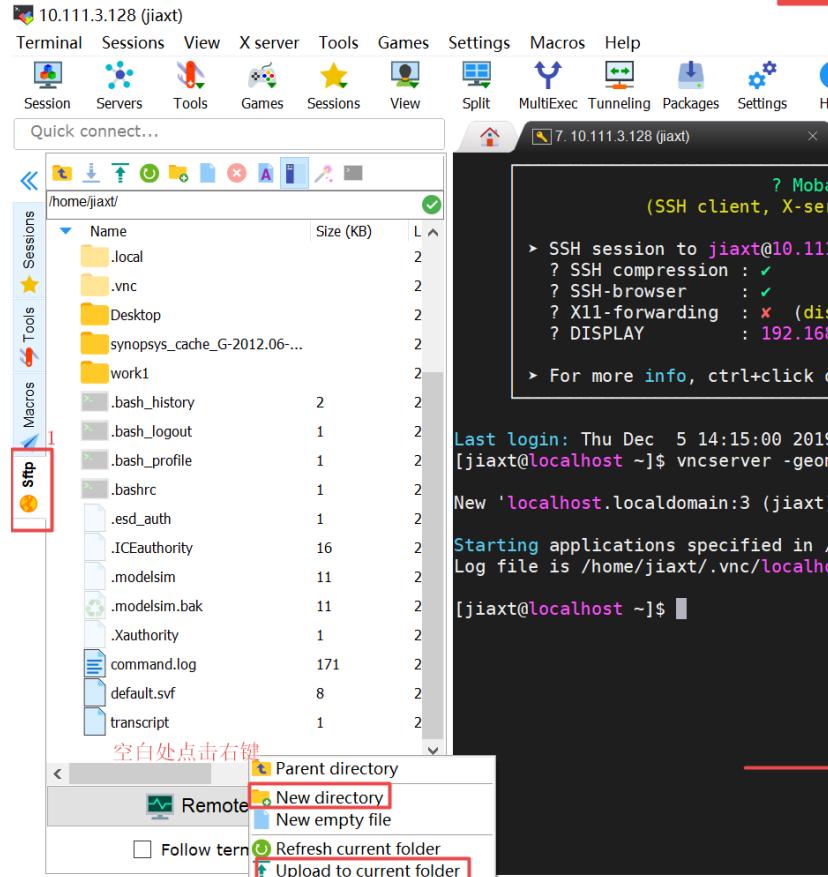


# Linux系统登陆方法





# Linux系统登陆方法





# Linux系统登陆方法

## ■ 远程桌面系统

### — VNC Viewer

- 不要使用VNC server
- <https://www.realvnc.com/en/connect/download/viewer/>

### — 启动端口

- vncserver –geometry 1920x1080 :xx
- You will require a password to access your desktops.
  - ✓ 第一次输入时会提示输入一个密码，是后续登录VNC时使用的
- Would you like to enter a view-only password (y/n)?
  - ✓ 输入n，回车

➤ 分辨率根据自己电脑设置

➤ 记住下面的端口号

➤ 不要多次申请

```
Last login: Thu Dec  5 14:15:00 2019 from 10.134.17.66
[jiaxt@localhost ~]$ vncserver -geometry 1920x1080

New 'localhost.localdomain:3 (jiaxt)' desktop is localhost.localdomain:3
Starting applications specified in /home/jiaxt/.vnc/xstartup
Log file is /home/jiaxt/.vnc/localhost.localdomain:3.log

[jiaxt@localhost ~]$
```



# Linux系统登陆方法

V VNC Viewer

File View Help

10.111.3.128:1

Sign in... ▾



10.111.3.128:2



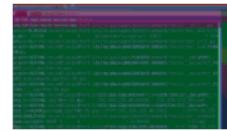
10.111.3.128:3



192.168.0.90:3



192.168.0.90:12



192.168.226.40:1



192.168.226.40:6



192.168.226.60:7



192.168.226.60:88



192.168.226.136:2



192.168.226.136:22



192.168.226.136:64



219.142.135.84:59...



# 作业

- 登录Linux系统，执行目录和文件操作