数字逻辑

第三章 组合逻辑电路分析与设计

北京理工大学 计算机学院

张磊

leizhang@bit.edu.cn

本章内容

一．设计过程

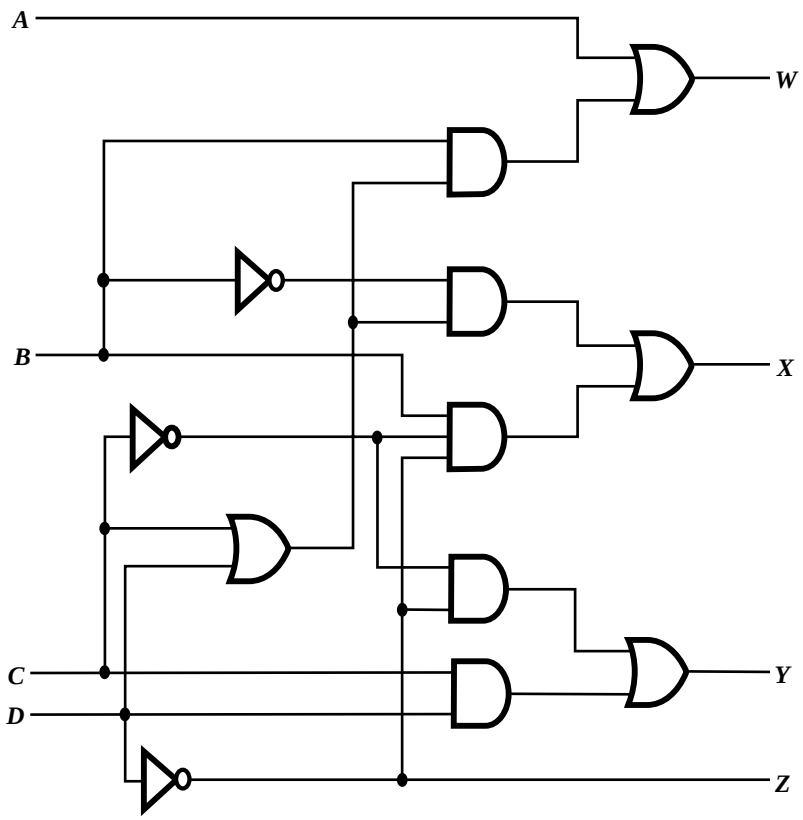
- 1. 规范化
- 2. 形式化
- 3. 优化
- 4. 工艺映射
- 5. 验证

1. 规范化
   - 指定组合电路行为
2. 形式化
   - 用真值表对输入输出形式化
3. 优化
   - 优化逻辑，减少门输入成本，如卡诺图优化
4. 工艺映射
   - 将优化后逻辑映射到实现工艺
5. 验证
   - 验证设计正确性

- 优化的结果：

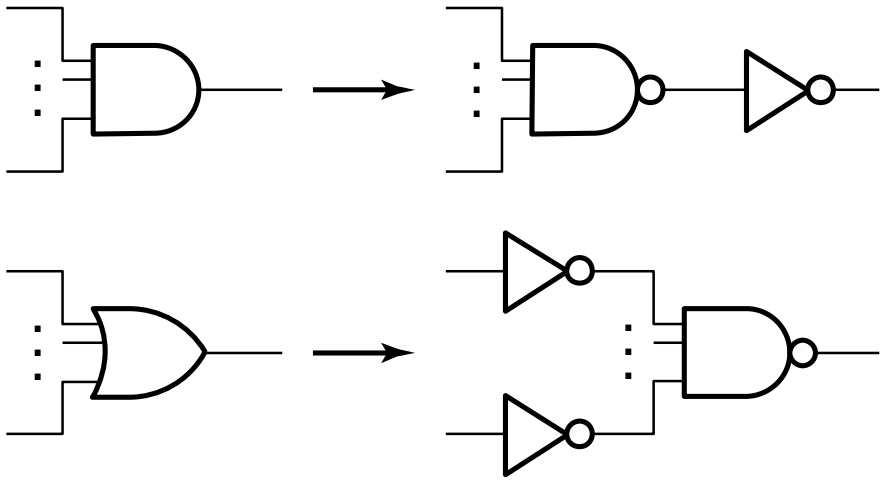
$$W = A + BT_1$$
$$X = \overline{B}T_1 + B\overline{T}_1$$
$$Y = CD + \overline{T}_1$$
$$Z = \overline{D}$$

等价与

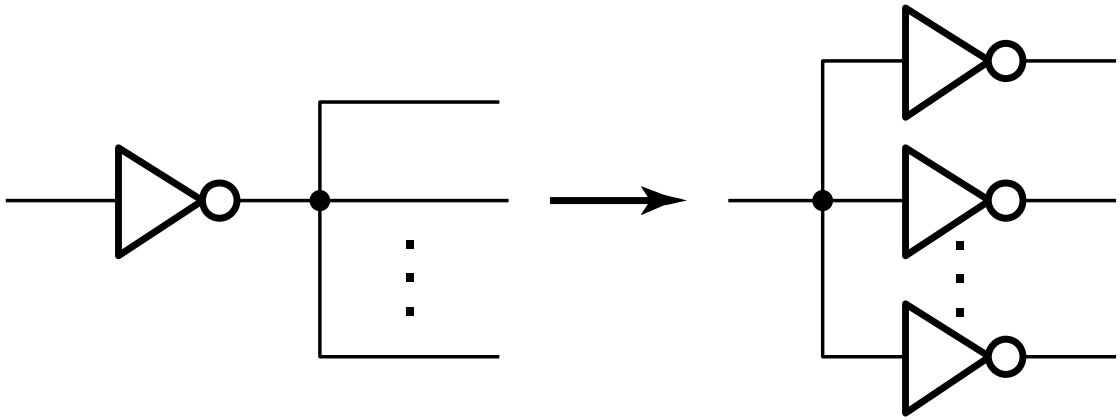


- 4. 工艺映射
  - 给定设计好的电路概要图
    - （含与门、或门、反相）
  - →到与非门
    - 库中含反相器和n-输入与非门，n = 2, 3, ...
  - →到或非门
    - 库中含反相器和n-输入或非门，n = 2, 3, ...

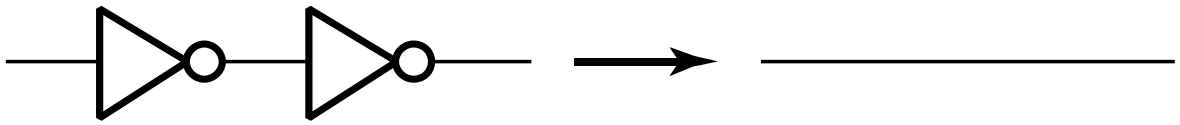
- 映射到与非门
  - 库中含反相器和n-输入与非门，n = 2, 3, ...
- 映射过程:
  - ① 用与非门替换掉与门和或门



- 映射到与非门
  - 库中含反相器和n-输入与非门，n = 2, 3, …
- 映射过程:
  - ② 将反相器推过电路中的扇出点

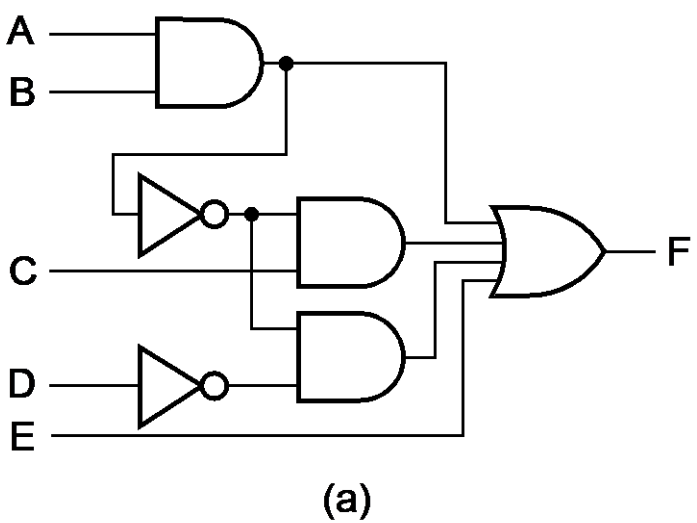


- 映射到与非门
  - 库中含反相器和n-输入与非门，$n = 2, 3, \ldots$;
- 映射过程:
  - ③ 抵消掉反相器对

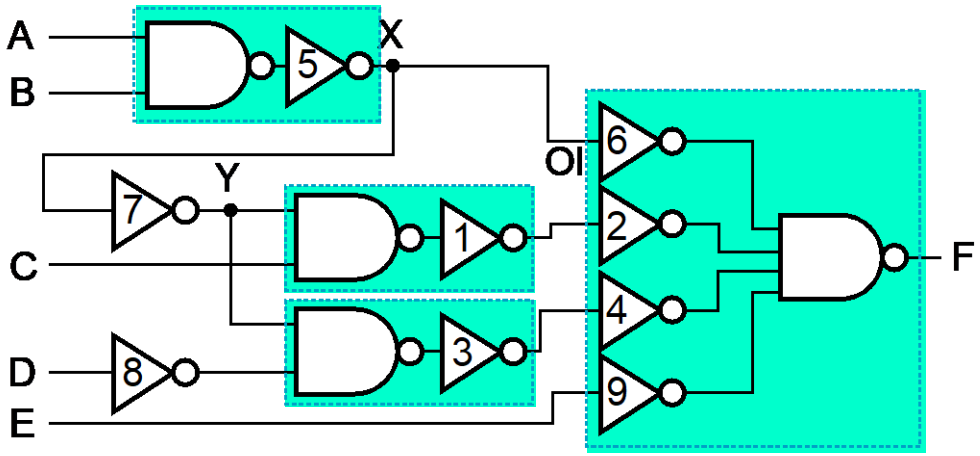


- 映射到与非门
  - 库中含反相器和n-输入与非门，$n = 2, 3, \ldots$
- 映射过程:
  - ④ 重复②和③直到在a和b之间只存在1个反相器:
    a. 电路输入或与非门的输出
    b. 与非门输入
  - 为什么要进行重复进行②和③?

例子

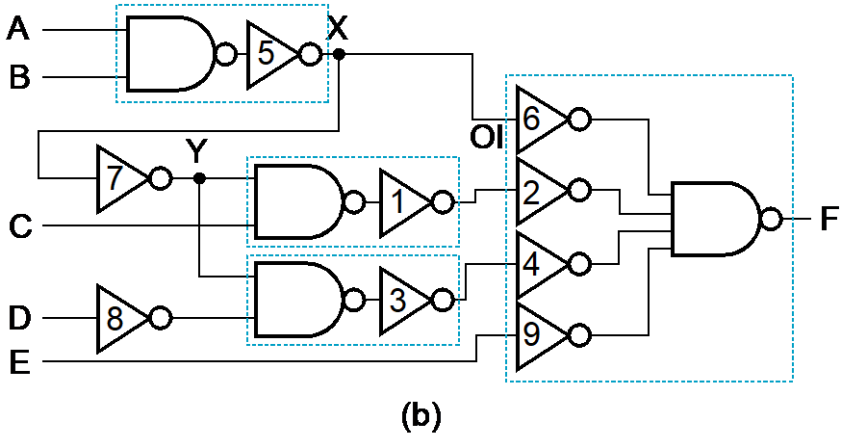


(a)

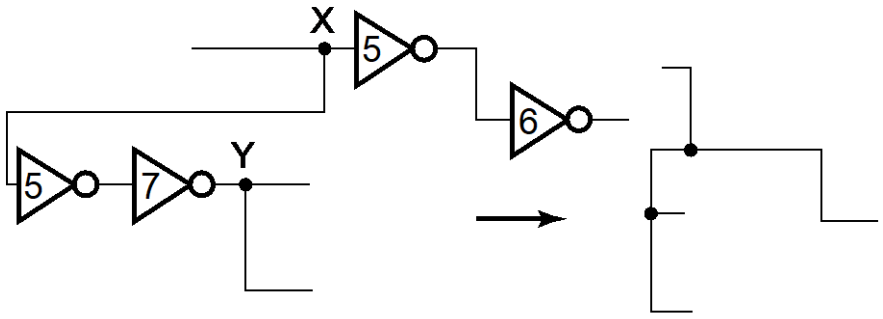


(b)

例子

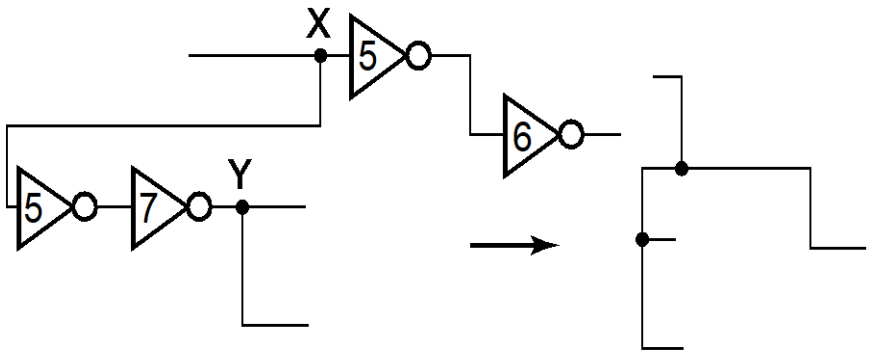


(b)

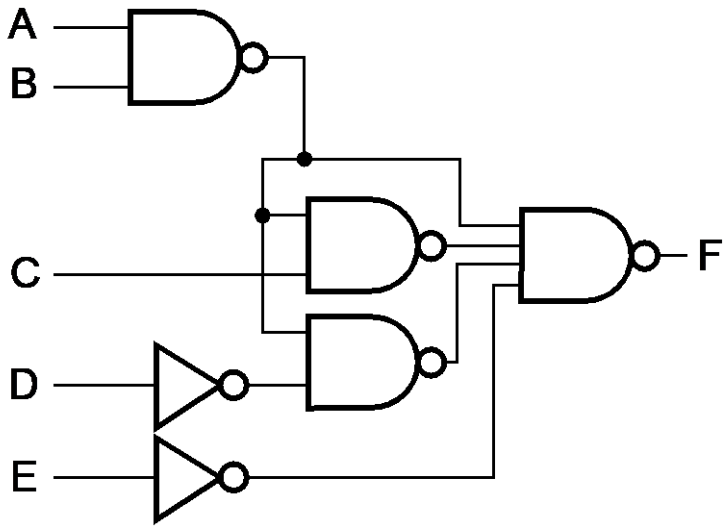


(c)

例子

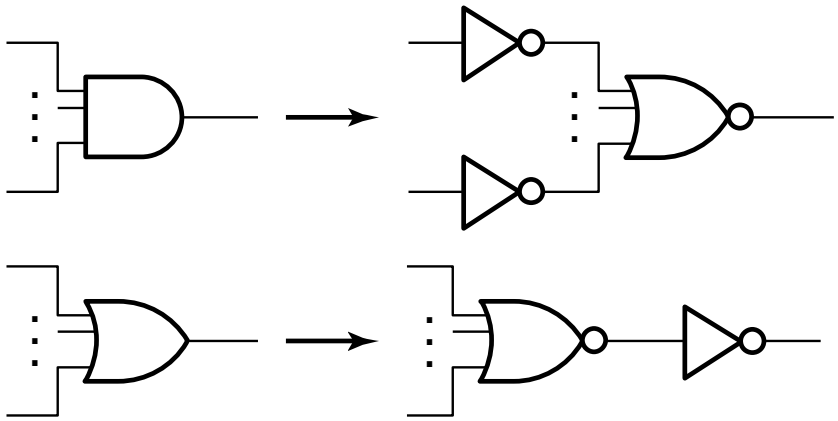


(c)



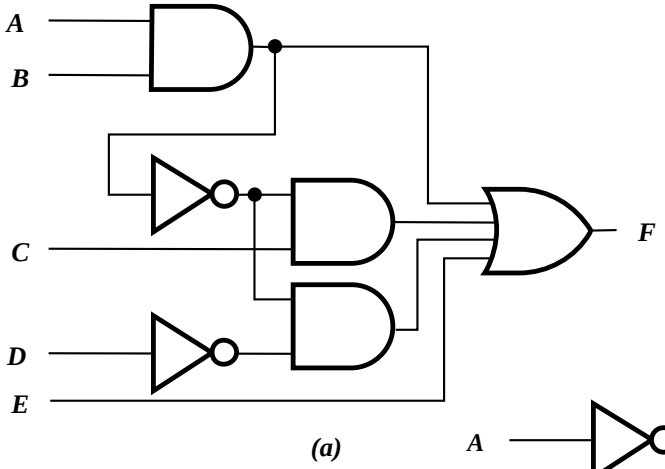
(d)

- 映射到或非门
  - 库中含反相器和n-输入或非门，n = 2, 3, ...
- 映射过程:
  - ① 用或非门替换掉与门和或门

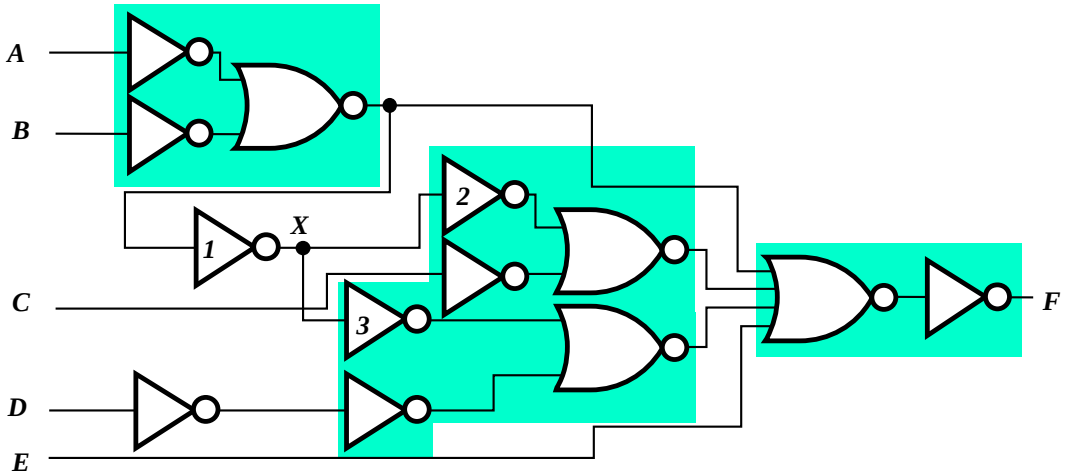


- 映射到或非门
  - 库中含反相器和n-输入或非门，$n = 2, 3, \ldots$
- 映射过程:
  - ② 将反相器推过电路中的扇出点
  - ③ 抵消掉反相器对
  - ④ 重复②和③直到在a和b之间只存在1个反相器:
    a. 电路输入或或非门的输出
    b. 或非门输入

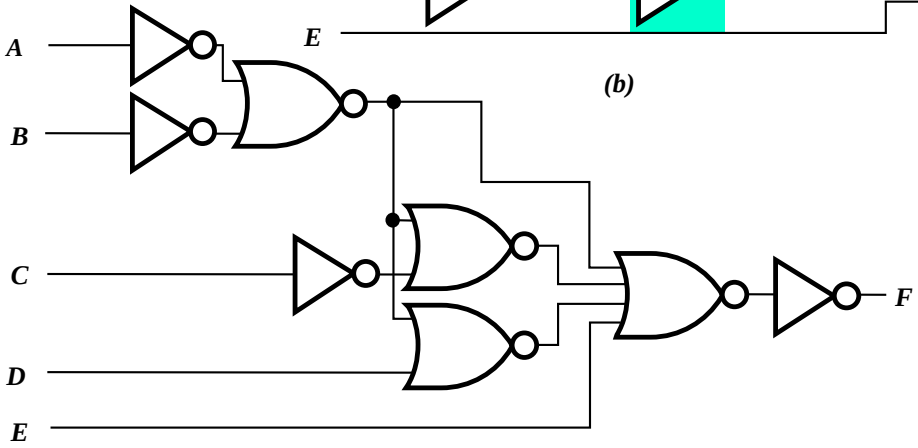
例子



(a)



(b)



(c)

- 5. 验证
  - 人工逻辑分析
  - 模拟

- 人工逻辑分析
  - 找出实现电路真值表或方程式
  - 将实现电路真值表和规范真值表进行比较，或
  - 证明实现电路的方程式和规范方程式等价

$$T_1 = \overline{\overline{C + D}} = C + D$$

$$W = \overline{\overline{A}\,\overline{(T_1 B)}} = A + B T_1$$

$$X = (T_1 B)(B\overline{C}\,\overline{D}) = \overline{B} T_1 + B\overline{C}\,\overline{D}$$

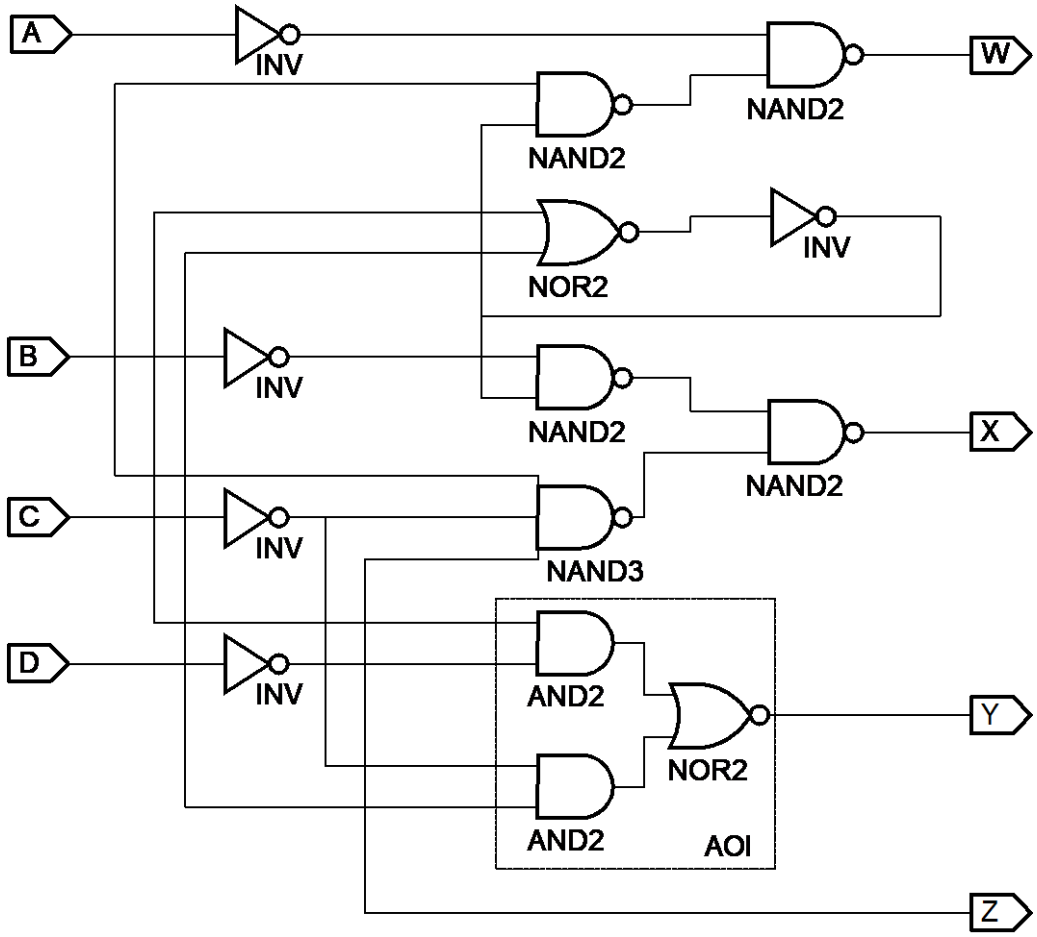
$$Y = \overline{C\overline{D} + \overline{C} D} = CD + \overline{C}\,\overline{D}$$

Input BCD A B C D	Output Excess-3 WXYZ
0 0 0 0	0 0 1 1
0 0 0 1	0 1 0 0
0 0 1 0	0 1 0 1
0 0 1 1	0 1 1 0
0 1 0 0	0 1 1 1
0 1 0 1	1 0 0 0
0 1 1 0	1 0 0 1
0 1 1 1	1 0 1 0
1 0 0 0	1 0 1 1
1 0 0 1	1 1 0 0

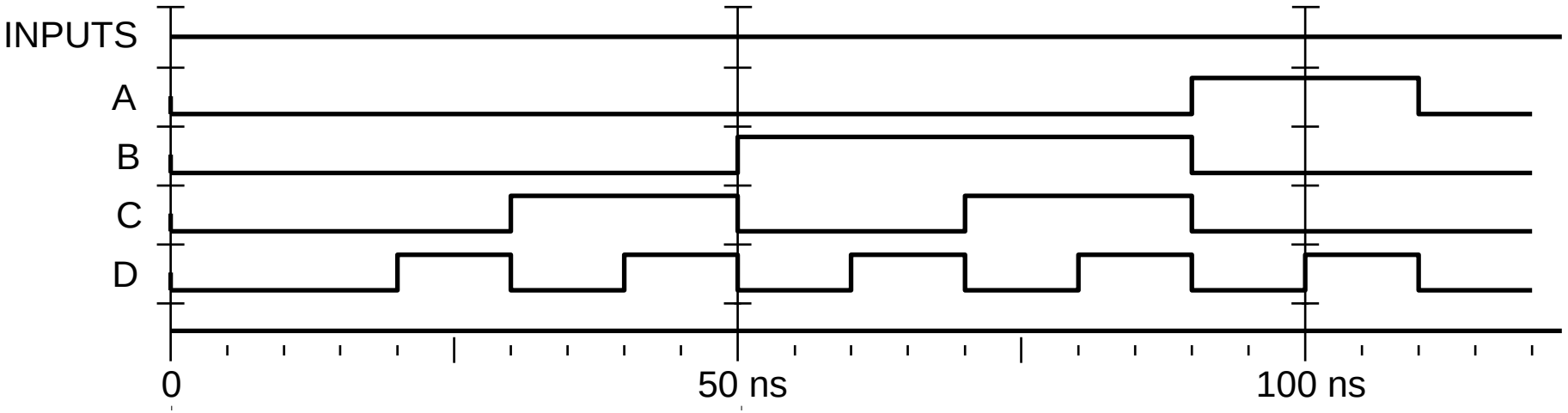
- 模拟
  - 以Verilog为例：Verilog是一种硬件描述语言
  1. 使用Verilog 对电路进行编程实现
  2. 编写测试程序，即TestBench
     - 产生模拟信号
     - 将产生信号加到实现电路上
  3. 将输出与期望值比较

- 模拟
  - 使用图编辑器或者文本编辑器对实现电路进行一个门级表示;
  - 采用波形编辑器或者文本编辑器来输入一个测试，这个测试是输入组合的一个序列;
    - 如果所有的响应是正确的，那就能保证实现电路的正确性;
    - 由于所有可能的输入组合的短缺，因此生成这样的一个测试有一定难度.

- 模拟
  - 输入BCD码到余三码转换器的原理图

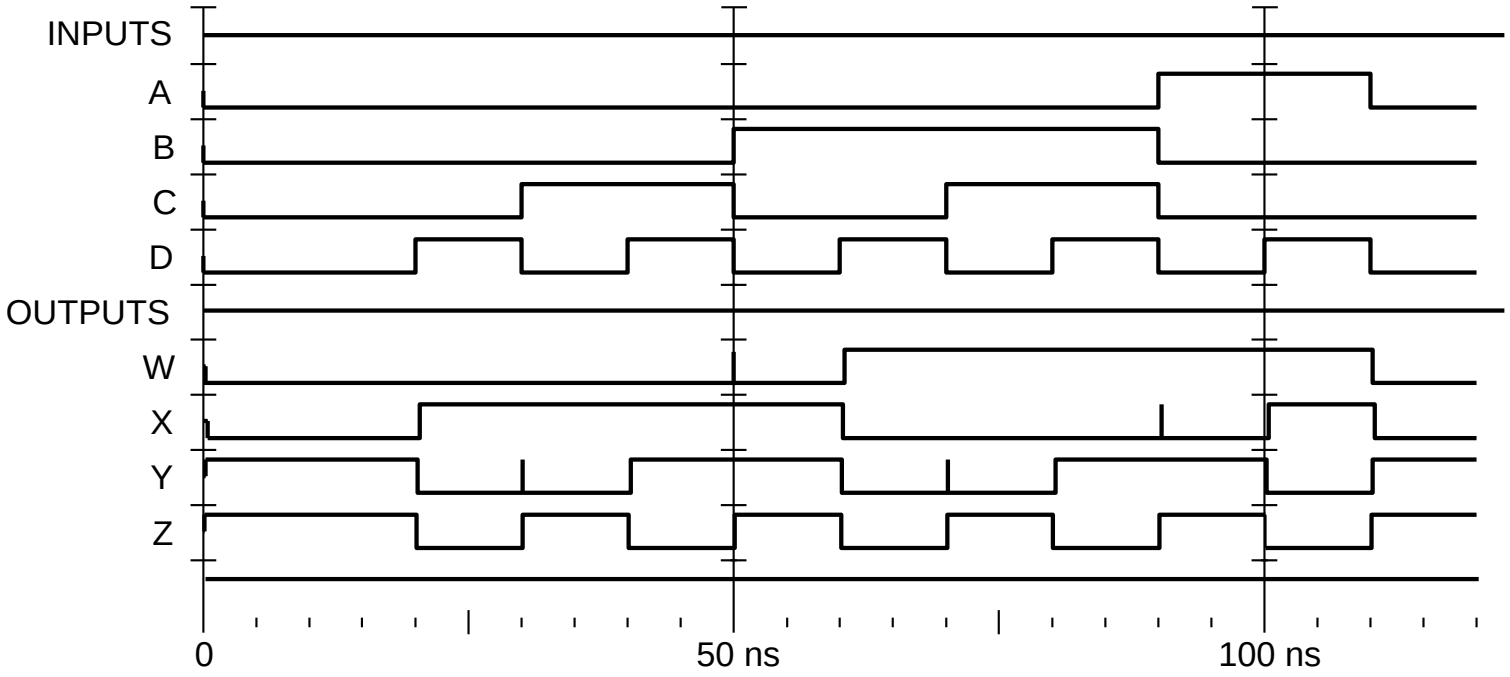


- 模拟
  - 输入一个波形，其包含所有可能的输入组合



  - 所有的BCD码组合是否都出现了？

- 模拟
  - 输入一个波形，其包含所有可能的输入组合



  - 所有的模拟输出组合是否和原始真值表一致?

- 例子：一个加法器
  - ① 使用Verilog 对电路进行编程实现

```
1  module add(a,b,c,d,e);// 模块接口
2  input [5:0] a; // 输入信号a
3  input [5:0] b; // 输入信号b
4  input [5:0] c; // 输入信号a
5  input [5:0] d; // 输入信号b
6
7  output[7:0] e; // 求和输出信号
```

- 例子：一个加法器
  - ② 编写TestBench: 产生模拟信号

```
initial
     begin     // initial 是仿真用的初始化关键词
       a=0 ;   // 必须初始化输入信号
       b=0 ;
       c=0 ;
       d=0
          for(i=1;i<31;i=i+1)
             begin
              #10 ; a = i; b = i; c = i; d = i;
             end
      end
```

- 例子：一个加法器
  ② 编写TestBench: 将产生信号加到实现电路上

```
// 调用被仿真模块模块
add uut (
.a(a),
.b(b),
.c(c),
.d(d),
.e(e));
```

- 例子：一个加法器
  - ③ 将输出与期望值比较