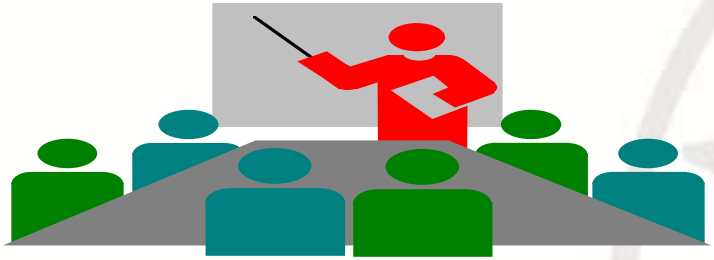


逻辑与计算机设计基础实验与课程设计

实验七

同步时序电路典型设计

施青松
Asso. Prof. Shi Qingsong
College of Computer Science and Technology, Zhejiang University
zjsqs@zju.edu.cn

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

实验目的

1. 掌握典型同步时序电路的工作原理和设计方法；
2. 掌握有限状态机描述与电路实现方法；
3. 掌握时序电路的状态图、状态方程和触发器激励函数；
4. 掌握用FPGA实现有限状态机设计、仿真与调试。

实验环境

实验设备

1. 计算机(Intel Core i5以上，4GB内存以上)系统

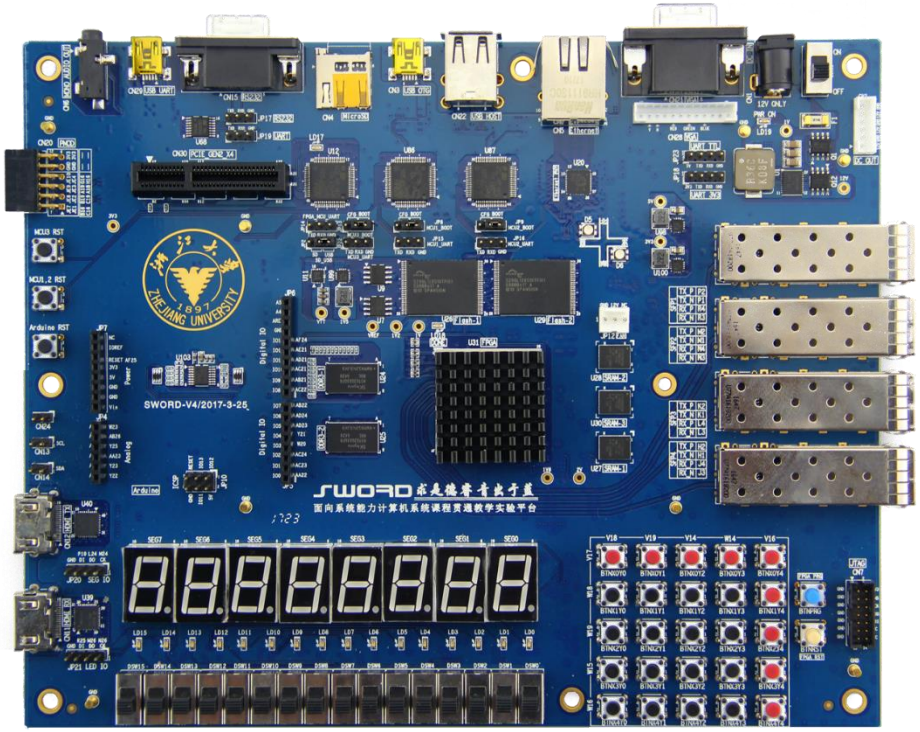
2. 计算机软硬件课程贯通教学实验系统(SWORD4.0)

3. Xilinx ISE14.7及以上开发工具

材料

无

计算机软硬件课程贯通教学实验系统



贯通教学实验平台主要参数

▼ **核心芯片**

Xilinx Kintex™-7系列的XC7K325资源：

162,240个，Slice：25350，片内存储：11.7Mb

▼ **存储体系** 支持32位存储层次体系结构

6MB SRAM静态存储器：支持32Data，16位TAG

512M BDDR3动态存储：支持32Data

32MB NOR Flash存储 ：支持32位Data

▼ **基本接口** 支持微机原理、SOC或微处理器简单应用

4×5+1矩阵按键、16位滑动开关、16位LED、8位七段数码管

▼ **标准接口** 支持基本计算机系统实现

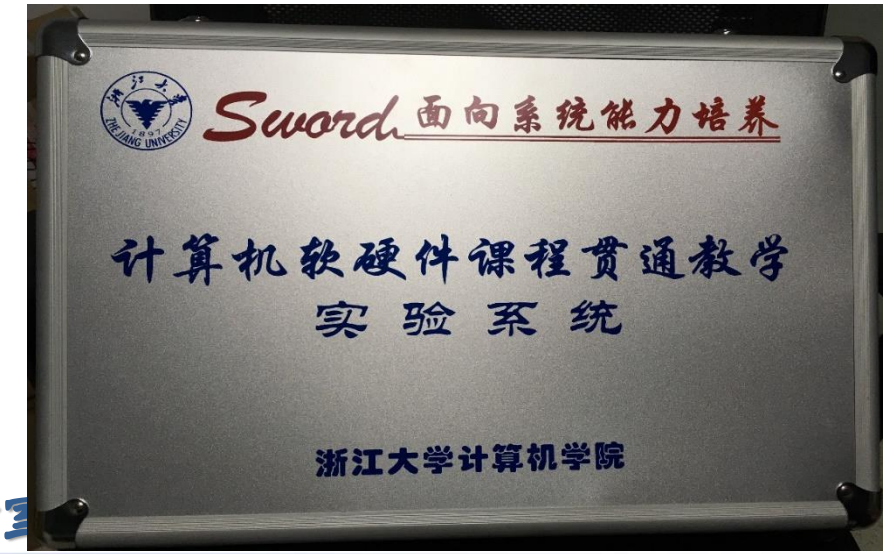
12位VGA接口（RGB656）、USB-HID（键盘）

▼ **通讯接口** 支持数据传输、调试和网络

UART接口、 10M/100M/1000M以太网、 SFP光纤接口

▼ **扩展接口** 支持外存、多媒体和个性化设备

MicroSD(TF）、PMOD、 HDMI、Arduino



Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

实验任务

1. 用原理图设计基于状态方程描述的4位二进制同步计数器；

2. 用HDL 行为描述设计32位同步二进制双向计数器；

3. 集成到实验环境(混合 计算器，Calculation)。

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

4位同步二进制计数器状态表

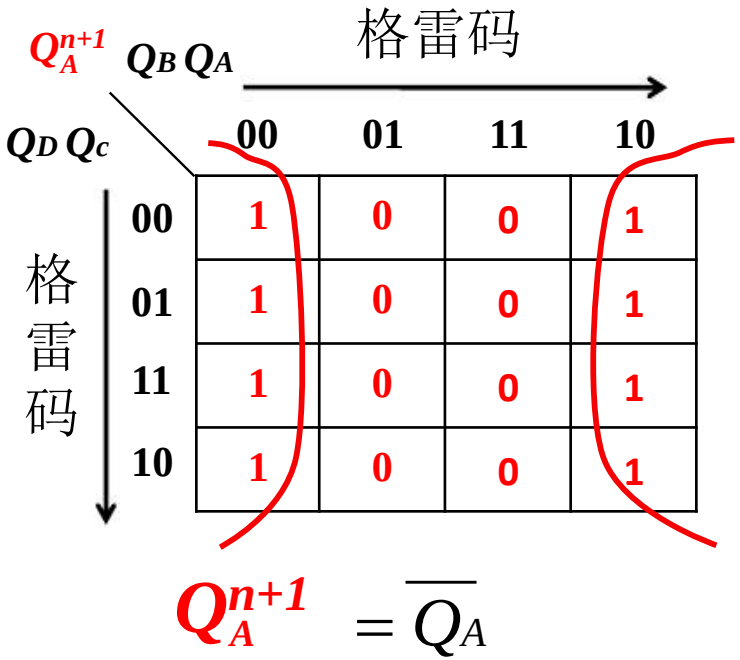
- **状态变化条件**
  - 计数触发
  - 无外部输入
- **输入激励**
  - 满足次态的输入要求
  - 输入方程
- **状态分配**
  - 计数值决定
- **触发器选择**
  - D触发器
  - 4个

	当前状态(现态)				下一状态(次态)				触发器激励(输入)			
	Q_A	Q_B	Q_C	Q_D	Q_A^{n+1}	Q_B^{n+1}	Q_C^{n+1}	Q_D^{n+1}	D_A	D_B	D_C	D_D
0	0	0	0	0	1	0	0	0	1	0	0	0
1	1	0	0	0	0	1	0	0	0	1	0	0
2	0	1	0	0	1	1	0	0	1	1	0	0
3	1	1	0	0	0	0	1	0	0	0	1	0
4	0	0	1	0	1	0	1	0	1	0	1	0
5	1	0	1	0	0	1	1	0	0	1	1	0
6	0	1	1	0	1	1	1	0	1	1	1	0
7	1	1	1	0	0	0	0	1	0	0	0	1
8	0	0	0	1	1	0	0	1	1	0	0	1
9	1	0	0	1	0	1	0	1	0	1	0	1
10	0	1	0	1	1	1	0	1	1	1	0	1
11	1	1	0	1	0	0	1	1	0	0	1	1
12	0	0	1	1	1	0	1	1	1	0	1	1
13	1	0	1	1	0	1	1	1	0	1	1	1
14	0	1	1	1	1	1	1	1	1	1	1	1
15	1	1	1	1	0	0	0	0	0	0	0	0

4位同步二进制计数器状态方程 Q_A^{n+1}

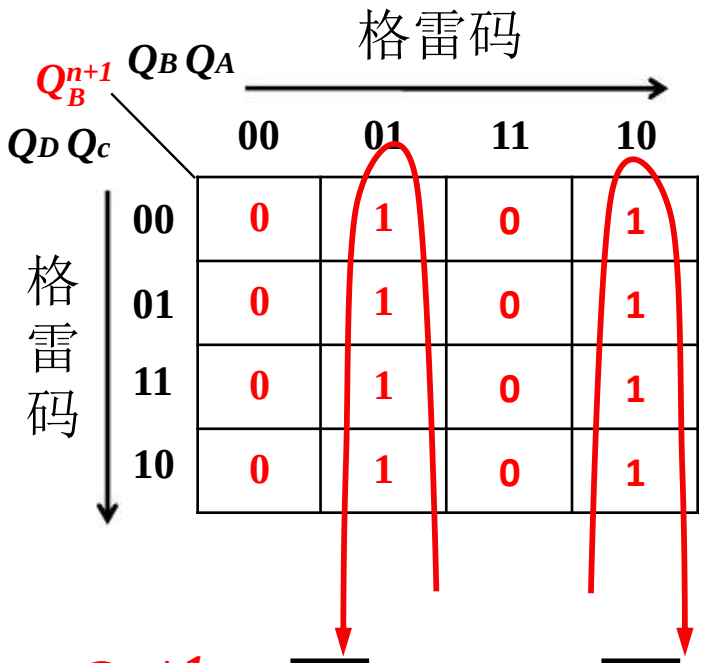
	Q_A	Q_B	Q_C	Q_D	Q_A^{n+1}	Q_B^{n+1}	Q_C^{n+1}	Q_D^{n+1}
0	0	0	0	0	1	0	0	0
1	1	0	0	0	0	1	0	0
2	0	1	0	0	1	1	0	0
3	1	1	0	0	0	0	1	0
4	0	0	1	0	1	0	1	0
5	1	0	1	0	0	1	1	0
6	0	1	1	0	1	1	1	0
7	1	1	1	0	0	0	0	1
8	0	0	0	1	1	0	0	1
9	1	0	0	1	0	1	0	1
10	0	1	0	1	1	1	0	1
11	1	1	0	1	0	0	1	1
12	0	0	1	1	1	0	1	1
13	1	0	1	1	0	1	1	1
14	0	1	1	1	1	1	1	1
15	1	1	1	1	0	0	0	0

根据状态真值表可写出触发器A的输出函数(状态方程) Q_A^{n+1} 的卡诺图和化简结果



4位同步二进制计数器状态方程 Q_B^{n+1}

	Q_A	Q_B	Q_C	Q_D	Q_A^{n+1}	Q_B^{n+1}	Q_C^{n+1}	Q_D^{n+1}
0	0	0	0	0	1	0	0	0
1	1	0	0	0	0	1	0	0
2	0	1	0	0	1	1	0	0
3	1	1	0	0	0	0	1	0
4	0	0	1	0	1	0	1	0
5	1	0	1	0	0	1	1	0
6	0	1	1	0	1	1	1	0
7	1	1	1	0	0	0	0	1
8	0	0	0	1	1	0	0	1
9	1	0	0	1	0	1	0	1
10	0	1	0	1	1	1	0	1
11	1	1	0	1	0	0	1	1
12	0	0	1	1	1	0	1	1
13	1	0	1	1	0	1	1	1
14	0	1	1	1	1	1	1	1
15	1	1	1	1	0	0	0	0

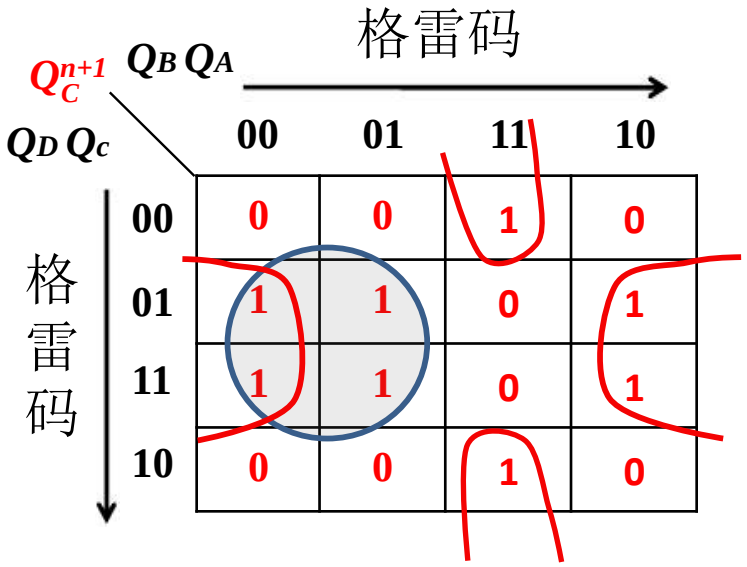


$$Q_B^{n+1} = \overline{Q_B} Q_A + Q_B \overline{Q_A}$$

$$= \overline{Q_A \oplus Q_B}$$

4位同步二进制计数器状态方程 Q_C^{n+1}

	Q_A	Q_B	Q_C	Q_D	Q_A^{n+1}	Q_B^{n+1}	Q_C^{n+1}	Q_D^{n+1}
0	0	0	0	0	1	0	0	0
1	1	0	0	0	0	1	0	0
2	0	1	0	0	1	1	0	0
3	1	1	0	0	0	0	1	0
4	0	0	1	0	1	0	1	0
5	1	0	1	0	0	1	1	0
6	0	1	1	0	1	1	1	0
7	1	1	1	0	0	0	0	1
8	0	0	0	1	1	0	0	1
9	1	0	0	1	0	1	0	1
10	0	1	0	1	1	1	0	1
11	1	1	0	1	0	0	1	1
12	0	0	1	1	1	0	1	1
13	1	0	1	1	0	1	1	1
14	0	1	1	1	1	1	1	1
15	1	1	1	1	0	0	0	0

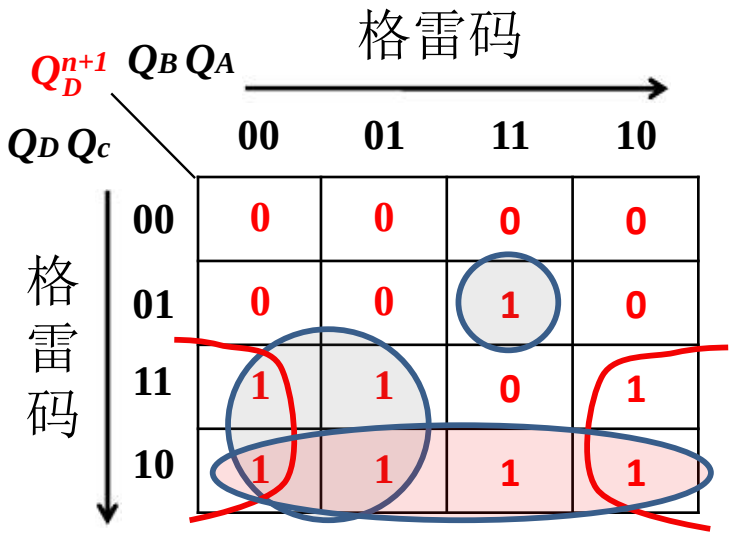


$$Q_C^{n+1} = \overline{Q_A}Q_C + \overline{Q_B}Q_C + Q_A Q_B \overline{Q_C}$$

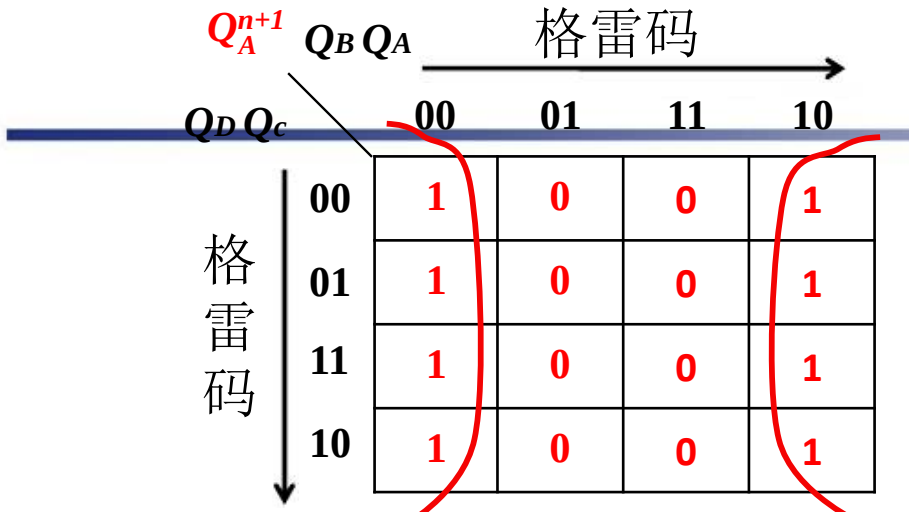
$$= \overline{\overline{(\overline{Q_A} + \overline{Q_B})} \oplus \overline{Q_C}}$$

4位二进制同步计数器设计原理：D_D

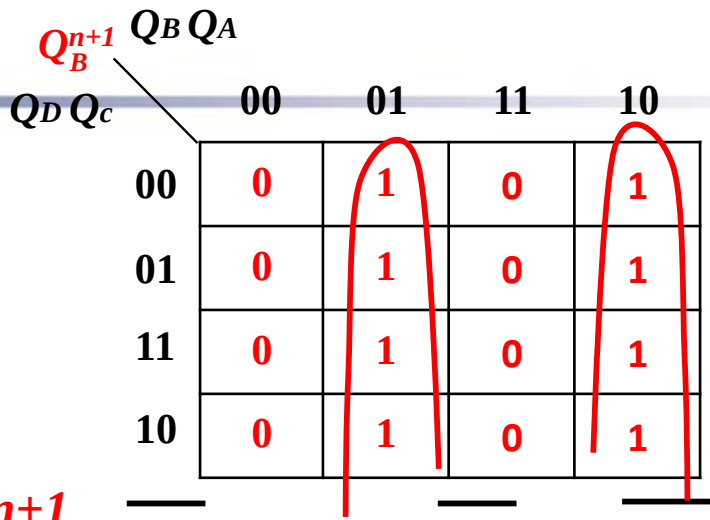
	Q_A	Q_B	Q_C	Q_D	Q_A^{n+1}	Q_B^{n+1}	Q_C^{n+1}	Q_D^{n+1}
0	0	0	0	0	1	0	0	0
1	1	0	0	0	0	1	0	0
2	0	1	0	0	1	1	0	0
3	1	1	0	0	0	0	1	0
4	0	0	1	0	1	0	1	0
5	1	0	1	0	0	1	1	0
6	0	1	1	0	1	1	1	0
7	1	1	1	0	0	0	0	1
8	0	0	0	1	1	0	0	1
9	1	0	0	1	0	1	0	1
10	0	1	0	1	1	1	0	1
11	1	1	0	1	0	0	1	1
12	0	0	1	1	1	0	1	1
13	1	0	1	1	0	1	1	1
14	0	1	1	1	1	1	1	1
15	1	1	1	1	0	0	0	0



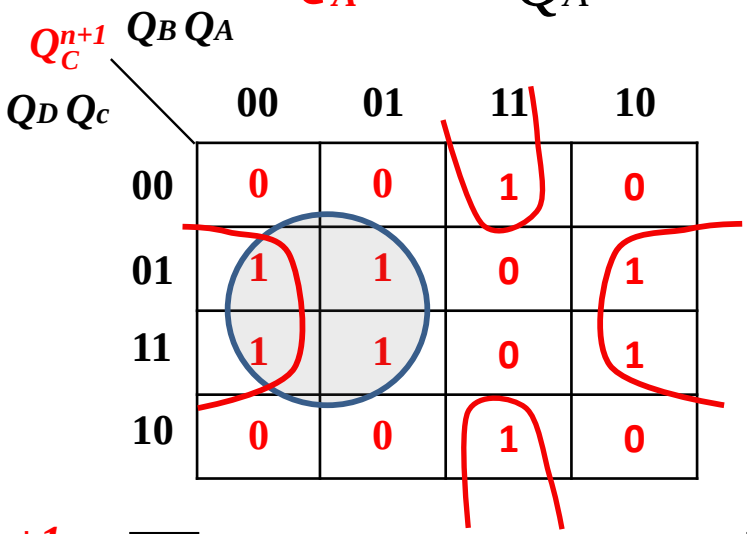
$$Q_D^{n+1} = \overline{Q_A}Q_D + \overline{Q_B}Q_D + \overline{Q_C}Q_D + Q_AQ_BQ_C\overline{Q_D}$$
$$= \overline{\overline{(Q_A + Q_B + Q_C)} \oplus \overline{Q_D}}$$



$$Q_A^{n+1} = \overline{Q_A}$$

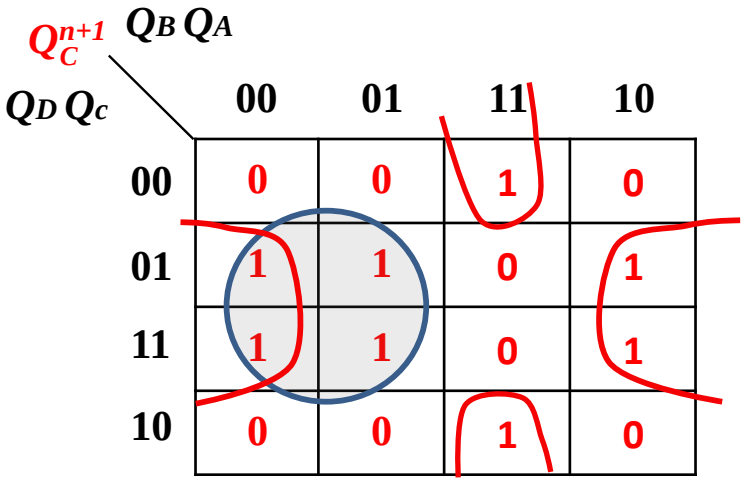


$$Q_B^{n+1} = \overline{Q_B}Q_A + Q_B\overline{Q_A} = \overline{Q_A \oplus Q_B}$$



$$Q_C^{n+1} = \overline{Q_A}Q_C + \overline{Q_B}Q_C + Q_AQ_B\overline{Q_C}$$

$$= \overline{\overline{(\overline{Q_A} + \overline{Q_B})} \oplus \overline{Q_C}}$$



$$Q_C^{n+1} = \overline{Q_A}Q_C + \overline{Q_B}Q_C + Q_AQ_B\overline{Q_C}$$

$$= \overline{\overline{(\overline{Q_A} + \overline{Q_B})} \oplus \overline{Q_C}}$$

激励函数(触发器输入方程)

◎ 同步时序电路采用D触发器实现，可以求出激励函数为：

- 根据D触发器特征方程：$Q_{n+1} = D_n$ 只有一个触发器的时序电路状态方程

$$D_A = \overline{Q_A}$$

$$D_B = \overline{Q_A}Q_B + Q_A\overline{Q_B} = \overline{Q_A \oplus Q_B}$$

$$D_C = \overline{Q_A}Q_C + \overline{Q_B}Q_C + Q_AQ_B\overline{Q_C} = \overline{(\overline{Q_A} + \overline{Q_B}) \oplus Q_C}$$

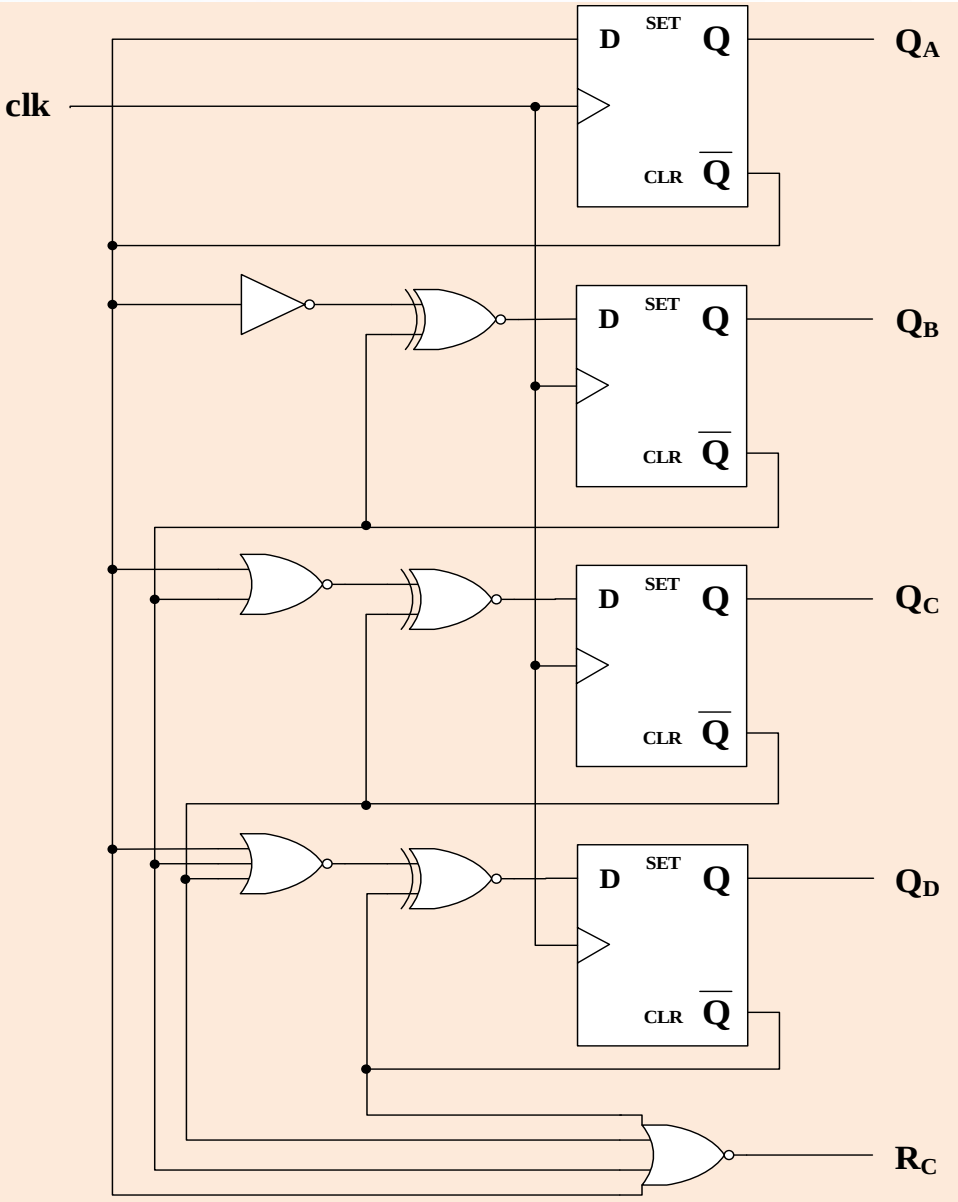
$$D_D = \overline{Q_A}Q_D + \overline{Q_B}Q_D + \overline{Q_C}Q_D + Q_AQ_BQ_C\overline{Q_D} = \overline{(\overline{Q_A} + \overline{Q_B} + \overline{Q_C}) \oplus Q_D}$$

- 增加一个4位进位输出R_C

$$R_C = \overline{\overline{Q_A} + \overline{Q_B} + \overline{Q_C} + \overline{Q_D}}$$

4位二进制同步计数器逻辑原理图

- **画出逻辑电路图**
  - 根据触发器输入方程
- **结构描述**
  - 根据电路结构写描述
  - 与电路图一一对应
  - 描述方法
    - 可用原理图直接描述
    - 也可以用HDL描述



模块结构描述

- **结构描述：与逻辑电路结构对应的描述**
  - 利用模块调用描述系统或部件
    1）调用内置门原语(在门级结构描述)；
    2）调用开关级原语(在晶体管开关级结构描述)；
    3）调用用户定义（UDP）的原语(在门级结构描述)；
    4）模块实例（创建层次结构结构描述)。
  - 与原理图对应的描述，加深时序电路的感性认识
- **实验中需要使用的主要门级单元模块(原语)**
  - 非门：

    名称：**INV**　　接口：**I，O**

    示例：**INV　　NotA (.I(a), .O(na));**
  - 二输入或非门：

    名称：**NOR2**　接口：**I0，I1，O**

    示例：**NOR2　G1 (.I0(a), .I1(b), .O(na_b));**

本实验使用到的原语汇总表

逻辑门	名称	接口	示例
非门	INV	I,O	INV NotA (.I(a), .O(na));
二输入或非门	NOR2	I0,I1,O	NOR2 G1(.I0(a), .I1(b), .O(c));
三输入或非门	NOR3	I0,I1,I2,O	
四输入或非门	NOR4	I0,I1,I2,I3,O	
二输入异或非门	XNOR2	I0,I1,O	XNOR2 G1(.I0(a), .I1(b), .O(c));
二输入与门	AND2	I0,I1,O	
二输入或门	OR2	I0,I1,O	
二输入与非门	NAND2	I0,I1,O	

D触发器	FD	C,clock
		D,data in
		Q dataout
			FD FFDA (.C(clk), .D(Da), .Q(Qa));
			defparam FFDA.INIT = 1'b0;
			// 定义D触发器的初值为0

4位同步计数器结构化描述：原语描述

```
module counter_4bit(clk, Qa, Qb, Qc, Qd, Rc);
……                    //端口及变量定义
FD                    FFDA (.C(clk),.D(Da),.Q(Qa)),
                      FFDB (.C(clk),.D(Db),.Q(Qb)),        也可调用实验九设计的D触发器
                      FFDC (.C(clk),.D(Dc),.Q(Qc)),
                      FFDD (.C(clk),.D(Dd),.Q(Qd));
defparam FFDA.INIT = 1'b0;              // define initial value of the D type Flip-Flop
defparam FFDB.INIT = 1'b0;
defparam FFDC.INIT = 1'b0;
defparam FFDD.INIT = 1'b0;
INV      GQa (.I(Qa), .O(nQa)),         //4个非门，  FD实例没有反向输出端
         GQb (.I(Qb), .O(nQb)),
         GQc (.I(Qc), .O(nQc)),
         GQd (.I(Qd), .O(nQd));
assign Da = nQa;                        //赋值描述

XNOR2    ODb (.I0(Qa), .I1(nQb), .O(Db)),                          //2输入异或非
         ODc (.I0(Nor_nQa_nQb), .I1(nQc), .O(Dc)),
         ODd (.I0(Nor_nQa_nQb_nQc), .I1(nQd), .O(Dd));

NOR4     ORc (.I0(nQa), .I1(nQb), .I2(nQc), .I3(nQd), .O(Rc));     //4输入或非门
NOR2     G1 (.I0(nQa), .I1(nQb), .O(Nor_nQa_nQb));                 //2输入或非门
NOR3     G2(.I0(nQa), .I1(nQb), .I2(nQc), .O(Nor_nQa_nQb_nQc));
endmodule
```

4位同步计数器结构化描述：门级描述

行为化结构描述

```
module counter_4bit(clk, rst, Qa, Qb, Qc, Qd, Rc);
…….                    //端口定义
wire Da, Db, Dc, Dd, nQa, nQb, nQc, nQc, Rc;
reg   Qa, Qb, Qc, Qd;

      assign Da = nQa;
      assign Db = ~(nQa^nQb);
      assign Dc = ~( (~(nQa | nQb)) ^ nQc);
      assign Dd = ~((~(nQa | nQb | nQc)) ^ nQd);
      assign Rc = ~(nQa | nQb | nQc | nQd));

     always @ (posedge clk)
            if (rst) {Qa,Qb,Qc,Qd} <= 4'b0000;          //同步清零
            else begin
                  Qa <= Da;
                  Qb <- Db;
                  Qc <= Dc;
                  Qd <= Dd;
            end
endmodule
```

四位同步二进制双向计数器激励方程

- 激励函数与结构化行为描述
  - 行为描述简单
  - 结构化有利于学习

```
module counter_4rev (clk, rst,s, cnt, Rc);
…….                          //端口定义
wire Da,Db,Dc,Dd,nQa,nQb,nQc,nQc, Rc;
reg  Qd , Qc, Qb, Qa;

     assign Da = ? ;
     assign Db = ? ;
     assign Dc = ? ;
     assign Dd = ? ;
     assign Rc = ? ;

     assign cnt = {Qd , Qc, Qb, Qa};

    always @ (posedge clk)
        if (rst) cnt <= 4'b0000; //同步清零
        else {Qd, Qc, Qb, Qa} <=
                              {Dd,Dc,Db,Da};

endmodule
```

S = 1时，正向计数，各触发器逻辑表达式同前面

S = 0时，反向计数，各触发器逻辑表达式如下式

根据理论课自行完成分析过程

$$D_A = \overline{Q_A}$$

$$D_B = \overline{S}(\overline{Q_A} \oplus \overline{Q_B}) + S(\overline{\overline{Q_A}} \oplus \overline{Q_B}) = \overline{S \oplus \overline{Q_A} \oplus \overline{Q_B}}$$

$$D_C = \overline{S}[(\overline{Q_A}\,\overline{Q_B}) \oplus \overline{Q_C}] + S[(\overline{\overline{Q_A}} + \overline{Q_B}) \oplus \overline{Q_C}]$$

$$= \overline{[\overline{S}\,\overline{Q_A}\,\overline{Q_B} + S(\overline{Q_A} + \overline{Q_B})] \oplus \overline{Q_C}}$$

$$= \overline{[\overline{S}(Q_A + Q_B) + S(\overline{Q_A} + \overline{Q_B})] \oplus \overline{Q_C}}$$

$$D_D = \overline{S}[(\overline{Q_A}\,\overline{Q_B}\,\overline{Q_C}) \oplus \overline{Q_D}] + S[(\overline{\overline{Q_A}} + \overline{Q_B} + \overline{Q_C}) \oplus \overline{Q_D}]$$

$$= \overline{[\overline{S}\,\overline{Q_A}\,\overline{Q_B}\,\overline{Q_C} + S(\overline{Q_A} + \overline{Q_B} + \overline{Q_C})] \oplus \overline{Q_D}}$$

$$= \overline{[\overline{S}(Q_A + Q_B + Q_C) + S(\overline{Q_A} + \overline{Q_B} + \overline{Q_C})] \oplus \overline{Q_D}}$$

$$R = \overline{S}\,\overline{Q_A}\,\overline{Q_B}\,\overline{Q_C}\,\overline{Q_D} + SQ_AQ_BQ_CQ_D$$

32位同步二进制可逆计数器：行为描述

◎ 同步二进制可逆计数器行为描述

```
module counter_32bit_rev(input wire clk, s,
                         output reg [31:0] cnt,    //32位计数器
                         output wire Rc);

// (~|cnt)：先对cnt的每一位进行“或”运算,再对结果取非。即cnt[31:0]=0时: (~|cnt)=1
// 反向计数，cnt=全0时借位：条件(~s & (~|cnt)) ==1，需s=0, cnt[3:0]=0。
// 正向计数，cnt=全1时进位：条件( s & ( &cnt)) ==1，需s=1, cnt[3:0]=F…
    assign Rc = (~s & (~|cnt)) | (s & (&cnt)); //进位或借位时
    always @ (posedge clk) begin
        if(s) cnt <= cnt + 1; //s==1时，正向计数
        else  cnt <= cnt - 1; //s==0时，反向计数
    end

endmodule
```

行为描述简单抽象：
不利于逻辑电路原理学习
实际设计中常采用行为描述

Course Outline

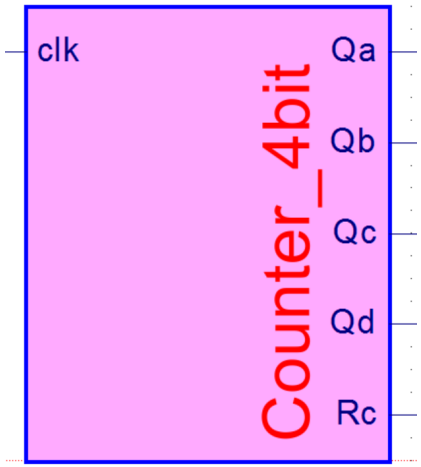
- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

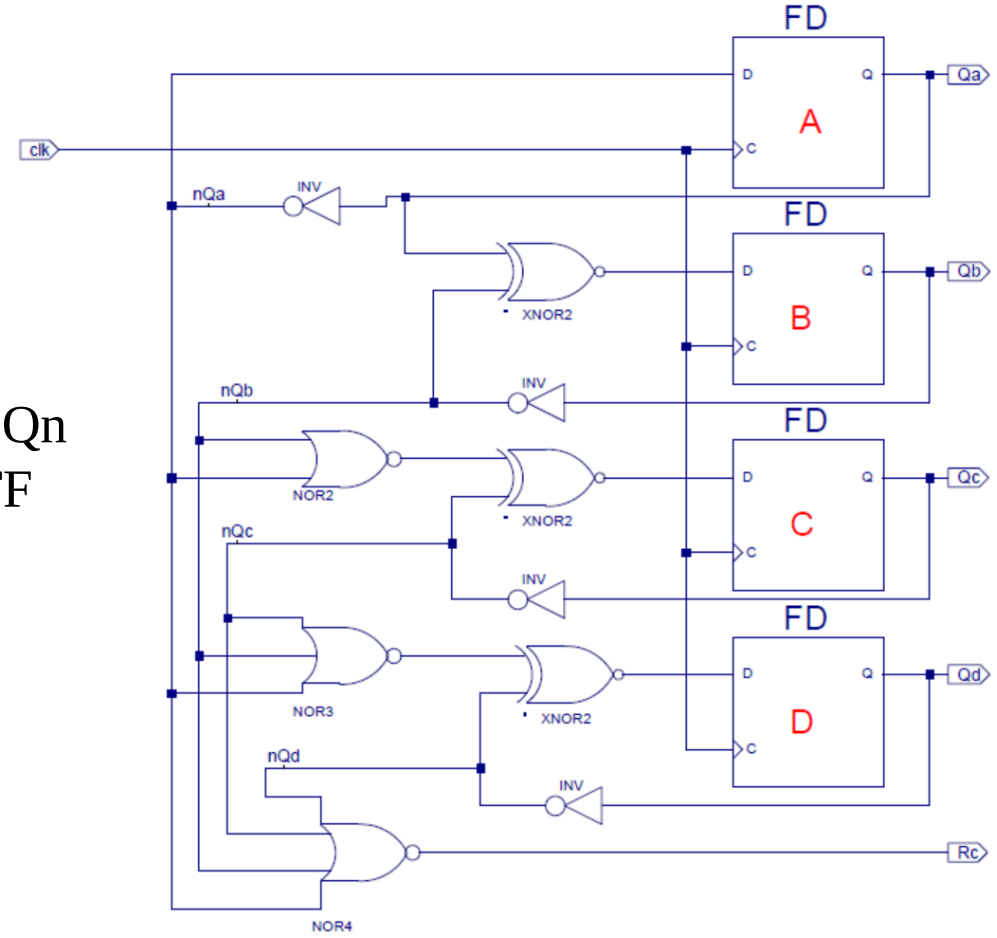
设计工程一：FSM

- **设计实现4位计数器**
  - 用状态机分析实现
  - 状态存储选用D触发器
  - 通过求解D触发器**激励方程**设计计数器时序逻辑电路
- **设计实现32位二进制双向计数器**
  - 增加计数器初始化功能
  - 初值用输入模块Ai，初值加载控制信号为Load
- **将上述二个计数模块集成到实验八的顶层模块**
  - 修改实验八顶层模块为：Top_FSM
  - 其余功能不变，新增功能：
    - 4位计数显示用LED(Sword/Arduino)
    - 32位双向计数器显示用通道3

设计要点

- **新建工程：FSM**
  - FSM=Finite state machine
- **设计4位计数器**
  - 用原理图输入实现
  - 选用D触发器实例：FD
    - **注意：**FD没有反向输出Qn
    - 也可用实验九的MB_DFF



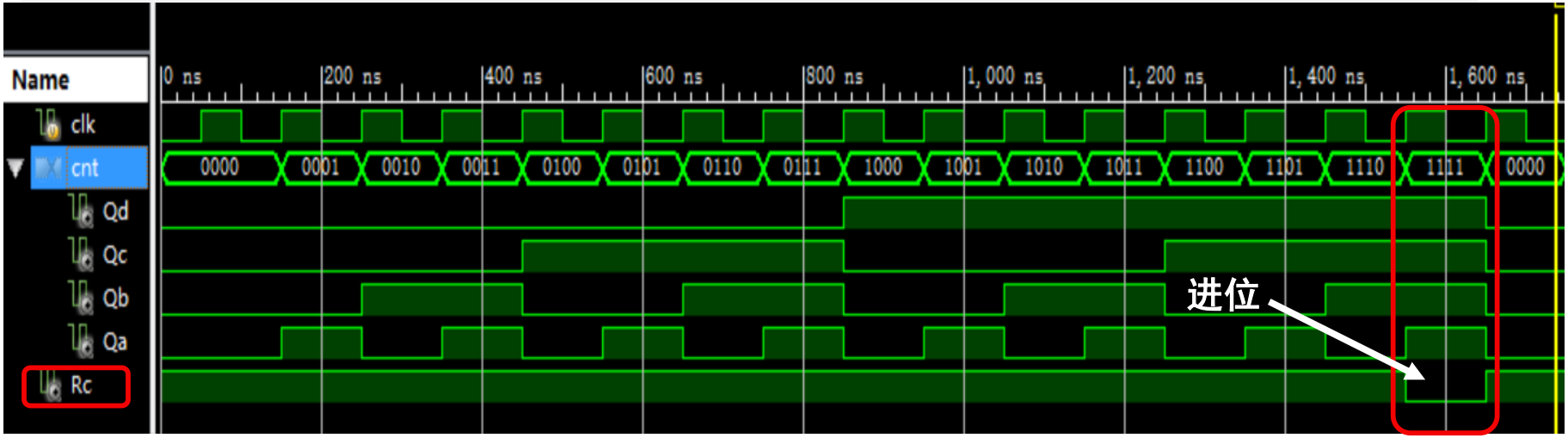


参考激励方程画逻辑图

4位同步二进制计数器激励与仿真

```
module counter_4bit_Test;
        ••••••
        initial begin                  //初始
        // Initialize Inputs
          clk = 0;
          forever #50 clk <= ~clk;
        end
endmodule
```

仿真时钟也可以这样获得

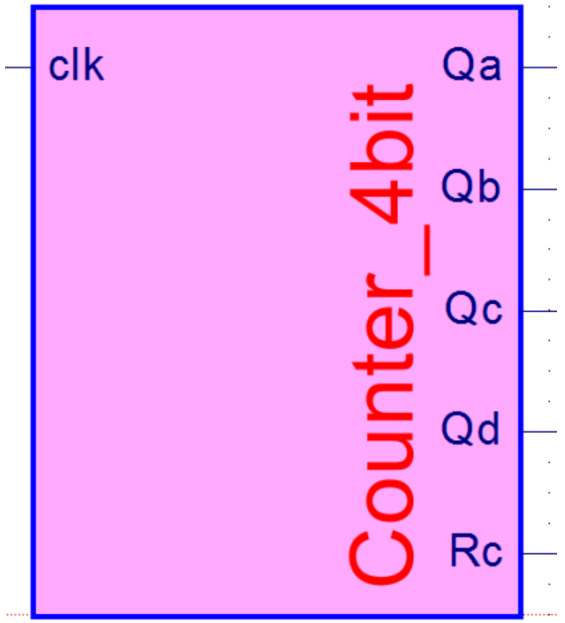


◎ 学习Veri代码描述

- 打开*View HDL Functional Model*分析学习模块的代码描述

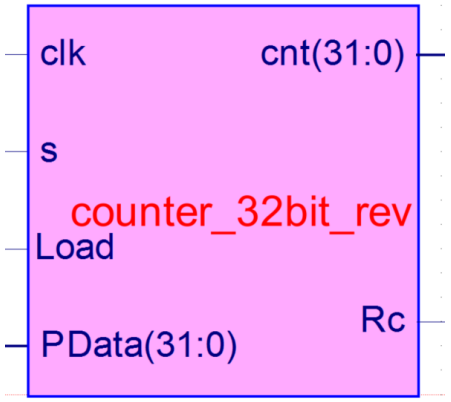
四位同步二进制计数器测试

- **输入时钟用1秒脉冲信号：**
  - 用通用分频输出：clkdiv[26]
  - 也可以设计1秒时钟辅助模块：clk_1s
- **仿真通过后封装逻辑符号 :Counter_4bit**
- **辅助模块：1秒定时器模块**
  - 输入100MHz的时钟信号
  - 输出周期为1秒的时钟信号
  - 用于计数器测试
- **顶层模块采用Top-FSM**
  - 采用实验八工程2的顶层模块
- **I/O引脚分配**
  - LED[6:2]=Rc_4,Qd,Qc,Qb,Qa



◎设计32位同步双向计数器

- 用行为描述实现：counter_32_rev.v
- 双向控制信号：s
- 增加计数器初始化功能
  - 初值Pdata[31:0]=输入模块Ai，
  - 初值加载控制信号为Load
- 参考描述




```
21  module counter_32bit_rev(input clk,              //时钟
22                           input s,                //计数方向
23                           input Load,             //计数初值加载控制
24                           input[31:0]PData,       //计数初值输入
25                           output reg[31:0]cnt,    //32计数器
26                           output reg Rc           //计数器溢出
27                           );
28
29  // assign Rc = (~s & (~|cnt)) | (s & (&cnt));
30     always @(posedge clk) begin
31        if(???)cnt <= ????;
32        else begin
33           if (s) cnt <= cnt + 1;      //s==1时，正向计数
34           else   cnt <= cnt - 1;      //s==0时，反向计数
35     //计数溢出也可以如下描述
36           if(条件1 |                  //cnt[31: 0]=32'h00000000,则|cnt=1
37              条件2) Rc<=1;            //cnt[31: 0]=32'hffffffff,则&cnt=1) Rc<=1;
38           else Rc<=0;
39        end
40     end
41
42  endmodule
```

32位同步二进制计数器激励与仿真

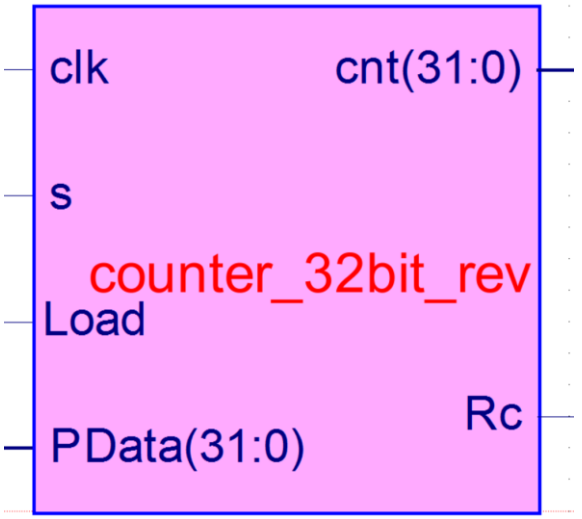
- 参考32位加法器仿真
- 仿真通过后封装逻辑符号

？ 仿真结果(请设计完成)

- 学习Veri代码描述
  - 打开*View HDL Functional Model*分析学习模块的代码描述

32位可逆同步计数器测试

- **输入时钟用1秒脉冲信号：**
  - 辅助模块：clk_1s
  - 也可以用通用分频输出：clkdiv[29]
- **仿真通过后封装逻辑符号 :Counter_32bit_rev**
- **顶层模块采用Top-FSM**
  - 采用实验八32位加法的顶层模块
- **I/O引脚分配**
  - S：控制计数方向，push_out(0)
  - Load：并行置入控制， push_out(1)
  - PData：并行输入数据Ai
  - cnt：32位计数输出接显示通道data3



辅助模块：1Hz的秒脉冲方波

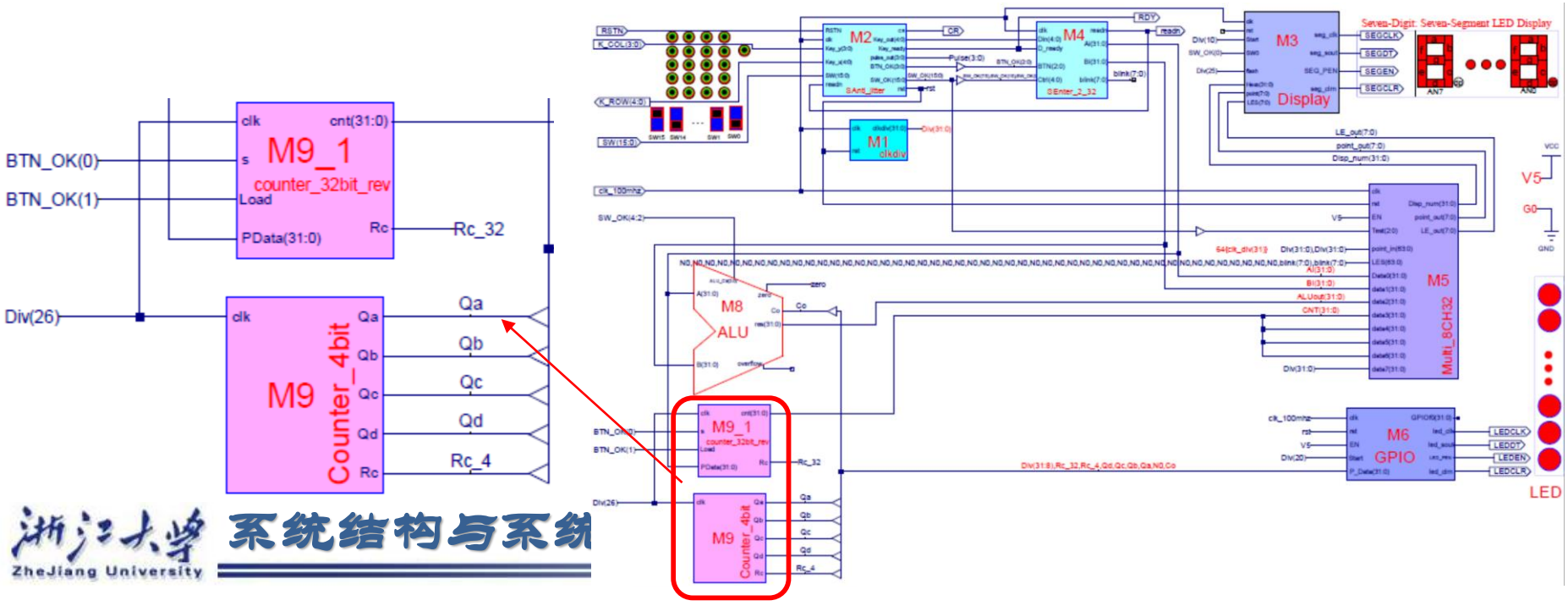
◎**100MHz**信号通过**50,000,000**次分频后，得到**1Hz**的秒脉冲方波，用于计数器的脉冲输入。

```
module counter_1s(clk, clk_1s);
  input wire clk;
  output reg clk_1s;
  reg [31:0] cnt;
    always @ (posedge clk) begin
        if (cnt < 50_000_000) begin    //50M*(1/100M)S=0.5S
            cnt <= cnt + 1;
        end else begin
            cnt <= 0;
            clk_1s <= ~clk_1s;         // clk_1s==1的时间=0.5s
            end                        // clk_1s==0的时间=0.5s
            end                        //周期T=1S
 endmodule
```

集成混合计算器：计数功能

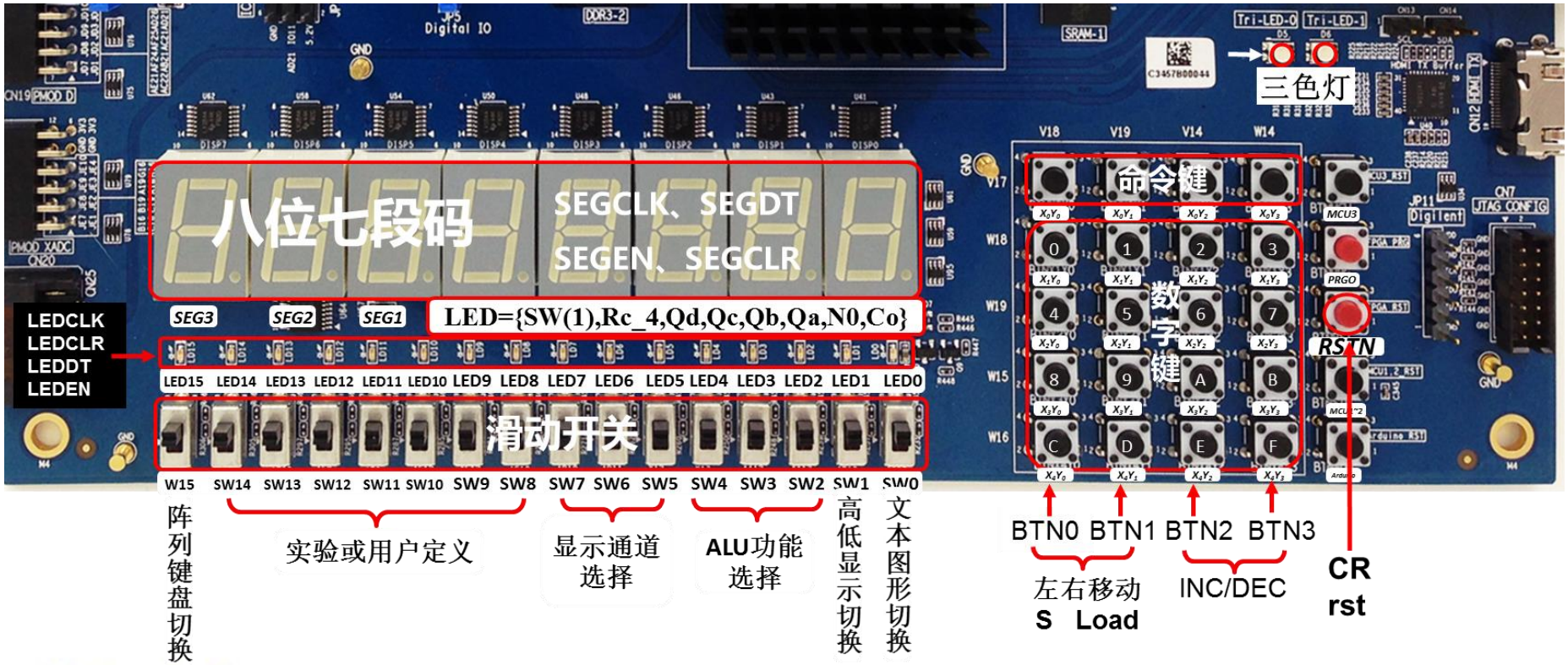
◎ 集成Calculation

- 复制实验八的顶层模块，并改名为：Top-FSM.sch
  - ⊙ 集成：4位计数器命名M9、32位计数器命名M9-1
- 接口分配
  - ⊙ M9：输入clk=Div[26]；输出LED={SW(1),Rc_4,Qd,Qc,Qb,Qa,N0,Co}
  - ⊙ M9-1：输入clk=Div[26]，s= BTN0 ， Load= BTN1 ， Pdata=Ai；输出cnt→显示通道3(data3)



物理验证

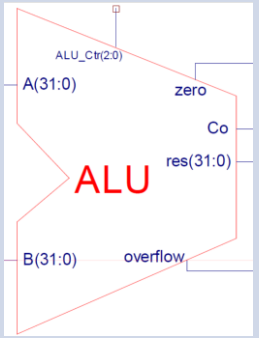
- 输入接口物理映射
  - SW[7:5] = 通道选择
    = 000：输入Ai；= 001：修改加数；=010： ALU输出
    = 011：32位计数输出



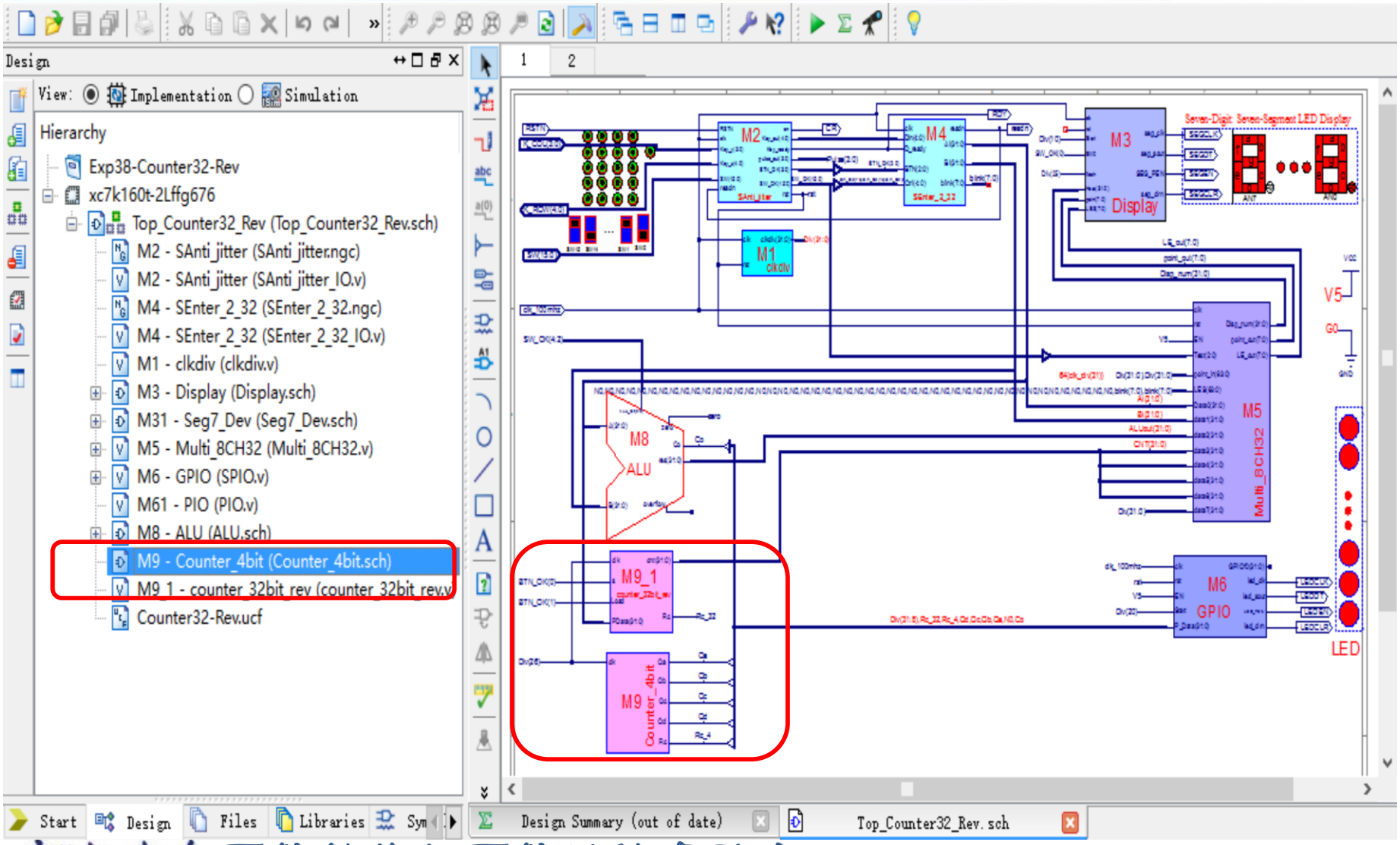
输入设备功能定义

开关定义	=0	=1	备注
SW[0]	未用		
SW[1]	32位二进制高16位	32位二进制低16位	
SW[4:2]	ALU功能选择	ALU功能选择	见ALU功能控制
SW[7:5]	通道选择 =000 =001 =010 …… =111	 通道0 通道1 通道2 **通道3** ……	 ***Ai*** ***Bi*** **RES(ALU_Out)** **Cnt**
按键定义	**=0**	**=1**	**备注**
Button[0]		计数方向	S
Button[1]		并行置入控制	**Load**
Button[2]			
Button[3]			

ALU及计数器功能控制

ALU功能选择	=0（XXX）	=1（功能）	备注
SW[4:2]= ALU_Ctr(2:0)	000 001 010 011 100 101 110 111	与 或 加 自定义 自定义 自定义 减 Slt	SW[4]=0  SW[4]=1 A<B，A=1，否则A=0
Button[0]=s	减1计数	加1计数	输入模块是移动
Button[1]=Load	计数	初始化	输入模块是移动

实验七的顶层结构


Design
View: Implementation Simulation
Hierarchy
Exp38-Counter32-Rev
xc7k160t-2Lffg676
Top_Counter32_Rev (Top_Counter32_Rev.sch)
M2 - SAnti_jitter (SAnti_jitter.ngc)
M2 - SAnti_jitter (SAnti_jitter_IO.v)
M4 - SEnter_2_32 (SEnter_2_32.ngc)
M4 - SEnter_2_32 (SEnter_2_32_IO.v)
M1 - clkdiv (clkdiv.v)
M3 - Display (Display.sch)
M31 - Seg7_Dev (Seg7_Dev.sch)
M5 - Multi_8CH32 (Multi_8CH32.v)
M6 - GPIO (SPIO.v)
M61 - PIO (PIO.v)
M8 - ALU (ALU.sch)
M9 - Counter_4bit (Counter_4bit.sch)
M9_1 - counter_32bit_rev (counter_32bit_rev.v)
Counter32-Rev.ucf
M2
M4
M3
Display
M1
clkdiv
M8
ALU
M5
Multi_8CH32
M9_1
M9
Counter_4bit
M6
GPIO
LED
V5
GND
Start
Design
Files
Libraries
Design Summary (out of date)
Top_Counter32_Rev.sch

同学们：每次做完实验请整理好实验台，放好仪器，理清桌面。


Thank you!