EDA技术实用教程

第3章

VHDL设计初步

3.1 组合电路的VHDL描述

3.1.1 2选1多路选择器及其VHDL描述1

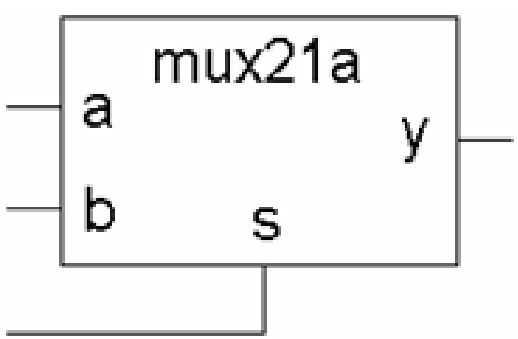


图 3-1 mux21a 实体

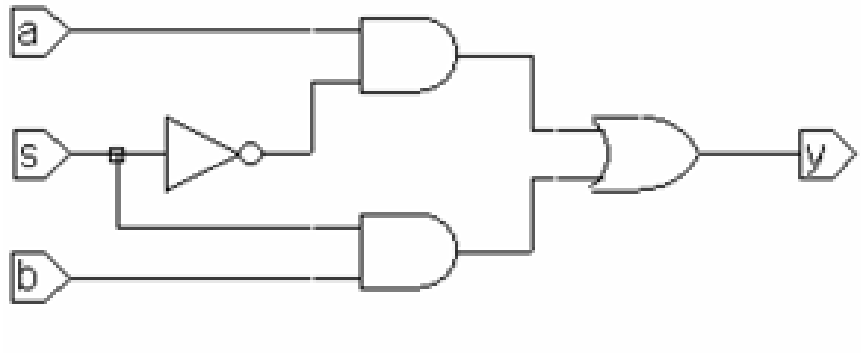


图 3-2 mux21a 结构体

3.1 组合电路的VHDL描述

3.1.1 2选1多路选择器及其VHDL描述1

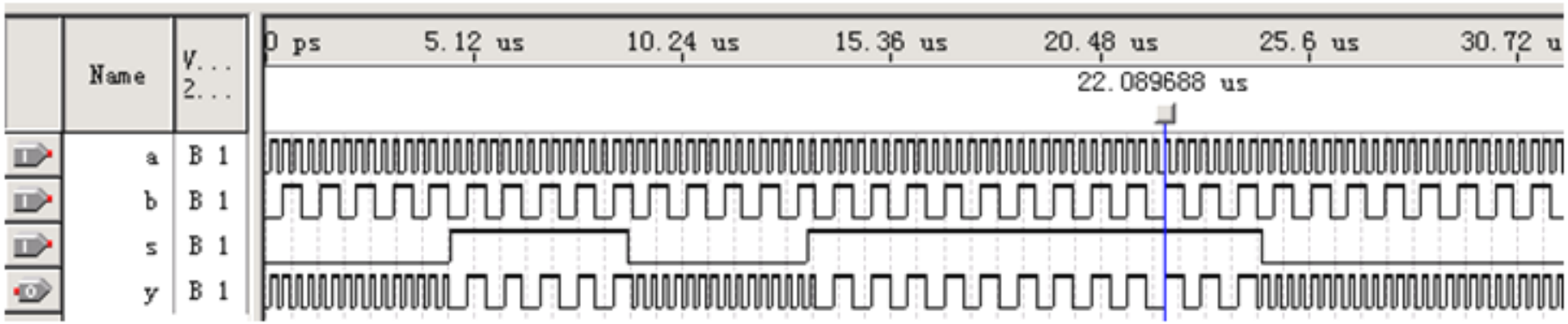


图 3-3　mux21a 电路的时序波形

3.1 组合电路的VHDL描述

【例 3-1】

VHDL表述的二选一多路选择器完整电路模块程序

VHDL实体描述部分

电路模块端口说明和定义段

```
ENTITY mux21a IS
   PORT (
          a : IN  BIT;
          b : IN  BIT;
          s : IN  BIT;
          y : OUT BIT
         );
END ENTITY mux21a;
```

VHDL结构体描述部分

电路模块逻辑功能描述部分

```
ARCHITECTURE one OF mux21a IS
   BEGIN
    y <= a  WHEN  s = '0'
            ELSE  b ;
END ARCHITECTURE one ;
```

3.1 组合电路的VHDL描述

3.1.1 2选1多路选择器及其VHDL描述1

1. 实体表达

```
ENTITY  e_name  IS
  PORT  ( p_name :  port_m   data_type;
                ...
            p_namei : port_mi   data_type );
END ENTITY e_name;
```

3.1 组合电路的VHDL描述

2. 实体名

【例 3-2】

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
ENTITY mux21a IS
   PORT ( a, b, s :  IN  STD_LOGIC;
                  y : OUT STD_LOGIC  );
END ENTITY mux21a;
ARCHITECTURE one OF mux21a IS
     SIGNAL e :  STD_LOGIC;
     SIGNAL d :  STD_LOGIC;
  BEGIN
    d <= a AND (NOT S) ;
    e <= b AND s ;
    y <= d OR e  ;
END ARCHITECTURE one ;
```

3.1 组合电路的VHDL描述

3. 端口语句和端口信号名

4. 端口模式

(1) IN：输入端口。

(2) OUT：输出端口。

(3) INOUT：双向端口。

(4) BUFFER：缓冲端口。

3.1 组合电路的VHDL描述

5. 数据类型

6. 结构体表达

```
ARCHITECTURE arch_name OF e_name IS
  [说明语句]
BEGIN
  (功能描述语句)
END ARCHITECTURE arch_name ;
```

3.1 组合电路的VHDL描述

7. 赋值符号和数据比较符号

```
RU  : OUT   BOOLEAN;
   ...
RU  <=  TRUE WHEN (C<D) ELSE FALSE ;  --C,D是端口输入的普通类型数据
```

8. WHEN_ELSE条件信号赋值语句

赋值目标 <= 表达式 WHEN 赋值条件 ELSE
表达式 WHEN 赋值条件 ELSE
...
表达式 ;

```
z  <= a WHEN p1 = '1' ELSE
      b WHEN p2 = '1' ELSE
      c  ;
```

3.1 组合电路的VHDL描述

9．关键字

10．标识符

11．规范的程序书写格式

12．文件取名和存盘

3.1 组合电路的VHDL描述

3.1.2 2选1多路选择器及其VHDL描述2

【例 3-2】

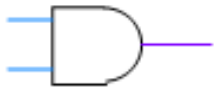
```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
ENTITY mux21a IS
   PORT ( a, b, s :  IN  STD_LOGIC;
                  y : OUT STD_LOGIC  );
END ENTITY mux21a;
ARCHITECTURE one OF mux21a IS
     SIGNAL e :  STD_LOGIC;
     SIGNAL d :  STD_LOGIC;
  BEGIN
    d <= a AND (NOT S) ;
    e <= b AND s ;
    y <= d OR e  ;
END ARCHITECTURE one ;
```

3.1 组合电路的VHDL描述

3.1.2 2选1多路选择器及其VHDL描述2

1. 逻辑操作符

表 3-1 VHDL 逻辑操作符

逻辑操作符	逻辑图形	逻辑功能	逻辑操作符	逻辑图形	逻辑功能
AND		逻辑与操作	NAND		逻辑与非操作
OR		逻辑或操作	NOR		逻辑或非操作
XOR		逻辑异或操作	XNOR		逻辑同或操作
			NOT		逻辑取非操作

3.1 组合电路的VHDL描述

3.1.2 2选1多路选择器及其VHDL描述2

2. 标准逻辑位数据类型STD_LOGIC

```
TYPE BIT IS('0','1');  --只有两种取值
```

```
TYPE STD_LOGIC IS ('U','X','0','1','Z','W','L','H','-'); --有九种取值
```

3.1 组合电路的VHDL描述

3.1.2 2选1多路选择器及其VHDL描述2

3. 设计库和标准程序包

```
LIBRARY  WORK ;
LIBRARY  STD ;
USE STD.STANDARD.ALL ;
```

```
LIBRARY  <设计库名>;
USE  < 设计库名>.<程序包名>.ALL ;
```

```
LIBRARY  IEEE ;
USE  IEEE.STD_LOGIC_1164.ALL ;
```

4. 信号定义和数据对象

3.1 组合电路的VHDL描述

3.1.3 2选1多路选择器及其VHDL描述3

【例 3-3】

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
ENTITY mux21a IS
  PORT ( a, b, s : IN  IN STD_LOGIC;
                 y : OUT IN STD_LOGIC  );
END ENTITY mux21a;
ARCHITECTURE one OF mux21a IS
 BEGIN
     PROCESS (a,b,s)    BEGIN
          IF s='0'  THEN  y <= a ;
             ELSE  y <= b ;
          END IF;
     END PROCESS;
END ARCHITECTURE one ;
```

3.1 组合电路的VHDL描述

3.1.3 2选1多路选择器及其VHDL描述3

【例 3-3】

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
ENTITY mux21a IS
  PORT ( a, b, s : IN  IN STD_LOGIC;
                 y : OUT IN STD_LOGIC  );
END ENTITY mux21a;
ARCHITECTURE one OF mux21a IS
 BEGIN
     PROCESS (a,b,s)    BEGIN
          IF s='0'  THEN  y <= a ;
             ELSE  y <= b ;
          END IF;
     END PROCESS;
END ARCHITECTURE one ;
```

3.1 组合电路的VHDL描述

3.1.3 2选1多路选择器及其VHDL描述3

1. 条件语句

```
IF a THEN ...                              -- 注意，a的数据类型必须是BOOLEAN
IF (s1='0')AND(s2='1')OR(c<b+1) THEN ...
```

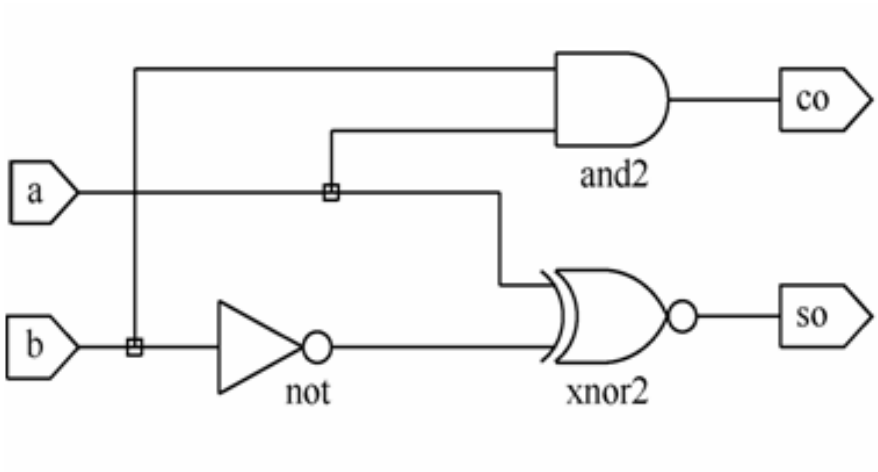
2. 进程语句和顺序语句

IF_THEN_ELSE_END IF；

PROCESS…END PROCESS

3.1 组合电路的VHDL描述

3.1.4 半加器及其VHDL的描述



a	b	so	co
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

图 3-4　半加器 h_adder 电路图及其真值表

3.1 组合电路的VHDL描述

3.1.4 半加器及其VHDL的描述

【例 3-4】

```
LIBRARY  IEEE;    --半加器描述(1)：布尔函数描述方法
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY h_adder IS
  PORT (a, b : IN STD_LOGIC;
        co, so : OUT STD_LOGIC);
END ENTITY  h_adder;
ARCHITECTURE fh1 OF h_adder  is
BEGIN
  so <= NOT(a XOR (NOT b)) ;   co <= a AND b ;
END ARCHITECTURE fh1;
```

【例 3-5】

```
LIBRARY  IEEE;   --半加器描述(2)：真值表描述方法
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY h_adder IS
PORT (a, b : IN STD_LOGIC;
      co, so : OUT STD_LOGIC);
END ENTITY h_adder;
ARCHITECTURE fh1 OF h_adder  is
  SIGNAL abc : STD_LOGIC_VECTOR(1 DOWNTO 0);--定义标准逻辑位矢量数据类型
BEGIN
    abc <= a & b ;   -- a 相并 b，即 a 与 b 并置操作，获得二维矢量数据类型
  PROCESS(abc)   BEGIN
      CASE abc IS                    -- 类似于真值表表述方式的 CASE 语句
       WHEN "00" => so<='0'; co<='0' ;
       WHEN "01" => so<='1'; co<='0' ;
       WHEN "10" => so<='1'; co<='0' ;
       WHEN "11" => so<='0'; co<='1' ;
       WHEN OTHERS => NULL ;
      END CASE;
   END PROCESS;
END ARCHITECTURE fh1 ;
```

3.1 组合电路的VHDL描述

3.1.4 半加器及其VHDL的描述

1. CASE语句

```
CASE <表达式> IS
When <选择值或标识符> => <顺序语句>; ... ; <顺序语句> ;
When <选择值或标识符> => <顺序语句>; ... ; <顺序语句> ;
...
WHEN OTHERS => <顺序语句>;
END CASE ;
```

3.1 组合电路的VHDL描述

3.1.4 半加器及其VHDL的描述

2. 标准逻辑矢量数据类型

```
        B : OUT  STD_LOGIC_VECTOR(7 DOWNTO 0);
或      SIGNAL A : STD_LOGIC_VECTOR(1 TO 4)


B <= "01100010" ;              -- 其中 B(7)为 '0'
B（4 DOWNTO 1）<= "1101" ;     -- 其中 B(4)为 '1'
B（7 DOWNTO 4）<= A ;          -- 其中 B(6)等于 A(2)

 SIGNAL C : BIT_VECTOR(3 DOWNTO 0);
```

3.1 组合电路的VHDL描述

3.1.4 半加器及其VHDL的描述

3. 并置操作符 &

```
SIGNAL a : STD_LOGIC_VECTOR (3 DOWNTO 0);
SIGNAL d : STD_LOGIC_VECTOR (1 DOWNTO 0);
...
a  <= '1'& '0' & d(1)& '1'  ; -- 元素与元素并置，并置后的数组长度为 4
...
IF a & d = "101011"  THEN ... -- 在 IF 条件句中可以使用并置符
```

3.1 组合电路的VHDL描述

3.1.5 一位二进制全加器及其VHDL描述

3. 并置操作符 &

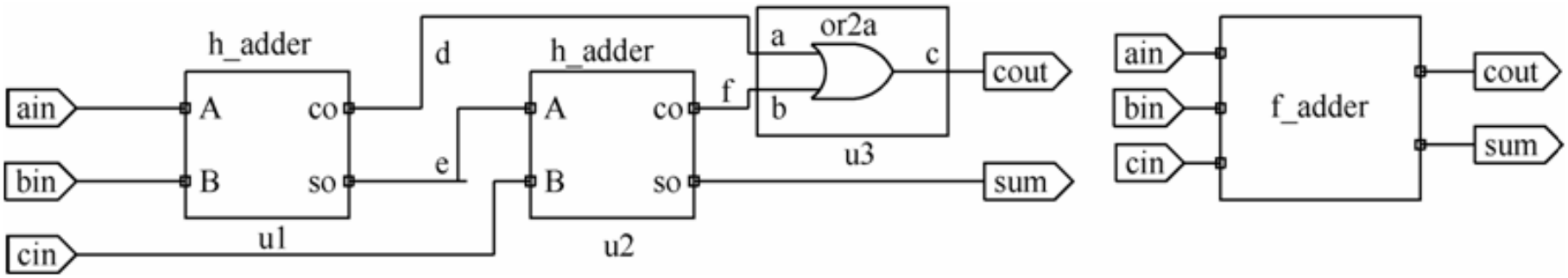


图 3-5 全加器 f_adder 电路图及其实体模块

3.1 组合电路的VHDL描述

3.1.5 一位二进制全加器及其VHDL描述

3. 并置操作符 &

【例 3-6】

```
LIBRARY  IEEE ;    --或门逻辑描述
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY or2a IS
  PORT (a, b :IN STD_LOGIC;
             c : OUT STD_LOGIC );
END ENTITY or2a;
ARCHITECTURE one OF or2a IS
  BEGIN
  c <= a OR b ;
END ARCHITECTURE one ;
```

【例 3-7】

```
LIBRARY  IEEE;   --1 位二进制全加器顶层设计描述
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY f_adder IS
  PORT (ain, bin, cin  : IN STD_LOGIC;
            cout, sum   : OUT STD_LOGIC );
END ENTITY f_adder;
ARCHITECTURE fd1 OF f_adder IS
  COMPONENT h_adder                    --调用半加器声明语句
    PORT (a, b :  IN STD_LOGIC;
          co, so :  OUT STD_LOGIC);
  END COMPONENT ;
  COMPONENT or2a
     PORT (a, b : IN STD_LOGIC;
              c : OUT STD_LOGIC);
  END COMPONENT;
SIGNAL d, e, f  :  STD_LOGIC; --定义 3 个信号作为内部的连接线。
 BEGIN
  u1 : h_adder PORT MAP(a=>ain, b=>bin, co=>d, so=>e);  --例化语句
  u2 : h_adder PORT MAP(a=>e,   b=>cin,  co=>f, so=>sum);
  u3 :   or2a   PORT MAP(a=>d,   b=>f,    c=>cout);
END ARCHITECTURE fd1;
```

3.1 组合电路的VHDL描述

3.1.6 VHDL例化语句

```
COMPONENT 元件名 IS
    PORT （端口名表);
END COMPONENT 文件名 ;
```

```
COMPONENT h_adder
    PORT （ c，d :  IN STD_LOGIC;
            e，f :  OUT STD_LOGIC）;
```

例化名 : 元件名 PORT MAP（ [端口名 =>] 连接端口名,...）;

3.2 基本时序电路的VHDL描述

3.2.1 D触发器的VHDL描述

【例 3-8】

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
ENTITY DFF1 IS
  PORT (CLK,D : IN STD_LOGIC ;
              Q : OUT STD_LOGIC );
END ;
ARCHITECTURE bhv OF DFF1 IS
 SIGNAL Q1 : STD_LOGIC ;   --类似于在芯片内部定义一个数据的暂存节点
 BEGIN
  PROCESS (CLK,Q1)        --由 PROCESS 进程语句引导一个顺序语句结构
   BEGIN
    IF  CLK'EVENT AND CLK = '1'
        THEN  Q1 <= D ;
    END IF;               --IF 语句结束
    END PROCESS ;         --PROCESS 进程语句结束
      Q <= Q1 ;           --将内部的暂存数据向端口输出（--是注释符号）
END bhv;
```

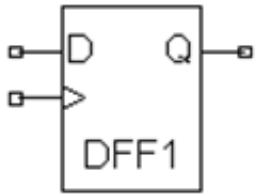


图 3-6 D 触发器

3.2 基本时序电路的VHDL描述

3.2.1 D触发器的VHDL描述

1. 上升沿检测表达式和信号属性函数EVENT

<信号名>'EVENT

2. 不完整条件语句与时序电路

【例 3-9】

```
ENTITY COMP_BAD IS
  PORT( , b  : IN BIT;   q  : OUT BIT    );
END ;
ARCHITECTURE one OF COMP_BAD IS
  BEGIN
CMP:  PROCESS (a,b)  BEGIN -- CMP 是当前进程的标号或名称，不参与综合
      IF  a > b  THEN  q <= '1' ;
     ELSIF a < b  THEN  q <= '0' ;-- 注意未提及当 a=b 时，q 作何操作
      END IF;
  END PROCESS ;
END ;
```

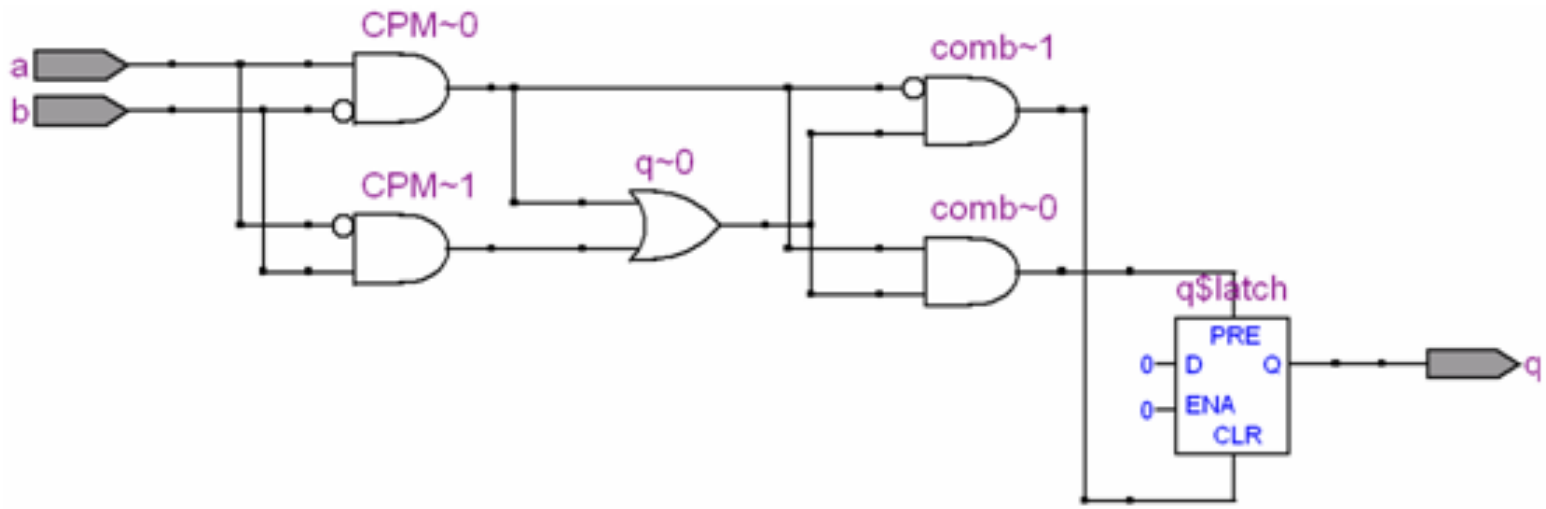


图 3-7 例 3-9 的电路图

3.2 基本时序电路的VHDL描述

【例 3-10】

```
IF  a > b  THEN  q <= '1' ;
           ELSE   q <= '0' ;   END IF;
```

图 3-8 例 3-10 的电路图

3.2 基本时序电路的VHDL描述

3.2.2 VHDL实现时序电路的不同表述

```
CLK'EVENT AND (CLK='1') AND (CLK'LAST_VALUE='0')
```

【例 3-11】

```
IF  (CLK'EVENT AND CLK='1') AND (CLK'LAST_VALUE='0')
    THEN  Q <= D ;    --确保 CLK 的变化是一次上升沿的跳变
END IF;
```

【例 3-12】

```
IF  CLK='1' AND CLK'LAST_VALUE ='0'
   THEN  Q <= D ;
END IF;
```

3.2 基本时序电路的VHDL描述

3.2.2 VHDL实现时序电路的不同表述

【例 3-13】

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
      ...
 IF  rising_edge(CLK) -- 注意使用此函数必须打开 STD_LOGIC_1164 程序包
    THEN  Q1 <= D ;
 END IF;
```

【例 3-14】

```
WREG: PROCESS
        BEGIN
        wait until CLK = '1'   ;     --利用 wait 语句
         Q <= D ;
      END PROCESS;
```

3.2 基本时序电路的VHDL描述

3.2.2 VHDL实现时序电路的不同表述

【例 3-15】

```
PROCESS (CLK)   BEGIN  --可以将 BEGIN 与 PROCESS 写在一行
    IF  CLK = '1'
    THEN  Q <= D ;     --利用进程的启动特性产生对 CLK 的边沿检测
    END IF;
END PROCESS ;
```

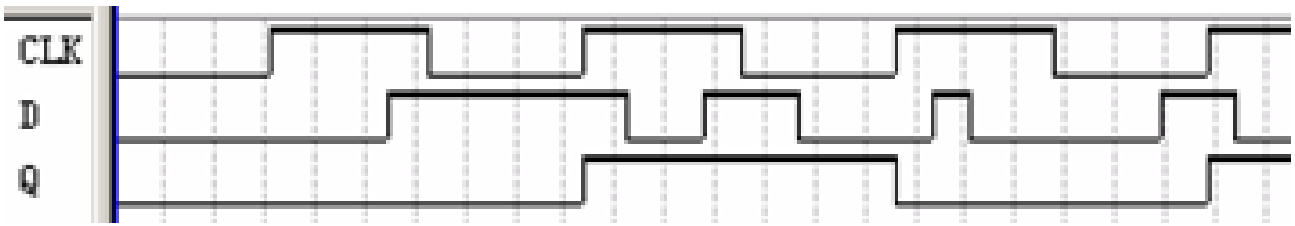


图 3-9　例 3-15 D 触发器时序波形

3.2 基本时序电路的VHDL描述

3.2.2 VHDL实现时序电路的不同表述

【例 3-16】

```
  PROCESS (CLK, D)                --注意敏感信号 D 的作用
   BEGIN
      IF  CLK = '1'               --电平触发型寄存器
      THEN  Q <= D ;
      END IF;
END PROCESS ;
```

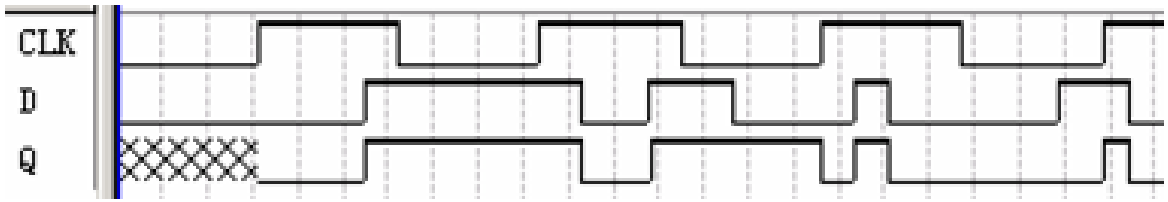


图 3-10 例 3-16 的时序波形

3.2 基本时序电路的VHDL描述

3.2.3 异步时序电路设计

【例 3-17】

```
    SIGNAL Q1,Q2 : STD_LOGIC;
    . . .
      QQ <= Q2 ;
PRO1: PROCESS（CLK）   BEGIN
       IF  CLK'EVENT AND CLK='1'
         THEN  Q1 <= NOT （Q2 OR A）;
       END IF;
      END PROCESS ;
PRO2: PROCESS （Q1）   BEGIN
       IF  Q1'EVENT AND Q1='1'   THEN  Q2 <= D;   END IF;
      END PROCESS ;
```

3.2 基本时序电路的VHDL描述

3.2.3 异步时序电路设计

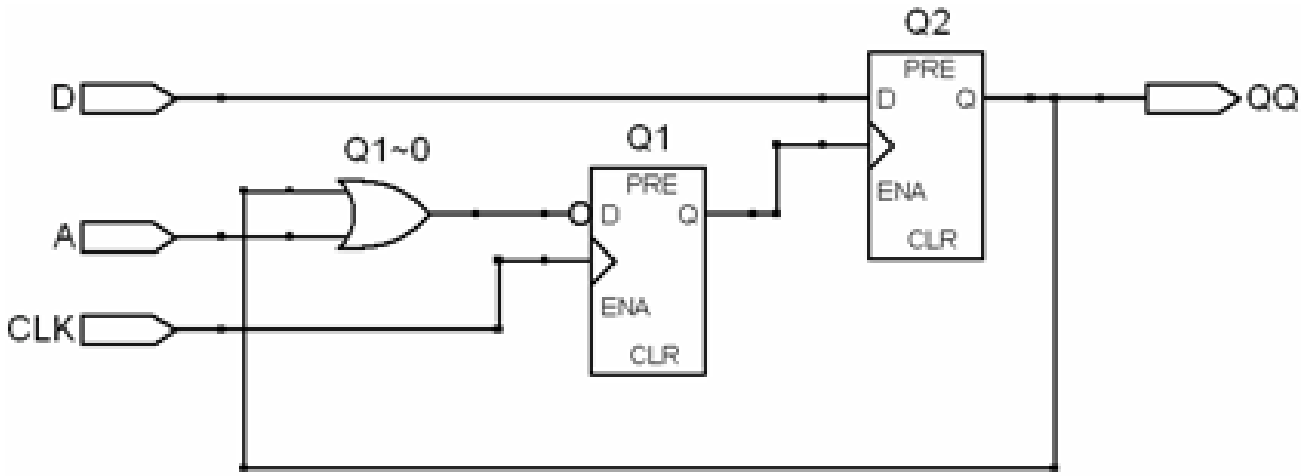


图 3-11 例 3-17 综合后的电路

3.3 计数器的VHDL设计

【例 3-18】

```
ENTITY CNT4 IS
  PORT （ CLK : IN BIT ;
          Q  : BUFFER INTEGER RANGE 15 DOWNTO 0  ）;
  END ;
ARCHITECTURE bhv OF CNT4 IS
  BEGIN
  PROCESS （CLK）  BEGIN
     IF  CLK'EVENT AND CLK = '1'  THEN
       Q <= Q + 1 ;   END IF;
  END PROCESS ;
END bhv;
```

3.3 计数器的VHDL设计

3.3.1 4位二进制加法计数器设计

3.3.2 整数类型

1	十进制整数
0	十进制整数
35	十进制整数
10E3	十进制整数，等于十进制整数 1000
16#D9#	十六进制整数，等于十六进制数 D9H
8#720#	八进制整数，等于八进制数 720O
2#11010010#	二进制整数，等于二进制数 11010010B

```
Q : BUFFER NATURAL RANGE 15 DOWNTO 0;
```

3.3 计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

【例 3-19】

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL ;
USE IEEE.STD_LOGIC_UNSIGNED.ALL ;
ENTITY CNT4 IS
PORT (CLK : IN STD_LOGIC ;
        Q  : OUT STD_LOGIC_VECTOR(3 DOWNTO 0) ) ;
END ;
ARCHITECTURE bhv OF CNT4 IS
   SIGNAL Q1 : STD_LOGIC_VECTOR(3 DOWNTO 0);
BEGIN
  PROCESS(CLK)   BEGIN
     IF  CLK'EVENT AND CLK = '1'
        THEN   Q1 <= Q1 + 1 ;
     END IF;
  END PROCESS ;
     Q <= Q1 ;
END bhv;
```

3.3 计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

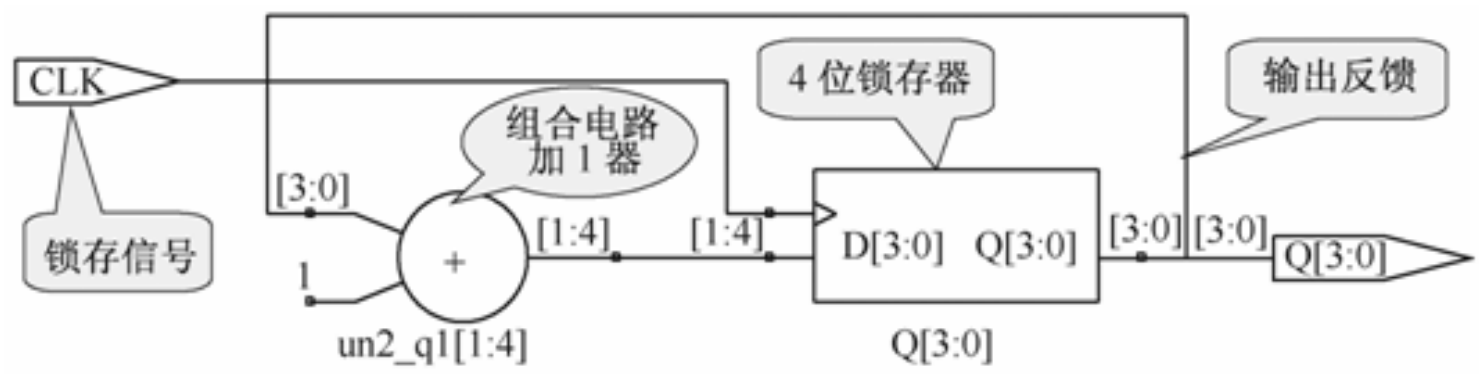


图 3-12　4 位加法计数器 RTL 电路（Symplify 综合）

3.3 计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

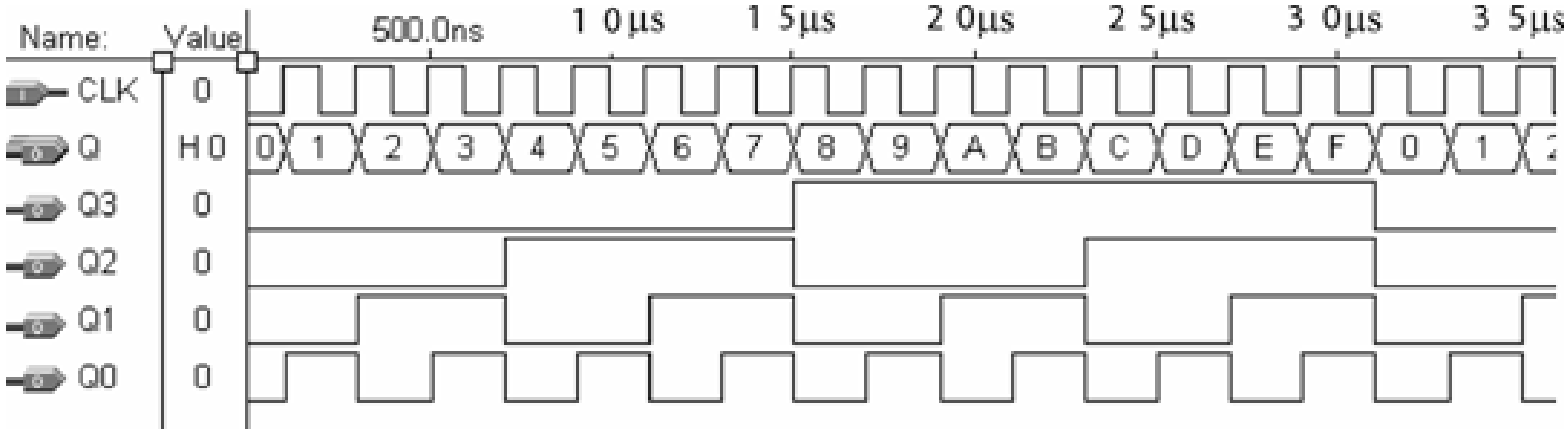


图 3-13　4 位加法计数器工作时序

3.4 实用计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

【例 3-20】

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY CNT10 IS
     PORT (CLK,RST,EN,LOAD : IN STD_LOGIC;
           DATA : IN STD_LOGIC_VECTOR(3 DOWNTO 0);  --4 位预置数
           DOUT : OUT STD_LOGIC_VECTOR(3 DOWNTO 0); --计数值输出
           COUT : OUT STD_LOGIC  );                 --计数进位输出
END CNT10;
ARCHITECTURE behav OF CNT10 IS
BEGIN
  PROCESS(CLK, RST, EN, LOAD)
     VARIABLE  Q : STD_LOGIC_VECTOR(3 DOWNTO 0);
  BEGIN
```

接下页

3.4 实用计数器的VHDL设计

接上页

```
IF RST='0' THEN   Q := (OTHERS=>'0'); --复位低电平时，计数寄存器清 0
   ELSIF CLK'EVENT AND CLK='1' THEN  --测试时钟上升沿
     IF EN='1' THEN                          --计数使能高电平，允许计数
        IF (LOAD='0') THEN  Q := DATA; ELSE  --预置控制低电平，允许加载
           IF Q<9 THEN   Q := Q + 1; --计数小于 9，继续累加
              ELSE   Q := (OTHERS=>'0'); --否则计数清 0
           END IF;
        END IF;
     END IF;
END IF;
IF Q="1001" THEN COUT<='1';              --当计数为 9 时，进位输出 1
         ELSE    COUT<='0';  END IF;    --否则进位输出 0
    DOUT <= Q;                           --计数寄存器的值输出端口
END PROCESS;
END behav;
```

3.4 实用计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

1. 十进制计数器相关语法

```
  VARIABLE  CQI: STD_LOGIC_VECTOR(3 DOWNTO 0)

SIGNAL    d1  : STD_LOGIC_VECTOR(4 DOWNTO 0);
VARIABLE  a1 : STD_LOGIC_VECTOR(15 DOWNTO 0);
...
d1 <= (OTHERS=>'1');   a1 := (OTHERS=>'0') ;

d1 <= (1=>e(3),3=>e(5), OTHERS=>e(1) );

d1 <= e(1) & e(5) & e(1) & e(3) & e(1) ;
```

3.4 实用计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

2. 程序分析

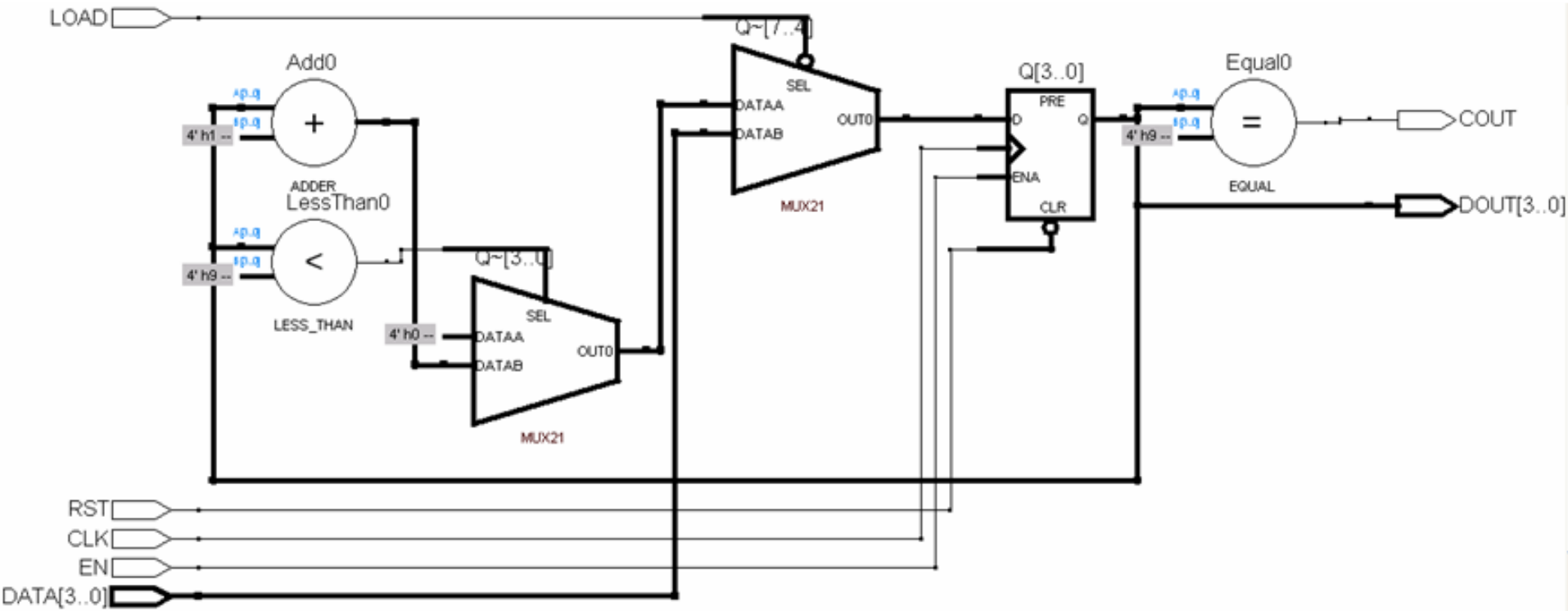


图 3-14　例 3-22 的 RTL 电路图

3.4 实用计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

2. 程序分析

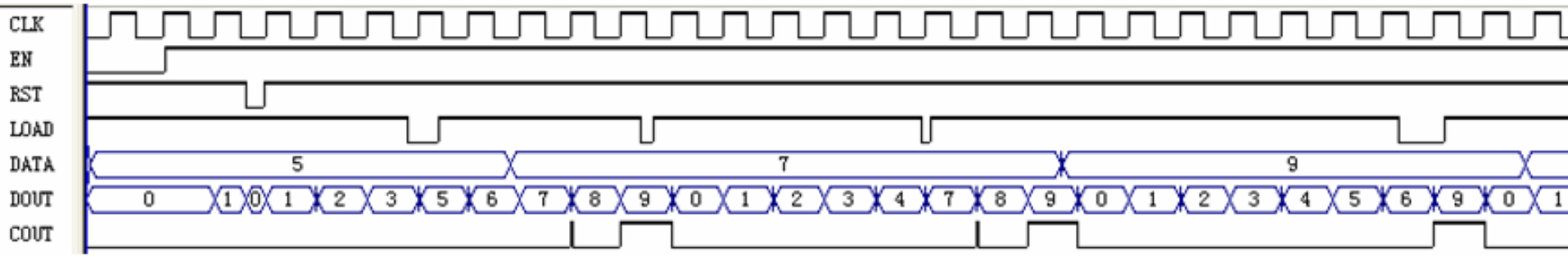


图 3-15　例 3-22 的工作时序

3.4 实用计数器的VHDL设计

3.3.3 计数器的其他VHDL表达方式

3. 时序模块中的同步控制信号和异步控制信号的构建

4. 另一种描述方式

【例 3-21】

```
SIGNAL  Q : STD_LOGIC_VECTOR(3 DOWNTO 0);
   . . .
  REG:  PROCESS(CLK, RST, EN,Q,LOAD)   BEGIN
     IF RST='0' THEN   Q <= (OTHERS=>'0') ;
       ELSIF CLK'EVENT AND CLK='1' THEN
        IF EN='1' THEN
           IF (LOAD='0') THEN  Q <= DATA;  ELSE
              IF Q<9 THEN   Q <= Q + 1;
                   ELSE   Q <= (OTHERS=>'0');
               END IF;
           END IF;
         END IF;
        END IF;
    END PROCESS;
       DOUT <= Q;
  COM:  PROCESS(Q)   BEGIN
        IF Q="1001" THEN COUT<='1';  ELSE    COUT<='0';  END IF;
   END PROCESS;
```

习　题

3-1 画出与以下实体描述对应的原理图符号元件：

```
ENTITY buf3s IS                              -- 实体 1：  三态缓冲器
    PORT （input : IN STD_LOGIC ;     -- 输入端
          enable : IN STD_LOGIC ;      -- 使能端
          output : OUT STD_LOGIC ）;  -- 输出端
END buf3x ;
ENTITY mux21 IS        --实体 2：  2 选 1 多路选择器
    PORT （in0,  in1, sel :  IN STD_LOGIC;
           output : OUT STD_LOGIC）;
```

习　题

3-2　图**3-16**所示的是**4**选**1**多路选择器，试分别用**IF_THEN**语句、**WHEN_ELSE**和**CASE**语句的表达方式写出此电路的**VHDL**程序，要求选择控制信号**s1**和**s0**的数据类型为**STD_LOGIC**；当**s1='0'**，**s0='0'**；**s1='0'**，**s0='1'**；**s1='1'**，**s0='0'**和**s1='1'**，**s0='1'**时，分别执行**y<=a**、**y<=b**、**y<=c**、**y<=d**。

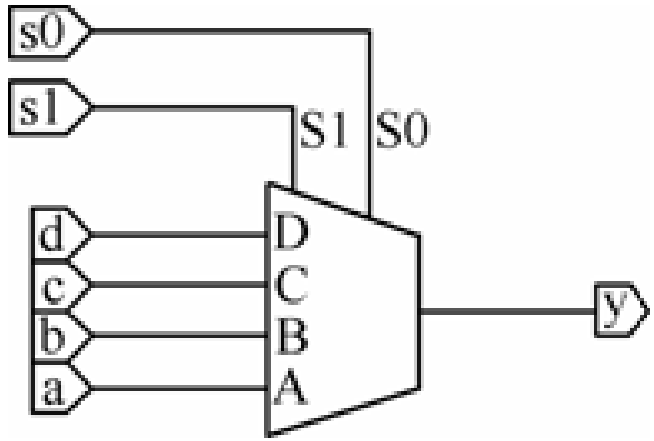


图 3-16　4 选 1 多路选择器

习　题

3-3　图**3-17**所示的是双**2**选**1**多路选择器构成的电路**MUXK**，对于其中**MUX21A**，当**s='0'**和**s='1'**时，分别有**y<='a'**和**y<='b'**。试在一个结构体中用两个进程来表达此电路，每个进程中用**CASE**语句描述一个**2**选**1**多路选择器**MUX21A**。

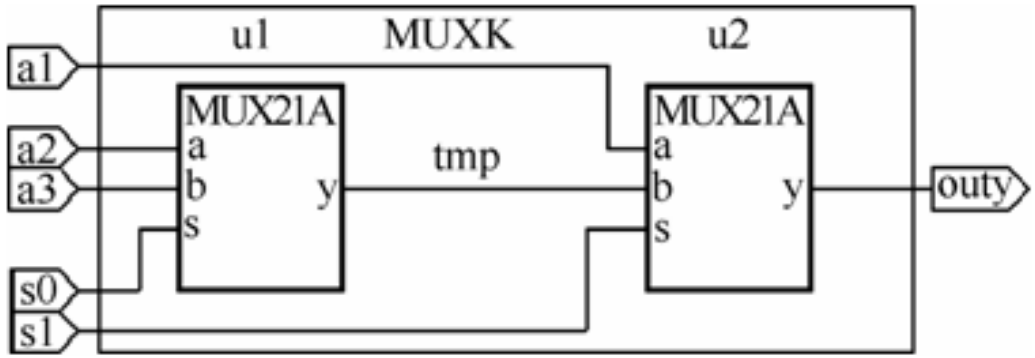


图 3-17　题 3-3 电路

习　题

3-4 将**3-20**程序的计数器改为**12**进制计数器，程序用例**3-21**的方式表述，并且将复位**RST**改为同步清**0**控制，加载信号**LOAD**改为异步控制方式。讨论例**3-20**与例**3-21**的异同点。

3-5 设计含有异步清零和计数使能的**16**位二进制加减可控计数器。

3-6 图**3-18**是一个含有上升沿触发的**D**触发器的时序电路，试写出此电路的**VHDL**设计文件。

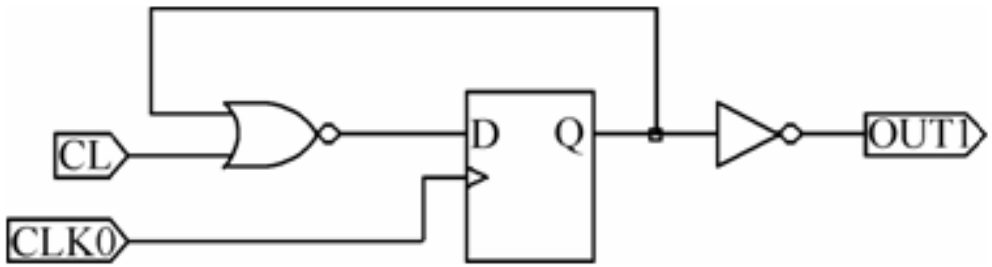


图 3-18 时序电路图

习　题

3-7 给出**1**位全减器的**VHDL**描述。要求：
（**1**）首先设计**1**位半减器，然后用例化语句将它们连接起来，图**3-19**中**h_ suber**是半减器，**diff**是输出差，**s _out**是借位输出，**sub _in**是借位输入。
（**2**）根据图**3-19**设计**1**位全减器。以**1**位全减器为基本硬件，构成串行借位的**8**位减法器，要求用例化语句来完成此项设计（减法运算是 **x – y – sun _in = diffr**）。

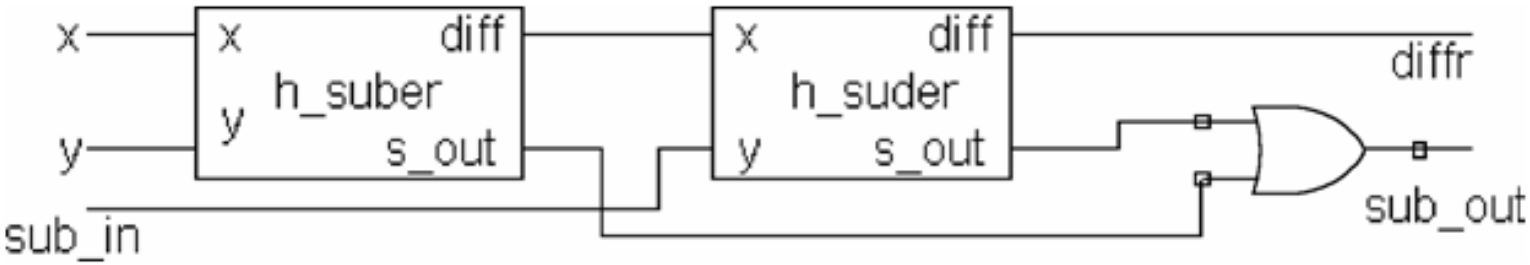


图 3-19　1 位全减器

习　题

3-8 给出一个**4**选**1**多路选择器的**VHDL**描述。选通控制端有**4**个输入：**S0**、**S1**、**S2**、**S3**。当且仅当**S0=0**时：**Y=A**；**S1=0**时：**Y=B**；**S2=0**时：**Y=C**；**S3=0**时：**Y=D**。

3-9 分频方法有多种，最简单的是二分频和偶数分频甚至奇数分频，这用触发器或指定计数模的计数器即可办到。但对于现场实现指定分频比或小数分频率的分频电路的设计就不是很简单了。
试对例**3-20**的设计稍作修改，将其进位输出**COUT**与异步加载控制**LOAD**连在一起，构成一个自动加载型**16**位二进制数计数器，也即一个**16**位可控的分频器，给出其**VHDL**表述，并说明工作原理。设输入频率**fi=4MHz**，输出频率**fo=516.5±1Hz**（允许误差±**0.1Hz**），**16**位加载数值=？。

3-10 用**VHDL**设计一个功能类似**74LS160**的计数器。

3-11 给出含有异步清零和计数使能的**16**位二进制加减可控计数器的**VHDL**描述。

习　题

3-12 分别给出以下2个**RTL**图的**VHDL**描述，注意其中的**D**触发器和锁存器的表述。