第5章

可编程器件实验——提高篇

本章学习目标

- 学习 VHDL 复杂程序设计。
- 锻炼 CPLD 综合开发设计能力。

本章通过 CPLD 综合性实验开发项目使设计者提高软硬件资源的综合应用能力。

5.1 实验一 组合逻辑电路设计

一、实验目的

(1) 掌握组合逻辑电路的设计方法。
(2) 掌握组合逻辑电路的静态测试方法。
(3) 熟悉 CPLD 设计的过程,比较原理图输入和文本输入的优劣。

二、实验硬件

(1) 输入:按键开关(常高)4 个;拨码开关 4 位。
(2) 输出:LED 灯。
(3) 主芯片:Altera EPM240T100C5N。

三、实验内容

(1) 设计一个四舍五入判别电路,其输入为 8421BCD 码,要求当输入大于或等于 5

时，判别电路输出为 1，反之为 0。

（2）设计四个开关控制一盏灯的逻辑电路，要求改变任意开关的状态能够引起灯亮灭状态的改变。（即任一开关的合断改变原来灯亮灭的状态）

（3）设计一个优先排队电路，其排队顺序如下：

A＝1　最高优先级

B＝1　次高优先级

C＝1　最低优先级

要求输出端最多只能有一端为 1，即只能是优先级较高的输入端所对应的输出端为 1。

四、实验原理

1. 四舍五入判别电路

（1）原理图：采用原理图输入方式，如图 5.1 所示。

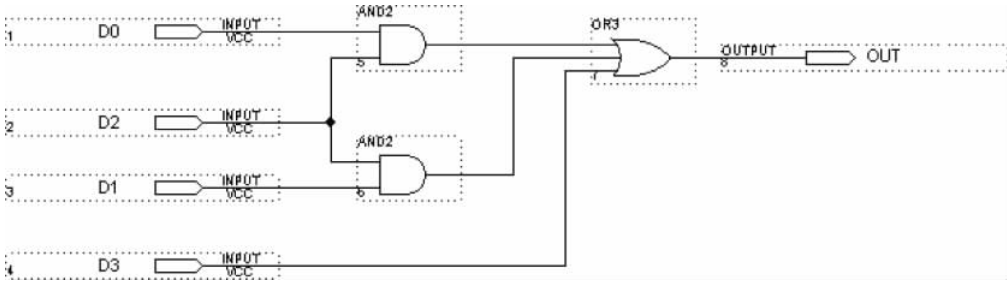

图 5.1　四舍五入判别电路原理图

（2）连线：四位拨码开关连 D3、D2、D1、D0 信号对应的管脚。OUT 输出信号管脚接 LED 灯。

2. 四个开关控制一盏灯的逻辑电路

（1）原理图：采用原理图输入方式，如图 5.2 所示。

（2）连线：四位按键开关分别连 K0、K1、K2、K3 信号对应的管脚。OUT 输出信号管脚接 LED 灯。

3. 优先排队电路

（1）原理图：采用原理图输入方式，如图 5.3 所示。

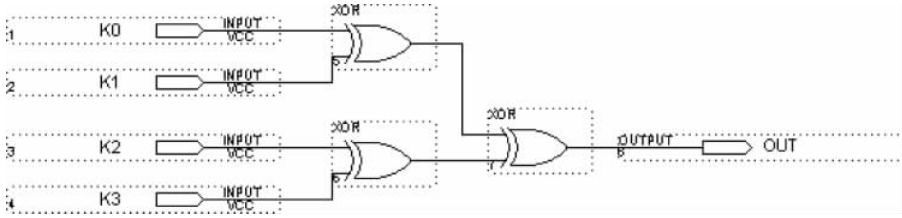

图 5.2　四个开关控制一盏灯的逻辑电路原理图

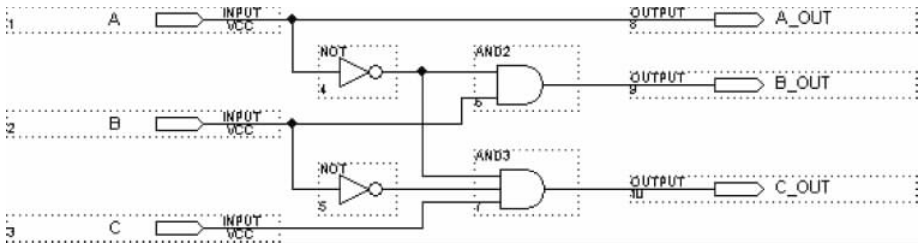

图 5.3　优先排队电路原理图

（2）连线：A、B、C 信号对应管脚分别连三个按键开关。输出 A_Out、B_Out、C_Out 信号对应的管脚分别连三个 LED 发光二极管。

注：具体管脚参数由底层管脚编辑决定。

五、实验报告要求

（1）描述实验的基本原理和步骤。

（2）用数据和图片给出各个步骤中取得的实验结果，并对实验结果及实验过程中出现的问题进行分析总结。

（3）完整的源代码。

六、参考源程序

1. 四舍五入判别电路 VHDL 程序

```
SUBDESIGN t5_1
(
    d0,d1,d2,d3:INPUT;
    out: OUTPUT;
```

```
)
BEGIN
    IF( (d3,d2,d1,d0) >= 5 ) THEN
        out = VCC;
    ELSE
        out = GND;
     END IF;
END;
```

2. 四个开关控制一盏灯的逻辑电路 VHDL 程序

```
SUBDESIGN t5_2
(
    k0,k1,k2,k3:INPUT;
    out: OUTPUT;
)
BEGIN
 TABLE
      (k3,k2,k1,k0)  =>  out;
      B"0000"        =>  GND;
      B"0001"        =>  VCC;
      B"0011"        =>  GND;
      B"0010"        =>  VCC;
      B"0110"        =>  GND;
      B"0111"        =>  VCC;
      B"0101"        =>  GND;
      B"0100"        =>  VCC;
      B"1100"        =>  GND;
      B"1101"        =>  VCC;
      B"1111"        =>  GND;
      B"1110"        =>  VCC;
      B"1010"        =>  GND;
      B"1011"        =>  VCC;
      B"1001"        =>  GND;
      B"1000"        =>  VCC;
  END TABLE;
END;
```

3. 优先排队电路 VHDL 程序

```
SUBDESIGN t5_3
```

```
(
  a,b,c                 :  INPUT;
  a_out,b_out,c_out     : OUTPUT;
)
BEGIN
  IF a THEN
    a_out = VCC; b_out = GND; c_out = GND;
  ELSIF b THEN
    a_out = GND; b_out = VCC; c_out = GND;
  ELSIF c THEN
    a_out = GND; b_out = GND; c_out = VCC;
  ELSE
    a_out = GND;
    b_out = GND;
    c_out = GND;
  END IF;
END;
```

5.2 实验二　计数器及时序电路设计

一、实验目的

(1) 了解时序电路的经典设计方法(D 触发器、JK 触发器和一般逻辑门组成的时序逻辑电路)。

(2) 了解同步计数器,异步计数器的使用方法。

(3) 了解同步计数器通过清零阻塞法和预显数法得到循环任意进制计数器的方法。

(4) 理解时序电路和同步计数器加译码电路的联系,设计任意编码计数器。

(5) 了解同步芯片和异步芯片的区别。

二、实验硬件

(1) 主芯片 Altera EPM240T100C5N。

(2) 时钟。

(3) 4 位八段数码管。

三、实验内容

(1) 用 D 触发器设计异步 4 位二进制加法计数器。

(2) 用 JK 触发器设计异步十进制减法计数器。

(3) 用 74161 两个宏连接成 8 位二进制同步计数器。

(4) 用 74390 两个宏连接成 8 位十进制异步计数器。

(5) 用 74161 以清零和置数法组成六进制和十二进制计数器。

(6) 分别用 D 触发器和同步计数器加译码电路的方法构成七进制电路,实现如下编码:

[0→2→5→3→4→6→1]循环

四、实验原理

实验内容中的六个实验均要通过“扫描显示电路”进行显示,具体连线根据每个实验内容完成时的管脚分配来定义,与相应的输入输出接口功能模块相连。

1. 异步 4 位二进制加法器

利用 D 触发器设计的计数器电路原理图,如图 5.4 所示。

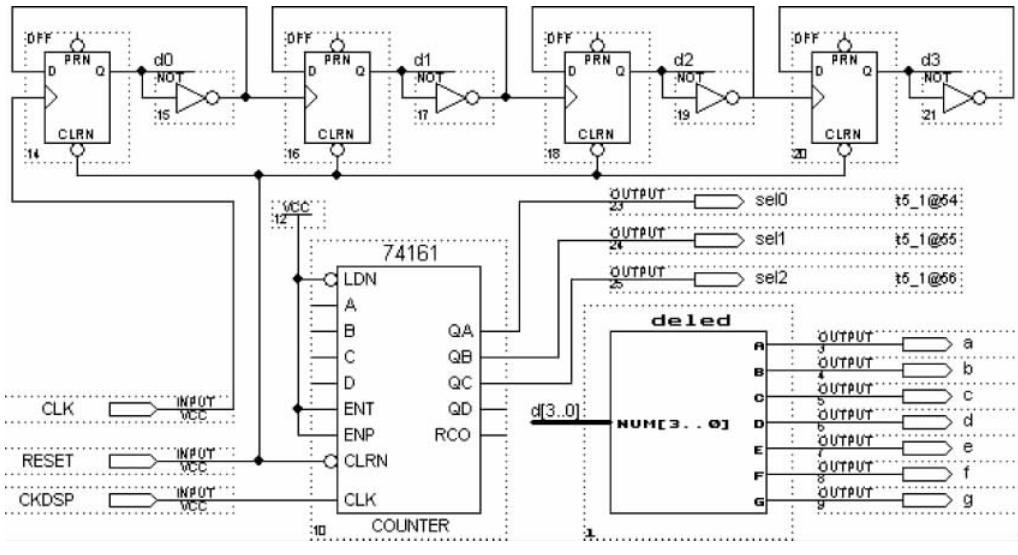

图 5.4　基于 D 触发器的计数器电路原理图

说明：计数时钟频率 CLK＜2Hz；扫描时钟频率＞40Hz；

4 位 D 触发器接成异步计数器；sel0～sel2 为扫描地址(控制 8 位数码管的扫描顺序和速度)；a～9 为显示译码输出，代表数码管的 8 个段位(a,b,c,d,e,f,g)。

8 位数码管同时顺序显示 0～F。

2. 异步十进制减法计数器

利用 JK 触发器设计的计数器电路原理图，如图 5.5 所示。

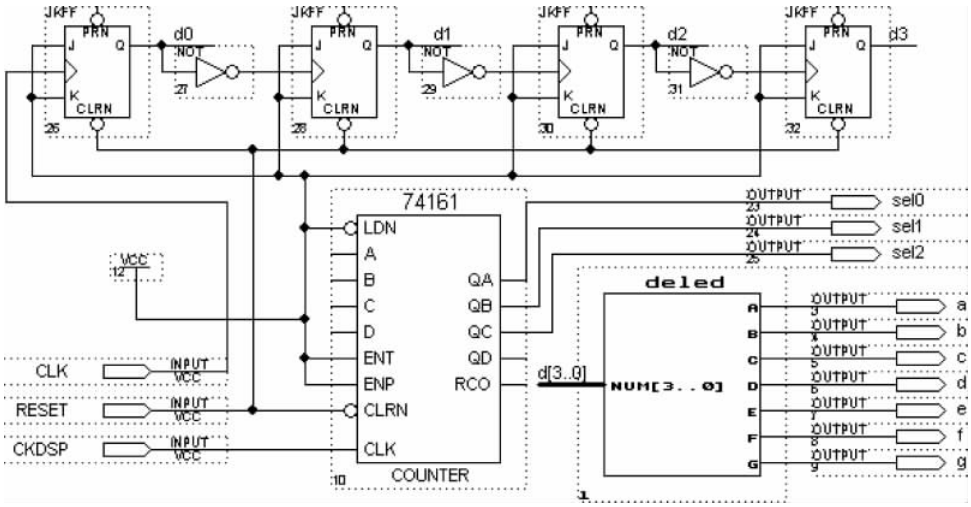

图 5.5　基于 JK 触发器的计数器电路原理图

3. 8 位二进制同步计数器

用 74161 两个宏连接成 8 位二进制同步计数器电路原理图，如图 5.6 所示。

说明：两个 74161 串接成典型的同步计数器；muxh14 完成扫描数据切换；两位数码管同时显示 00～FF。

4. 8 位十进制异步计数器

用 74390 两个宏连接成 8 位十进制异步计数器原理图，如图 5.7 所示。

说明同前；

两位数码管同时顺序显示十进制 00～99。

5. 六进制和十二进制计数器

用 74161 以清零和置数法组成六进制和十二进制计数器实验原理图，如图 5.8 所示。

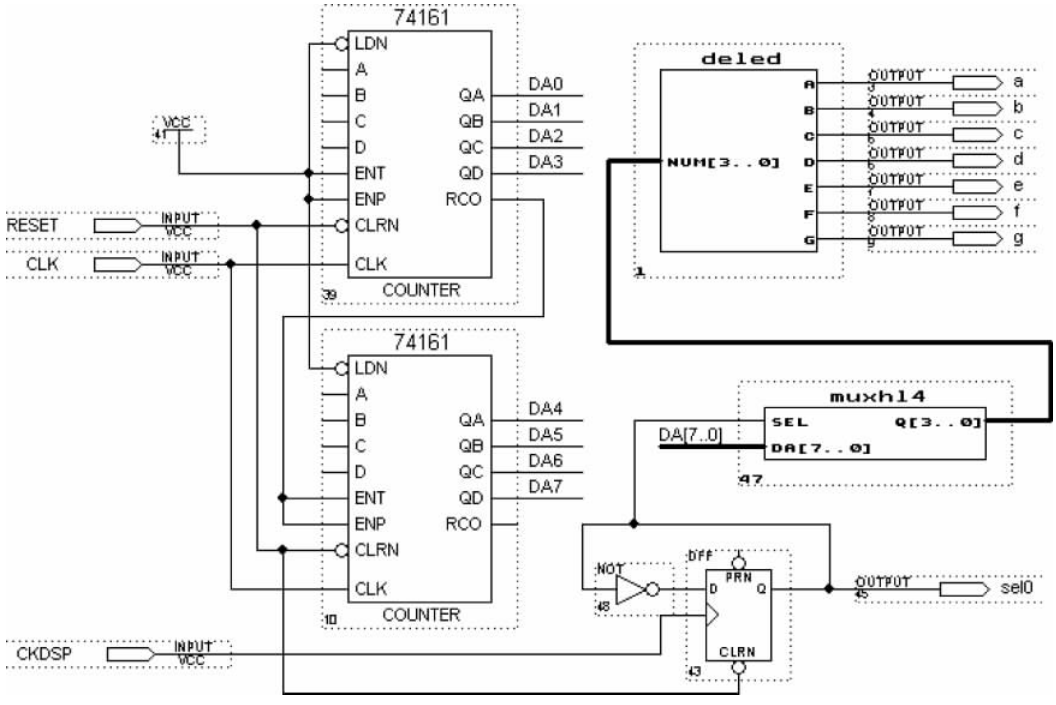

图 5.6　基于 74161 的同步计数器电路原理图

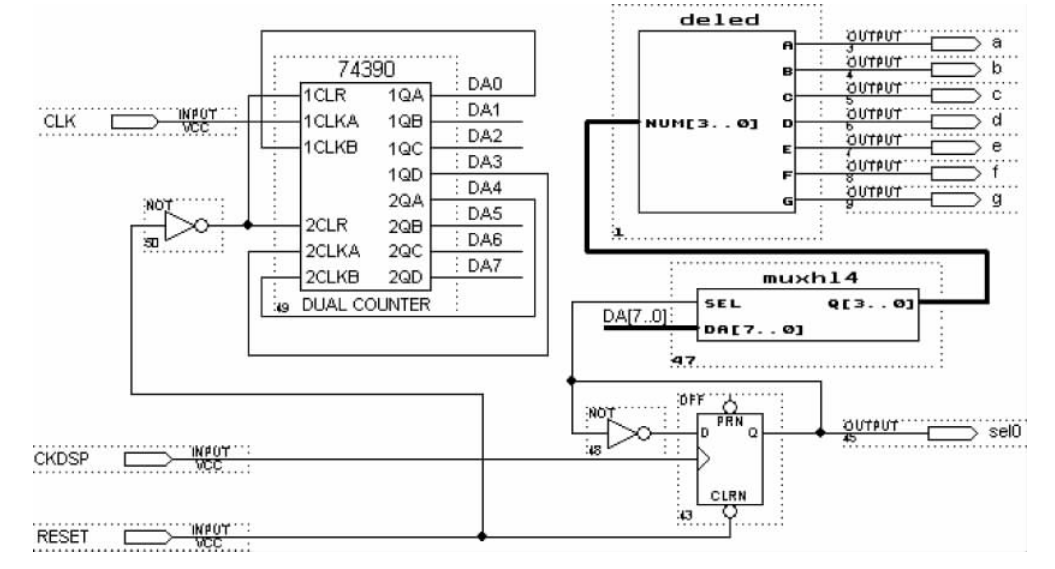

图 5.7　基于 74390 的异步计数器电路原理图

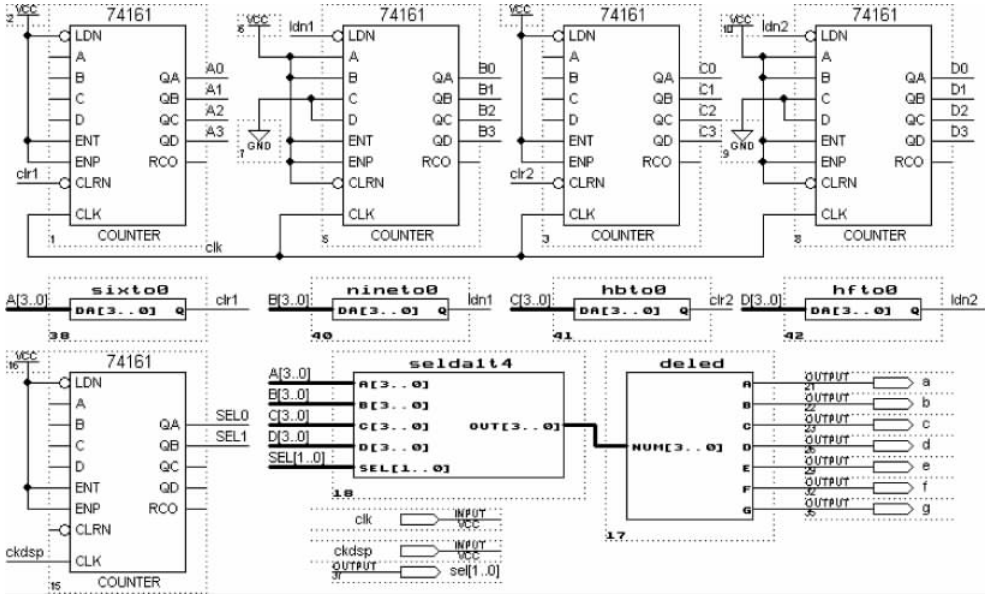

图 5.8　基于 74161 的六进制和十二进制计数器电路原理图

说明：

① 清零法分别完成 0～6、0～B 的顺序计数；

② 置位法分别完成 3～9，3～F 的顺序计数，用 4 个数码管显示 4 个计数状态。

6. 七进制电路

分别用 D 触发器和同步计数器加译码电路的方法构成七进制电路实验参考原理图，如图 5.9 所示。

说明：这是按 0→2→5→3→4→6→1 变化的七进制计时器；图中包括两个独立的实现方法，一种为异步清零，一种为同步清零，两种方法同时显示。

用 74161 计数器加译码的方法实现异步清零七进制计数器的设计；同时用状态机的方法实现同步清零七进制计数器的设计。

五、实验报告

(1) 描述实验的基本原理和步骤。

(2) 用数据和图片给出各个步骤中取得的实验结果，并对实验结果及实验过程中出现的问题进行分析总结。

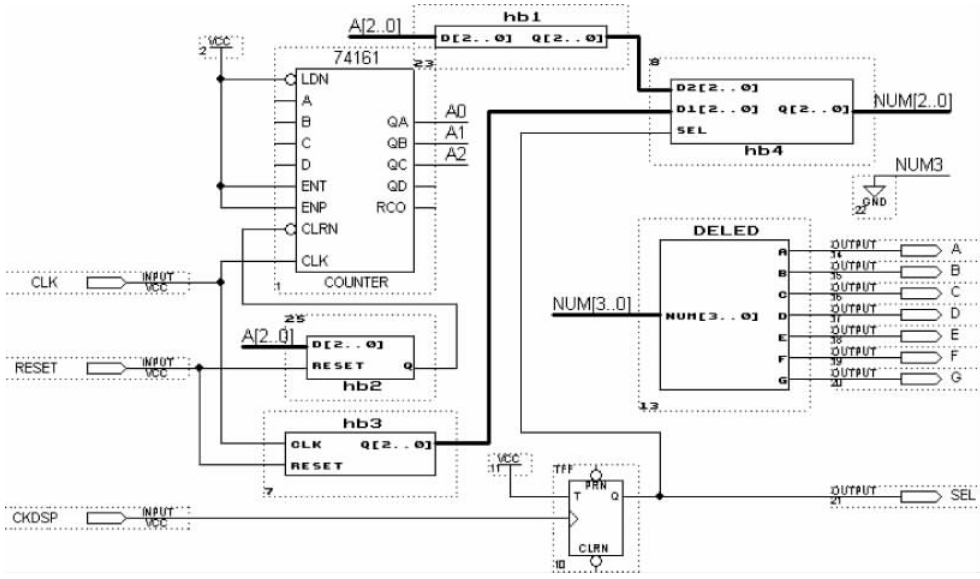

图 5.9　基于 D 触发器和同步计数器的七进制电路原理图

(3) 完整的源代码。

六、参考源程序

1. 异步 4 位二进制加法计数器 VHDL 源程序

```
SUBDESIGN muxhl4
(
SEL,DA[7..0]:INPUT;
  Q[3..0]:OUTPUT;
)
BEGIN
IF SEL == B"0" then
        q[3..0] = da[3..0];
   ELSIFSEL == B"1" then
        q[3..0] = da[7..4];
   END IF;
END;
```

2. 异步十进制减法计数器 VHDL 源程序

```
SUBDESIGN nineto0
(
  DA[3..0]:INPUT;
  Q:OUTPUT;
)
BEGIN
   if DA[3..0] == B"1010" then
       q = B"0";
   ELSE
       q = B"1";
   END IF;
END;
```

3. 8 位二进制同步计数器 VHDL 源程序

```
SUBDESIGN hb1
(
  d[2..0]:INPUT;
  q[2..0]:OUTPUT;
)
BEGIN
   CASE d[ ] IS
      WHEN 0  = >     q[ ] = 0;
      WHEN 1  = >     q[ ] = 2;
      WHEN 2  = >     q[ ] = 5;
      WHEN 3  = >     q[ ] = 3;
      WHEN 4  = >     q[ ] = 4;
      WHEN 5  = >     q[ ] = 6;
      WHEN 6  = >     q[ ] = 1;
   END CASE;
END;
```

4. 8 位十进制异步计数器 VHDL 源程序

```
SUBDESIGN hb2
(
```

```
  reset,d[2..0]:INPUT;
  q:OUTPUT;
)
BEGIN
   IF(reset) THEN
     IF(d[ ]>6) THEN
        q = GND;
     ELSE
        q = VCC;
     END IF;
   ELSE
q = GND;
   END IF;
END;
```

5. 六进制和十二进制计数器 VHDL 源程序

```
SUBDESIGN hb3
(
  clk,reset:INPUT;
  q[2..0]:OUTPUT;
)
VARIABLE
   ss: MACHINE OF BITS (q[2..0])
    WITH STATES (
                 s0 = 0,
                 s1 = 2,
                 s2 = 5,
                 s3 = 3,
                 s4 = 4,
                 s5 = 6,
                 s6 = 1    );
BEGIN
   ss.clk = clk;
   ss.reset = !reset;
   TABLE
      ss   =>  ss;
      s0   =>  s1;
      s1   =>  s2;
```

```
        s2   =>  s3;
        s3   =>  s4;
        s4   =>  s5;
        s5   =>  s6;
        s6   =>  s0;
    END TABLE;
END;
```

6. 七进制电路 VHDL 源程序

```
SUBDESIGN hb4
(
  sel,d1[2..0],d2[2..0]:INPUT;
  q[2..0]:OUTPUT;
)
BEGIN
    IF(sel) THEN
        q[ ] = d1[ ];
    ELSE
        q[ ] = d2[ ];
    END IF;
END;
```

5.3 实验三　数字钟设计

一、实验目的

(1) 掌握多位计数器相连的设计方法。
(2) 掌握十进制,六进制,二十四进制计数器的设计方法。
(3) 继续巩固多位共阴极扫描显示数码管的驱动及编码。
(4) 掌握扬声器的驱动。
(5) 学习 LED 灯的花样显示。
(6) 掌握 CPLD 技术的层次化设计方法。

二、实验硬件

(1) 主芯片 Altera EPM240T100C5N。

(2) 8 个 LED 灯。

(3) 扬声器。

(4) 8 位 8 段共阴极扫描数码显示管。

(5) 三个按键开关(清零,调小时,调分钟)。

三、实验内容

在同一 CPLD 芯片上集成了如下电路模块:

(1) 时钟计数:秒……60 进制 BCD 码计数;

分……60 进制 BCD 码计数;

时……24 进制 BCD 码计数。

同时整个计数器有清零、调时、调分功能。在接近整数时能提供报时信号。

(2) 具有驱动 8 位 8 段共阴极扫描数码管的片选驱动信号输出和 8 段字形译码输出。

(3) 扬声器在整点时有报时驱动信号产生。

(4) LED 灯在整点时有花样显示信号产生。

四、实验原理

数字钟设计原理图如图 5.10 所示(模块化设计)。

模块说明:

(1) 各种进制的计数及时钟控制模块(十进制、六进制、24 进制)。

(2) 扫描分时显示,译码模块。

(3) 彩灯,扬声器编码模块。

输入接口:

(1) 代表清零信号 RESET、调时信号 SETHOUR、调分信号 SETMIN 的管脚分别连接按键开关。

(2) 代表计数时钟信号 CLK 和扫描时钟信号 ckdsp 的管脚分别同 1Hz 时钟源和 32Hz(或更高)时钟源相连。

输出接口:

(1) 代表扫描显示的驱动信号管脚 SEL2、SEL1、SEL0 和 a、b、c、d、e、f、g 参照 4.4 实验四的连法。

(2) 代表扬声器驱动信号的管脚 SPEAK 同扬声器驱动接口 SPEAKER 相连。

(3) 代表花样灯显示信号管脚 LAMP0、LAMP1、LAMP2、LAMP3 同 4 个 LED 灯相连。

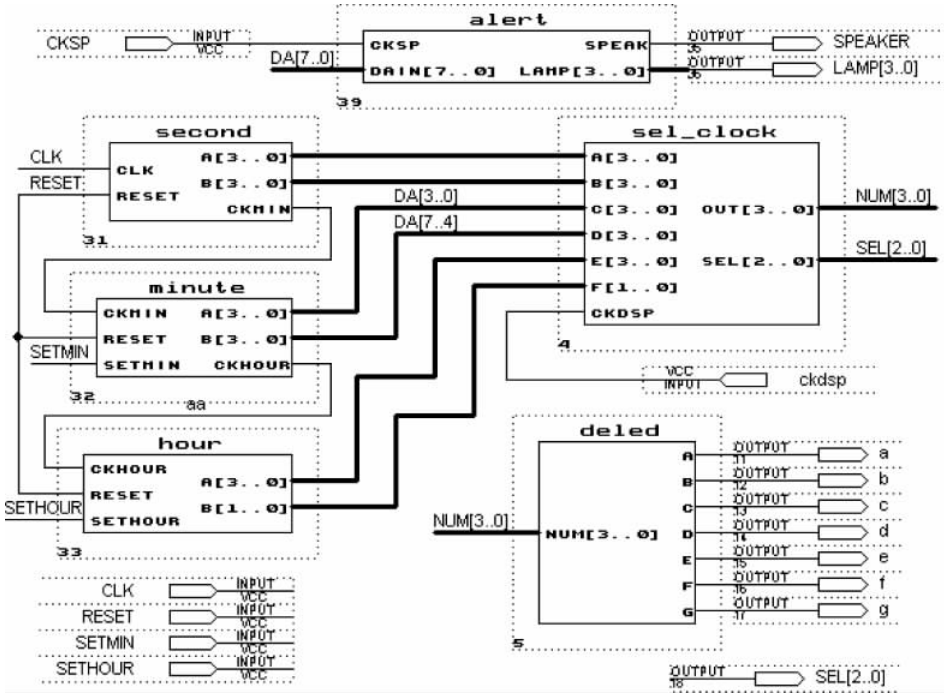

图 5.10　数字钟设计原理图

五、实验报告

(1) 描述实验的基本原理和步骤。

(2) 用数据和图片给出各个步骤中取得的实验结果,并对实验结果及实验过程中出现的问题进行分析总结。

(3) 完整的源代码。

六、参考源程序

1. 秒钟模块(60 进制)VHDL 源程序

```
SUBDESIGN    Second
(
CLK,RESET                : INPUT;
A[3..0],B[3..0],CKMIN    : OUTPUT;
```

```
)
VARIABLE
count1[3..0]: DFF;
count2[3..0]: DFF;
SS:DFF;
BEGIN
count1[].clk = clk;
count1[].clrn = RESET;
count2[].clk = clk;
count2[].clrn = RESET;
SS.CLK = CLK;
SS.CLRN = RESET;
SS = B"0";

IF    COUNT1[] == B"1001" AND
      COUNT2[] == B"0101" THEN
SS = B"1";
END IF;

IF count1[] == B"1001" THEN
count1[] = B"0000";
ELSE
count1[].d = count1[].q + 1;
END IF;

IF count1[] == B"1001" THEN
IF count2[] == B"0101" THEN
count2[] = B"0000";
ELSE
count2[].d = count2[].q + 1;
END if;
ELSE
count2[].d = count2[].q;
END IF;

a[3..0] = count1[];
b[3..0] = count2[];
CKMIN = SS;
END;
```

2. 分钟模块(60 进制)VHDL 源程序

```
SUBDESIGN Minute
(
CKMIN,RESET,SETMIN: INPUT;
A[3..0],B[3..0],CKHOUR: OUTPUT;
)
VARIABLE
count1[3..0]: DFF;
count2[3..0]: DFF;
count3: DFF;
BEGIN
count1[].clk = CKMIN OR !SETMIN;
count1[].clrn = RESET;
count2[].clk = CKMIN OR !SETMIN;
count2[].clrn = RESET;
count3.clk = CKMIN OR !SETMIN;
count3.clrn = RESET;
count3 = B"0";

IF    COUNT1[] == B"1001" AND
      COUNT2[] == B"0101" THEN
COUNT3 = B"1";
END IF;

IF    count1[] == B"1001" then
count1[] = B"0000";
ELSE
count1[].d = count1[].q + 1;
END IF;

IF    count1[] == B"1001" then
IF    count2[] == B"0101" THEN
count2[] = B"0000";
ELSE
count2[].d = count2[].q + 1;
END if;
ELSE
count2[].d = count2[].q;
```

```
END IF;

a[3..0] = count1[];
b[3..0] = count2[];
CKHOUR = count3;
END;
```

3. 小时模块(24 进制)VHDL 源程序

```
SUBDESIGN Hour
(
CKHOUR,RESET,SETHOUR : INPUT;
A[3..0],B[1..0]: OUTPUT;
)
VARIABLE
count1[3..0]: DFF;
count2[1..0]: DFF;

BEGIN
count1[].clk = CKHOUR OR !SETHOUR;
count1[].clrn = RESET;
count2[].clk = CKHOUR OR !SETHOUR;
count2[].clrn = RESET;

IF count2[]< B"10" THEN
IF count1[] == B"1001" THEN
count1[] = B"0000";
count2[].d = count2[].q + 1;
ELSE
count1[].d = count1[].q + 1;
count2[].d = count2[].q;
END IF;
ELSIF    count2[] == B"10" THEN
IF    count1[] == B"0011" THEN
count1[] = B"0000";
count2[] = B"00";
ELSE
count1[].d = count1[].q + 1;
count2[].d = count2[].q;
END IF;
```

```
END IF;

a[3..0] = count1[];
b[1..0] = count2[];
END;
```

4. 显示扫描模块 VHDL 源程序

```
SUBDESIGN SEL_clock
(
A[3..0],B[3..0],C[3..0] : INPUT;
    D[3..0],E[3..0],F[1..0] : INPUT;
CKDSP: INPUT;
OUT[3..0],SEL[2..0] :OUTPUT;
)

VARIABLE
COUNT[2..0] : DFF;
BEGIN
count[].clk = CKDSP;

IF COUNT[] == B"101" THEN
COUNT[] = B"000";
ELSE
count[].d = count[].q + 1;
END IF;

IF COUNT[] == B"000" then
out[3..0]= A[3..0];
ELSIF COUNT[] == B"001" then
out[3..0]信号与系统学习及解题指导 = B[3..0];
ELSIF COUNT[] == B"010" then
out[3..0]信号与系统学习及解题指导 = C[3..0];
ELSIF COUNT[] == B"011" then
out[3..0]信号与系统学习及解题指导 = D[3..0];
ELSIF COUNT[] == B"100" then
out[3..0]信号与系统学习及解题指导 = E[3..0];
ELSIF COUNT[] == B"101" then
out[1..0]信号与系统学习及解题指导 = F[1..0];
out[3..2]信号与系统学习及解题指导 = B"00";
```

```
END IF;

SEL[2..0] = COUNT[];
END;
```

5. 报时模块 VHDL 源程序

```
SUBDESIGN  alert
(
CKSP,DAIN[7..0]: INPUT;
SPEAK,LAMP[3..0]: OUTPUT;
)

VARIABLE
s: DFF;
ss:MACHINE OF BITS (lamp[3..0])
WITH STATES(
s0 = B"0000",
s1 = B"0001",
s2 = B"0010",
s3 = B"0100",
s4 = B"1000");
BEGIN
IF dain[7..0] == B"00000000" THEN
SS.clk = cksp;
s.clk = cksp;
s.d = !s.q;
speak = s.q;
CASE ss IS
WHEN s0  => ss = s1;
WHEN s1  => ss = s2;
WHEN s2  => ss = s3;
WHEN s3  => ss = s4;
WHEN s4  => ss = s1;
WHEN OTHERS  => ss = s0;
END CASE;
ELSE
SS = S0;
SPEAK = GND;
END IF;
END;
```

5.4 实验四 函数信号发生器设计

一、实验目的

(1) 了解 DAC 的工作原理。
(2) 熟悉 AD558 的使用方法。

二、实验硬件

(1) 主芯片 Altera EPM240T100C5N。
(2) 模拟功能块 AD558。
(3) 4 位八段扫描显示数码管。
(4) 示波器。
(5) 拨码开关。

三、实验内容

CPLD 芯片将一路时钟分频产生可选模式(四种)循环 8 位二进制计数值 DAOUT[7..0],将之接入 AD558 的 D[7..0],用示波器来观察 DAC 的波形输出。

连线说明:

(1) EPM7128SLC84-15 模块的 CLK 接时钟源;
(2) Model1、Model2 接拨码开关;
(3) RESET 接按键开关;
(4) DAOUT[7..0]接 DAC 的 D[7..0]输入;
(5) DAC 模块的/CE、/CS 接逻辑 0;
(6) DAC 模块的 D/AOUT 接示波器观察。

四、实验原理

AD558 可将输入的数字量(8 位)转化成 0～2.56V 的模拟电压量;用 CPLD 器件产生四种循环变化的数据量:

(1) 0～255(8 位)循环加法计数;

(2) 255～0(8 位)循环减法计数；

(3) 0～255～0(8 位)循环加减法计数；

(4) 0,20H,40H,60H,80H,A0H,C0H,E0H 编码计数器。

将计数器的八位输出接到 DAC 的八位输入,可以产生四种波形(频率相同)：

(1) 递增斜波；

(2) 递减斜波；

(3) 三角波；

(4) 递增阶梯波。

五、实验报告

(1) 描述实验的基本原理和步骤。

(2) 用数据和图片给出各个步骤中取得的实验结果,并对实验结果及实验过程中出现的问题进行分析总结。

(3) 完整的源代码。

六、参考源程序

函数发生器 VHDL 程序：

```
SUBDESIGN t11
(
    clk,reset,model[1..0]: INPUT;
    daout[7..0]: OUTPUT;
)

VARIABLE
    count[7..0],subadd:DFF;
--   subadd:NODE;
BEGIN
    count[].clk = clk;
    count[].clrn = reset;
    subadd.clk = clk;
    subadd.clrn = reset;
    daout[] = count[].q;
-- sa[] = subadd[].q;
    CASE model[] IS
```

```
    WHEN 0 =>
        count[].d = count[].q + 1;
    WHEN 1 =>
        count[].d = count[].q - 1;
    WHEN 2 =>
        IF(subadd.q == GND) THEN
           count[].d = count[].q + 1;
             IF(count[].q == 254)
        THEN
                subadd.d = VCC;
             ELSE
                subadd.d = GND;
                END IF;
        ELSE
           count[].d = count[].q - 1;
           IF(count[].q <= 1) THEN
                subadd.d = GND;
           ELSE
                subadd.d = VCC;
           END IF;
         END IF;
    WHEN 3 =>

    count[].d = count[].q + H"20";
    END CASE;
END;
```

5.5 实验五 模拟信号检测电路设计

一、实验目的

(1) 学习并掌握利用 CPLD 检测模拟信号的设计方法。
(2) 强化 ADC、DAC 在 CPLD 设计中的应用。

二、硬件要求

(1) 主芯片 EPM240T100C5N。

(2) 模拟功能块 AD558。

(3) 模拟功能块双运放 LM358。

(4) 可变电阻。

三、实验内容

用 DAC+比较器的方法对外界模拟信号进行检测,同时这种联合装置加上 CPLD 可以代替低频 ADC 的功能。

连线说明:

(1) EPM7128SLC84-15 模块的 CLK、CKDSP 接时钟源;

(2) RESET 接按键开关;

(3) Jmp 接集成运放比较输出端;

(4) DA[7..0]接 DAC 的数据输入;

(5) DAC 模块的/CE、/CS 接逻辑 0;

(6) SPEAKER:接 Jmp 信号。

四、实验原理

CPLD 芯片产生 8 位二进制循环加法计数值 D7～D0,它们与 AD558 数据端相连,使 DAC 产生 0～2.56V 的斜波电压,经一级运放放大一倍后成为 0～5.12V(理论值,实际的动态范围是 0～3.78V 左右,在高饱和位 3.78V)。

可变电阻器模拟 0～5V 的模拟输入量,这个值被接入另一个运放的反向输入端,而 DAC 放大的 0～5.12V 的电压量被接入运放的同相输入端,比较器运放的输出端接 CPLD 的输入管脚 jmp。

工作时,可变电阻器输入一个 Vtest 值,而 DAC 从 0～5.12V 循环扫描,当扫描值 Vscan 低于 Vtest 时,比较器输出的 jmp 为 0,当扫描值 Vscan 高于 Vtest 时,比较器输出为 1,而使 jmp 产生正向跳变的 DAC 数字量输入值 DAOUT 就代表模拟量 Vtest 的数值。

显示用习惯的十进制,并在芯片内部完成两倍显示转换。

以上各模块的大部分连线都已接好,只保留少量端口,以适应实验的灵活性。

参考原理图如图 5.11 所示。

说明:

hb1:00H～FFH 循环加法计数器;

hb2:000～512 十进制加法计数器(用于显示);

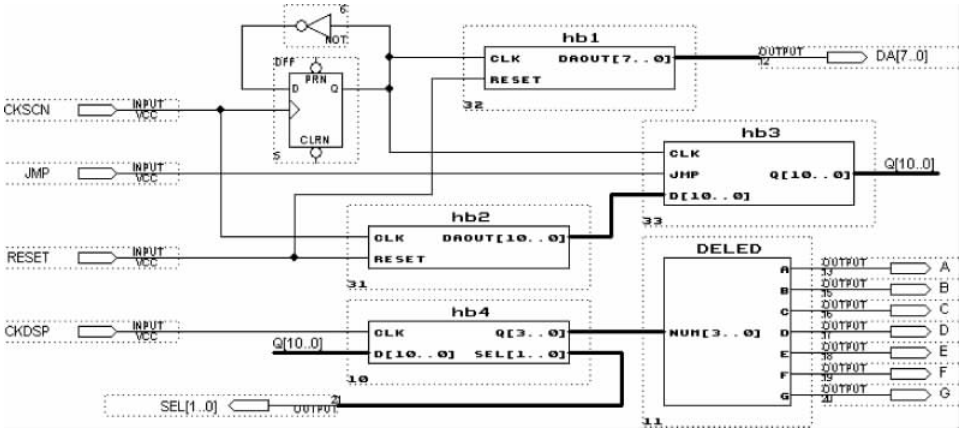

图 5.11　模拟信号检测原理图

hb3：显示锁存；

hb4：分时总线切换；

DELED：BCD 数显示译码。

五、实验报告

（1）描述实验的基本原理和步骤。

（2）用数据和图片给出各个步骤中取得的实验结果，并对实验结果及实验过程中出现的问题进行分析总结。

（3）完整的源代码。

六、参考源程序

1. hb1.tdf

```
SUBDESIGN hb1
(
   clk,reset:INPUT;
   daout[7..0]:OUTPUT;
)
VARIABLE
   count[7..0]:DFF;
BEGIN
   count[].clk = clk;
```

```
    count[ ].clrn = reset;
    daout[ ] = count[ ].q;
    count[ ].d = count[ ].q + 1;
END;
```

2. hb2.tdf

```
SUBDESIGN hb2
(
    clk,reset   :INPUT;
    daout[10..0]     :OUTPUT;
)
VARIABLE
    count[10..0]   :DFF;
BEGIN
    count[ ].clk = clk;
    count[ ].clrn = reset;
daout[ ] = count[ ].q;

    IF(count[3..0].q == 9) THEN
        IF(count[7..4].q == 9) THEN
            count[ ].d = count[ ].q + H"67";
        ELSE
            count[ ].d = count[ ].q + 7;
        END IF;
    ELSE
        IF(count[ ].q!= H"511") THEN
            count[ ].d = count[ ].q + 1;
        END IF;
    END IF;
END;
```

3. hb3.tdf

```
SUBDESIGN hb3
(
    clk,jmp,d[10..0]:INPUT;
    q[10..0]:OUTPUT;
)

VARIABLE
    count[10..0],s:DFF;
BEGIN
    s.clk = clk;
```

```
s.d = jmp;
   count[].clk = s.q;
   count[].d = d[];
   q[] = count[].q;
END;
```

4. hb4.tdf

```
SUBDESIGN hb4
(
   clk,d[10..0]:INPUT;
   q[3..0],sel[1..0]:OUTPUT;
)
VARIABLE
   count[1..0]:DFF;
BEGIN
count[].clk = clk;
   sel[] = count[].q;

   IF(count[].q == 2) THEN
      count[].d = 0;
   ELSE
      count[].d = count[].q + 1;
   END IF;

   CASE count[].q IS
      WHEN 0  => q[] = d[3..0];
      WHEN 1  => q[] = d[7..4];
      WHEN 2  => q[2..0] = d[10..8];
   END CASE;
END;
```

5.6 本章小结

本章设置了五个典型的综合性可编程器件(CPLD)实验项目,给出了实验原理图和实验步骤,提供了 VHDL 源程序作为参考。相对于第 4 章的基础实验项目,本章中的实验项目具有一定难度,能够锻炼读者复杂 VHDL 程序编制和 CPLD 综合开发能力,帮助读者进一步提高数字系统设计技术。

参 考 文 献

[1] 薛冰,沈锋,凌欢等. 零点起步：Altera CPLD/FPGA 轻松入门与开发实例. 北京：机械工业出版社,2012

[2] 黄平,王伟,周广涛. 基于 Quartus Ⅱ的 FPGA/CPLD 数字系统设计与应用. 北京：电子工业出版社,2014

[3] 方易圆. 可编程逻辑器件与 EDA 技术. 北京：清华大学出版社,2014

[4] 周润景,姜攀. 基于 Quartus Ⅱ的数字系统 Verilog HDL 设计实例详解. 第 2 版. 北京：电子工业出版社,2014

[5] 吴厚航. FPGA/CPLD 边练边学：快速入门 Verilog/VHDL. 北京：北京航空航天大学出版社,2013

[6] 杨健. EDA 技术与 VHDL 基础. 北京：清华大学出版社,2013

[7] 刘昌华. EDA 技术与应用：基于 Quartus Ⅱ和 VHDL. 北京：北京航空航天大学出版社,2012.

[8] 张艳花,牛晋川. 数字电子技术基础学习指导及习题详解. 北京：电子工业出版社,2011

[9] 郑燕,赫建国. 基于 VHDL 与 Quartus Ⅱ软件的可编程逻辑器件应用与开发. 第 2 版. 北京：国防工业出版社,2011

[10] 李民权. 数字电路与逻辑设计学习指导及习题详解. 北京：国防工业出版社,2014

[11] 吴文全. 电子技术基础实验与学习指导. 北京：电子工业出版社,2013

[12] 李景华,杜玉远. 可编程逻辑器件及 EDA 技术：数字系统设计与 SOPC 技术. 沈阳：东北大学出版社,2014

[13] 江小安,朱贵宪. 数字逻辑简明教程. 西安：西安电子科技大学出版社,2015

[14] 徐志军,王金明,尹廷辉等. EDA 技术与 VHDL 设计. 北京：电子工业出版社,2015