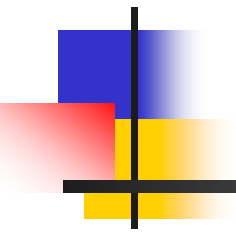

第7章

系统设计优化

7.1 资源优化

7.1.1 资源共享

【例 7-1】

```
module multmux (A0, A1, B, S, R);
   input[3:0] A0, A1, B;    input S;
   output[7:0] R;   reg[7:0] R;
   always @(A0 or A1 or B or S)   begin
     if (S==1'b0)  R<=A0 * B ;  else  R<=A1 * B ;   end
endmodule
```

【例 7-2】

```
module multmux (A0, A1, B, S, R);
     input[3:0] A0, A1, B;    input S;
     output[7:0] R;   wire [7:0] R; reg [3:0] TEMP;
     always @(A0 or A1 or B or S)   begin
       if (S==1'b0)  TEMP <= A0 ;  else  TEMP <= A1  ;   end
       assign R=TEMP * B;
 endmodule
```

7.1 资源优化

7.1.1 资源共享

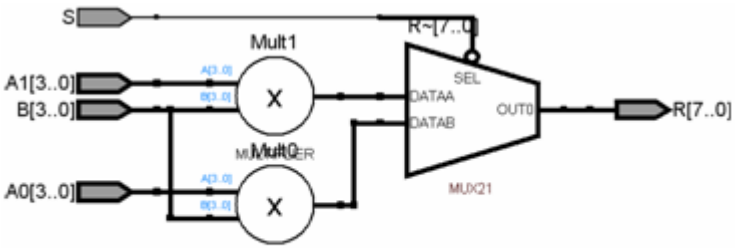

图 7-1 先乘后选择的设计方法 RTL 结构

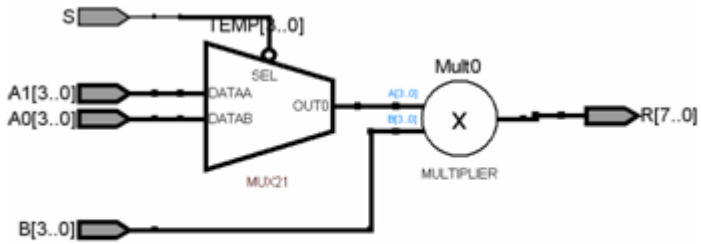

图 7-2 先选择后乘设计方法 RTL 结构

7.1 资源优化

7.1.1 资源共享

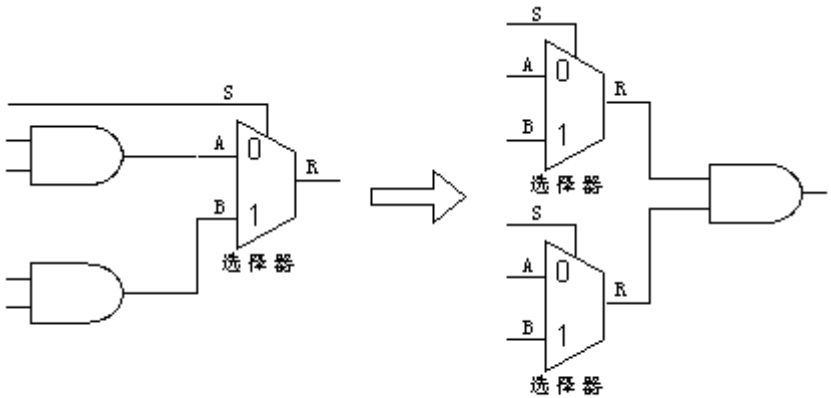

图 7-3　资源共享反例

7.1 资源优化

7.1.2 逻辑优化

【例 7-3】

```
module mult1 (clk, ma, mc);
   input clk;  input[11:0] ma;  output[23:0] mc;
   reg[23:0] mc;   reg[11:0] ta, tb;
   always @(posedge clk)   begin
      ta <= ma; mc <= ta * tb;  tb <= 12'b100110111001;  end
endmodule
```

【例 7-4】

```
module mult2  (clk, ma, mc);
   input clk;  input[11:0] ma;   output[23:0] mc;
   reg[23:0] mc;   reg[11:0] ta;
   parameter tb = 12'b100110111001;
   always @(posedge clk)
      begin  ta<=ma ; mc<=ta * tb ;    end
endmodule
```

7.1 资源优化

7.1.3 串行化

$$yout = a_0 \times b_0 + a_1 \times b_1 + a_2 \times b_2 + a_3 \times b_3$$

【例 7-5】

```
module pmultadd  (clk, a0, a1, a2, a3, b0, b1, b2, b3, yout);
   input clk;   input[7:0] a0, a1, a2, a3, b0, b1, b2, b3;
   output[15:0] yout;    reg[15:0] yout;
   always @(posedge clk)   begin
      yout <= ((a0 * b0)+(a1 * b1))+((a2 * b2)+(a3 * b3)) ;   end
endmodule
```

7.1 资源优化

7.1.3 串行化

【例 7-6】

```
module smultadd  (clk, start, a0, a1, a2, a3, b0, b1, b2, b3, yout);
    input clk, start;  input[7:0] a0, a1, a2, a3, b0, b1, b2, b3;
    output[15:0] yout;    reg[15:0] yout, ytmp;    reg[2:0] cnt;
    wire[7:0] tmpa, tmpb; wire[15:0] tmp;
 assign tmpa=(cnt==0)? a0:(cnt==1)? a1:(cnt==2)? a2:(cnt==3)? a3:a0;
 assign tmpb=(cnt==0)? b0:(cnt==1)? b1:(cnt==2)? b2:(cnt==3)? b3:b0;
  assign tmp = tmpa * tmpb ;
    always @(posedge clk)  begin
      if (start==1'b1)  begin  cnt<=3'b000 ; ytmp<={16{1'b0}} ;  end
       else if (cnt<4)     begin  cnt<=cnt+1  ; ytmp<=ytmp+tmp ;  end
       else if (cnt==4)    begin  yout<=ytmp ;   end    end
endmodule
```

7.2 速度优化

7.2.1 流水线设计

图 7-4 未使用流水线

$$F_{max} \approx F_{max1} \approx F_{max2} \approx 1/T_1$$

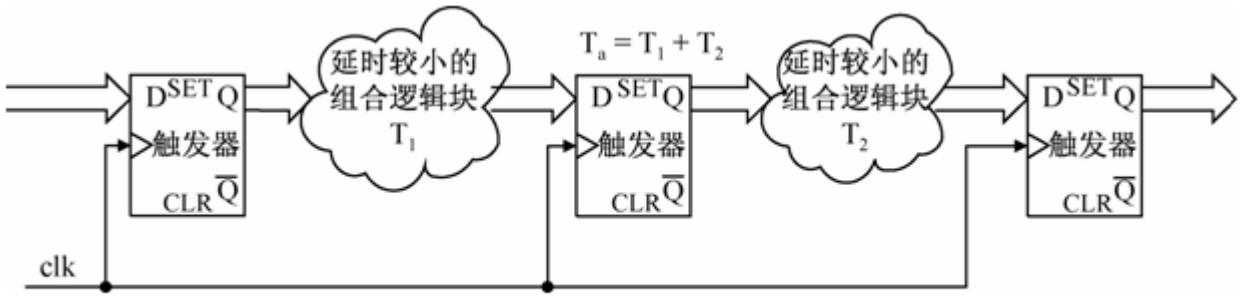

图 7-5 使用流水线结构

7.2 速度优化

7.2.1 流水线设计

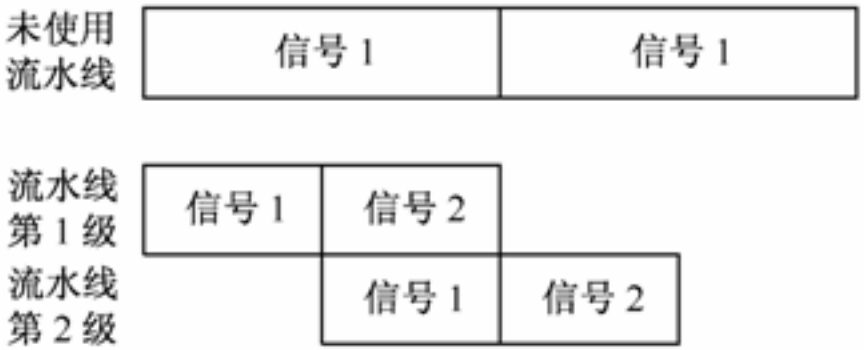

图 7-6 流水线工作图示

7.2 速度优化

7.2.1 流水线设计

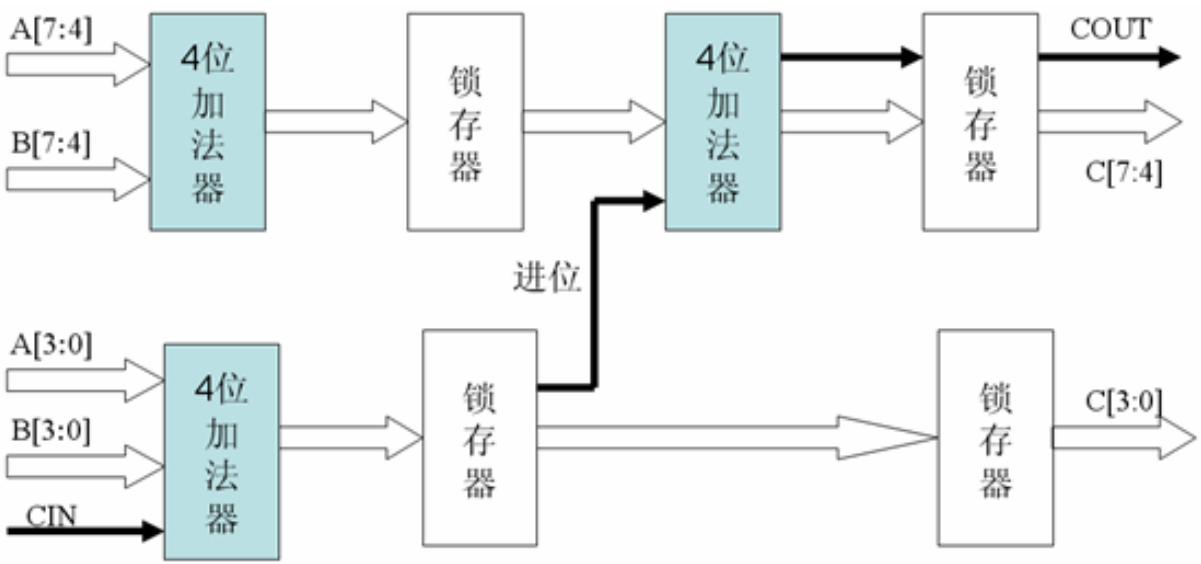

图 7-7　8 位加法器流水线工作图示

7.2 速度优化

7.2.1 流水线设计

【例 7-7】普通加法器，EP3C5 综合结果：LCs=10,REG=0,T=7.748ns.

```
module ADDER8(CLK,SUM,A,B,COUT,CIN);
   input [7:0] A,B;  input CLK,CIN;  output COUT;  output [7:0] SUM;
   reg COUT;   reg [7:0] SUM;
    always @(posedge CLK)
    {COUT,SUM[7:0]} <= A + B + CIN;
  endmodule
```

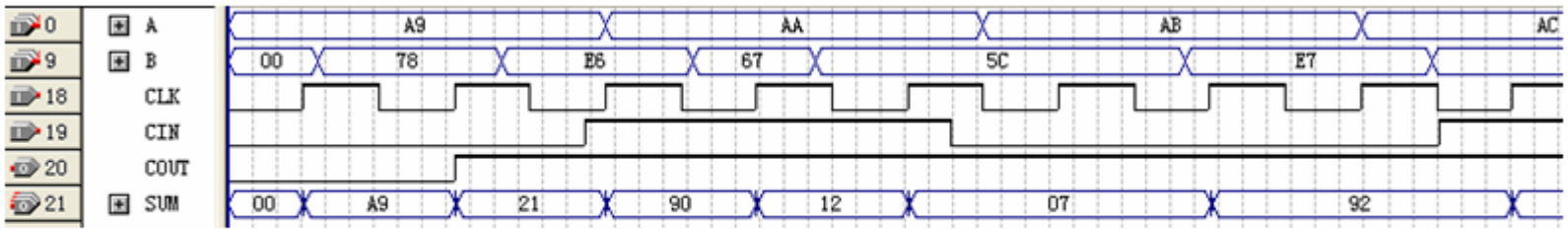

图 7-8　例 7-7 的时序仿真波形

7.2 速度优化

7.2.1 流水线设计

【例 7-8】流水线加法器，EP3C5 综合结果：T=3.63ns，LCs=24，REG=22。

```
module ADDER8(CLK,SUM,A,B,COUT,CIN);
   input [7:0] A,B;  input CLK,CIN;  output COUT;   output[7:0] SUM;
    reg TC,COUT;   reg[3:0] TS,TA,TB;    reg[7:0] SUM;
     always @(posedge CLK)  begin
       {TC,TS} <= A[3:0]+B[3:0]+CIN ;   SUM[3:0]<=TS;    end
     always @(posedge CLK)  begin
       TA <= A[7:4];   TB <= B[7:4];
      {COUT,SUM[7:4]} <= TA+TB+TC;     end
endmodule
```

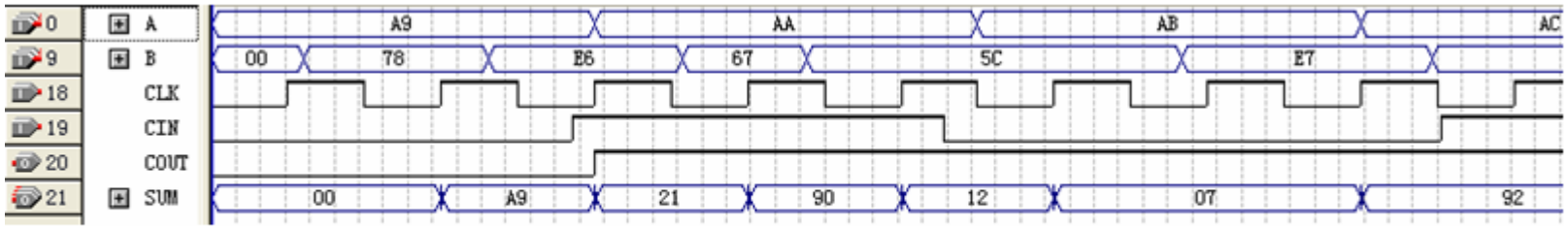

图 7-9　例 7-8 的时序仿真波形

7.2 速度优化

7.2.2 寄存器配平

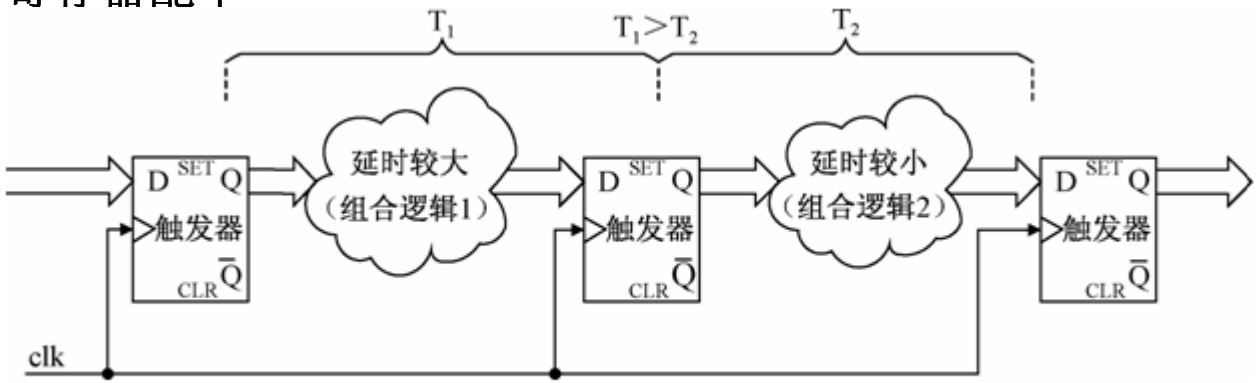

图 7-10 不合理的电路结构

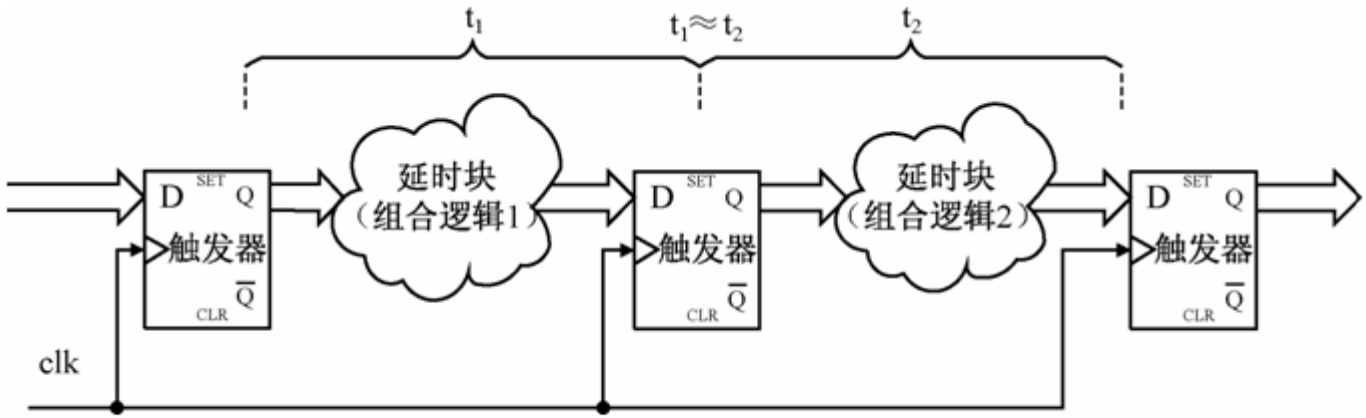

图 7-11 寄存器配平后的结构

7.2 速度优化

7.2.3 关键路径法

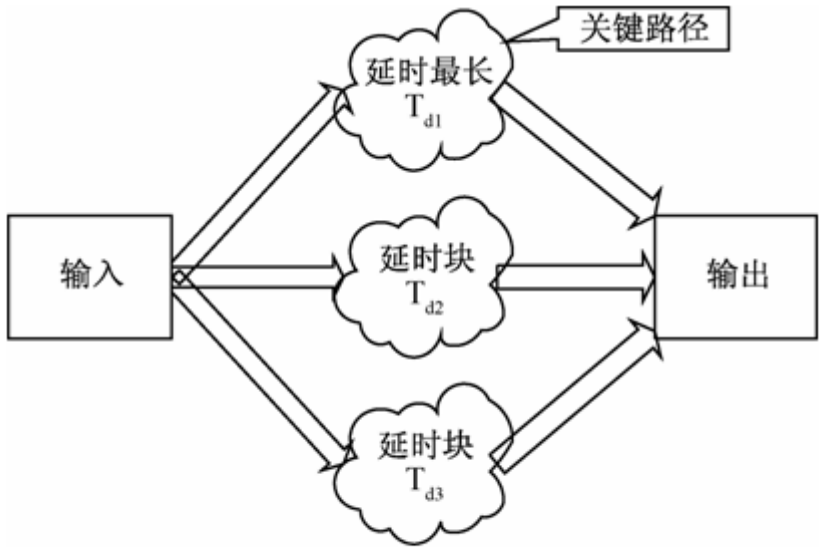

图 7-12 关键路径示意

7.2 速度优化

7.2.4 乒乓操作法

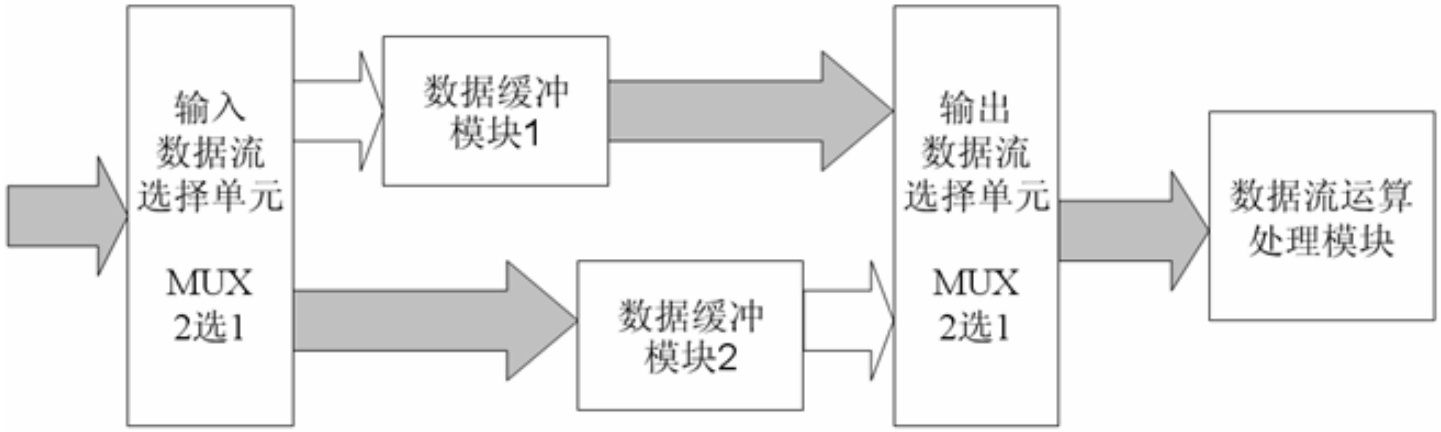

图 7-13　乒乓操作数据缓存结构示意图

7.2.5 加法树法

7.3 优化设置与分析

7.3.1 增量布局布线控制

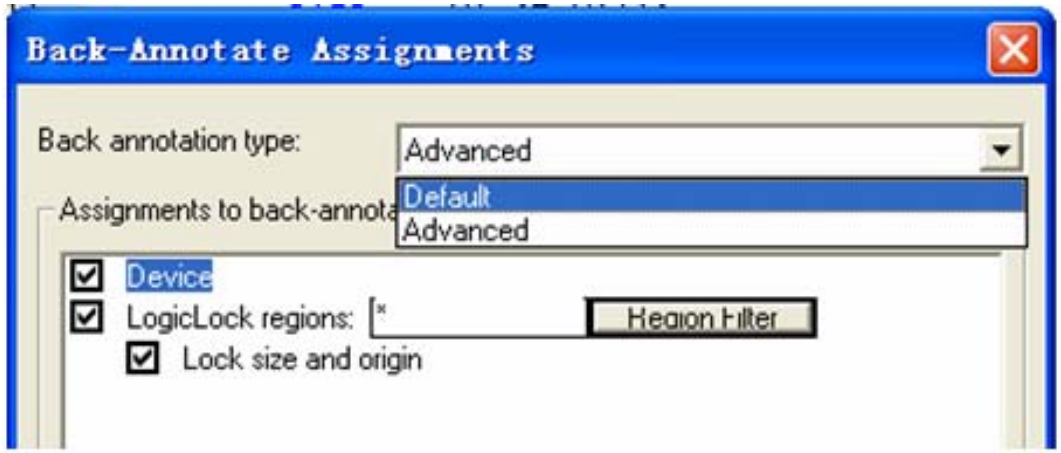

图7-14 反标设置

7.3 优化设置与分析

7.3.2 检查设计可靠性

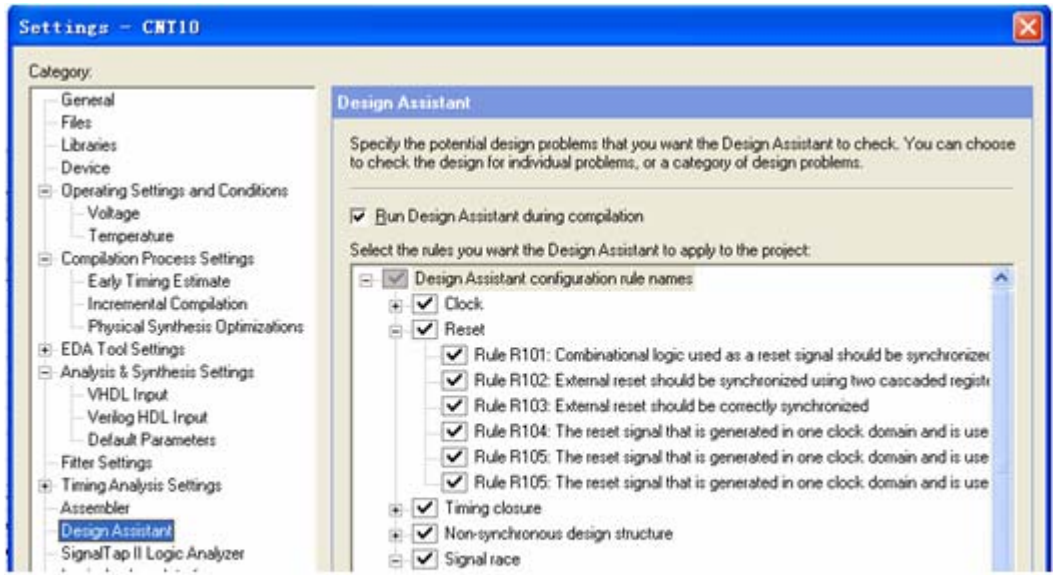

图 7-15 Design Assistant 设置

7.3 优化设置与分析

7.3.3 时序设置与分析

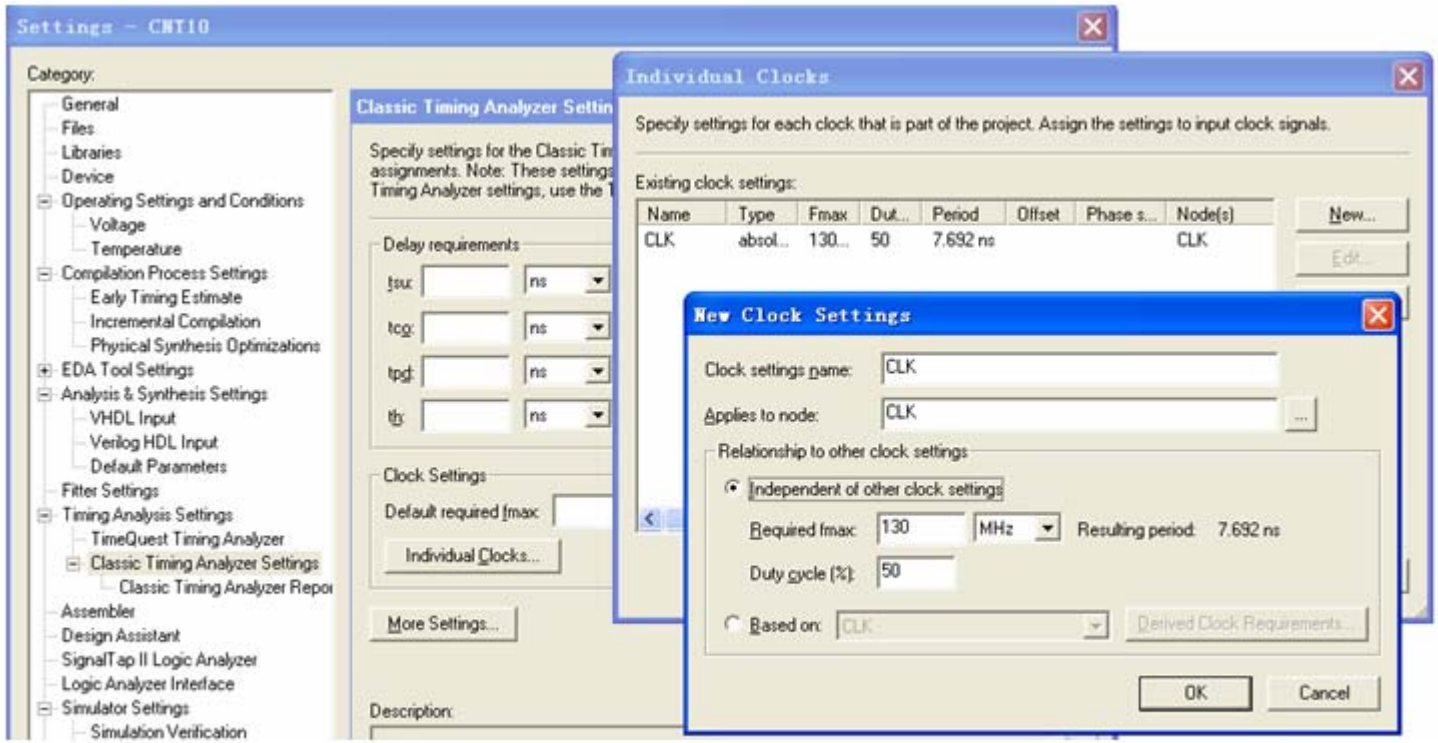

图 7-16 全编译前时序条件设置（设置时钟信号 CLK 不低于 130MHz）

7.3 优化设置与分析

7.3.4 查看时序分析结果

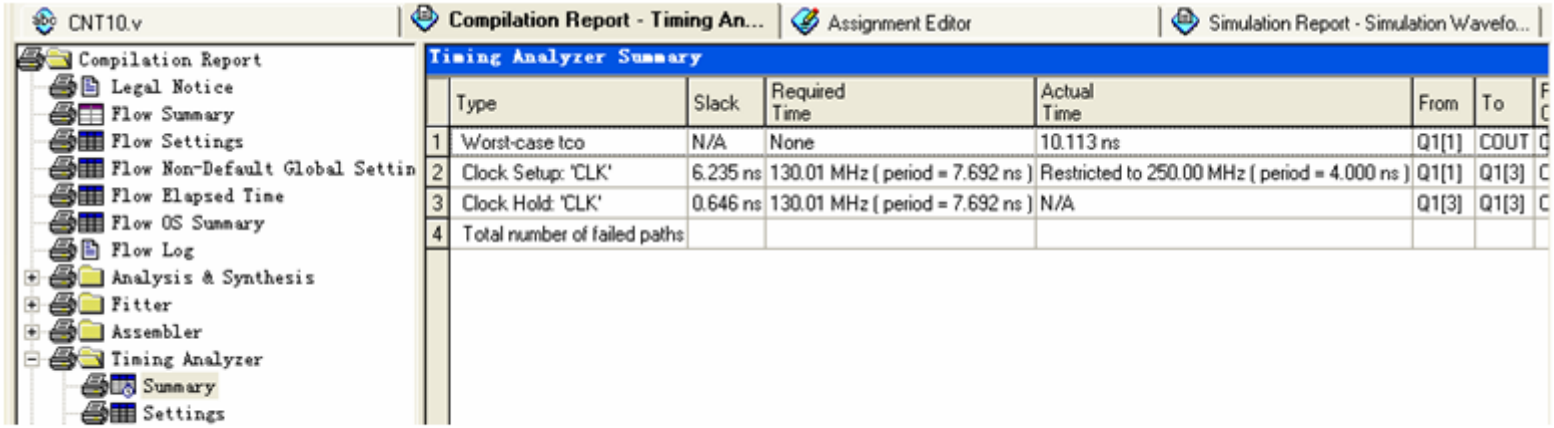

图 7-17 时序分析报告窗口

7.3 优化设置与分析

7.3.5 适配优化设置

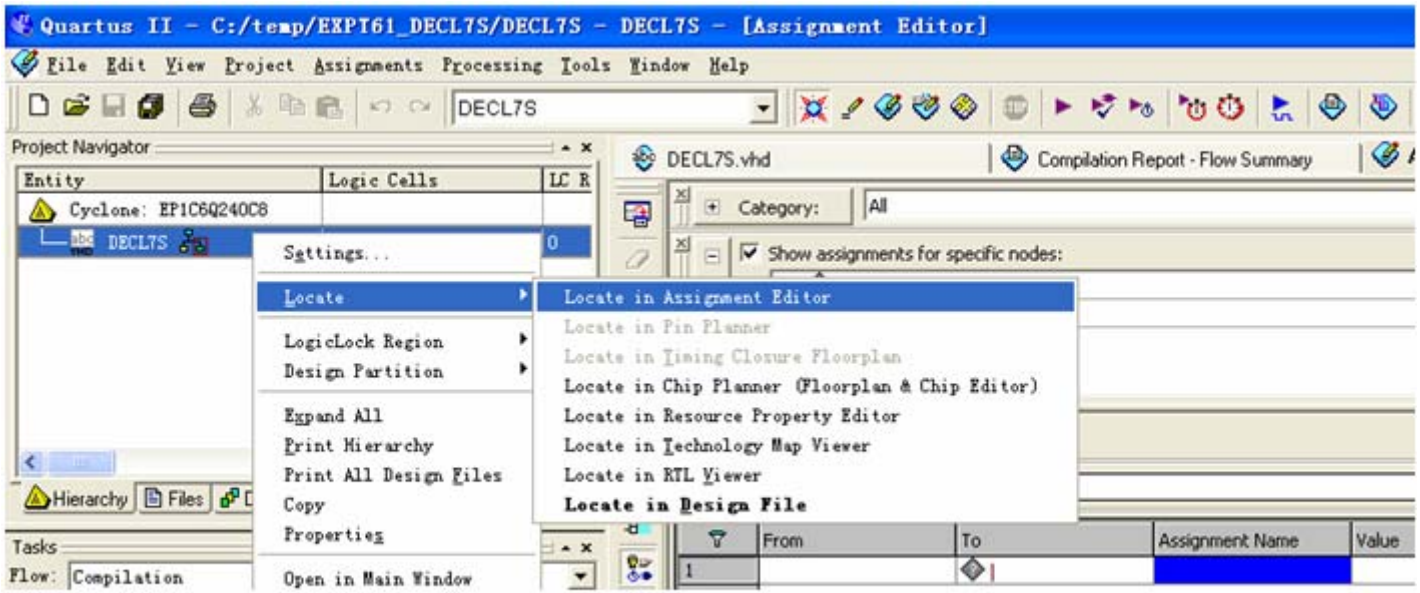

图 7-18 打开 Assignment Editor

7.3 优化设置与分析

7.3.5 适配优化设置

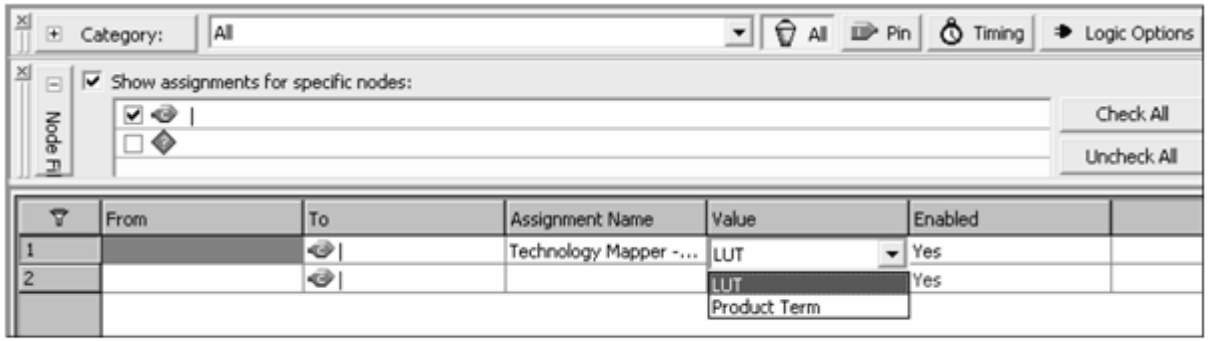

图 7-19 选用乘积项逻辑优化

7.3.6 LogicLock优化技术

习　题

```
module addmux (A, B, C, D, sel, Result);
  input[7:0] A, B, C, D;  input sel;
  output[7:0] Result;   reg[7:0] Result;
  always @(A or B or C or D or sel)  begin
    if (sel==1'b0)  Result <= A + B ;
    else  Result <= C + D ;  end
endmodule
```

$$y(n)=x(n)h(0)+x(n-1)+x(n-2)h(2)+x(n-3)h(3)$$

习　题

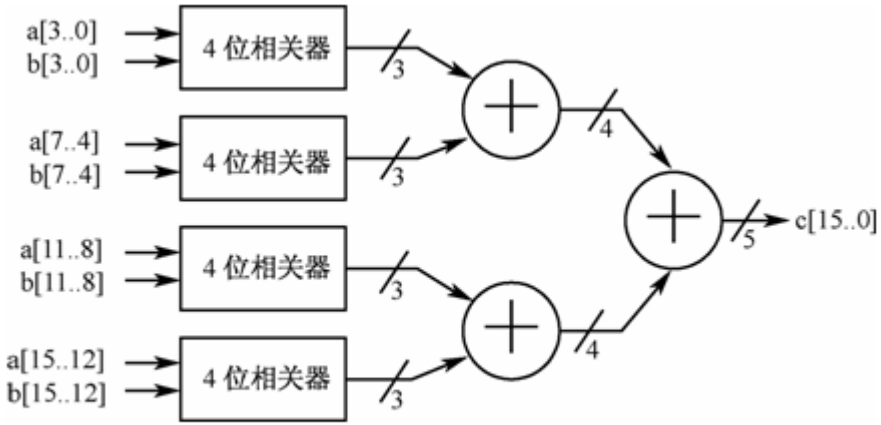

图 7-20　习题 7-5 图

实验与设计

7-1 SPWM脉宽调制控制系统设计

【例 7-9】三角波发生模块

```
module TRANG3 (ADR, OUTD);
    input[9:0] ADR;  output[9:0] OUTD;  wire[9:0] OUTD;
    reg[9:0] OT1;    reg[10:0] CC;
    always @(ADR or CC)   begin
       if (ADR < 10'b1000000000)
       begin  OT1[9:1] <= ADR[8:0] ; OT1[0] <= 1'b0 ;  end
       else   begin   CC <= 11'b10000000000 + (~ADR) ;
        OT1[9:1] <= CC[8:0] ;  OT1[0] <= 1'b0 ;      end    end
      assign OUTD = OT1 ;
endmodule
```

实验与设计

7-1 SPWM脉宽调制控制系统设计

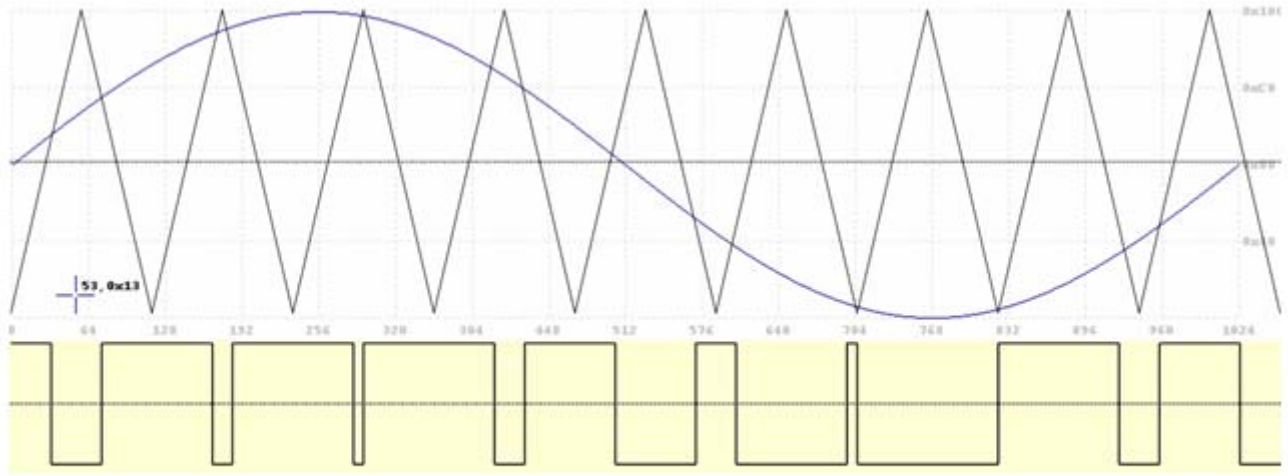

图 7-21 SPWM 波生成原理图

实验与设计

7-1 SPWM脉宽调制控制系统设计

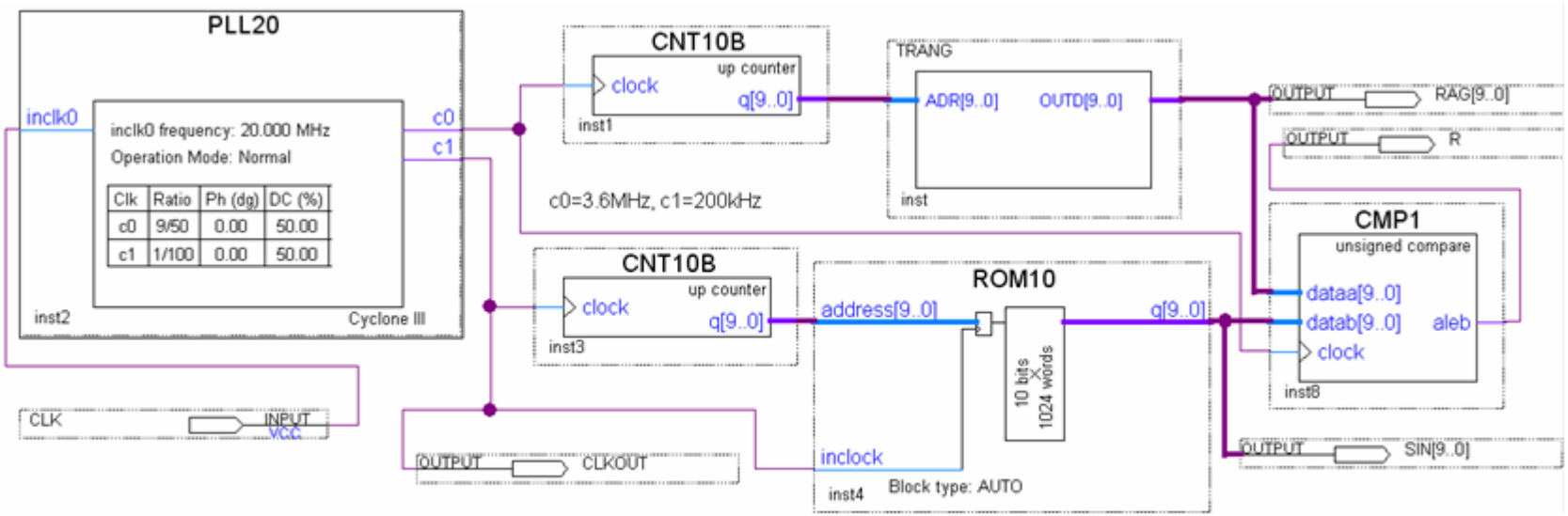

图 7-22　SPWM 波发生器基本电路图

实验与设计

7-1 SPWM脉宽调制控制系统设计

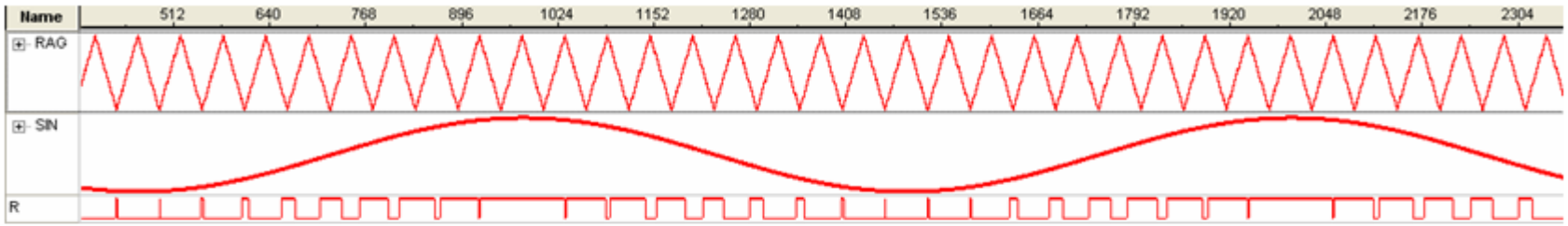

图 7-23　图 7-22 电路的 SignalTap II 实测波形

实验与设计

7-1 SPWM脉宽调制控制系统设计

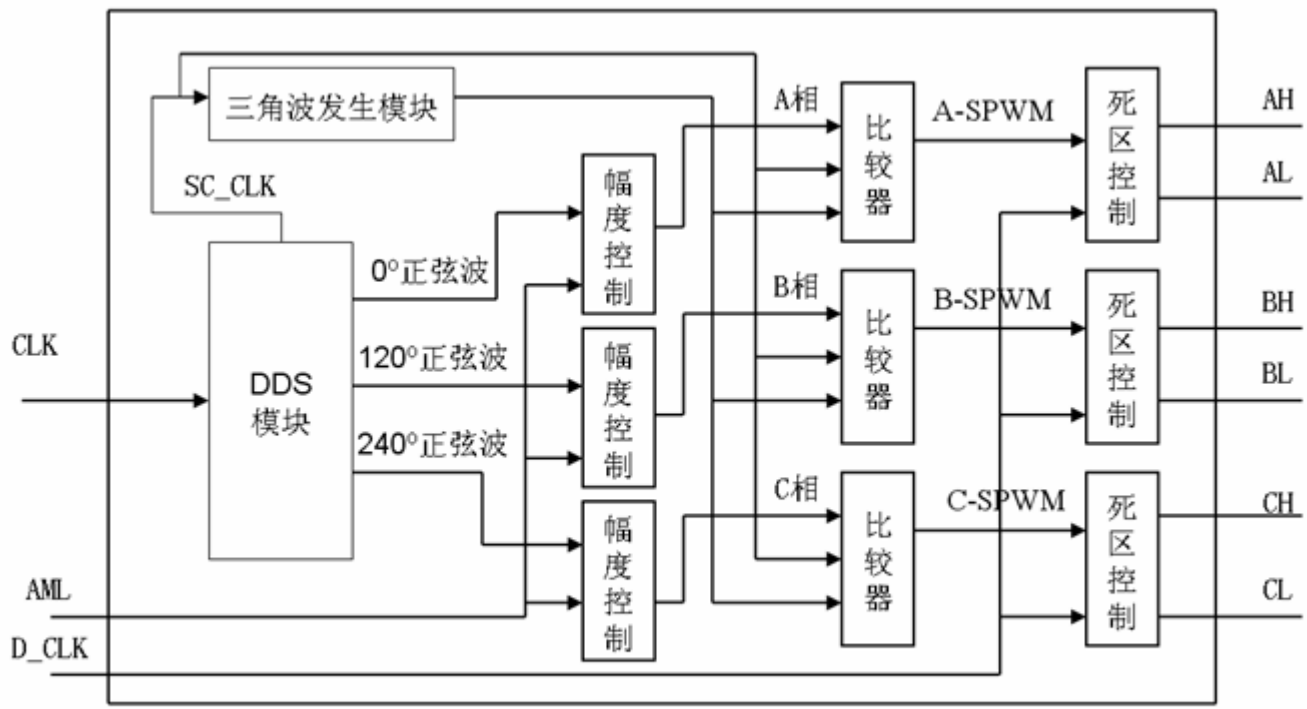

图 7-24　三相 SPWM 控制器电路模块图

实验与设计

7-2 基于DES数据加密标准的加解密系统设计

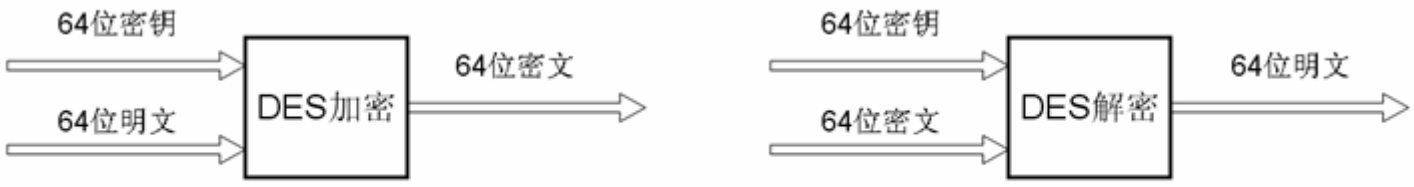

图 7-25 DES 加解密模块示意

实验与设计

7-4 线性反馈移位寄存器设计

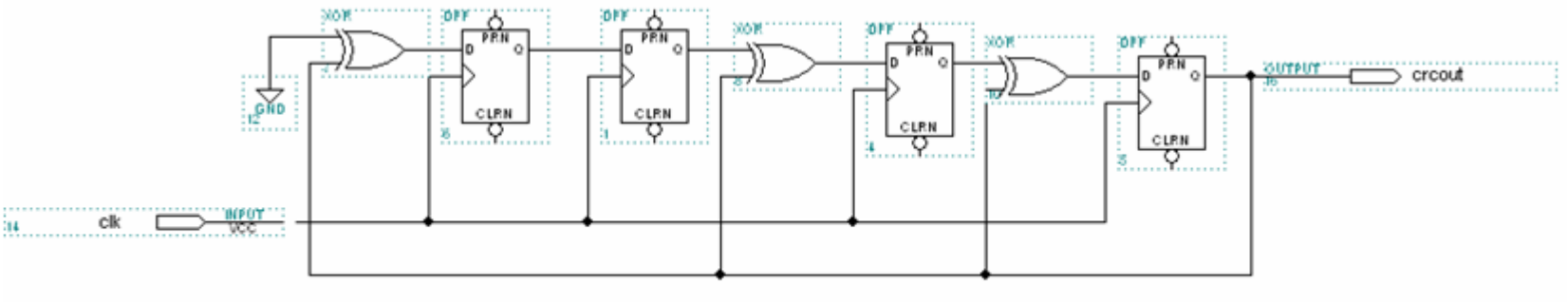

图 7-26　LFSR 举例

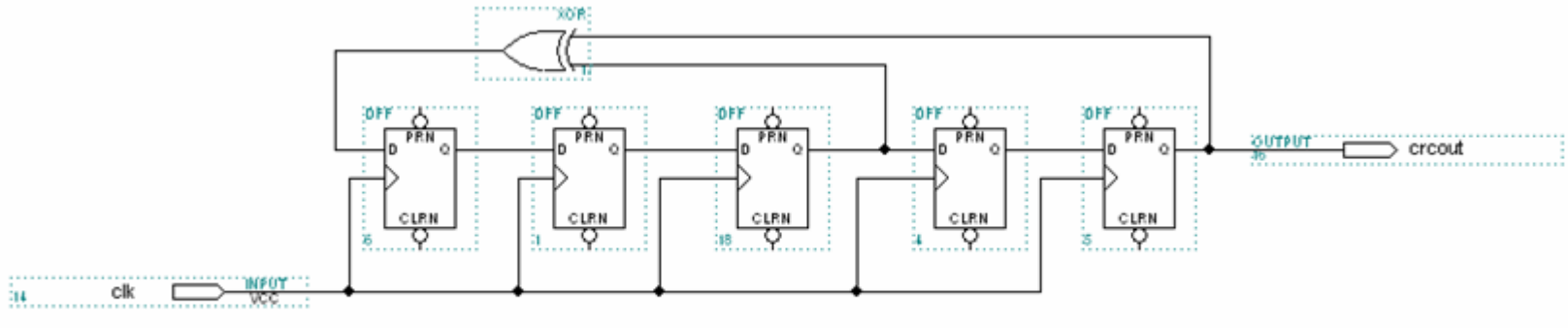

图 7-27　另一种 LFSR 结构

实验与设计

7-5 步进电机细分控制电路设计

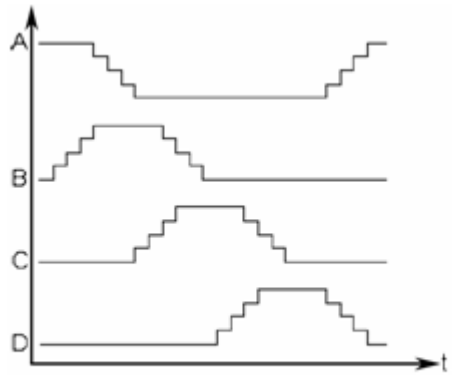

图 7-28 四相步进电机八细分电流波形

实验与设计

7-5 步进电机细分控制电路设计

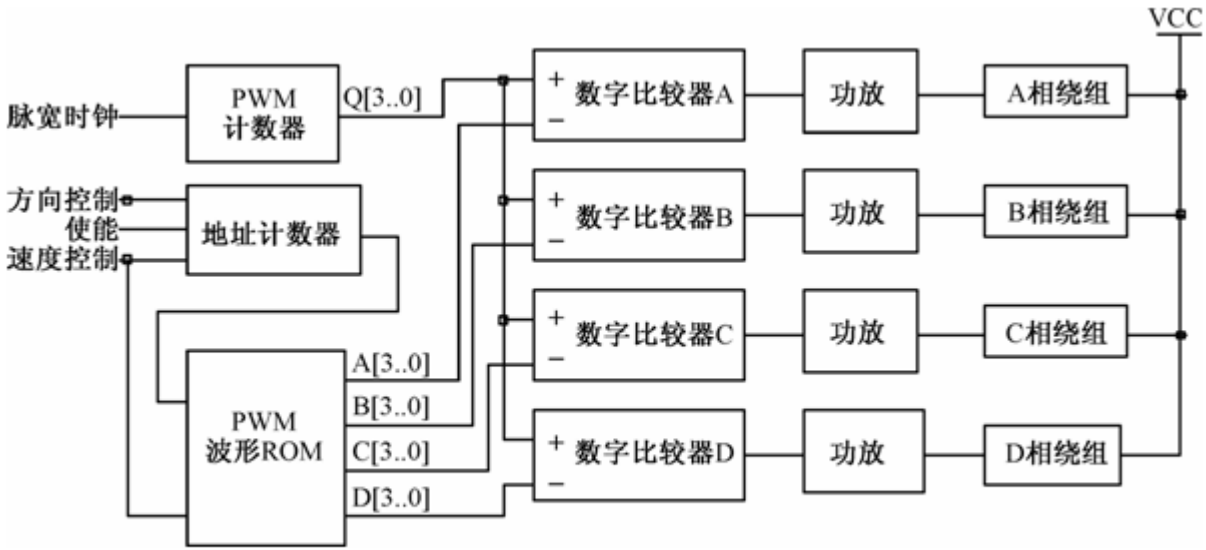

图 7-29 步进电机细分驱动电路结构图

实验与设计

7-5 步进电机细分控制电路设计

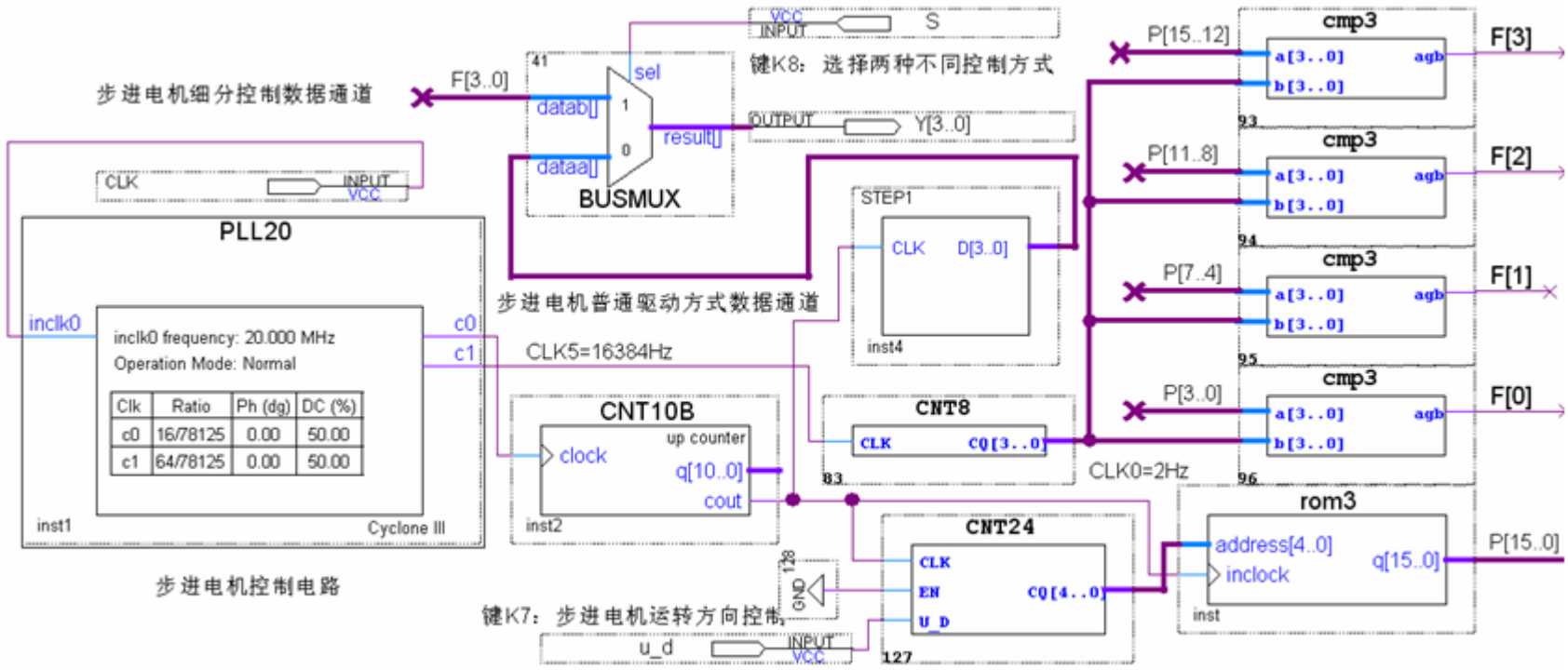

图 7-30 步进电机细分控制电路

实验与设计

7-5 步进电机细分控制电路设计

【例 7-10】步进电机非细分控制程序。

```
module STEP1 (CLK, D);
   input CLK ;  output[3:0] D;  reg[3:0] D;  reg[1:0] CQ;
   always @(CQ)   begin
      case (CQ)
         2'b00 :  D <= 4'b1000 ;     2'b01 :  D <= 4'b0100 ;
         2'b10 :  D <= 4'b0010 ;     2'b11 :  D <= 4'b0001 ;
      endcase    end
   always @(posedge CLK)
     begin  CQ <= CQ + 1 ; end
 endmodule
```

7-7 直流电机综合测控系统设计

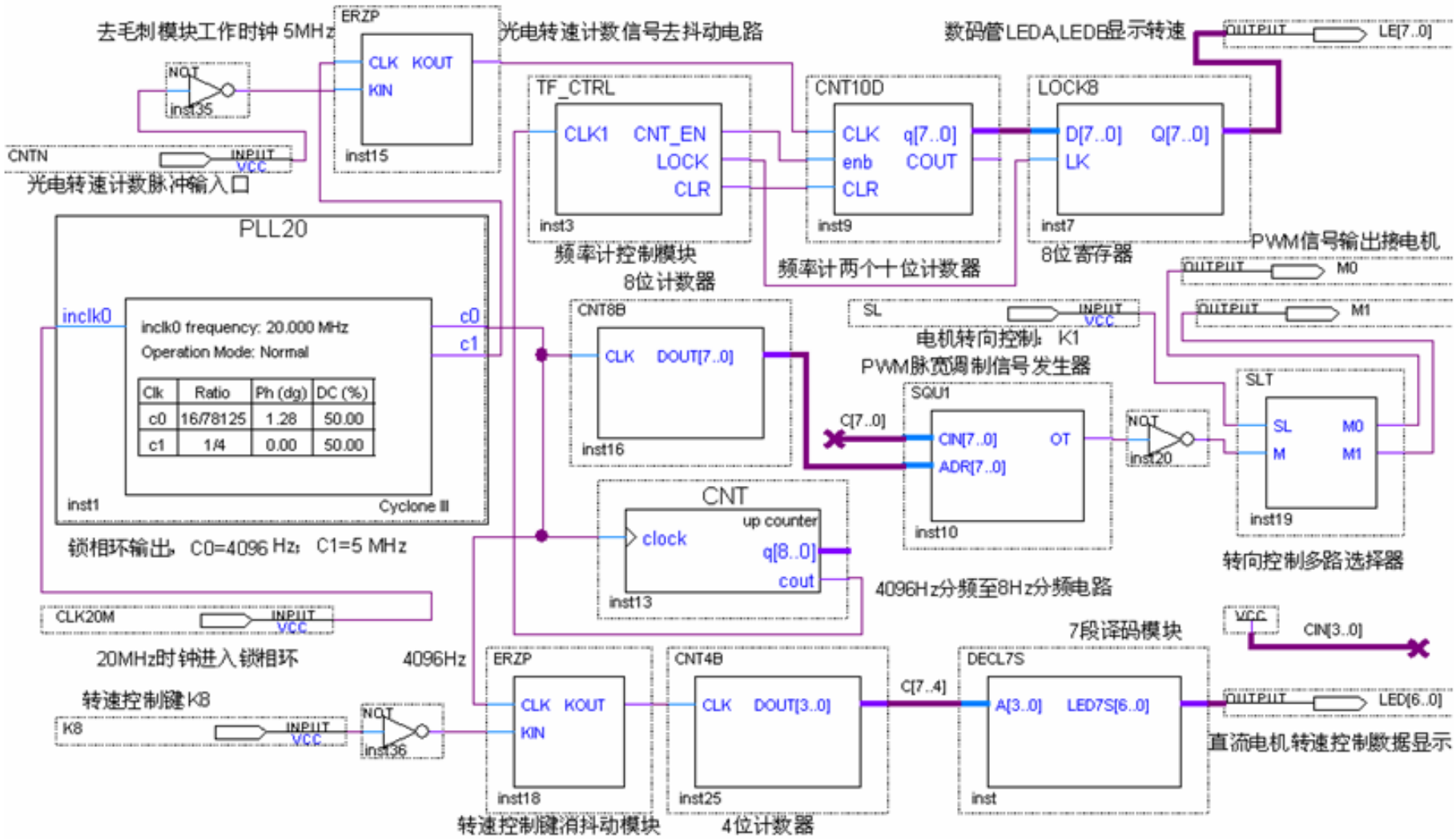

图 7-31 直流电机驱动控制电路顶层设计

实验与设计

7-7 直流电机综合测控系统设计

【例 7-11】

```
module SQU1 (CIN, ADR, OT);
   input[7:0] CIN,ADR;   output OT;  reg OT;
   always @(CIN)   begin
      if (ADR < CIN)  OT <= 1'b0 ;
      else   OT <= 1'b1 ;  end     endmodule
```

实验与设计

7-9 AM幅度调制信号发生器设计

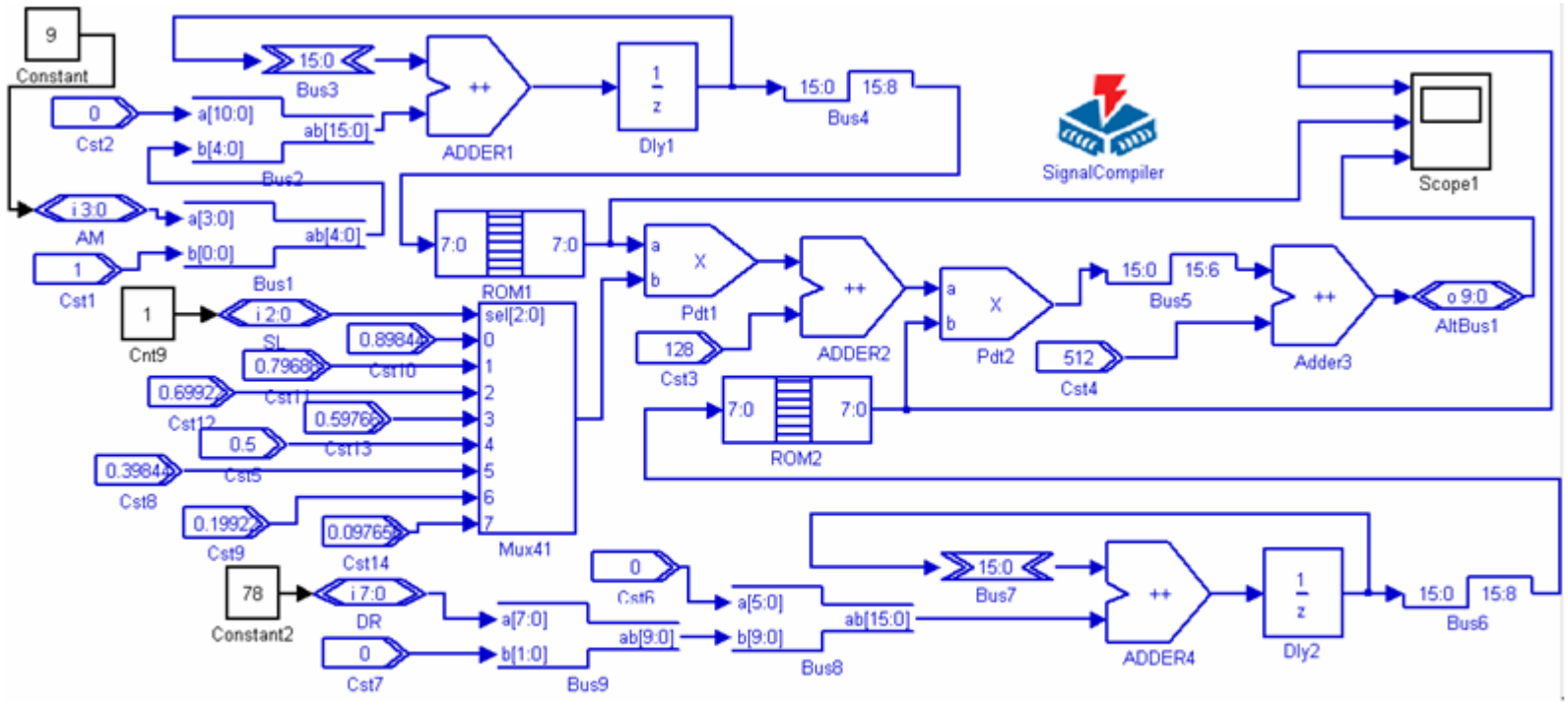

图 7-32　AM 信号发生器 DSP Builder/MATLAB Simulink 模型

实验与设计

7-9 AM幅度调制信号发生器设计

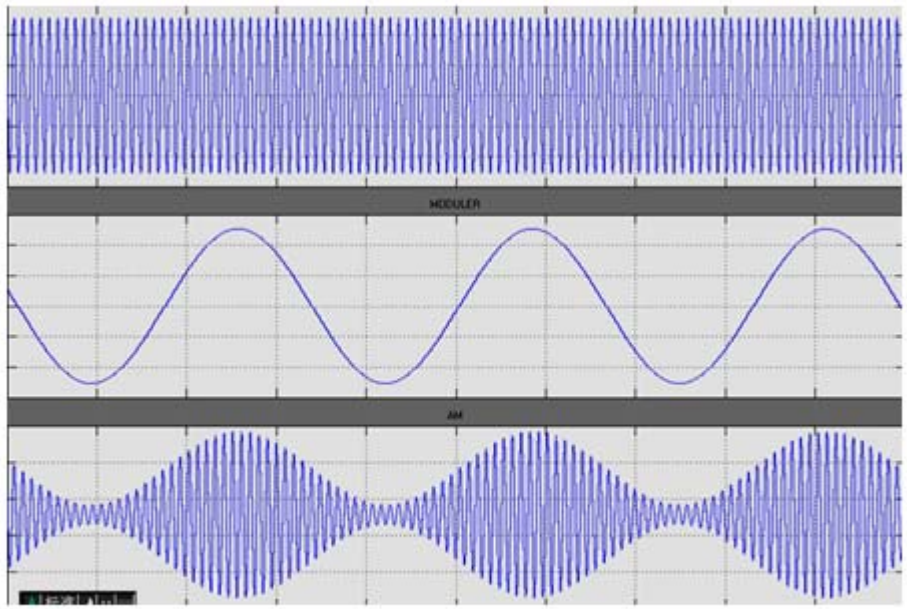

图 7-33　AM 模型仿真波形