西安电子科技大学

集成电路设计导论 课程实验报告

实验名称 ASIC 仿真实验 1-4

人工智能 学院 23200XX 班

姓名 X X X 学号 2300920XXXX

同作者 无

实验日期 2025 年 12 月 19、21 日

成 绩

指导教师评语:

指导教师:

年 月 日

实验报告内容基本要求及参考格式

一、实验目的

二、实验所用仪器（或实验环境）

三、实验基本原理及步骤（或方案设计及理论计算）

四、实验数据记录（或仿真及软件设计）

五、实验结果分析及回答问题（或测试环境及测试结果）

一、实验目的

1. 掌握集成电路设计的基本流程，涵盖需求分析、架构设计、硬件描述语言编程、仿真验证等关键环节，形成从理论到实践的完整认知。

2. 深入理解各类核心数字/模拟电路的工作原理，包括MOS管特性、CMOS反相器延迟机制、简单状态机逻辑、运算电路（超前进位加法器、加/减法器）、线性反馈移位寄存器（LFSR）及移位式二进制乘法器的设计原理与实现方法。

3. 熟练运用Verilog HDL进行RTL级电路设计,精通ModelSim EDA、NI MultiSIM14.0等仿真软件的操作，能够独立完成电路搭建、参数配置、仿真运行及结果分析。

4. 培养电路设计的优化与扩展能力，能够根据实验要求对现有电路进行功能改进，如调整彩灯闪烁模式、扩展LFSR阶数、优化CMOS反相器延迟参数等。

5. 提升实验数据处理、问题分析及报告撰写能力，能够编制完备的测试向量，全面验证电路功能，并清晰阐述实验原理、步骤、结果及结论。

二、实验所用仪器（或实验环境）

（一）硬件设备

微型计算机

（二）软件环境

1. NI MultiSIM14.0 EDA软件: 用于实验1（MOS管特性测试、CMOS反相器延迟参数测试），提供虚拟仪器（IV分析仪、示波器）、元器件库及仿真环境支持。

2. ModelSim EDA软件: 用于实验2（简单状态机）、实验3（运算和通信电路）、实验4（移位式二进制乘法器），支持Verilog HDL代码编译、仿真验证及波形分析。

三、实验基本原理及步骤（或方案设计及理论计算）

（一）实验1: 晶体MOS管及其电路（MOS管特性/反相器延迟参数）

1. 实验原理

（1）MOS管输出特性

MOS管的输出特性反映漏极电流(I_D)与漏源电压(V_DS)的关系,受栅源电压(V_GS)、沟道宽长比（W/L）、阈值电压（VTO）等参数影响。当V_GS大于VTO时，MOS管导通，形成导电沟道，I_D随V_DS变化呈现饱和区、线性区等特征。通过IV分析仪可精准测量不同V_GS下的I_D-V_DS曲线，分析参数对器件性能的影响。

（2）CMOS反相器延迟参数

CMOS反相器由NMOS管（下拉管）和PMOS管（上拉管）串联组成，延迟参数（上升

时间 t_r、下降时间 t_f、传输延迟 t_pd）是衡量电路开关速度的核心指标。t_r 是输出从低电平（0.1V_CC）上升到高电平（0.9V_CC）的时间，t_f 是输出从高电平（0.9V_CC）下降到低电平（0.1V_CC）的时间，t_pd 为输入输出信号跳变的平均延迟。延迟参数与 MOS 管跨导（KP）、沟道尺寸、负载电容等密切相关。

2. 实验步骤

1. MOS 管输出特性测试:

- 打开 NI MultiSIM14.0，搭建 NMOS 管输出特性测试电路，选择“TRANSISTORS_VIRTUAL”库中的 MOS_N_4T 晶体管，连接 IV 分析仪（通过“仿真”→“仪器”→“IV 分析仪”调用）。

- 配置 IV 分析仪参数：V_DS 起始值 0V、停止值 24V、增量 10mV；V_GS 起始值 0V、停止值 12V、步数 13，确认器件类型为“NMOS”。

- 设置 NMOS 管模型参数:双击晶体管,点击“编辑模型”,设置阈值电压 VTO=0.7V，跨导参数 KP=0.2A/V²，沟道宽度 W 分别为 100um、150um、200um、250um，长度 L 固定为 100um。

- 运行仿真：点击“运行”按钮，仿真完成后暂停，通过图示仪观察并记录不同 W 下的 I_D-V_DS 曲线。

2. CMOS 反相器延迟参数测试:

- 搭建 CMOS 反相器电路，VCC 设置为 5V，GND 为接地端，信号源选用“BIPOLAR_VOLTAGE”(20MHz),负载电容 C=0.3pF,NMOS/PMOS 管分别选用 MOS_N 和 MOS_P。

- 配置信号源参数：正脉冲电压 5V、负脉冲电压 0V、上升/下降时间 1ps、频率 20MHz。

- 场景仿真:

- 场景：调整 MOS 管沟道尺寸，使 t_r 和 t_f 趋于相等，记录最终参数配置。

（二）实验 2：简单状态机实验（循环彩灯控制电路）

1. 实验原理

本实验基于“计数器+解码器”架构设计循环彩灯控制电路，核心是通过状态机逻辑实现 16 个彩灯的有序控制。

- 分频计数器（cnt1）：4 位同步计数器，通过 load 信号加载 data_in 值设定分频系数，计数至 15（4′b1111）后产生 ena 信号，实现时钟分频，控制彩灯闪烁频率。

- 模式计数器（cnt2）：4 位同步计数器，ena 信号有效时计数，生成 0¯15 的序列，为解码器提供输入。

- 4-16 解码器：将 cnt2 的 4 位二进制输入转换为 16 位彩灯控制信号（lamp_ctl），“1”表示点亮，“0”表示关闭，通过 case 语句定义点亮顺序。

- 复位逻辑：reset 信号（高有效）初始化计数器和输出，确保电路启动状态一致。

2. 实验步骤

1. 电路设计：使用 Verilog HDL 编写彩灯控制电路代码（module exercise_1），定义 reset、clk、load、data_in（输入）和 lamp_ctl（输出）端口，实现分频计数器、模式计数器、解码器及输出锁存逻辑。

2. 测试向量设计:编写 testbench 文件,生成复位信号(t_reset)、时钟信号(t_clk,周期 1000ns)、置数控制信号（t_load）及置数数据（t_data_in），通过组件例化连接被测电路。

3. 仿真验证:在 ModelSim 中加载设计文件和 testbench,运行仿真,观察 lamp_ctl 波形，验证彩灯顺时针点亮（0→1→…→F）功能。

4. 功能扩展:

   - 修改 cnt2 为递减计数，实现彩灯逆时针点亮（F→E→…→0）。

（三）实验 3：运算和通信电路设计（16-bit CLA + LFSR）

1. 实验原理

（1）16-bit 超前进位加法器（CLA）

超前进位加法器通过提前计算进位信号，解决串行进位延迟问题。核心是定义进位生成项（G = A & B）和进位传递项（P = A ^ B），基于 4 位 CLA 模块递归扩展为 16 位:

-4 位 CLA 模块:通过超前进位逻辑计算 C1¯C4,输出 4 位和 S 及组进位生成项(GG)、组进位传递项(PG),其中 GG = G3 + G2 ·P3 + G1 ·P3 ·P2 + G0 ·P3 ·P2 ·P1,PG = P3 ·P2 ·P1 ·P0。

-16 位 CLA 扩展：4 个 4 位 CLA 模块级联，通过顶层超前进位逻辑计算 C4、C8、C12、C16，实现 16 位加法运算。

（2）16-bit 加/减法器

基于二进制补码规则，在 16 位 CLA 基础上添加加减控制逻辑：Add 模式(Cin=0)时，直接执行 A+B；Sub 模式（Cin=1）时，B 经异或门翻转（B = B ^ 1），执行 A + (-B)，实现减法功能。

2. 实验步骤

1. 层次化设计:

   - 用 Verilog HDL 实现 4 位 CLA 模块，验证进位计算和加法功能。

- 基于 4 位 CLA 递归设计 16 位 CLA 电路，编写顶层模块。

- 扩展 16 位加/减法器，添加加减控制信号，实现 Add/Sub 模式切换。

2. 仿真测试:

- 对 4 位 CLA 和 16 位 CLA 进行仿真，输入不同操作数（如 0x0001+0x0001、0xFFFF+0x0001），验证加法结果正确性。

- 测试 16 位加/减法器，覆盖正数相加、负数相加、正负相减等场景，验证补码运算准确性。

（四）实验 4: 算数运算电路实验（实用型移位式二进制乘法器电路）

1. 实验原理

移位式二进制乘法器采用“部分积生成-右移累加”策略,实现 16-bit × 16-bit 乘法，结果存储于 32-bit 寄存器:

- 部分积生成: 根据乘数最低有效位（LSB）判断，若 LSB 为 1，部分积为被乘数; 若为 0，部分积为 0。

- 累加与移位: 部分积与累加器（32-bit）左半部分相加，结果右移 1 位; 同时乘数右移 1 位，重复操作 16 次，累加器最终存储乘积。

- 控制电路: 协调部分积生成、加法、移位的时序，确保乘法过程有序完成，进位位包含在累加结果中，保证数据动态范围。

2. 实验步骤

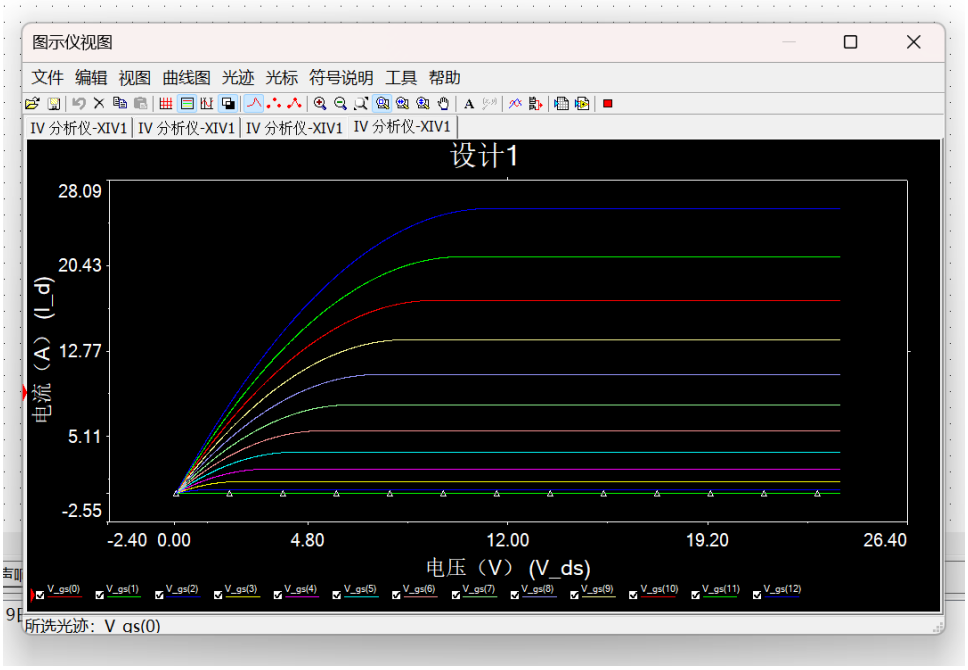
1. 电路设计:用 Verilog HDL 描述移位式乘法器,包含被乘数寄存器、乘数寄存器、32-bit 部分积累加器、16-bit 加法器及控制逻辑模块。

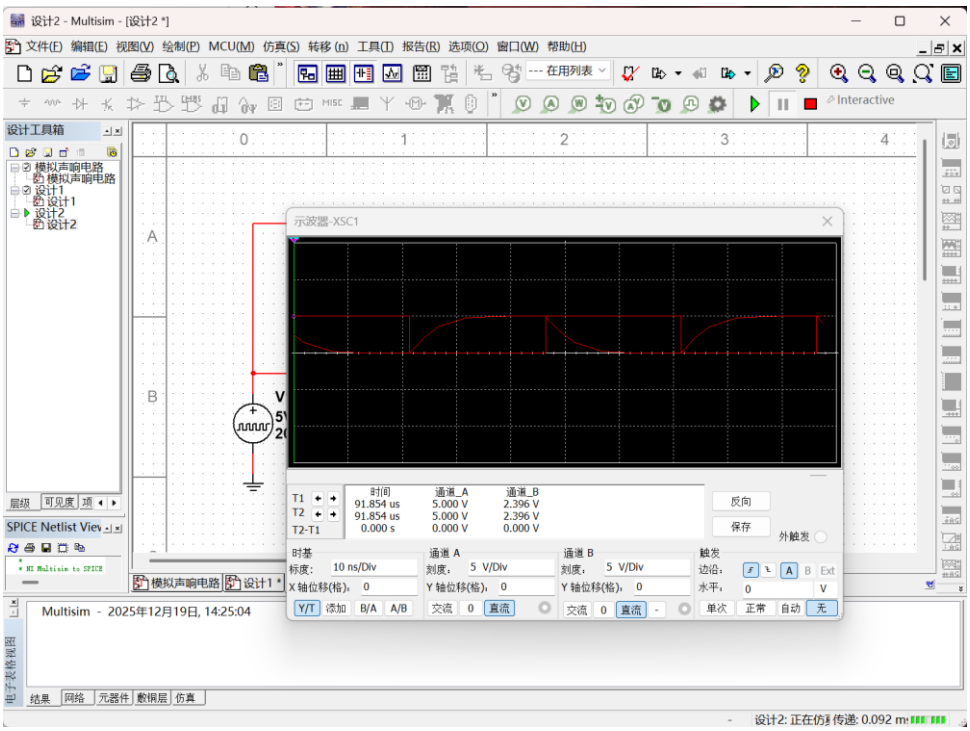
2. 仿真验证: 在 ModelSim 中搭建仿真环境，输入不同被乘数和乘数（如 0x0002 × 0x0003、0xFFFF × 0xFFFF），观察 32-bit 乘积输出，验证结果正确性。

3. 完备性测试: 更改测试参数，覆盖小数值、大数值、零值、负数等场景，确保乘法器在全输入范围内功能可靠。

四、实验数据记录（或仿真及软件设计）

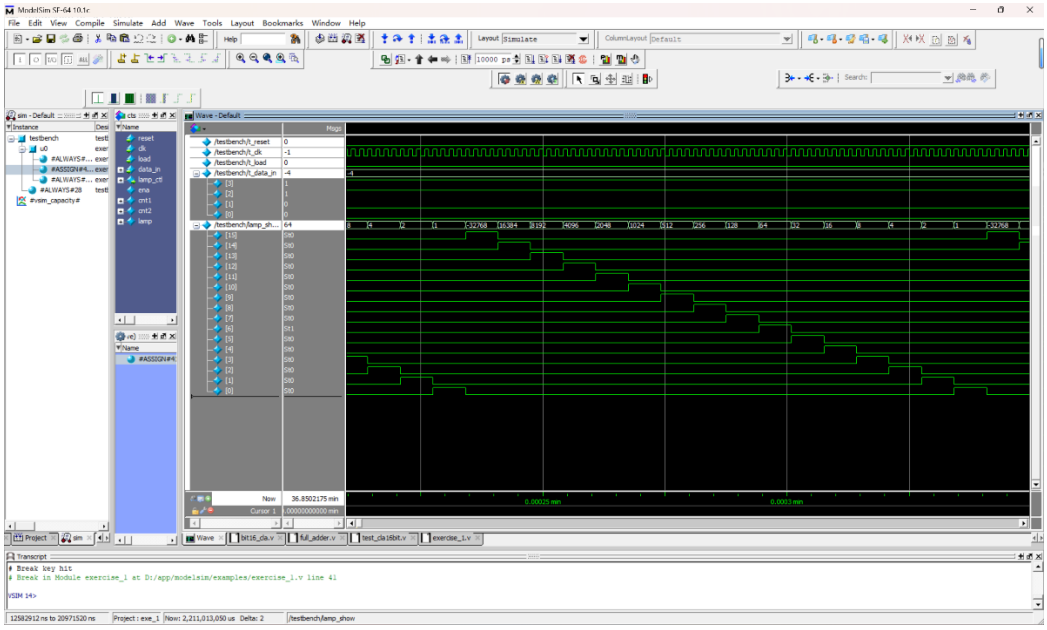
（一）实验 1: 晶体 MOS 管及其电路





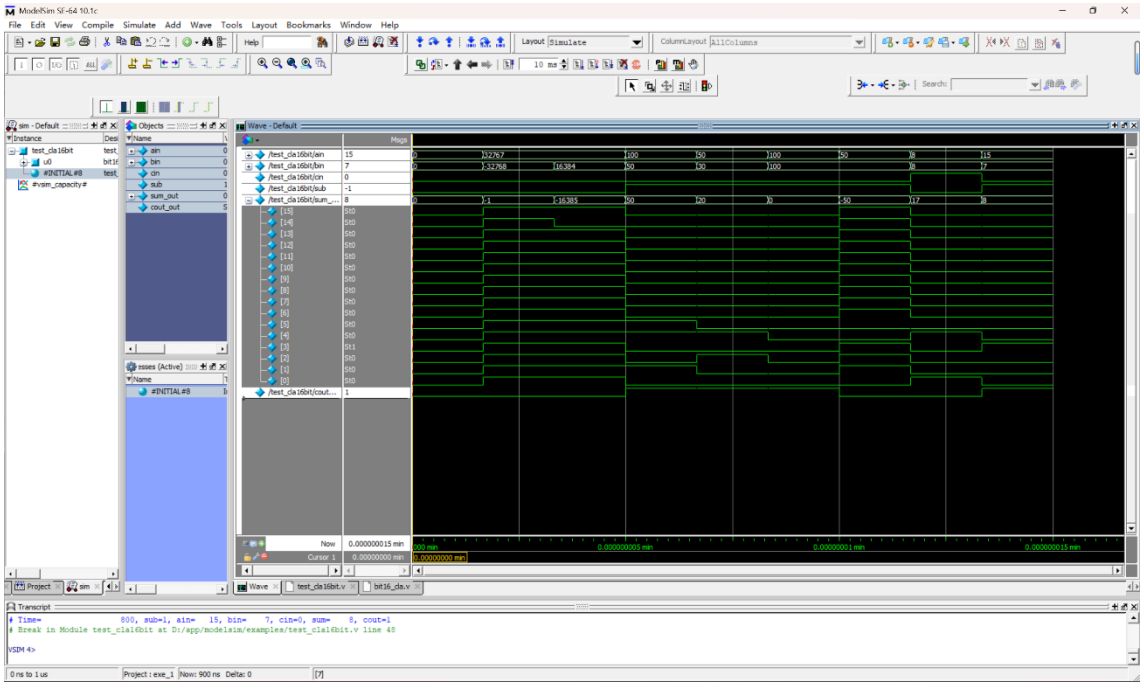
通过增加调整 MOS 管沟道宽度，即可使 t_r 和 t_f 趋于相等，记录最终参数配置。

（二）实验 2：简单状态机实验



逆时针点亮：修改 cnt2 为递减计数后，点亮顺序为 F→E→…→0，波形显示 lamp_ct1 从 16'b1000_0000_0000_0000 依次向低位翻转。或修改亮灯循环次序，根据灯的序号由高到低进行亮灯。

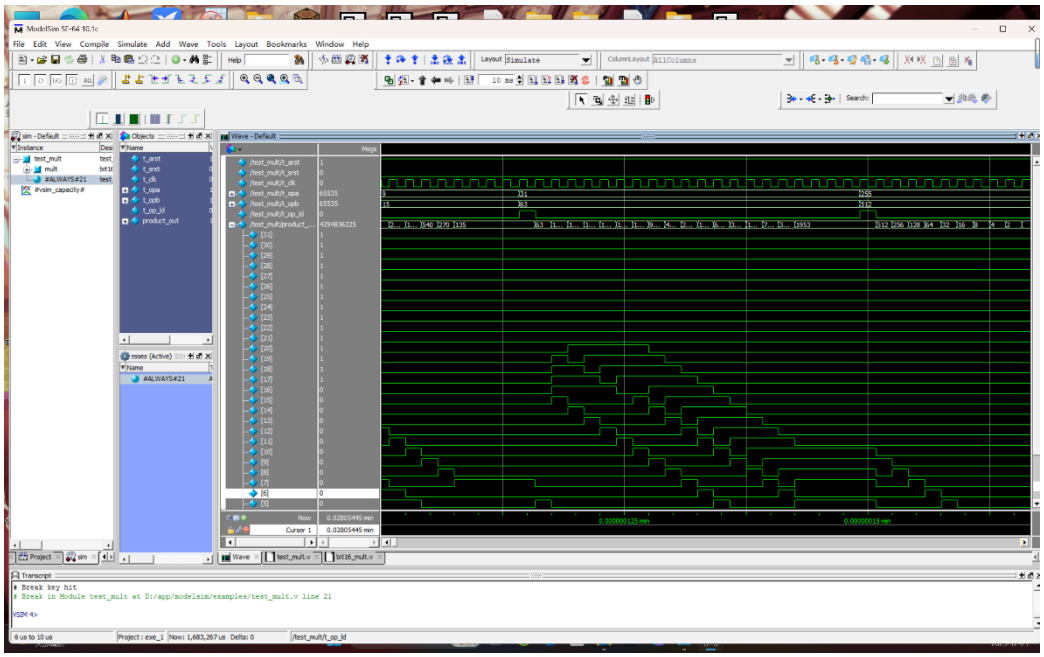
（三）实验 3：运算和通信电路设计



由于算法主要采取了补码运算的原理，主要通过添加了一个参数 sub 作为判断：

如果为加法，则直接相加；如果为减法，则最终结果加一。

（四）实验 4：移位式二进制乘法器实验



在原有代码的基础上，添加了乘法器功能。完成了乘法运算。

五、实验结果分析及回答问题（或测试环境及测试结果）

（一）实验 1：晶体 MOS 管及其电路

1. MOS 管宽长比对性能的影响

由实验数据可知，当 V_GS 和 V_DS 固定时，漏极电流 I_D 随沟道宽度 W 的增大而线性增加。原因是 W 增大使沟道导电截面积扩大，载流子（电子）传输能力增强，器件导通性能提升。但 W 过大会增加器件面积和寄生电容，导致开关速度下降，实际设计中需在导电能力与寄生参数间权衡。

2. CMOS 反相器延迟参数分析

- 延迟参数分析方法：通过示波器观察输入输出波形，拖动时标测量 t_r（0.1V_CC→0.9V_CC）和 t_f（0.9V_CC→0.1V_CC），t_pd 取两者平均值。

- 影响因素：

-MOS 管参数：NMOS 管 KP 增大，t_f 缩短；PMOS 管 KP 增大，t_r 缩短；VTO 增大，延迟时间延长。

- 沟道尺寸：调整 N/P 管宽长比可平衡 t_r 和 t_f，如实验中设置 NMOS W=200um、PMOS W=400um，使 t_r ≈ t_f ≈ 7.6ns。

- 负载电容：负载电容越大，充放电时间越长，延迟参数越大，实验中固定 C=0.3pF 以排除其干扰。

（二）实验 2：简单状态机实验

1. 4-bit 递减计数器设计（Verilog 代码）

代码:

```
//***** ASIC Exercise - 1 *****
//*** Counter + Decoder ***
//*** can be used to control colour lamps ***
//*** colour lamps arrangement ***
//*******************
//*** d e f 0 1 2 ***
//*** c         3 ***
//*** b         4 ***
//*** a 9 8 7 6 5 ***
//*******************

module exercise_1 (reset,clk,load,data_in,

                   lamp_ctl);

//* input/output ports definition *
input           reset,clk,load;
input [3:0]     data_in;

output  [15:0]  lamp_ctl;
reg     [15:0]  lamp_ctl;

//* in-circuit signals definition *
wire            ena;
reg     [3:0]   cnt1,cnt2;
reg     [15:0]  lamp;

//* circuit RTL description *
//* 4-bit mode-16 counter with sync_data load*
//* used as freq_divider *
always @(posedge reset or posedge clk)
  if (reset)
    cnt1 <= 4'b0000;
  else if (load)
    cnt1 <= data_in;
  else if (cnt1 == 4'b1111)
    cnt1 <= data_in;
  else
    cnt1 <= cnt1 + 1'b1;

assign ena = &cnt1;

//* 4-bit mode-16 counter *
always @(posedge reset or posedge clk)
  if (reset)
    cnt2 <= 4'b1111;
  else if (ena)
    begin
      if (cnt2 == 4'b0000)
        cnt2 <= 4'b1111;
      else
        cnt2 <= cnt2 - 1'b1;
    end
```

```
//* 4-16 decoder *
always @(reset or cnt2)
  begin
    if (reset)
      lamp = 16'h0000;
    else
      case (cnt2)
        4'b0000 : lamp = 16'b0000_0000_0000_0001;
        4'b0001 : lamp = 16'b0000_0000_0000_0010;
        4'b0010 : lamp = 16'b0000_0000_0000_0100;
        4'b0011 : lamp = 16'b0000_0000_0000_1000;
        4'b0100 : lamp = 16'b0000_0000_0001_0000;
        4'b0101 : lamp = 16'b0000_0000_0010_0000;
        4'b0110 : lamp = 16'b0000_0000_0100_0000;
        4'b0111 : lamp = 16'b0000_0000_1000_0000;
        4'b1000 : lamp = 16'b0000_0001_0000_0000;
        4'b1001 : lamp = 16'b0000_0010_0000_0000;
        4'b1010 : lamp = 16'b0000_0100_0000_0000;
        4'b1011 : lamp = 16'b0000_1000_0000_0000;
        4'b1100 : lamp = 16'b0001_0000_0000_0000;
        4'b1101 : lamp = 16'b0010_0000_0000_0000;
        4'b1110 : lamp = 16'b0100_0000_0000_0000;
        4'b1111 : lamp = 16'b1000_0000_0000_0000;
        default : lamp = 16'b0000_0000_0000_0000;
      endcase
    end

//* latch the output control signal *
always @(posedge reset or posedge clk)
  if (reset)
    lamp_ctl <= 16'h0000;
  else
    lamp_ctl <= lamp;

endmodule
```

5. 彩灯闪烁方式更改及效果

- 更改方法：除修改计数器为递减模式外，可直接调整解码器映射关系，如 cnt2=4'b0000 对应 lamp=16'b1000_0000_0000_0000，cnt2=4'b0001 对应 lamp=16'b0100_0000_0000_0000，无需修改计数器逻辑。

（三）实验 3：运算和通信电路设计

1. 测试向量编制方法

代码：

```
//************************************************
//*** Modified bit16_cla with Add/Sub support ****
//************************************************

module bit16_cla_add_sub (
    input [15:0] ain, bin,
    input cin,
    input add_sub,      // ????????
    output [15:0] sum,
    output cout
```

```
);

//*** ?????? ***
wire [15:0] bin_actual;
wire cin_actual;

//*** ??????? ***
assign bin_actual = add_sub ? ~bin : bin;      // ?????
assign cin_actual = add_sub ? 1'b1 : cin;      // ???cin=1

//*** ??4?4?CLA??????? ***
wire p15_12, p11_8, p7_4, p3_0;
wire g15_12, g11_8, g7_4, g3_0;
wire c11, c7, c3;

//*** ???4?4?CLA?? ***
bit4_cla t3 (
    .ain(ain[15:12]),
    .bin(bin_actual[15:12]),
    .cin(c11),
    .sum(sum[15:12]),
    .cas_p(p15_12),
    .cas_g(g15_12)
);

bit4_cla t2 (
    .ain(ain[11:8]),
    .bin(bin_actual[11:8]),
    .cin(c7),
    .sum(sum[11:8]),
    .cas_p(p11_8),
    .cas_g(g11_8)
);

bit4_cla t1 (
    .ain(ain[7:4]),
    .bin(bin_actual[7:4]),
    .cin(c3),
    .sum(sum[7:4]),
    .cas_p(p7_4),
    .cas_g(g7_4)
);

bit4_cla t0 (
    .ain(ain[3:0]),
    .bin(bin_actual[3:0]),
    .cin(cin_actual),  // ??????cin
    .sum(sum[3:0]),
    .cas_p(p3_0),
    .cas_g(g3_0)
);

//*** ?????? ***
bit4cla_logic cla_logic (
    .cin(cin_actual),   // ??????cin
    .p3(p15_12), .g3(g15_12),
    .p2(p11_8), .g2(g11_8),
```

```
    .p1(p7_4), .g1(g7_4),
    .p0(p3_0), .g0(g3_0),
    .c3(cout), .c2(c11), .c1(c7), .c0(c3)
);

endmodule
```

（四）实验 4：移位式二进制乘法器实验

1. 乘法器设计

代码:

```
//************************************************
//*** first generated on 28th,may,2017 ***********
//************************************************

module  bit16_mult  (arst,srst,clk,
                     opa,opb,op_ld,

                     mult_out);

input           arst,srst,clk;
input   [15:0]  opa,opb;
input           op_ld;

output  [31:0]  mult_out;
reg     [31:0]  mult_out;

//************************************************

//*** generate calculate enanle signal ***
reg   cal_enb;
wire  cal_end;

always @(posedge clk or negedge arst)
  if (~arst)
    cal_enb <= 1'b0;
  else if (srst || cal_end)
    cal_enb <= 1'b0;
  else if (op_ld)
    cal_enb <= 1'b1;

//*** shift - add cycle control ***
reg [3:0] cal_cnt;

always @(posedge clk or negedge arst)
  if (~arst)
    cal_cnt <= 4'h0;
  else if (srst || op_ld)
    cal_cnt <= 4'h0;
  else if (cal_enb)
    cal_cnt <= cal_cnt + 1'b1;

assign  cal_end = &cal_cnt;

//*** partial-product select ***
wire  [15:0]  par_product;
```

```
reg   [15:0]  multiplicand;

assign par_product = (mult_out[0] == 1'b1) ? multiplicand : 16'h0000;

//*** sum of partial-product ***
wire  [16:0] sum_pproduct;

assign  sum_pproduct = par_product + mult_out[31:16];

//*** operands loda-in & shift operation ***


always @(posedge clk or negedge arst)
  if (~arst)
    begin
      multiplicand <= 16'h0000;
      mult_out <= 32'h00000000;
    end
  else if (srst)
    begin
      multiplicand <= 16'h0000;
      mult_out <= 32'h00000000;
    end
  else if (op_ld)
    begin
      multiplicand <= opa;
      mult_out <= {16'h0000,opb};
    end
  else if (cal_enb)
    mult_out <= ({sum_pproduct,mult_out[15:0]}) >> 1;

endmodule


//================================================
//*** ?????4-bit?????? ****************
//================================================

module  bin_cnt_4bit  (
  input   async_rst,
  input   sync_rst,
  input   clk,
  input   enb,
  input   [3:0] d_in,
  input   d,

  output  [3:0] cnt_out,
  output  cout
);

reg [3:0] cnt;

always @(posedge clk or posedge async_rst)
  if (async_rst)
    cnt <= 4'b0000;
  else if (sync_rst)
    cnt <= 4'b0000;
```

```
  else if (d)
    cnt <= d_in;
  else if (enb)
    cnt <= cnt + 4'b0001;

assign cnt_out = cnt;
assign cout = (cnt == 4'b1111) ? 1'b1 : 1'b0;

endmodule
```

六、实验总结与体会

本次集成电路设计系列实验按实验 1 至实验 4 的顺序，系统覆盖了 MOS 管特性、状态机、运算电路及乘法器设计，通过理论学习与实践操作相结合，全面提升了集成电路设计能力。

实验 1 中，通过 MOS 管特性测试理解了器件参数对导电性能的影响，通过 CMOS 反相器延迟测试掌握了电路时序优化方法，深刻认识到模拟电路参数与数字电路性能的关联。实验 2 的状态机设计让我熟练运用 Verilog HDL 实现时序逻辑，通过修改计数器和解码器逻辑,体会到数字电路设计的灵活性,学会了通过测试向量验证电路功能。实验 3 的 16-bit CLA 设计中，层次化设计方法显著提高了设计效率，超前进位逻辑的递归实现让我深入理解了进位优化的核心思想。实验 4 的移位式乘法器设计，通过“部分积生成-右移累加”的逻辑实现，理解了复杂运算电路的时序协调方法，完备性测试确保了电路在全输入范围内的可靠性。

在实验过程中，也遇到了诸多问题：如 LFSR 反馈逻辑设计错误导致序列周期不符合预期，通过查阅生成多项式原理修正了反馈节点；乘法器时序控制逻辑调试中，通过波形分析定位了移位与累加的时序冲突。这些问题的解决，培养了我的问题分析和调试能力。

本次实验不仅巩固了集成电路设计的基础知识，还提升了硬件描述语言编程、仿真软件操作及实验数据处理能力，深刻体会到理论与实践结合的重要性。未来，我将进一步学习复杂电路设计技巧，优化电路性能，为后续专业学习和工程实践打下坚实基础。