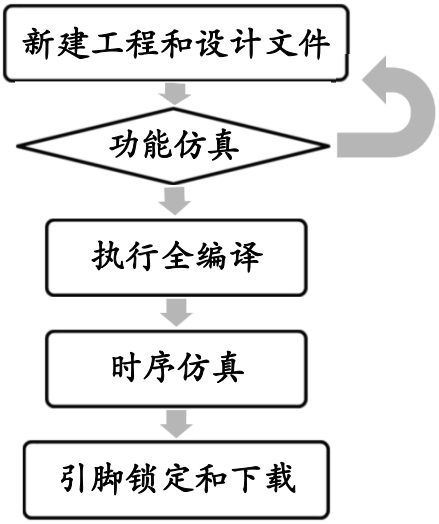
EDA讲座2

- 时序仿真和层次化设计
- 硬件描述语言2
- EDA作业二

设计流程



原理图、硬件描述语言

建立波形文件
检查逻辑功能是否正确

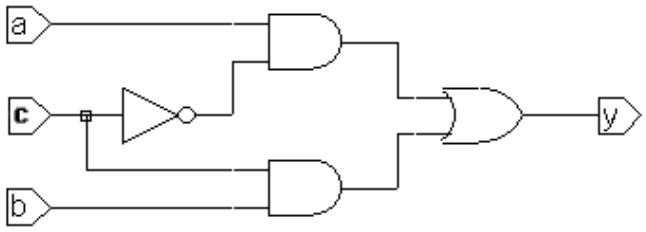
综合、布局布线

验证电路的时序

下载验证

2

二选一数据选择器



a	b	c	y
X	X	0	a
X	X	1	b

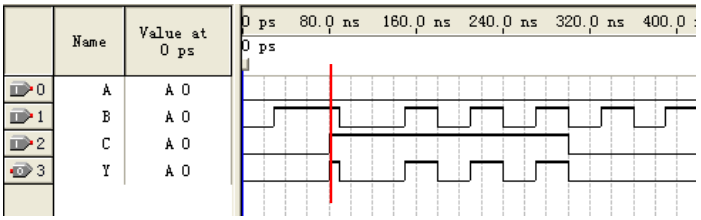
3

1.时序仿真

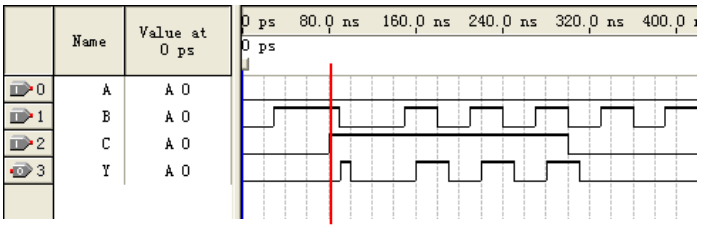
在全编译之后验证设计的时序性能

- 打开波形文件*.VWF，修改激励
  Edit → Set End Time: 10ns ~ 100us
- 在Simulation → Options中指定仿真工具
  选择 QuartusII Simulator
- 执行时序仿真 Simulation → Run timing Simulation
- 观察仿真结果

4



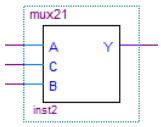
Functional



Timing

5

2.封装元件和层次化设计



- 封装成同名的元件符号

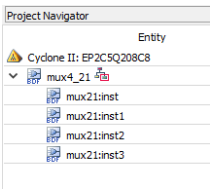
  File→Creat/Update →Creat Symbol Files For Current File
- 新建原理图，添加封装好的元件mux21
- 绘图后，将原理图命名为mux4_21并保存
- 将其设定为顶层文件：

  导航栏Files页签：选中mux4_21.bdf，

  右键菜单Set as Top-level Entity

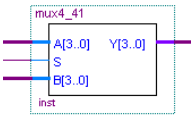
  导航栏Hierarchy页签：顶层文件即改为mux4_21

编译后导航栏中的层次：



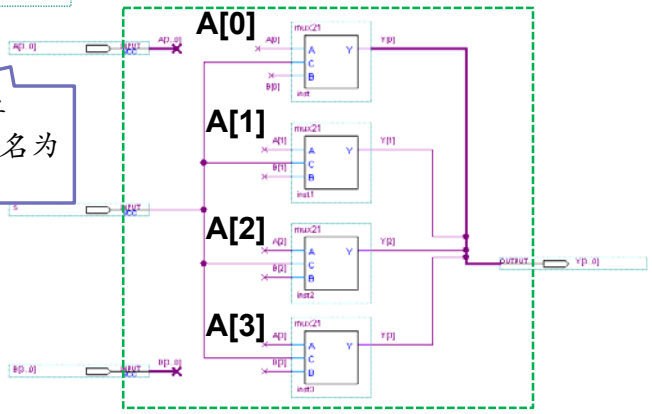
6

扩展：4位二选一

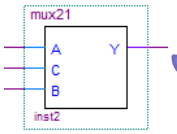


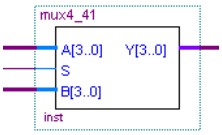
A[3..0]

选中连线，右键菜单Properties，name命名为A[3..0]

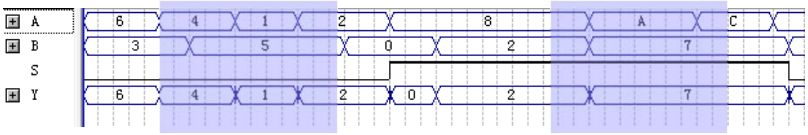


扩展：4位二选一

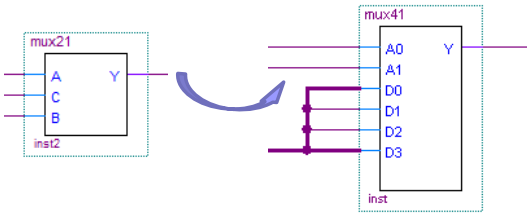





```
module mux4_21 (A,B,S,Y);
  input [3:0] A;
  input [3:0] B;
  input S;
  output [3:0] Y;

  reg [3:0] Y;

  always @ (A,B,S)
    begin
      if (S==0)   Y<=A;
      else        Y<=B;
    end
endmodule
```

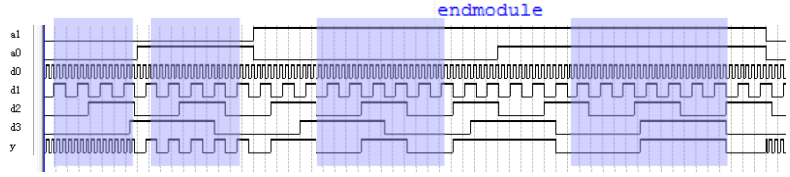


扩展：四选一




```
module mux41 (d0,d1,d2,d3,a0,a1,y);
  input d0,d1,d2,d3,a0,a1;
  output y;
  reg y;

  always @ (d0,d1,d2,d3,a0,a1)
    begin
      case ({a1,a0})
        2'b00: y<=d0;
        2'b01: y<=d1;
        2'b10: y<=d2;
        2'b11: y<=d3;
      endcase
    end
endmodule
```

☞拼接运算符{ }



3.硬件描述语言2

一、组合电路（以二选一为例）

1. 电路模块的基本结构
2. 电路模块的描述方式

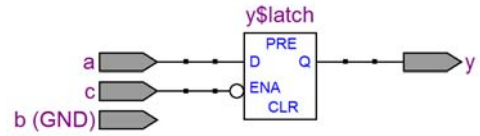
二、测试平台

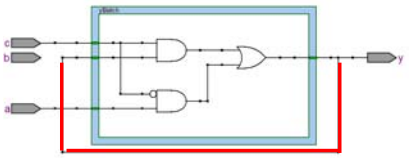
三、时序电路

1. 同步复位和异步复位的D触发器
2. 状态机

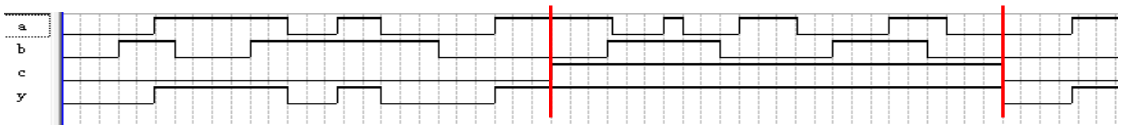
不写全所有分支未加default语句带来的问题：

```
module mux21 (a,b,c,y);
  input a,b,c;
  output y;

  reg y;

  always @ (a,b,c)
    begin
      if (c==0)  y<=a;
      //else       y<=b;
    end
endmodule
```





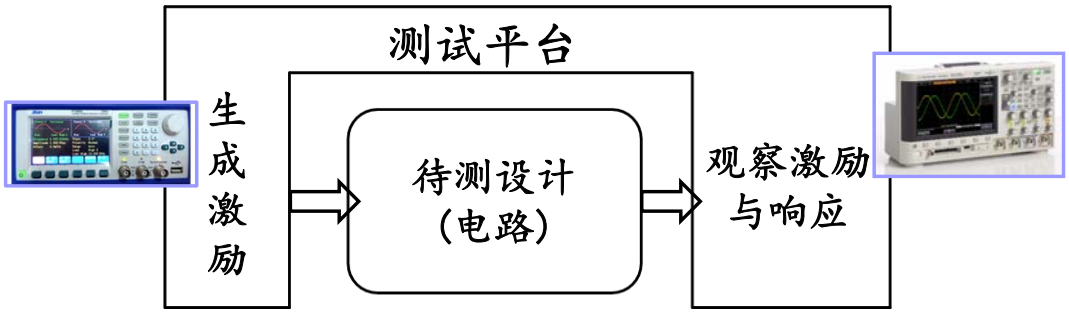
电平触发的D锁存器



使用不完整的条件语句，产生了时序电路

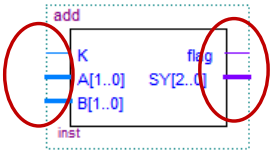
测试平台（Testbench）

- 为系统设计提供激励的HDL描述
- 测试和观察在此激励下的响应



电路文件vs.测试文件

电路文件add.bdf/add.v



测试文件add_tb.v

```
`timescale 10ns/1ns
module add_tb;
  wire [2:0]so;
  wire F;
  reg [1:0]a,b;
  reg k;
  add u1(.A(a),.B(b),.K(k),.SY(so),.flag(F));
  initial
    begin
      a=00;b=00;k=0;
      #20 a=00;b=01;k=0;
      #20 a=00;b=10;k=0;
      #20 a=00;b=11;k=0;
      #20 a=01;b=00;k=0;
      #20 a=01;b=01;k=0;
      #20 a=01;b=10;k=0;
        ......
      #20 a=10;b=11;k=1;
      #20 a=11;b=00;k=1;
      #20 a=11;b=01;k=1;
      #20 a=11;b=10;k=1;
      #20 a=11;b=11;k=1;
    end
   initial
    begin
    $monitor ("k=%d a=%d b=%d so=%d F=%d",k,a,b,so,F);
    end
  endmodule
```

测试文件的结构

```
`timescale 10ns/1ns
module add_tb;
  wire [2:0]so;
  wire F;
  reg [1:0]a,b;
  reg k;
  add u1(.A(a),.B(b),.K(k),.SY(so),.flag(F));
  initial
    begin
      a=00;b=00;k=0;
      #20 a=00;b=01;k=0;
      #20 a=00;b=10;k=0;
      #20 a=00;b=11;k=0;
      #20 a=01;b=00;k=0;
      #20 a=01;b=01;k=0;
      #20 a=01;b=10;k=0;
        ......
      #20 a=10;b=11;k=1;
      #20 a=11;b=00;k=1;
      #20 a=11;b=01;k=1;
      #20 a=11;b=10;k=1;
      #20 a=11;b=11;k=1;
    end
   initial
    begin
    $monitor ("k=%d a=%d b=%d so=%d F=%d",k,a,b,so,F);
    end
  endmodule
```

仿真时间单位/仿真精度

端口声明

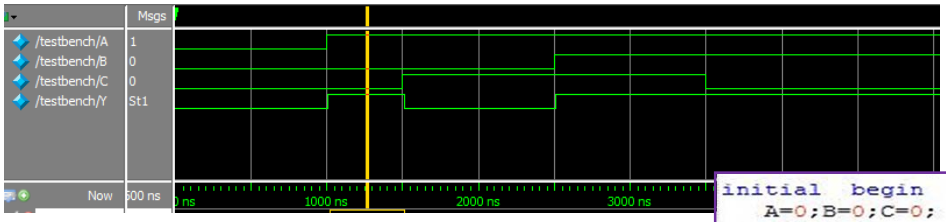
实例引用被测模块
（引入待测设计）

产生激励

**输出仿真数据
（观察激励与响应）

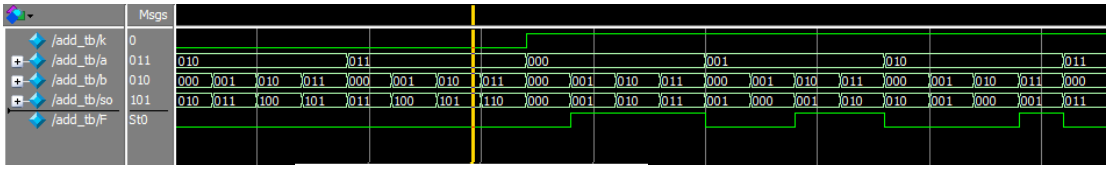
常用激励

- 由一组指定值组成的波形（1）




```
initial  begin
   A=0;B=0;C=0;
   #10 A=1;
   #5  C=1;
   #10 B=1;
   #10 C=0;
   #20 A=0;
   #10 B=1;
end
```

- 由一组指定值组成的波形（2）




```
initial
  begin
    a=0;
    b=0;
    k=0;
    for (k=0;k<=1;k=k+1)begin
     for (a=0;a<4;a=a+1)begin
      for (b=0;b<4;b=b+1)begin
      #20;
      end
     end
    end
  end
```

- 不断重复的波形，如时钟波形

时钟激励的几种写法

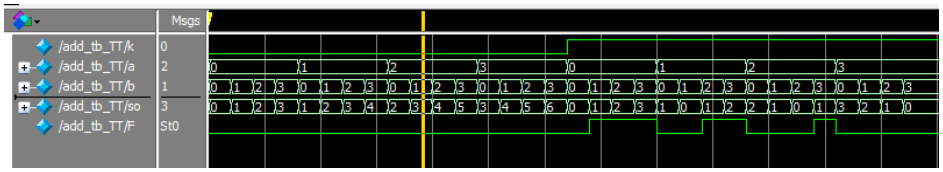
```
always #2 clk=~clk;
```
1

```
parameter Tburst=100,Ton=2,Toff=2;
  initial
    begin
      repeat (Tburst)
        begin
          #Toff clk=1'b1;
          #Ton  clk=1'b0;
        end
    end
```
3

```
parameter tON=2,tOFF=2;
always
  begin
    #tON clk=0;
    #tOFF clk=1;
  end
```
2

验证结果

- 波形



- 数据

```
# k=0 a=2 b=3 so=5 F=0
# k=0 a=3 b=0 so=3 F=0
# k=0 a=3 b=1 so=4 F=0
# k=0 a=3 b=2 so=5 F=0
# k=0 a=3 b=3 so=6 F=0
# k=1 a=0 b=0 so=0 F=0
# k=1 a=0 b=1 so=1 F=1
# k=1 a=0 b=2 so=2 F=1
# k=1 a=0 b=3 so=3 F=1
# k=1 a=1 b=0 so=1 F=0
# k=1 a=1 b=1 so=0 F=0
# k=1 a=1 b=2 so=1 F=1
# k=1 a=1 b=3 so=2 F=1
# k=1 a=2 b=0 so=2 F=0
# k=1 a=2 b=1 so=1 F=0
# k=1 a=2 b=2 so=0 F=0
```

Quartus II & Modelsim
联合仿真流程

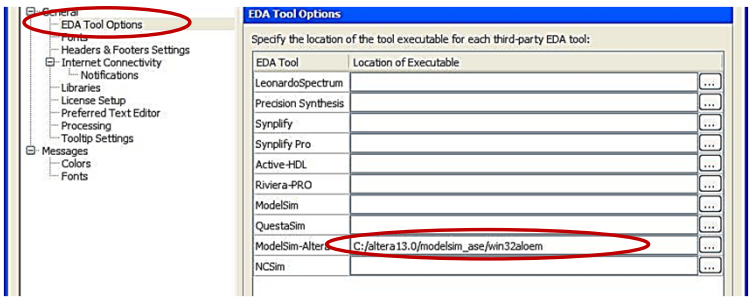
参见网络学堂“EDA视频V”

——熟悉流程、设置步骤

准备工作

- 检查第三方仿真工具Modelsim

  Tools→Options窗口中，EDA Tool Options 中修改路径如下：



C:\Altera\13.0\modelsim_ase\win32aloem

准备工作（续）

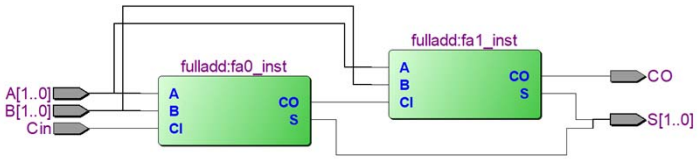
- 电路设计和测试文件

  如：*.v和testbench.v
- 也可以使用模板生成测试文件*.vt
  - 执行Processing→Start→Start TestBench Template Writer
  - 生成的模板文件存放路径：project文件夹中/simulation/modelsim/*.vt
  - 打开并修改*.vt文件

新建工程

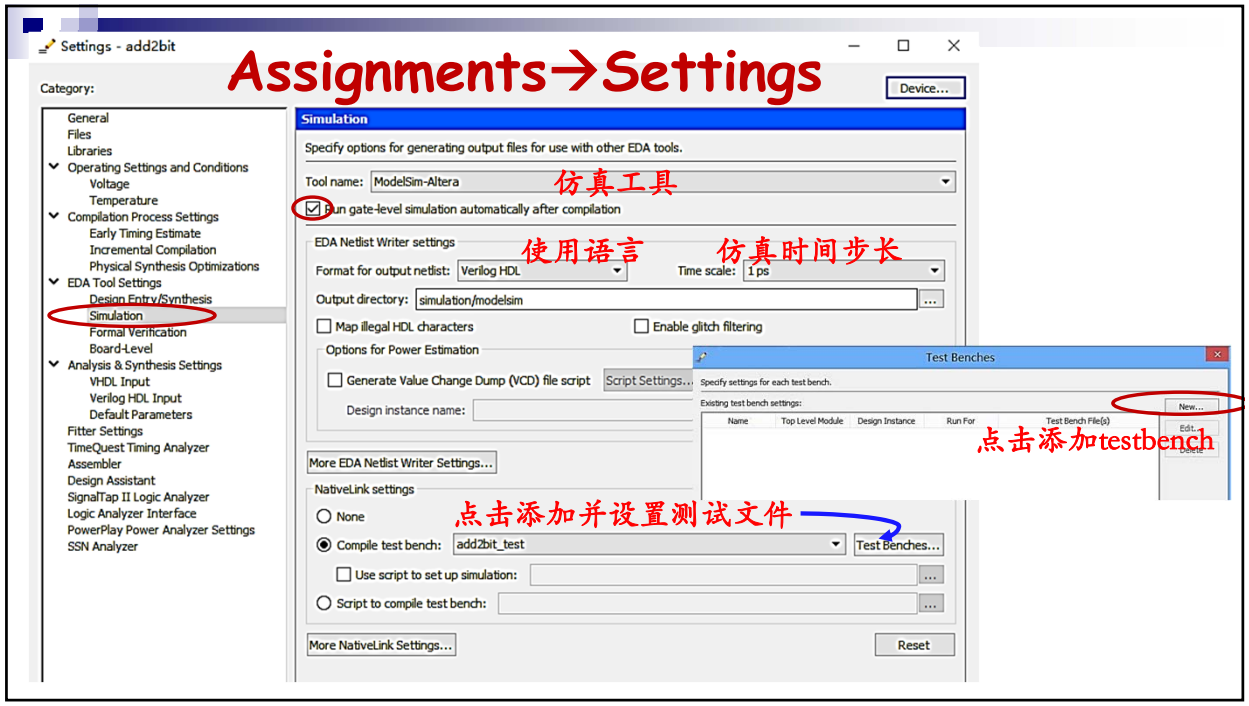
- 新建project，新建或加载电路文件和测试文件，并选中第三方仿真工具

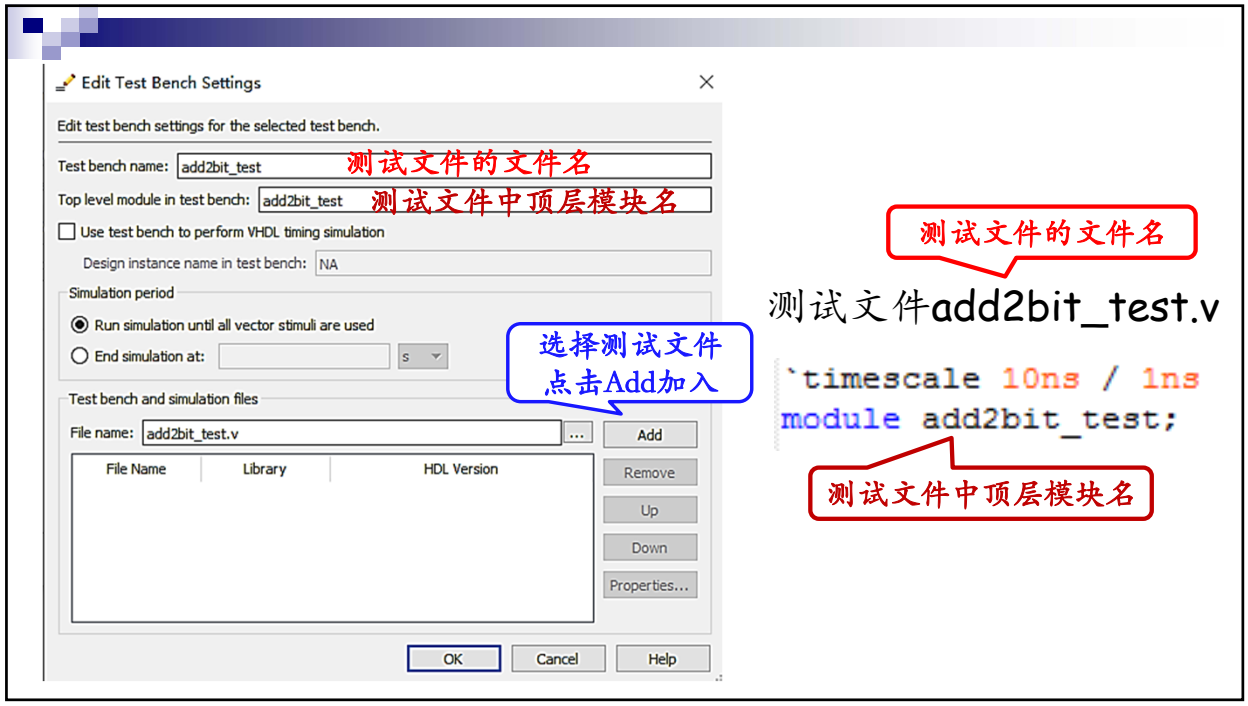
  Processing→Start→Start Analysis and Sythesis Tools →Netlist Viewers →RTL Viewer



关联仿真

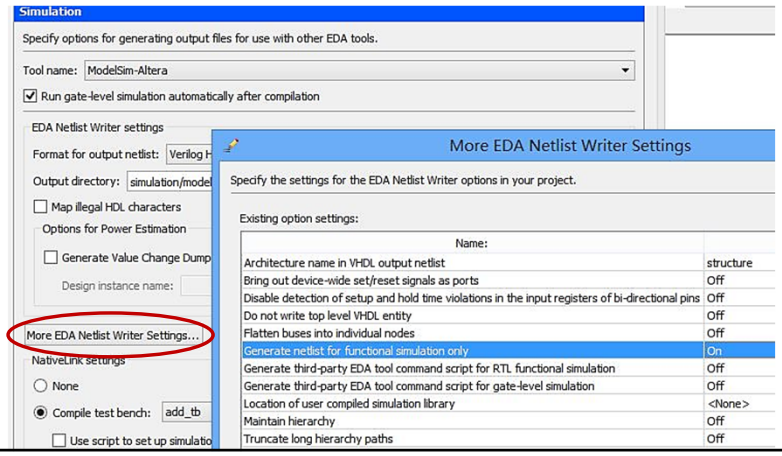
- 仿真设置，Assignments→Settings
  - 仿真工具
  - 语言及仿真步长
  - 设置并添加测试文件





执行功能仿真

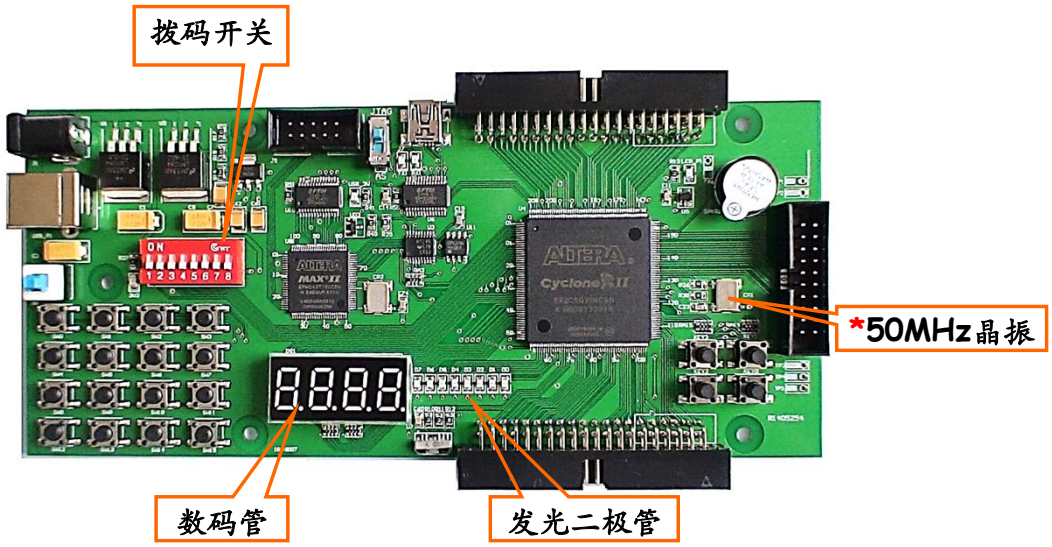
- Settings页中，点击More EDA Netlist Writer Settings…“生成功能仿真所需网表文件”设置为ON
- 设置完成后全编译并启动 modelsim



执行时序仿真

- Settings页中，点击More EDA Netlist Writer Settings…“生成功能仿真所需网表文件”设置为OFF
- 设置完成后全编译并启动modelsim

EDA作业二



实验任务:

- 在可编程逻辑器件上实现一个运算电路，以实现S=M+N。（M、N为3位有符号数）

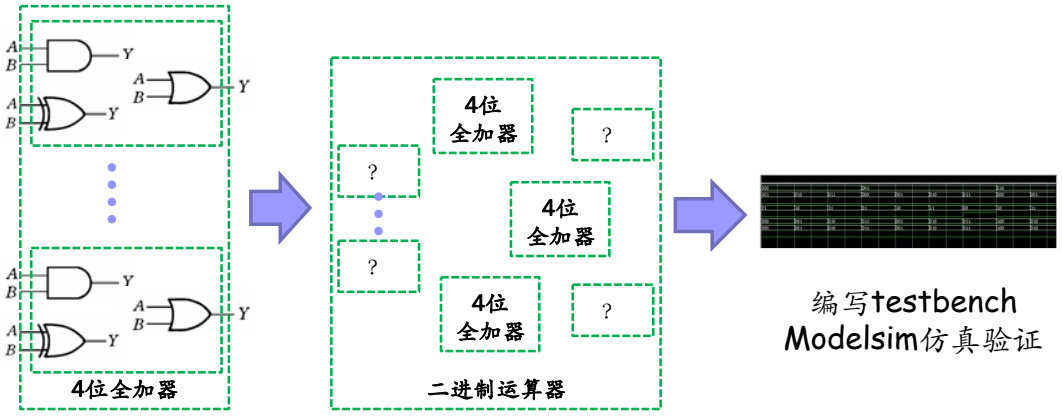
  运算数输入：拨码开关、按键

  运算结果显示：数码管、发光二极管

- 任务分解：
  - ✓二进制运算器
  - ✓4位数码管驱动电路

任务：二进制运算器（原理图）

自下而上的模块化、层次化的设计



任务：4位数码管的驱动电路

S=M+N

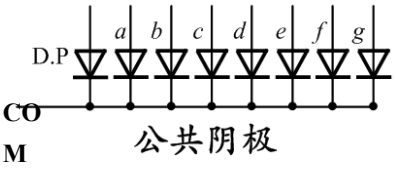
DIP1 DIP2	数码管3	数码管2	数码管1	数码管0
00	M	不亮	不亮	不亮
01	不亮	N	不亮	不亮
10	不亮	不亮	S(正负标志)	不亮
11	不亮	不亮	不亮	S(运算结果)

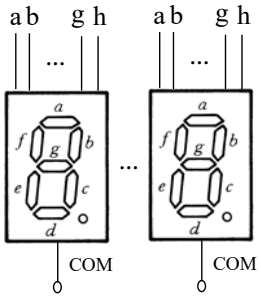
- 数码管静态显示方式

  共阴极数码管，高电平有效

  段选线、位选线由FPGA引脚直接驱动

  ☞显示位数较多时，占用I/O





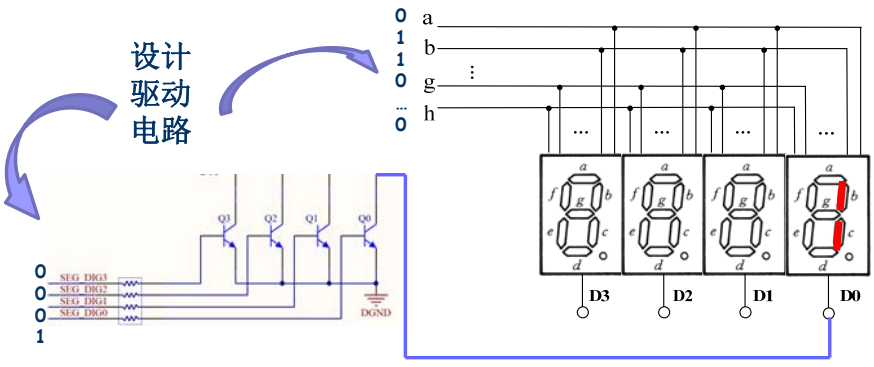
- 数码管动态显示方式

  段选线并联在一起，共阴极为选通端

  驱动电路端轮流加以高电平，点亮相应的数码管

  采用动态扫描的方式，使多位数码管“同时”被点亮

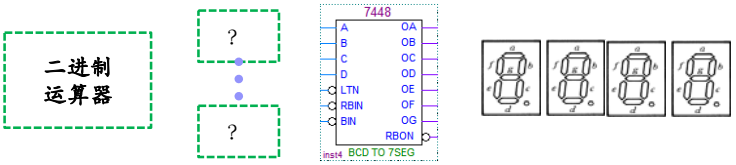
选做



任务：4位数码管的驱动电路

S=M+N

DIP1 DIP2	数码管3	数码管2	数码管1	数码管0
00	M	不亮	不亮	不亮
01	不亮	N	不亮	不亮
10	不亮	不亮	S(正负标志)	不亮
11	不亮	不亮	不亮	S(运算结果)



预告：

- 预先设计，课上以答疑、调试、验收为主。
- 验收步骤：
  - 第9周：二进制运算器及其Modelsim仿真
  - 第11周：数码管显示电路、整体下载
- 提供资源：
  - 第9周讲解验收细则
  - 提供教学视频（课程文件--EDA视频）
  - 提供文档资料（课程文件-- 实验文档）