第六章 数字逻辑实验设计

6.1 实验一 基本门电路与数据扩展描述

1.实验介绍

在本次实验中，我们将练习使用 Verilog HDL 语言，并采用三种不同的描述方式设计基本门电路，实现数据扩展。

2.实验目标

- 学习用不同的描述方式设计基本门电路并实现数据扩展。
- 学习设计仿真工具的使用方法。
- 学习如使用开发板。

3.实验原理

1) 基本门电路实验

（1） 基本与或非门实验

图 6.1.1 给出了所要建模的基本与或非门实验的电路原理图。

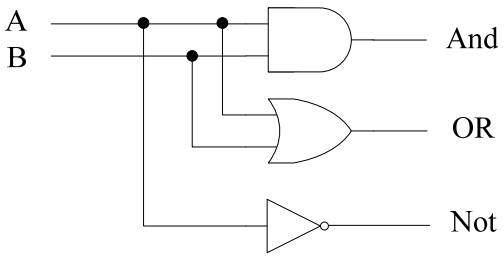


图 6.1.1 与或非门实验的电路原理图

● 接口定义：

```
module logic_gates_1(iA,iB,oAnd,oOr,oNot);//结构描述
     input iA,iB;           //输入信号 A、B
     output oAnd,oOr,oNot; //与、或、非输出信号
```

```
module logic_gates_2(iA,iB,oAnd,oOr,oNot); //数据流描述
     input iA,iB;           //输入信号 A、B
     output oAnd,oOr,oNot; //与、或、非输出信号
```

```
module logic_gates_3(iA,iB,oAnd,oOr,oNot);//行为描述
     input iA,iB;           //输入信号 A、B
     output oAnd,oOr,oNot; //与、或、非输出信号
```

- Verilog 代码描述：

a） 结构型描述

```
module logic_gates_1(iA,iB,oAnd,oOr,oNot);
   input iA, iB;
   output oAnd,oOr,oNot;
   and and_inst(oAnd, iA,iB);
   or  or_inst(oOr, iA,iB);
   not not_inst(oNot, iA);
endmodule
```

b） 数据流型描述

```
module logic_gates_2(iA,iB,oAnd,oOr,oNot);
   input iA, iB;
   output oAnd,oOr,oNot;
   assign oAnd = iA & iB;
   assign oOr = iA | iB;
   assign oNot = ~iA;
endmodule
```

c） 行为描述

```
module logic_gates_3(iA,iB,oAnd,oOr,oNot);
   input iA, iB;
   output oAnd,oOr,oNot;
   reg oAnd, oOr, ONot;
   always @ (*)
   begin
      oAnd = iA & iB;
      oOr = iA | iB;
      oNot = ~ iA;
   end
endmodule
```

- TestBench 代码描述：

```
`timescale 1ns/1ns
module logic_gates_tb;
  reg iA;
  reg iB;
  wire oAnd;
  wire oOr;
  wire oNot;
```

```
   initial
   begin
   iB = 0;
   #40 iB = 0;
   #40 iB = 1;
   #40 iB = 1;
   #40 iB = 0;
   end
```

```
initial
begin
  iA = 0;
  #40 iA = 1;
  #40 iA = 0;
  #40 iA = 1;
  #40 iA = 0;
end
```

```
logic_gates_1
logic_gates_inst(
   .iA(iA),
   .iB(iB),
   .oAnd(oAnd),
   .oOr(oOr),
   .oNot(oNot)
   );
endmodule
```

● Modelsim 仿真

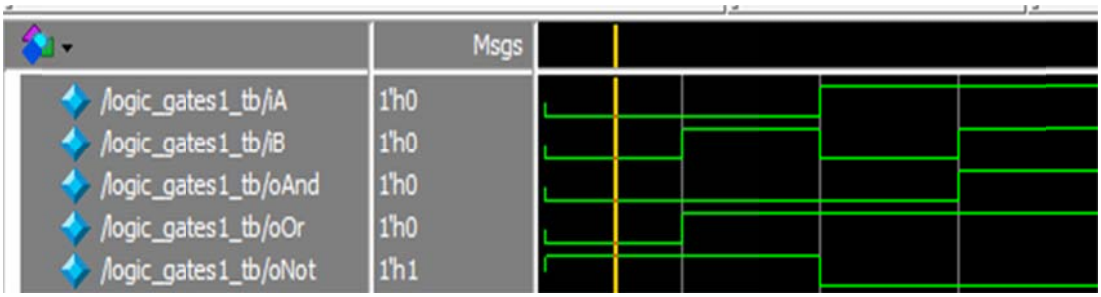


图 6.1.2 仿真结果

● XDC 文件配置

变量	iA	iB	oAnd	oOr	oNot
N4 板上的管脚	SW0（J15）	SW1（L16）	LD0（H17）	LD1（K15）	LD2（J13）

● 综合下板

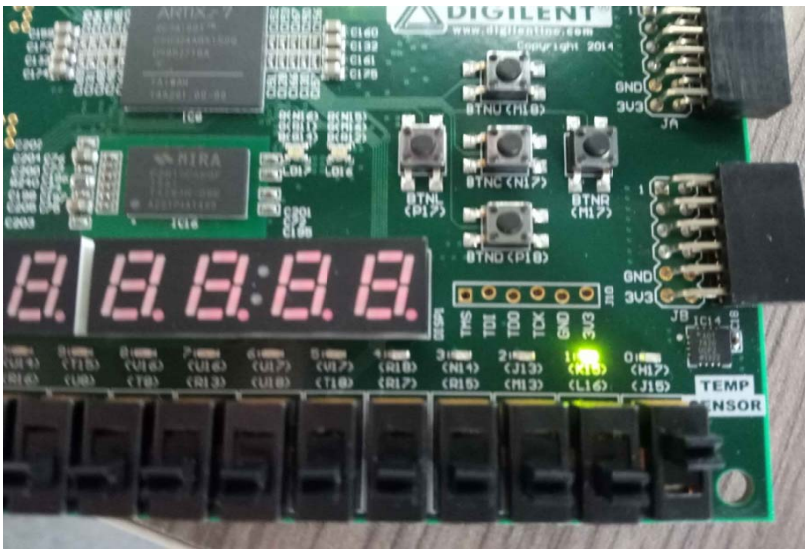
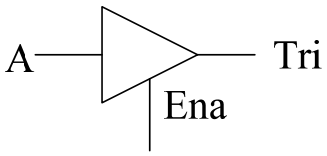


图 6.1.3 综合下板结果

（2） 三态门实验

图 6.1.4 给出了所要建模的三态门的逻辑符号和真值表。



A	Ena	Tri
0	0	z
1	0	z
0	1	0
1	1	1

图 6.1.4 三态门的逻辑符号和真值表

- 接口定义：

```
module three_state_gates(iA,iEna,oTri);
        input iA;          //输入信号 A
        input iEna;        //使能信号 Ena,高电平有效
        output oTri;       //三态输出信号 Tri
```

- Verilog 代码描述：

```
module three_state_gates(iA,iEna,oTri);
   input iA;
   input iEna;
   output oTri;
   assign oTri = (iEna==1)? iA:'bz;
endmodule
```

- TestBench 代码描述：

```
`timescale 1ns/1ns
Module three_state_gates_tb;
  reg iA;
  reg iEna;
  wire oTriState;

    three_state_gates uut (
.iA(iA),
.iEna(iEna),
.oTri(oTriState)
);
    initial
    begin
    iA = 0;
       #40 iA = 1;
    #40 iA = 0;
    #40 iA = 1;
     end
     initial
     begin
    iEna = 1;
    #20 iEna = 0;
    #40 iEna= 1;
    #20 iEna = 0;
     end
endmodule
```

● Modelsim 仿真

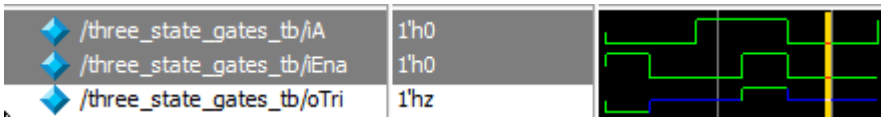


图 6.1.5 仿真结果

● XDC 文件配置

变量	iA	iEna	oTri
N4 板上的管脚	SW0 （J15）	BTNR （M17）	LD0 （H17）

● 综合下板

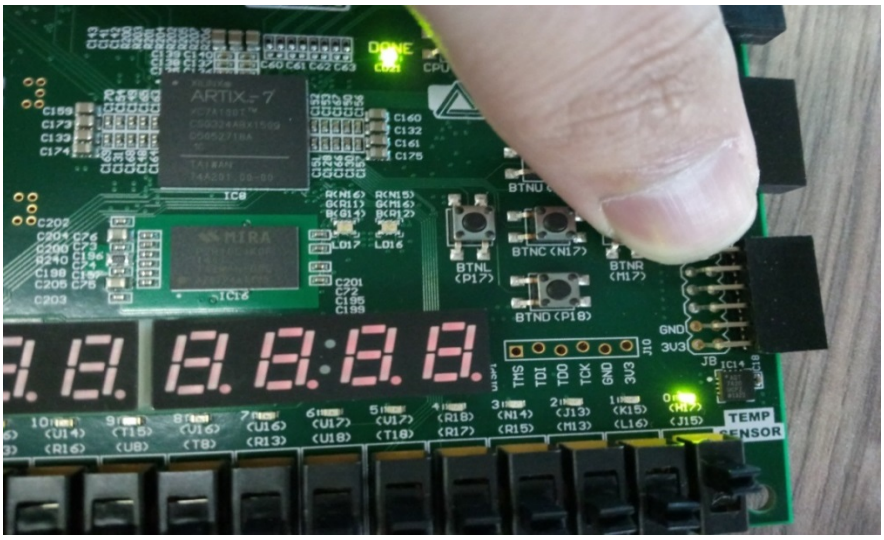


图 6.1.6 综合下板结果

2) 数据扩展实验

本实验所建模的扩展模块的功能为将输入的数据扩展为 32 位数据。

● 接口定义：

```
module extend #(parameter WIDTH = 16)(
    input [WIDTH - 1:0] a, //输入数据，数据宽度可以根据需要自行定义
    input sext,            //sext 有效为符号扩展，否则 0 扩展
    output [31:0]b        //32 位输出数据
);
```

● Verilog 代码描述：

```
module extend #(parameter WIDTH = 16)(
    input [WIDTH-1:0] a,
    input sext, //sext 有效是为符号扩展，否则为 0 扩展
    output [31:0] b
);
```

```
   assign b=sext? {{(32-WIDTH){a[WIDTH-1]}},a} : {{(32-
                    WIDTH){1'b0}},a};
endmodule
```

● TestBench 代码描述：

```
`timescale 1ns/1ns
module extend_tb;
    reg [15:0] a;
    reg sext;
    wire [31:0] b;
// Instantiate the Unit Under Test (UUT)
    extend uut (.a(a),.sext(sext),.b(b));
    initial
      begin
       // Initialize Inputs
       a = 0;
       sext = 0;
       // Wait 100 ns for global reset to finish
       #100;
       // Add stimulus here
       sext = 1;
       a = 16'h0000;
       #100;
       sext = 0;
       a = 16'h8000;
       #100;
       sext = 1;
       a = 16'h8000;
       #100;
       sext = 0;
       a = 16'hffff;
       #100;
       sext = 1;
       a = 16'hffff;
       #100;
     end
endmodule
```

● Modelsim 仿真

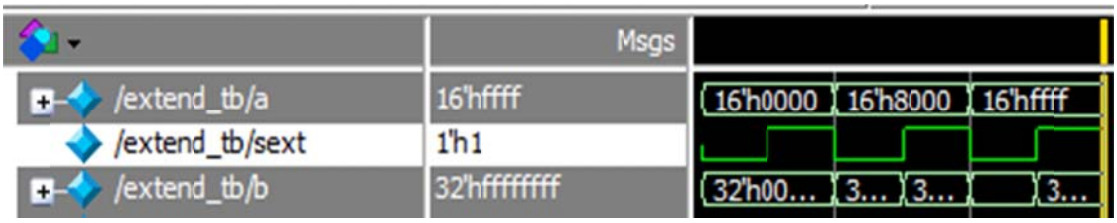


图 6.1.7 仿真结果

4.实验步骤：

1. 用 logisim 画出基本与或非门实验电路原理图，验证逻辑。
2. 新建 Vivado 工程，编写各个模块。
3. 编写各个模块的 test bench，并调用 ModelSim 仿真测试各模块。
4. 配置 XDC 文件，综合下板，并观察实验现象。
5. 按照要求书写实验报告。