

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。

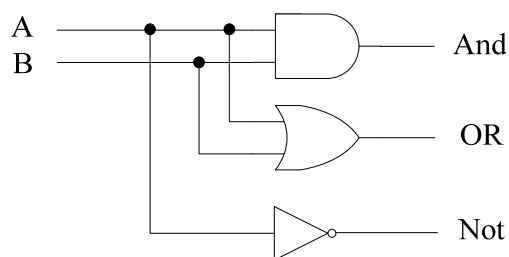


图 6.1.1 与或非门实验的电路原理图

● 接口定义:

```
module logic_gates(iA,iB,oAnd,oOr,oNot);
    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
```

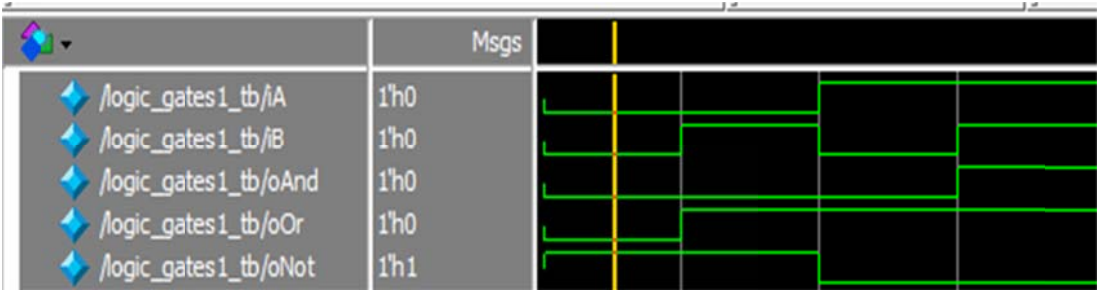
c) 行为描述

```
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:

<pre>`timescale 1ns/1ns module logic_gates_tb; reg iA; reg iB; wire oAnd; wire oOr; wire oNot; initial begin iA = 0; #40 iA = 1; #40 iA = 0; #40 iA = 1; #40 iA = 0; end end</pre>	<pre>initial begin iB = 0; #40 iB = 0; #40 iB = 1; #40 iB = 1; #40 iB = 0; end logic_gates_1 logic_gates_inst(.iA(iA), .iB(iB), .oAnd(oAnd), .oOr(oOr), .oNot(oNot)); endmodule</pre>
---	---

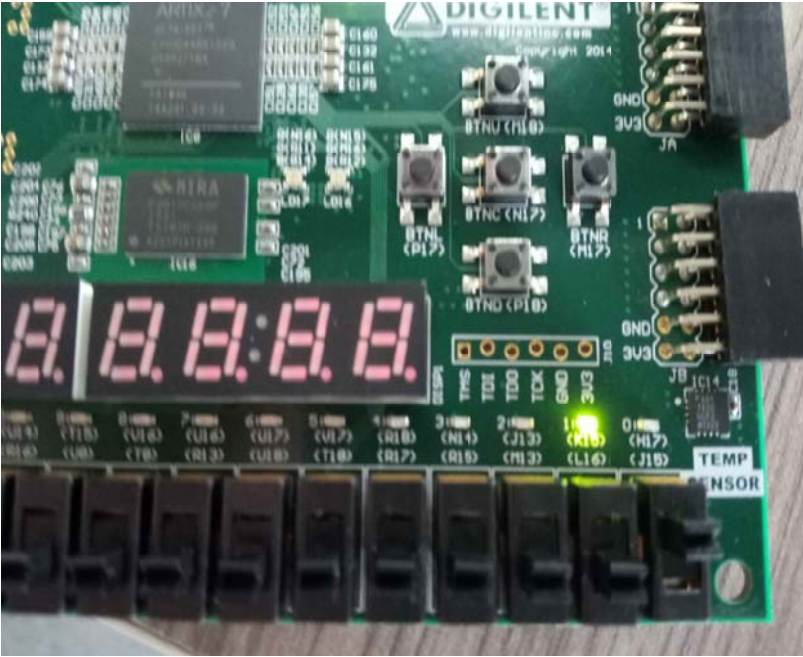
● Modelsim 仿真



● 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



(2) 三态门实验

图 6.1.2 给出了所要建模的三态门的逻辑符号和真值表。

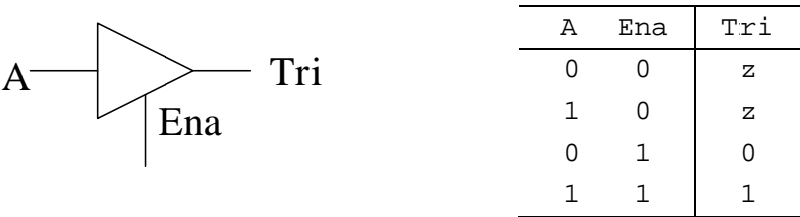


图 6.1.2 三态门的逻辑符号和真值表

● 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
endmodule
```

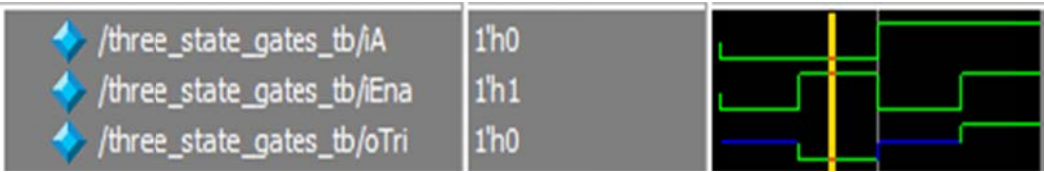
● Verilog 代码描述:

```
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:

<pre>`timescale 1ns/1ns Module three_state_gates_tb; reg iA; reg iEna; wire oTriState; three_state_gates uut (.iA(iA), .iEna(iEna), .oTriState(oTriState)); initial begin</pre>	<pre> 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</pre>
--	--

● Modelsim 仿真



● XDC 文件配置

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

The image shows a custom PCB assembly with the following components and labels:

- Top Left:** ARTIX-7 FPGA (K7A1000), C200, C201, C202, C203, C204, C205, C206, C207, C208, C209, C210, C211, C212, C213, C214, C215, C216, C217, C218, C219, C220, C221, C222, C223, C224, C225, C226, C227, C228, C229, C230, C231, C232, C233, C234, C235, C236, C237, C238, C239, C240, C241, C242, C243, C244, C245, C246, C247, C248, C249, C250, C251, C252, C253, C254, C255, C256, C257, C258, C259, C260, C261, C262, C263, C264, C265, C266, C267, C268, C269, C270, C271, C272, C273, C274, C275, C276, C277, C278, C279, C280, C281, C282, C283, C284, C285, C286, C287, C288, C289, C290, C291, C292, C293, C294, C295, C296, C297, C298, C299, C300, C301, C302, C303, C304, C305, C306, C307, C308, C309, C310, C311, C312, C313, C314, C315, C316, C317, C318, C319, C320, C321, C322, C323, C324, C325, C326, C327, C328, C329, C330, C331, C332, C333, C334, C335, C336, C337, C338, C339, C340, C341, C342, C343, C344, C345, C346, C347, C348, C349, C350, C351, C352, C353, C354, C355, C356, C357, C358, C359, C360, C361, C362, C363, C364, C365, C366, C367, C368, C369, C370, C371, C372, C373, C374, C375, C376, C377, C378, C379, C380, C381, C382, C383, C384, C385, C386, C387, C388, C389, C390, C391, C392, C393, C394, C395, C396, C397, C398, C399, C400, C401, C402, C403, C404, C405, C406, C407, C408, C409, C410, C411, C412, C413, C414, C415, C416, C417, C418, C419, C420, C421, C422, C423, C424, C425, C426, C427, C428, C429, C430, C431, C432, C433, C434, C435, C436, C437, C438, C439, C440, C441, C442, C443, C444, C445, C446, C447, C448, C449, C450, C451, C452, C453, C454, C455, C456, C457, C458, C459, C460, C461, C462, C463, C464, C465, C466, C467, C468, C469, C470, C471, C472, C473, C474, C475, C476, C477, C478, C479, C480, C481, C482, C483, C484, C485, C486, C487, C488, C489, C490, C491, C492, C493, C494, C495, C496, C497, C498, C499, C500, C501, C502, C503, C504, C505, C506, C507, C508, C509, C510, C511, C512, C513, C514, C515, C516, C517, C518, C519, C520, C521, C522, C523, C524, C525, C526, C527, C528, C529, C530, C531, C532, C533, C534, C535, C536, C537, C538, C539, C540, C541, C542, C543, C544, C545, C546, C547, C548, C549, C550, C551, C552, C553, C554, C555, C556, C557, C558, C559, C560, C561, C562, C563, C564, C565, C566, C567, C568, C569, C570, C571, C572, C573, C574, C575, C576, C577, C578, C579, C580, C581, C582, C583, C584, C585, C586, C587, C588, C589, C590, C591, C592, C593, C594, C595, C596, C597, C598, C599, C600, C601, C602, C603, C604, C605, C606, C607, C608, C609, C610, C611, C612, C613, C614, C615, C616, C617, C618, C619, C620, C621, C622, C623, C624, C625, C626, C627, C628, C629, C630, C631, C632, C633, C634, C635, C636, C637, C638, C639, C640, C641, C642, C643, C644, C645, C646, C647, C648, C649, C650, C651, C652, C653, C654, C655, C656, C657, C658, C659, C660, C661, C662, C663, C664, C665, C666, C667, C668, C669, C670, C671, C672, C673, C674, C675, C676, C677, C678, C679, C680, C681, C682, C683, C684, C685, C686, C687, C688, C689, C690, C691, C692, C693, C694, C695, C696, C697, C698, C699, C700, C701, C702, C703, C704, C705, C706, C707, C708, C709, C710, C711, C712, C713, C714, C715, C716, C717, C718, C719, C720, C721, C722, C723, C724, C725, C726, C727, C728, C729, C730, C731, C732, C733, C734, C735, C736, C737, C738, C739, C740, C741, C742, C743, C744, C745, C746, C747, C748, C749, C750, C751, C752, C753, C754, C755, C756, C757, C758, C759, C760, C761, C762, C763, C764, C765, C766, C767, C768, C769, C770, C771, C772, C773, C774, C775, C776, C777, C778, C779, C780, C781, C782, C783, C784, C785, C786, C787, C788, C789, C790, C791, C792, C793, C794, C795, C796, C797, C798, C799, C800, C801, C802, C803, C804, C805, C806, C807, C808, C809, C810, C811, C812, C813, C814, C815, C816, C817, C818, C819, C820, C821, C822, C823, C824, C825, C826, C827, C828, C829, C830, C831, C832, C833, C834, C835, C836, C837, C838, C839, C840, C841, C842, C843, C844, C845, C846, C847, C848, C849, C850, C851, C852, C853, C854, C855, C856, C857, C858, C859, C860, C861, C862, C863, C864, C865, C866, C867, C868, C869, C870, C871, C872, C873, C874, C875, C876, C877, C878, C879, C880, C881, C882, C883, C884, C885, C886, C887, C888, C889, C890, C891, C892, C893, C894, C895, C896, C897, C898, C899, C900, C901, C902, C903, C904, C905, C906, C907, C908, C909, C910, C911, C912, C913, C914, C915, C916, C917, C918, C919, C920, C921, C922, C923, C924, C925, C926, C927, C928, C929, C930, C931, C932, C933, C934, C935, C936, C937, C938, C939, C940, C941, C942, C943, C944, C945, C946, C947, C948, C949, C950, C951, C952, C953, C954, C955, C956, C957, C958, C959, C960, C961, C962, C963, C964, C965, C966, C967, C968, C969, C970, C971, C972, C973, C974, C975, C976, C977, C978, C979, C980, C981, C982, C983, C984, C985, C986, C987, C988, C989, C990, C991, C992, C993, C994, C9

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} : {27'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

	Msgs	
+ /extend_tb/a	16'hffff	16'h0000 16'h8000 16'hffff
/extend_tb/sext	1'h1	
+ /extend_tb/b	32'hffffffff	32'h00... 3... 3... 3...

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。