CHAPTER 3

# 第3章 组合逻辑电路设计与应用

本章重点介绍以下组合逻辑电路的设计：基本逻辑门、比较器、数据选择器、编码器、译码器、ALU 等电路；同时，结合上述项目介绍 ISE 开发环境中常用的一些工具，包括 ISim、FPGA Editor、PlanAhead、Design Summary 等。

学习组合逻辑电路的设计与应用，主要有 4 个目标：①通过实践，掌握 ISE 和 Vivado 工具软件的使用方法；②通过实践，进一步掌握 HDL 语言结构；③通过实践，掌握组合逻辑电路的设计方法；④通过实践，掌握开发板在组合逻辑电路设计中的应用方法。

**实战项目 5　设计基本门电路**

**【项目描述】** 设计完成 2 输入与门、与非门、或门、或非门和异或门和同或门。

要求：

（1）将拨码开关 SW0 和 SW1 分别作为输入变量 a 和 b；分别实现与门、与非门、或门、或非门、异或门、同或门；输出 y 接到 6 个灯 LD0～LD5，通过拨动拨码开关来观察输出的状态变化。

（2）使用 LSim 对设计完成的门电路进行功能仿真。

实战项目 5.mp4
(3.50MB)

**【知识点】**

（1）组合逻辑电路设计的一般方法。

（2）门电路的实现方法。

（3）使用 ISim 进行功能仿真的步骤和方法。

（4）开发板在组合逻辑电路设计中的应用方法。

## 3.1　基本门电路

### 3.1.1　基本门电路设计

设计组合逻辑，使用数据流描述来实现，如例 3-1 所示。

【例 3-1】 2 输入逻辑门实现代码。

```
module gate2(a,b,y);
    input a;
    input b;
    output[5:0] y;
    assign y[0] = a&b;                  //and
    assign y[1] = ~(a&b);               //nand
    assign y[2] = a|b;                  //or
    assign y[3] = ~(a|b);               //nor
    assign y[4] = a^b;                  //xor
    assign y[5] = ~(a^b);               //xnor
endmodule
```

为了引脚锁定的统一和方便,给例 3-1 增加顶层模块。

【例 3-2】 对例 3-1 增加顶层模块。

```
module P5_Gate_top(SW,LED);
    input[1:0] SW;
    output[5:0] LED;
    gate2 U1(.a(SW[0]),
             .b(SW[1]),
             .y(LED));
endmodule
```

对例 3-2 增加引脚锁定,约束文件如例 3-3 所示。

【例 3-3】 对例 3-2 设计的引脚约束文件。

```
NET "SW[0]" LOC = P11;              //SW0
NET "SW[1]" LOC = L3;               //SW1
NET "LED[0]" LOC = M5;              //LD0
NET "LED[1]" LOC = M11;             //LD1
NET "LED[2]" LOC = P7;              //LD2
NET "LED[3]" LOC = P6;              //LD3
NET "LED[4]" LOC = N5;              //LD4
NET "LED[5]" LOC = N4;              //LD5
```

最后对已经约束引脚的设计进行综合、实现、生成配置文件、编程到 Basys2 开发板。在 Basys2 开发板上,拨动 SW0 和 SW1 这两个拨码开关,可以看到 LD0～LD5 这 6 个 LED 灯相应地变化,并指示当前的输出状态。

### 3.1.2 约束文件

在使用开发板时,需要为输出/输入信号指定引脚,因此将 Basys2 开发板的所有输入和输出引脚都整理在 basys2.ucf 文件中,作为 ISE 工程的约束文件,如例 3-4 所示。

【例 3-4】 basys2.ucf 文件。

```
#pin assignment for clock
NET "clk" LOC = B8;                          //MCLK
#pin assignment for slide switches
NET "SW[0]" LOC = P11;                       //SW0
NET "SW[1]" LOC = L3;                        //SW1
NET "SW[2]" LOC = K3;                        //SW2
NET "SW[3]" LOC = B4;                        //SW3
NET "SW[4]" LOC = G3;                        //SW4
NET "SW[5]" LOC = F3;                        //SW5
NET "SW[6]" LOC = E2;                        //SW6
NET "SW[7]" LOC = N3;                        //SW7
#pin assignment for LEDs
NET "LED[0]" LOC = M5;                       //LD0
NET "LED[1]" LOC = M11;                      //LD1
NET "LED[2]" LOC = P7;                       //LD2
NET "LED[3]" LOC = P6;                       //LD3
NET "LED[4]" LOC = N5;                       //LD4
NET "LED[5]" LOC = N4;                       //LD5
NET "LED[6]" LOC = P4;                       //LD6
NET "LED[7]" LOC = G1;                       //LD7
#pin assignment for 7 - segment displays
NET "SEG[0]" LOC = L14;                      //CA
NET "SEG[1]" LOC = H12;                      //CB
NET "SEG[2]" LOC = N14;                      //CC
NET "SEG[3]" LOC = N11;                      //CD
NET "SEG[4]" LOC = P12;                      //CE
NET "SEG[5]" LOC = L13;                      //CF
NET "SEG[6]" LOC = M12;                      //CG
NET "SEG[7]" LOC = N13;                      //DP
NET "AN[0]" LOC = F12;                       //AN0
NET "AN[1]" LOC = J12;                       //AN1
NET "AN[2]" LOC = M13;                       //AN2
NET "AN[3]" LOC = K14;                       //AN3
#pin assignment for pushbotton switches
NET "BTN[0]" LOC = G12;                      //BTN0
NET "BTN[1]" LOC = C11;                      //BTN1
NET "BTN[2]" LOC = M4;                       //BTN2
NET "BTN[3]" LOC = A7;                       //BTN3
#pin PS2 interface
NET "PS2C" LOC = B1;                         //PS2C
NET "PS2D" LOC = C3;                         //PS2D
#pin VGA interface
NET "R[0]" LOC = C14;                        //red0
NET "R[1]" LOC = D13;                        //red1
NET "R[2]" LOC = F13;                        //red2
NET "G[0]" LOC = G14;                        //green0
```

```
NET "G[1]" LOC = G13;                    //green1
NET "G[2]" LOC = F14;                    //green2
NET "B[0]" LOC = J13;                    //blue0
NET "B[1]" LOC = H13;                    //blue1
NET "HS" LOC = J14;                      //hs
NET "VS" LOC = K13;                      //vs
```

在例 3-4 的 basys2.ucf 文件中,在每个引脚指定行的最后都有一个注释。该注释对应该引脚在开发板上的资源名称。basys2.ucf 文件涵盖了开发板上常用的可用资源,但在实际应用中,可能只会用到其中的一部分资源。用到的资源指定相应的引脚,未用到的资源不需要指定引脚。例如,例 3-2 仅用到 2 个拨码开关和 6 个 LED 灯,所以进行引脚锁定时,仅锁定这些资源,如例 3-3 所示。

对于例 3-1,可以不用例 3-2,直接对例 3-1 指定引脚,然后进行综合、实现、生成配置文件、编程到 Basys2 开发板。使用例 3-2 的好处在于:一是规范设计中与 FPGA 的引脚连接的信号的名称;二是可以方便地使用例 3-4 所示的约束文件。

同样地,可以将 Basys3 开发板的所有输入和输出引脚都整理在 basys3.xdc 文件中,作为 Vivado 工程的约束文件,如例 3-5 所示。

**【例 3-5】** basys3.xdc 文件。

```
#pin assignment for clock
set_property PACKAGE_PIN W5 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
#pin assignment for slide switches
set_property PACKAGE_PIN V17 [get_ports sw[0]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[0]]
set_property PACKAGE_PIN V16 [get_ports sw[1]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[1]]
set_property PACKAGE_PIN W16 [get_ports sw[2]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[2]]
set_property PACKAGE_PIN W17 [get_ports sw[3]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[3]]
set_property PACKAGE_PIN W15 [get_ports sw[4]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[4]]
set_property PACKAGE_PIN V15 [get_ports sw[5]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[5]]
set_property PACKAGE_PIN W14 [get_ports sw[6]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[6]]
set_property PACKAGE_PIN W13 [get_ports sw[7]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[7]]
set_property PACKAGE_PIN V2 [get_ports sw[8]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[8]]
set_property PACKAGE_PIN T3 [get_ports sw[9]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[9]]
set_property PACKAGE_PIN T2 [get_ports sw[10]]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports sw[10]]
set_property PACKAGE_PIN R3 [get_ports sw[11]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[11]]
set_property PACKAGE_PIN W2 [get_ports sw[12]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[12]]
set_property PACKAGE_PIN U1 [get_ports sw[13]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[13]]
set_property PACKAGE_PIN T1 [get_ports sw[14]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[14]]
set_property PACKAGE_PIN R2 [get_ports sw[15]]
set_property IOSTANDARD LVCMOS33 [get_ports sw[15]]
#pin assignment for leds
set_property PACKAGE_PIN U16 [get_ports led[0]]
set_property IOSTANDARD LVCMOS33 [get_ports led[0]]
set_property PACKAGE_PIN E19 [get_ports led[1]]
set_property IOSTANDARD LVCMOS33 [get_ports led[1]]
set_property PACKAGE_PIN U19 [get_ports led[2]]
set_property IOSTANDARD LVCMOS33 [get_ports led[2]]
set_property PACKAGE_PIN V19 [get_ports led[3]]
set_property IOSTANDARD LVCMOS33 [get_ports led[3]]
set_property PACKAGE_PIN W18 [get_ports led[4]]
set_property IOSTANDARD LVCMOS33 [get_ports led[4]]
set_property PACKAGE_PIN U15 [get_ports led[5]]
set_property IOSTANDARD LVCMOS33 [get_ports led[5]]
set_property PACKAGE_PIN U14 [get_ports led[6]]
set_property IOSTANDARD LVCMOS33 [get_ports led[6]]
set_property PACKAGE_PIN V14 [get_ports led[7]]
set_property IOSTANDARD LVCMOS33 [get_ports led[7]]
set_property PACKAGE_PIN V13 [get_ports led[8]]
set_property IOSTANDARD LVCMOS33 [get_ports led[8]]
set_property PACKAGE_PIN V3 [get_ports led[9]]
set_property IOSTANDARD LVCMOS33 [get_ports led[9]]
set_property PACKAGE_PIN W3 [get_ports led[10]]
set_property IOSTANDARD LVCMOS33 [get_ports led[10]]
set_property PACKAGE_PIN U3 [get_ports led[11]]
set_property IOSTANDARD LVCMOS33 [get_ports led[11]]
set_property PACKAGE_PIN P3 [get_ports led[12]]
set_property IOSTANDARD LVCMOS33 [get_ports led[12]]
set_property PACKAGE_PIN N3 [get_ports led[13]]
set_property IOSTANDARD LVCMOS33 [get_ports led[13]]
set_property PACKAGE_PIN P1 [get_ports led[14]]
set_property IOSTANDARD LVCMOS33 [get_ports led[14]]
set_property PACKAGE_PIN L1 [get_ports led[15]]
set_property IOSTANDARD LVCMOS33 [get_ports led[15]]
#pin assignment for 7-segment displays
set_property PACKAGE_PIN U2 [get_ports an[0]]
set_property IOSTANDARD LVCMOS33 [get_ports an[0]]
set_property PACKAGE_PIN U4 [get_ports an[1]]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports an[1]]
set_property PACKAGE_PIN V4 [get_ports an[2]]
set_property IOSTANDARD LVCMOS33 [get_ports an[2]]
set_property PACKAGE_PIN W4 [get_ports an[3]]
set_property IOSTANDARD LVCMOS33 [get_ports an[3]]
set_property PACKAGE_PIN W7 [get_ports seg[0]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[0]]
set_property PACKAGE_PIN W6 [get_ports seg[1]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[1]]
set_property PACKAGE_PIN U8 [get_ports seg[2]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[2]]
set_property PACKAGE_PIN V8 [get_ports seg[3]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[3]]
set_property PACKAGE_PIN U5 [get_ports seg[4]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[4]]
set_property PACKAGE_PIN V5 [get_ports seg[5]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[5]]
set_property PACKAGE_PIN W3 [get_ports seg[6]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[6]]
set_property PACKAGE_PIN U7 [get_ports seg[7]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[7]]
set_property PACKAGE_PIN V7 [get_ports seg[8]]
set_property IOSTANDARD LVCMOS33 [get_ports seg[8]]
#pin assignment for pushbotton switches
set_property PACKAGE_PIN T18 [get_ports btn[0]]
set_property IOSTANDARD LVCMOS33 [get_ports btn[0]]
set_property PACKAGE_PIN T17 [get_ports btn[1]]
set_property IOSTANDARD LVCMOS33 [get_ports btn[1]]
set_property PACKAGE_PIN U17 [get_ports btn[2]]
set_property IOSTANDARD LVCMOS33 [get_ports btn[2]]
set_property PACKAGE_PIN W19 [get_ports btn[3]]
set_property IOSTANDARD LVCMOS33 [get_ports btn[3]]
set_property PACKAGE_PIN U18 [get_ports btn[4]]
set_property IOSTANDARD LVCMOS33 [get_ports btn[4]]
#pin VGA interface
set_property PACKAGE_PIN G19 [get_ports R[0]]
set_property IOSTANDARD LVCMOS33 [get_ports R[0]]
set_property PACKAGE_PIN H19 [get_ports R[1]]
set_property IOSTANDARD LVCMOS33 [get_ports R[1]]
set_property PACKAGE_PIN J19 [get_ports R[2]]
set_property IOSTANDARD LVCMOS33 [get_ports R[2]]
set_property PACKAGE_PIN N19 [get_ports R[3]]
set_property IOSTANDARD LVCMOS33 [get_ports R[3]]
set_property PACKAGE_PIN J17 [get_ports G[0]]
set_property IOSTANDARD LVCMOS33 [get_ports G[0]]
set_property PACKAGE_PIN H17 [get_ports G[1]]
set_property IOSTANDARD LVCMOS33 [get_ports G[1]]
set_property PACKAGE_PIN G17 [get_ports G[2]]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports G[2]]
set_property PACKAGE_PIN D17 [get_ports G[3]]
set_property IOSTANDARD LVCMOS33 [get_ports G[3]]
set_property PACKAGE_PIN N18 [get_ports B[0]]
set_property IOSTANDARD LVCMOS33 [get_ports B[0]]
set_property PACKAGE_PIN L18 [get_ports B[1]]
set_property IOSTANDARD LVCMOS33 [get_ports B[1]]
set_property PACKAGE_PIN K18 [get_ports B[2]]
set_property IOSTANDARD LVCMOS33 [get_ports B[2]]
set_property PACKAGE_PIN J18 [get_ports B[3]]
set_property IOSTANDARD LVCMOS33 [get_ports B[3]]
set_property PACKAGE_PIN P19 [get_ports HS]
set_property IOSTANDARD LVCMOS33 [get_ports HS]
set_property PACKAGE_PIN R19 [get_ports VS]
set_property IOSTANDARD LVCMOS33 [get_ports VS]
```

对于本书中的项目，除特殊说明外，都适用于 Basys2 和 Basys3 开发板。开发 Basys2 开发板的应用项目时，使用 ISE 集成开发环境；开发 Basys3 开发板的应用项目时，使用 Vivado 集成开发环境；当使用 Vivado 开发环境开发 Basys3 开发板的应用项目时，参考例 3-5 所示编写引脚的约束文件。

### 3.1.3　使用 ISim 进行功能仿真

在代码编写完毕后，需要借助测试平台来验证所设计的模块是否满足要求。ISE 测试平台的建立，是利用 HDL 语言实现的。

首先在工程管理区的任意位置右击，并在弹出的菜单中选择 New Source 命令；然后选中 Verilog Test Fixture 类型，输入文件名 gate2_test；再单击 Next 按钮，进入下一页。这时，工程中所有 Verilog Module 的名称都会显示出来。设计人员需要指定要测试的模块。

用鼠标选中 gate2 模块，然后单击 Next 按钮进入下一页。直接单击 Finish 按钮，ISE 会在源代码编辑区显示自动生成的测试模块的代码。

**【例 3-6】** 自动生成测试模块。

```
module gate2_test;
    //Inputs
    reg a;
    reg b;
    //Outputs
    wire [5:0] y;
    //Instantiate the Unit Under Test (UUT)
    gate2 uut (
        .a(a),
        .b(b),
        .y(y)
    );
```

```
    initial begin
        //Initialize Inputs
        a = 0;
        b = 0;
        //Wait 100 ns for global reset to finish
        #100;
        //Add stimulus here
    end
endmodule
```

由此可见,ISE 自动生成了测试平台的完整架构,包括所需信号、端口声明以及模块调用。测试人员的工作就是在 initial...end 模块中的"//Add stimulus here"后面添加测试向量生成代码。可以通过改变输入信号来观察输出信号的变化。添加后的测试代码如例 3-7 所示。

**【例 3-7】** 例 3-1 的 testbench。

```
module gate2_test;
    //Inputs
    reg a;
    reg b;
    //Outputs
    wire [5:0] y;
    //Instantiate the Unit Under Test (UUT)
    gate2 uut (
        .a(a),
        .b(b),
        .y(y)
    );
    initial begin
        //Initialize Inputs
        a = 0;
        b = 0;
        //Wait 100 ns for global reset to finish
        #100;
        //Add stimulus here
        repeat(2) begin
            a = 0;
            b = 0;
            #100;
            a = 0;
            b = 1;
            #100;
            a = 1;
            b = 0;
            #100;
            a = 1;
```

```
            b = 1;
            #100;
        end
    end
endmodule
```

完成测试平台后，将 Design 界面设置为 Simulation 选项，这时在过程管理区显示与仿真有关的进程，如图 3-1 所示。

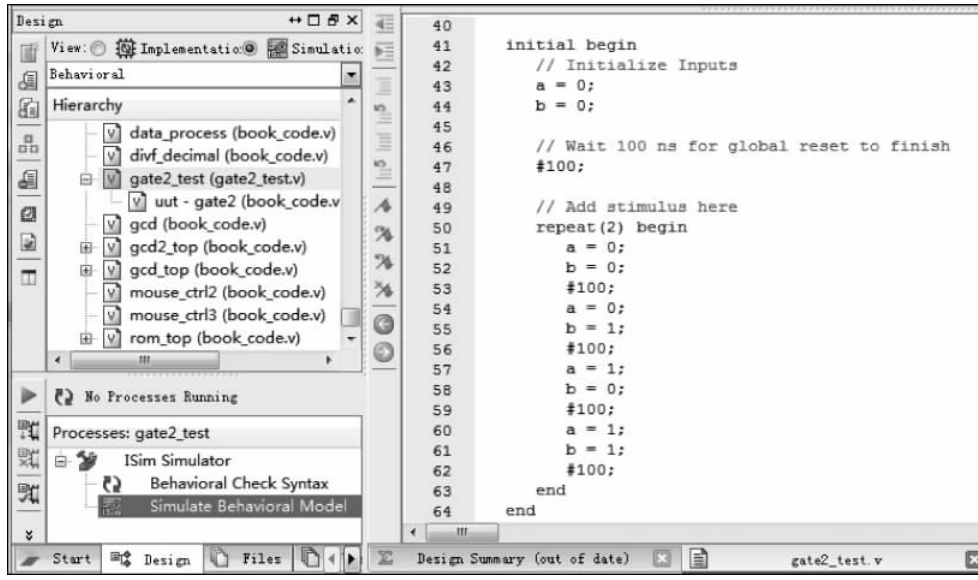


图 3-1　选择待测模块对话框

选中图 3-1 中 Xilinx ISE Simulator 下的 Simulate Behavioral Model 项，然后右击，选择弹出菜单的 Properties 项，弹出如图 3-2 所示的属性设置对话框。其中的一行 Simulation Run Time 用于设置仿真时间，可将其修改为任意时长。本例采用默认值。

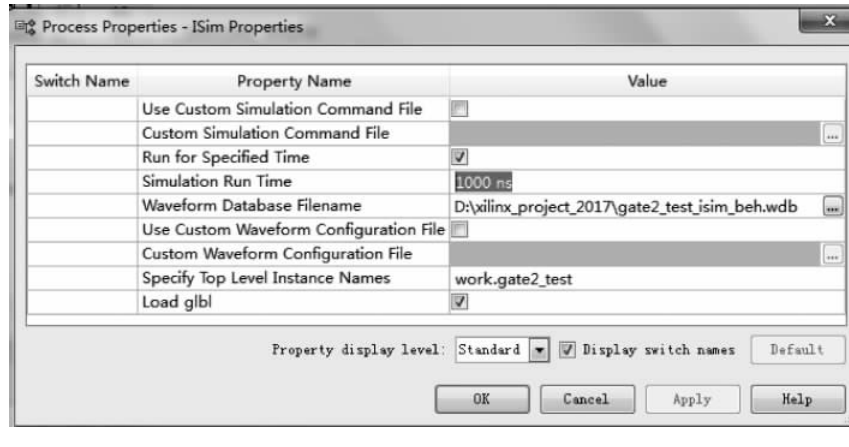


图 3-2　仿真过程示意图

仿真参数设置完毕,就可以进行仿真了。直接双击 ISE Simulator 软件中的 Simulate Behavioral Model,ISE 自动启动 ISE Simulator 软件,得到的仿真波形如图 3-3 所示。

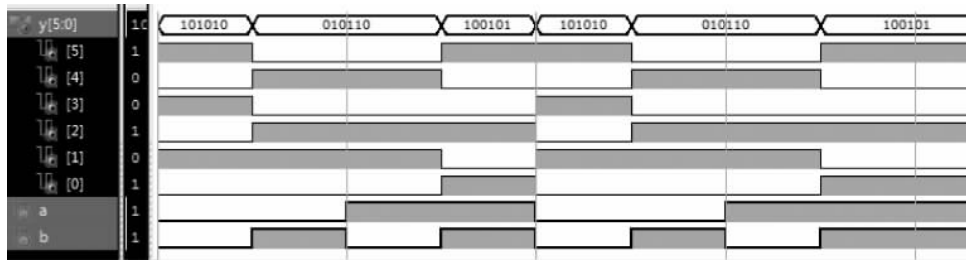


图 3-3　例 3-7 对应的仿真波形图

从仿真波形图可以看出,输出按照例 3-1 中设定的逻辑随着输入的变化而变化。

下面介绍 ISim 仿真软件的一些常用的、实用的功能。

(1) 更改数据显示格式。

ISim 在仿真时默认是二进制格式。为了便于使用,可以更改其显示格式。右击需要更改显示格式的数据,然后单击 Radix,弹出可选的显示格式,如图 3-4 所示,包括 Binary(二进制)、Hexadecimal(十六进制)、Unsigned Decimal(无符号十进制数)、Signed Decimal(有符号十进制数)、Octal(八进制)和 ASCII(ASCII 码)。

如果发现数据高、低位反向,选择 Reverse Bit Order(反转 bit 顺序,即高位和地位对换)。

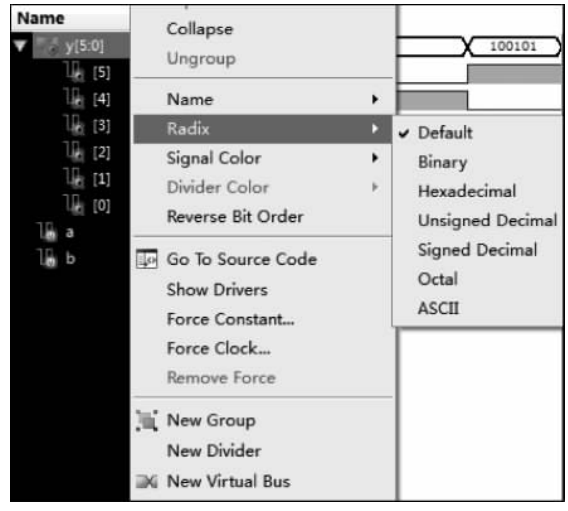


图 3-4　更改显示格式和位序

对于 1bit 数据,是一条线,如图 3-5 中变量 a 的波形。也可以选中该数据后右击。选择 New Virtual Bus,然后修改名字为原来的信号名,将一条线变成虚拟的总线形式,如图 3-5 中的变量 b。