# 第 3 章　乘法器和除法器的设计和验证

在现代中央处理单元（Central Processing Unit，CPU）中，不但具有最基本的算术逻辑单元（Arithmetic Logic Unit，ALU），而且还额外提供了专用的乘法器和除法器，以提高 CPU 在进行乘法和除法运算时的整体性能。

本章将介绍乘法器和除法器的算法原理，在此基础上使用高云云源软件对乘法器和除法器设计进行综合，并使用 ModelSim 软件对乘法器和除法器的逻辑功能进行验证。

## 3.1　乘法器的设计和验证

布斯算法是一种乘法算法，它以二进制补码的形式将两个有符号二进制数进行相乘。

本节将基于布斯算法使用 Verilog HDL 设计与实现布斯乘法器的硬件结构，并使用云源软件和 ModelSim 软件对设计进行综合和仿真。

### 3.1.1　基-2 布斯算法的设计

本小节将介绍基-2 布斯算法的原理，在此基础上使用 Verilog HDL 对该算法进行描述，最后介绍使用高云云源软件对设计进行综合的方法。

**1．基-2 布斯算法的原理**

考虑一个 $n$ 位宽度的被乘数 M，用二进制数表示为 $M_{n-1}M_{n-2}\cdots M_2M_1M_0$；$n$ 位宽度的乘数 R，用二进制数表示为 $R_{n-1}R_{n-2}\cdots R_2R_1R_0$，被乘数 M 和乘数 R 均为有符号的数。将乘数 R 用二进制补码表示，可以表示为

$$R=-R_{n-1}\times 2^{n-1}+\left(\sum_{i=0}^{n-2}R_i\times 2^i\right)$$

下面给 $n$ 位乘数 R 的最低有效位（Least Significant Bit，LSB）后面添加一个额外的“0”，将该位称为 $R_{-1}$，即 $R_{-1}=0$。进一步将乘数 R 表示为

$$\begin{aligned}R&=-R_{n-1}2^{n-1}+R_{n-2}2^{n-2}+R_{n-3}2^{n-3}+R_{n-4}2^{n-4}+\cdots+R_32^3+R_22^2+R_12^1+R_02^0+R_{-1}\\&=-R_{n-1}2^{n-1}+(2-1)R_{n-2}2^{n-2}+(2-1)R_{n-3}2^{n-3}+(2-1)R_{n-4}2^{n-4}+\cdots+(2-1)R_32^3\\&\quad+(2-1)R_22^2+(2-1)R_12^1+(2-1)R_02^0+R_{-1}\\&=(R_{n-2}-R_{n-1})2^{n-1}+(R_{n-3}-R_{n-2})2^{n-2}+(R_{n-4}-R_{n-3})2^{n-3}+\cdots+(R_3-R_4)2^4\\&\quad+(R_2-R_3)2^3+(R_1-R_2)2^2+(R_0-R_1)2^1+(R_{-1}-R_0)2^0\end{aligned}$$

进一步将被乘数 M 和乘数 R 的乘法运算表示为 M×R，得到的乘积结果用符号 P 表示：

$$\begin{aligned}P&=M\times[(R_{n-2}-R_{n-1})2^{n-1}+(R_{n-3}-R_{n-2})2^{n-2}+(R_{n-4}-R_{n-3})2^{n-3}+\cdots\\&\quad+(R_3-R_4)2^4+(R_2-R_3)2^3+(R_1-R_2)2^2+(R_0-R_1)2^1+(R_{-1}-R_0)2^0]\\&=M\times[(S_{n-1}\times 2^{n-1}+S_{n-2}\times 2^{n-2}+S_{n-3}\times 2^{n-3}+\cdots+S_4\times 2^4+S_3\times 2^3+S_2\times 2^2+S_1\times 2^1+S_0\times 2^0]\\&=M\times S_{n-1}\times 2^{n-1}+M\times S_{n-2}\times 2^{n-2}+M\times S_{n-3}\times 2^{n-3}+\cdots+M\times S_4\times 2^4+M\\&\quad\times S_3\times 2^3+M\times S_2\times 2^2+M\times S_1\times 2^1+M\times S_0\times 2^0\end{aligned}$$

式中，$S_i = R_{i-1} - R_i$。显然，$S_i$ 和 $R_{i-1}$ 与 $R_i$ 之间存在下面的关系（见表 3.1）。

**表 3.1　相邻两位 $R_i$ 和 $R_{i-1}$ 与 $S_i$ 之间的对应关系**

| $R_i$ | $R_{i-1}$ | $S_i$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | +1 |
| 1 | 0 | −1 |
| 1 | 1 | 0 |

将上式进一步简写为

$$P=\sum_{i=0}^{n-1} M \times S_i \times 2^i$$

对该式进行进一步分析可知下面的事实：

（1）当 $S_i$=0 时，乘积累加器 P 的值保持不变；

（2）当 $S_i$=1 时，将被乘数 M 乘以 $2^i$ 得到的部分乘积加到乘积累加器 P 中；

（3）当 $S_i$=−1 时，从乘积累加器 P 中减去被乘数 M 乘以 $2^i$ 得到的部分乘积。

这就是基−2 布斯算法的核心思想。进一步观察上面的式子可知，与 $2^i$ 进行相乘运算的操作可以转换为移位操作。

在实际实现时，基−2 布斯算法可以通过将两个预定的值 A 和 S 中的一个重复加（使用普通的无符号二进制加法）到乘积 P，然后对 P 执行向右的算术移位来实现。设 m 和 r 分别为 *x* 位二进制表示的被乘数和 *y* 位二进制数表示的乘数。其乘法的具体实现过程如下所示。

（1）确定 A 和 S 的值，以及乘积 P 的初始值。所有这些数字的长度都应该等于(*x*+*y*+1)。

① 对于 A，用 m 的值填充最高有效位（Most Significant Bit，MSB），即最左边的位，用“0”填充剩余的(*y*+1)位。

② 对于 S，以二进制补码表示法，用（−m）的值填充最高有效位，用“0”填充剩余的(*y*+1)位。

③ 对于 P，用 0 填充最高有效的 *x* 位，在其右侧，添加 r 的值。用“0”填充最低有效（最右边）的位。

（2）确定 P 的两个最低有效位。

① 如果它们的组合为“01”，则计算 P+A 的值，忽略任何溢出。

② 如果它们的组合为“10”，则计算 P+S 的值，忽略任何溢出。

③ 如果它们的组合为“00”，不做任何操作，下一步直接使用 P。

④ 如果它们的组合为“11”，不做任何操作，下一步直接使用 P。

（3）将第（2）步得到的值向右算术移动一位。移动的结果作为 P 的新值。

（4）重复步骤（2）和步骤（3），直到完成 *y* 次。

（5）从得到的乘积中删除 LSB，即最右边的位。这将作为 *m* 和 *r* 的乘积。

**【例 3.1】** 两个有符号的整数 4 和−5 相乘，采用 4 位位宽表示这两个数。根据上面的计算规则可知：

（1）m=“0100”（二进制数表示），r=“1011”（二进制数表示），−m=“1100”（二进制数表示），且 *x*=4，*y*=4。

（2）A=“0100 0000 0”，S=“1100 0000 0”，P=“0000 1011 0”。

（3）执行下面的循环，总共 4 次。

① P=“0000 1011 0”，最低两位为“10”，P+S 的值为“1100 1011 0”，算术右移一位，得到

P=“1110 0101 1”。

② P=“1110 0101 1”，最低两位为“11”，不做任何操作，算术右移一位，得到 P=“1111 0010 1”。

③ P=“1111 0010 1”，最低两位为“01”，P+A 的值为“0011 0010 1”，算术右移一位，得到 P=“0001 1001 0”。

④ P=“0001 1001 0”，最低两位为“10”，P+S 的值为“1101 1001 0”，算术右移一位，得到 P=“1110 1100 1”。

（4）去掉最低有效位后的 P=“1110 1100”，等效于十进制数−20。

**【例 3.2】** 两个有符号的整数−7 和−6 相乘，采用 4 位位宽表示这两个数，根据上面的计算规则可知：

（1）m=“1001”（二进制数表示），r=“1010”（二进制数表示），−m=“0111”（二进制数表示），且 x=4，y=4。

（2）A=“1001 0000 0”，S=“0111 0000 0”，P=“0000 1010 0”。

（3）执行下面的循环，总共 4 次。

① P=“0000 1010 0”，最低两位为“00”，不做任何操作，算术右移一位，得到 P=“0000 0101 0”。

② P=“0000 0101 0”，最低两位为“10”，P+S 的值为“0111 0101 0”，算术右移一位，得到 P=“0011 1010 1”。

③ P=“0011 1010 1”，最低两位为“01”，P+A 的值为“1100 1010 1”，算术右移一位，得到 P=“1110 0101 0”。

④ P=“1110 0101 0”，最低两位为“10”，P+S 的值为“0101 0101 0”，算术右移一位，得到 P=“0010 1010 1”。

（4）去掉最低有效位后的 P=“0010 1010”，等效于十进制数 42。

**2．建立新的设计工程**

（1）在 Windows 11 操作系统桌面中，找到并双击名字为“Gowin_V1.9.9 (64-bit)”的图标，启动高云云源软件（以下简称云源软件）。

（2）在云源软件主界面主菜单下，选择 File->New。

（3）弹出“New”对话框。在该对话框中，找到并展开“Projects”条目。在展开条目中，找到并选择“FPGA Design Project”条目。

（4）单击该对话框右下角的“OK”按钮，退出“New”对话框。

（5）弹出“Project Wizard-Project Name”对话框。在该对话框中，按如下设置参数。

① Name：example_3_1。

② Create in：E:\cpusoc_design_example。

（6）弹出“Project Wizard-Select Device”对话框。在该对话框中，按如下设置参数。

① Series（系列）：GW2A；

② Package（封装）：PBGA484；

③ Device（器件）：GW2A-55；

④ Speed（速度）：C8/I7；

⑤ Device Version（器件版本）：C。

选中器件型号（Part Numer）为“GW2A-LV55PG484C8/I7”的一行。

（7）单击该对话框右下角的“Next”按钮。

（8）弹出“Project Wizard-Summary”对话框。在该对话框中，总结了建立工程时设置的参数。

（9）单击该对话框右下角的“Finish”按钮，退出“Project Wizard-Summary”对话框。

### 3．创建基-2 布斯算法设计文件

本部分将介绍如何在当前设计工程中创建并添加基-2 布斯算法设计文件，主要步骤如下所述。

（1）在云源软件当前工程主界面左侧的“Design”标签页中，找到并选中 example_3_1 或 GW2A-LV55PG484C8/I7，单击鼠标右键，出现浮动菜单。在浮动菜单中，选择 New File。

（2）弹出“New”对话框。在该对话框中，选择“Verilog File”条目。

（3）单击该对话框右下角的“OK”按钮，退出“New”对话框。

（4）弹出“New Verilog file”对话框。在该对话框中，在“Name”右侧的文本框中输入 radix_2_booth，即该文件的名字为 radix_2_booth.v。

（5）单击该对话框右下角的“OK”按钮，退出“New Verilog file”对话框。

（6）自动打开 radix_2_booth.v 文件，在该文件中添加设计代码，如代码清单 3-1 所示。

**代码清单 3-1　radix_2_booth.v 文件**

```
'timescale 1ns/1ps
module radix_2_booth(                    // module 关键字定义模块 radix_2_booth
    input [31 : 0] x,                    //  定义被乘数 x 的宽度为 32 位
    input [31 : 0] y,                    //  定义乘数 y 的宽度为 32 位
    output [63:0] z                      //  定义乘积宽度为 64 位
);
reg [64: 0] product;                     // reg 关键字定义变量 product，位宽 65 位
integer i;                               // integer 关键字定义整数 i
reg[31:0] temp;                          // reg 关键字定义变量 temp，位宽 32 位
assign z=product[64:1];
always @(*)                              // always 关键字定义过程语句参考基-2 布斯算法的实现规则
begin                                    // begin 关键字标识过程语句的开始
  product={32'h0,y,1'b0};                //  给 product 分配初值
  for(i=0;i<32;i=i+1)                    // for 关键字定义循环结构
  begin                                  // begin 关键字标识 for 循环结构的开始
    if(product[1:0]==1)                  // if 关键字定义第一个判断条件分支
      begin                              // begin 关键字标识第一个判断条件分支的开始
        temp=product[64:33]+x;           //  加法操作
        product={temp,product[32:0]};    //  并置/连接操作
      end                                // end 关键字标识第一个判断条件分支的结束
    else if(product[1:0]==2)             // else if 关键字定义另一个判断条件分支
      begin                              // begin 关键字标识令一个判断条件分支的开始
        temp=product[64:33]-x;           //  减法操作
        product={temp,product[32:0]};    //  并置/连接操作
      end                                // end 关键字标识令一个判断条件分支的结束
    product={product[64],product[64:1]};
  end                                    // end 关键字标识 for 循环结构的结束
end                                      // end 关键字标识 always 过程语句的结束
endmodule                                // endmodule 关键字标识模块 radix_2_booth 的结束
```



（7）按 Ctrl+S 组合键，保存该设计文件。

### 4．设计综合

本部分将介绍如何对设计执行综合，并查看综合后的结果，主要步骤如下所述。

（1）在云源软件当前工程主界面左侧的窗口中，单击“Process”标签。

（2）在“Process”标签页中，找到并双击“Synthesize”条目，执行对该设计的综合。

（3）在“Synthesize”条目下，找到并单击“Synthesis Report”条目。

（4）在云源软件右侧的窗口中，打开 Synthesis Messages 界面。在该界面中，给出了该设计所使用的逻辑资源。

思考与练习 3-1：查看基-2 布斯乘法器所使用的 GW2A-LV55PG484C8/I7 器件的资源量。

思考与练习 3-2：在高云云源软件主界面主菜单下，选择 Tools->Schematic Viewer->RTL Design Viewer，查看 RTL 网表结构。

### 3.1.2　基-2 布斯算法的验证

本小节将介绍使用 ModelSim 软件对基-2 布斯算法进行验证的方法。

**1．建立新的验证工程**

本部分将介绍如何在 ModelSim 软件中建立新的工程用于综合后仿真，主要步骤如下所述。

（1）启动 ModelSim 软件，进入 ModelSim SE-64 10.4c（以下简称 ModelSim）主界面。

（2）在 ModelSim 主界面主菜单下，选择 File->New->Project。

（3）弹出“Create Project”对话框。在该对话框中，按如下设置参数。

① Project Name：postsynth_sim。

② Project Location：E:/cpusoc_design_example/example_3_2。

（4）单击“OK”按钮，退出“Create Project”对话框。

（5）如果预先没有创建所指向的目录，则弹出“Create Project”对话框。该对话框中提示信息“The project directory does not exist. OK to create this directory?”。

（6）单击该对话框中的“是”按钮。

**2．添加已存在的文件**

本部分将介绍如何添加已经存在的网表文件 example_3_1.vg。添加该文件的主要步骤如下所述。

（1）在完成上部分的操作后，弹出“Add Items to the Project”对话框。在该对话框中，单击名字为“Add Existing File”的图标。

（2）弹出“Add file to Project”对话框。在该对话框中，单击“File Name”标题栏右侧的“Browse”按钮。

（3）弹出“Select files to add to project”对话框。在该对话框中，按如下设置参数。

① 文件类型：All Files（*.*）(通过下拉框设置)。

② 将路径定位到 E:/cpusoc_design_example/example_3_1/impl/gwsynthesis，选中该目录下的 example_3_1.vg 文件，并单击该对话框右下角的“打开”按钮，自动退出“Select files to add to project”对话框。

③ 勾选“Add file to Project”对话框右下角“Copy to project directory”前面的复选框。

**3．创建新的测试文件**

本部分将介绍如何创建新的测试文件，主要步骤如下所述。

（1）再次单击“Add Items to the Project”对话框中名字为“Create New File”的图标。

（2）弹出“Create Project File”对话框。在该对话框中，按如下设置参数。

① File Name：test_radix_2_booth。（文本框输入）。

② Add file as type：Verilog（下拉框选择）。

（3）单击“Create Project File”对话框右下角的“OK”按钮，退出该对话框。

（4）单击“Add items to the Project”对话框右下角的“Close”按钮，退出该对话框。

（5）在 ModelSim 当前工程主界面的 Project 窗口中，找到并双击 test_radix_2_booth.v，打开该文件。在该文件中输入测试代码，如代码清单 3-2 所示。

**代码清单 3-2　test_radix_2_booth.v 文件**

```
'timescale 1ns/1ps                          // `timescale 关键字
module test_radix_2_booth;                  // module 关键字定义模块 test_radix_2_booth
reg [31:0] x;                               // reg 关键字定义变量 x，位宽 32 位
reg [31:0] y;                               // reg 关键字定义变量 y，位宽 32 位
wire [63:0] z;                              // wire 关键字定义网络 z，位宽 64 位
radix_2_booth Inst_radix_2_booth(           //  例化模块 radix_2_booth
      .x(x),                                //  模块 radix_2_booth 的端口 x 连接到变量 x
      .y(y),                                //  模块 radix_2_booth 的端口 y 连接到变量 y
      .z(z)                                 //  模块 radix_2_booth 的端口 z 连接到网络 z
);

always                                      // always 关键字声明过程语句
begin                                       //  给被乘数和乘数赋不同的值
   x=-80;                                   //  给 x 分配值-80
   y=-90;                                   //  给 y 分配值-90
   #100;                                    //  保持 100ns
   x=255;                                   //  给 x 分配值 255
   y=255;                                   //  给 y 分配值 255
   #100;                                    //  保持 100ns
   x=-101;                                  //  给 x 分配值-101
   y=202;                                   //  给 y 分配值 202
   #100;                                    //  保持 100ns
   x=443;                                   //  给 x 分配值 443
   y=-978;                                  //  给 y 分配值-978
   #100;                                    //  保持 100ns
end                                         // end 关键字标识 always 过程的结束
endmodule                                   // endmodule 关键字标识模块 test_radix_2_booth 的结束
```



（8）按 Ctrl+S 组合键，保存该设计文件。

**4．执行综合后仿真**

本部分将介绍如何对设计执行综合后仿真，主要步骤如下所述。

（1）在 ModelSim 主界面主菜单下，选择 Compile->Compile All，对这两个文件成功编译后，在 Status 一列中显示✔标记，如图 3.1 所示。

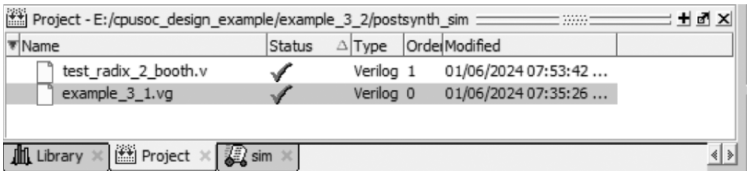


图 3.1　对设计编译后的结果

（2）在 ModelSim 主界面主菜单下，选择 Simulate->Start Simulation。

（3）弹出“Start Simulation”对话框，如图 3.2 所示。在该对话框中，单击“Libraries”标签。在该标签页中有两个子窗口，即 Search Libraries 和 Search Libraries First。

① 单击 Search Libraries 子窗口右侧的“Add”按钮，弹出“Select Library”对话框，如图 3.3 所示。在该对话框中，首先在“Select Library”标题栏下，通过下拉框，将“Select Library”设置为 gw2a；然后单击该对话框右下角的“OK”按钮，退出“Select Library”对话框。

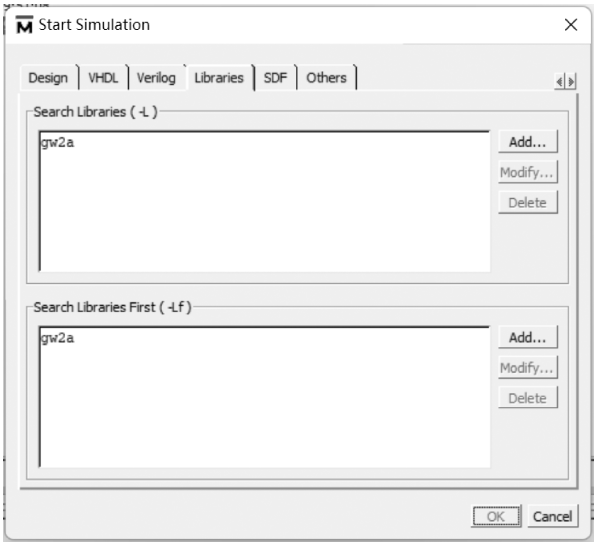


图 3.2 “Start Simulation”对话框中的“Libraries”标签页


图 3.3 “Select Library”对话框

② 单击 Search Libraries First 子窗口右侧的“Add”按钮，弹出“Select Library”对话框。在该对话框中，首先在“Select Library”标题栏下，通过下拉框，将“Select Library”设置为 gw2a；然后单击该对话框右下角的“OK”按钮，退出“Select Library”对话框。

（4）在“Start Simulation”对话框中，单击“Design”标签。在该标签页中，找到并展开“work”条目。在展开条目中，找到并选中“test_radix_2_booth”条目，如图 3.4 所示。在该标签页中，不要勾选“Enable optimization”前面的复选框。

（5）单击该对话框右下角的“OK”按钮，退出“Start Simulation”对话框。

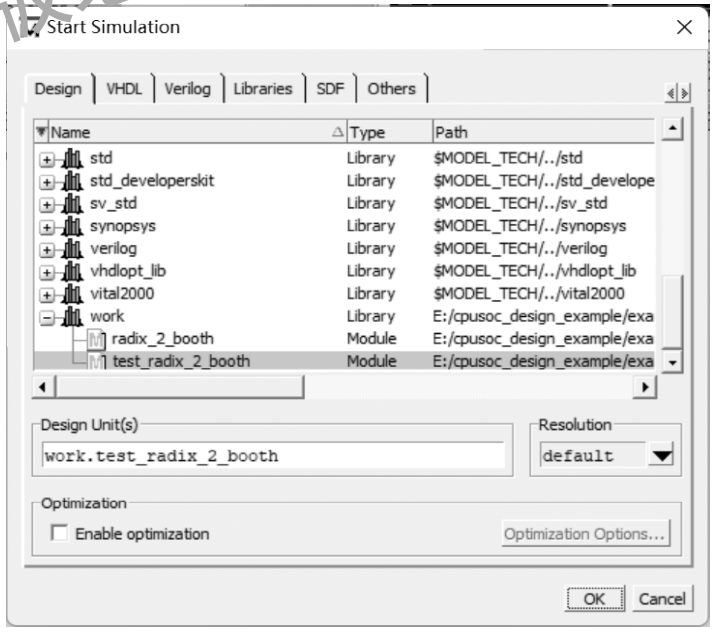


图 3.4 “Start Simulation”对话框中的“Design”标签页

（6）进入仿真界面，如图 3.5 所示。在该界面左侧的 Instance 窗口中，默认选择“test_radix_2_booth”条目，在 Objects 窗口中，按住 Ctrl 按键，分别单击 x、y 和 z 条目，以选中这 3 个信号。

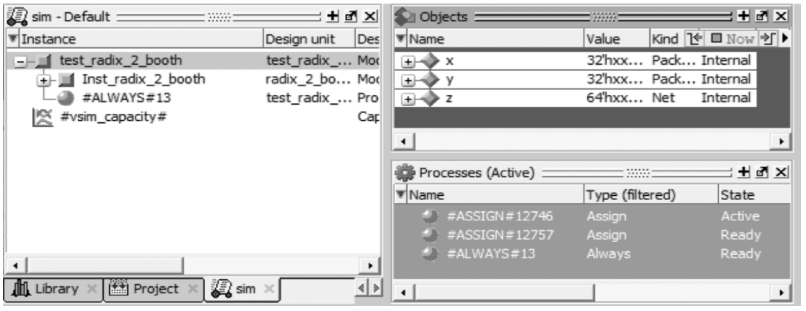


图 3.5　仿真界面

（7）单击鼠标右键，出现浮动菜单。在浮动菜单内，选择 Add Wave。将这 3 个信号自动添加到 Wave 窗口中，如图 3.6 所示。

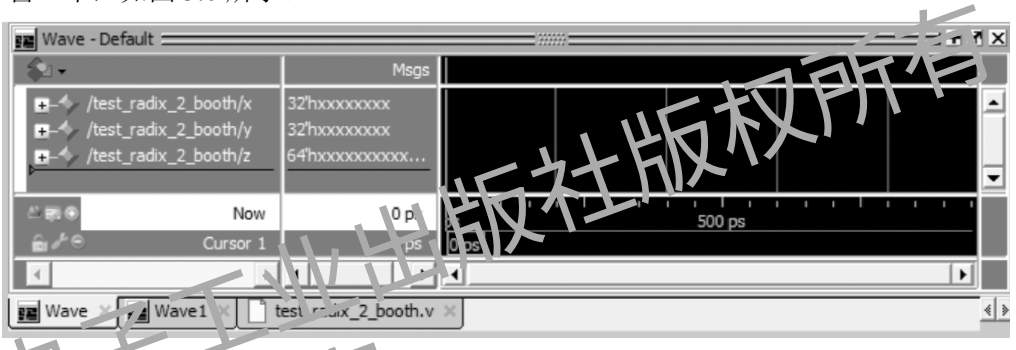


图 3.6　将 x、y 和 z 添加到 Wave 窗口中



（8）找到 ModelSim 主界面最下面的 Transcript 窗口。在该窗口的 VSIM>提示符后，输入下面的一行命令：

```
run 400ns
```

并按回车键。该命令表示运行仿真的时间长度为 400ns。

（9）在 Wave 窗口中，看到当给输入端口 x 和 y 分配不同的值时，端口 z 的输出结果，如图 3.7 所示。

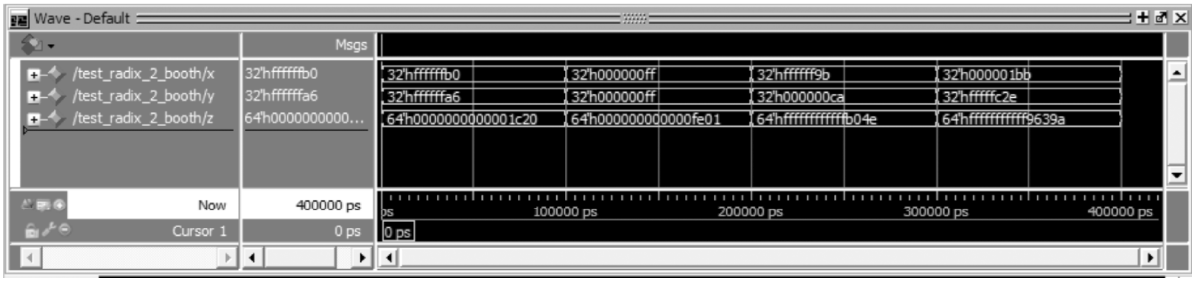


图 3.7　执行仿真后 Wave 窗口中给出的仿真波形

**注：**通过单击工具栏中的放大按钮或缩小按钮，使得在 Wave 窗口中正确的显示波形。

（10）图 3.7 使用 16 进制数显示不够直观，将其改为有符号的 10 进制数进行显示。在图 3.7