CMP科技经典

PLD 可编程逻辑器件实用开发技术丛书

通信收发信机的Verilog实现与仿真

姜宇柏 黄志强 等编著

机械工业出版社
CHINA MACHINE PRESS

可编程逻辑器件实用开发技术丛书

通信收发信机的Verilog实现与仿真

姜宇柏　黄志强　等编著

机械工业出版社

本书简要介绍了 Verilog 硬件描述语言的基础知识，列举了大量实例，每个实例都经过精心选择，非常利于读者掌握 Verilog 硬件描述语言本身及其建模方法。这些例子实用性都很强，其中有很多已经应用到实际的软件无线电收发信机中，对初学者和中级水平的工程师都很有帮助。

本书内容丰富、实用性很强，可以作为高等院校通信电子类高年级本科生、研究生的教材或教学参考书，同时对于通信、雷达、信号处理等各个领域的硬件设计工程师来说，也是一本具有实用价值和指导意义的技术参考书。

图书在版编目（CIP）数据

通信收发信机的 Verilog 实现与仿真/姜宇柏，黄志强等编著．—北京：机械工业出版社，2006.10
（可编程逻辑器件实用开发技术丛书）
ISBN 7-111-20106-X

Ⅰ.通… Ⅱ.①姜…②黄… Ⅲ.①硬件描述语言，VHDL—程序设计②数字通信 Ⅳ.①TP312②TN914.3

中国版本图书馆 CIP 数据核字（2006）第 124239 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
责任编辑：张俊红 版式设计：霍永明 责任校对：张晓蓉
封面设计：马精明 责任印制：洪汉军
北京汇林印务有限公司印刷
2007 年 1 月第 1 版第 1 次印刷
184mm×260mm · 27.25 印张 · 672 千字
0001—4000 册
定价：47.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
本社购书热线电话(010)68326294
编辑热线电话(010)88379768
封面无防伪标均为盗版

可编程逻辑器件实用开发技术丛书

编 委 会

丛 书 序

随着科学技术的迅猛发展，电子工业界经历了巨大的飞跃。集成电路的设计正朝着速度快、性能高、容量大、体积小和微功耗的方向发展。这种发展必将导致集成电路的设计规模日益增大，复杂程度日益增高。基于这种情况，可编程逻辑器件的出现和发展大大改变了传统的系统设计方法，这种方法使得电子系统设计变得更加简单方便、灵活快速，因此掌握可编程逻辑器件和相应的设计技术已经成为从事电子系统设计的设计工程师和科研人员的一项重要设计手段和技能。

可编程逻辑器件和相应的设计技术体现在三个主要方面：一是可编程逻辑器件的芯片技术；二是适用于可编程逻辑器件的硬件编程技术，即 VHDL 技术和 Verilog 技术；三是可编程逻辑器件设计的 EDA 开发工具，它主要用来进行可编程逻辑器件应用的具体实现。

可编程逻辑器件（Programmable Logic Device，PLD）的逻辑功能是由设计人员根据系统设计的具体要求，通过相应的器件编程来实现的。另外，由于 PLD 的集成度很高，因此它可以满足大多数数字系统设计的需要。历史上，可编程逻辑器件经历了 PROM、PLA、PAL、GAL、EPLD 到 CPLD 和 FPGA 的发展过程，在结构、制造工艺、集成度、逻辑功能、速度和功耗上都有了很大的提高和改进。其中，CPLD 和 FPGA 由于集成度非常高，因此这两种器件成为目前可编程逻辑器件的主流。

可编程逻辑器件的硬件编程技术主要体现在硬件描述语言的应用中，目前广泛使用的硬件描述语言是 VHDL 和 Verilog。这两种描述语言具有强大的功能和硬件描述能力，易于共享和复用，同时还具有独立于器件和工艺的设计能力，因此它们得到了各种 EDA 工具和集成电路厂商的普遍认同和推广，目前正在全球范围内的电子系统设计领域获得广泛应用。如今，国内外一些用户在购买和使用各种 EDA 工具时，通常都把是否支持 VHDL 和 Verilog 作为 EDA 工具是否先进的标准之一。

EDA 开发工具主要包括编辑器、仿真工具、检查/分析工具和优化/综合工具等。其中，编辑器用来对设计输入进行图形或者文本等方面的编辑操作；仿真工具是用来完成设计仿真操作的 EDA 开发工具，主要包括逻辑仿真工具和时序仿真工具；检查/分析工具用来对设计的逻辑产生可能性、电路的电气特性以及时序关系等进行检查和分析；优化/综合工具用来把一种硬件描述转化为底层描述，在转化的过程中伴随着设计的某种优化。现在，高级的 EDA 开发工具都是一种集成的开发环境，即集成了上述的所有开发工具，这样就可以用一种集成开发环境来完成所有的设计工作。

可见，对于可编程逻辑器件的设计来说，上面三个主要方面是相辅相成不可分割

的，一个高水平的可编程逻辑设计人员必须掌握这三个方面的技术，这样才能够满足实际设计工作的需要，从而实现成本低、设计简单和资源优化的完美设计。现在，可编程逻辑器件获得了极其广泛的应用，无论是电子设计工程师还是高等院校的学生，都迫切需要系统地来学习相应的技术，因此也就需要一套理论严谨、内容新颖、实用性较强的可编程逻辑器件丛书来满足广大读者的学习需要。基于这一点，机械工业出版社的领导和编辑组织了这套“可编程逻辑器件实用开发技术丛书”，这套丛书重点介绍了 Xilinx 公司和 Altera 公司的 CPLD/FPGA、相应的 EDA 开发工具 ISE 和 MAX + plus II/Quartus II 以及相应的 VHDL 和 Verilog 设计技术。本套丛书具体包括：

- 《VHDL 设计实例与仿真》
- 《通信收发信机的 Verilog 实现与仿真》
- 《面向 CPLD/FPGA 的 Verilog 设计》
- 《面向 CPLD/FPGA 的 VHDL 设计》
- 《MAX + plus II 和 Quartus II 应用与开发技巧》
- 《ISE 应用与开发技巧》
- 《Xilinx 可编程逻辑器件的应用与设计》
- 《Altera 可编程逻辑器件的应用与设计》

为了保证这套丛书的高质量和实用性，特组织了一批具有丰富可编程逻辑器件设计经验的工程师来进行相应丛书的编写。这套丛书从实际应用的角度出发，全面系统、由浅入深地介绍了可编程逻辑器件的各个相关技术，可以使广大读者快速高效地掌握可编程逻辑器件的知识。本套丛书读者范围十分广泛，它既可作为高等学校计算机和电子工程专业的研究生和本科生的教材或教学参考书，也可作为广大电子电路设计工程师、ASIC 设计人员和系统设计人员的参考书。

由于可编程逻辑器件技术发展十分迅速，加上编写时间相对紧张，书中难免存在不足，恳请广大读者和专家批评指正，联系信箱：buptzjh@163.com。

丛书编委会

前　言

Verilog HDL是20世纪90年代在美国等先进工业国家兴起的一种设计复杂数字系统的技术，它是由GDA（Gateway Design Automation）公司的Phil Moorby在1983年末首创的，最初只设计了一个仿真与验证工具，之后又陆续开发了相关的故障模拟与时序分析工具。1985年，Moorby推出他的第三个商用仿真器Verilog－XL，获得了巨大的成功，从而使得Verilog HDL迅速得到推广应用。近几年来，随着电子系统中专用集成电路和FPGA的广泛应用，Verilog HDL在工业界的应用也越来越广泛，美国大多数公司的RTL级代码都是用Verilog HDL编写的。

Verilog HDL作为一种硬件描述语言，既具有一般编程语言的共性，又有自己的一套语法和风格，初学者应该首先掌握这门语言的基本语法，打好基础。然而一旦初学者基本掌握了Verilog HDL的基本语法，成为一个比较熟练的设计师后，就会发现只掌握其语法实际上远远不够，掌握了语法仅仅是Verilog HDL学习和应用的第一步，而对整个系统的设计和结合实际硬件进行时序的精细设计才是最需要Verilog HDL工程师下功夫的地方。

本书的写作目标是通过实例让初学者快速入门，让读者正确分析写好的Verilog HDL程序，进一步掌握Verilog HDL的设计流程和建模方法。实践是学好编程语言的最佳途径，大量的使用实例并带有详细的注解是本书的特色之一。书中提供了大量的应用实例，目的是让初学者很快掌握系统的建模方法，能够尽快开始自己试探着写出符合编程风格的程序。

Verilog HDL不断发展，而硬件工程师在进行任何开发的时候，首要的问题就是考虑硬件的可实现性。任何符合HDL语法标准的代码都是对硬件行为的一种描述，但不一定是可直接对应成电路的设计信息。而众多初学者试图做的，却是想让软件去综合算法级或者更加抽象的硬件行为描述，这些行为级描述往往都是不能很简单就用硬件实现的。实际上就是因为初学者把Verilog HDL当成一门普通的编程语言来学习，而忽略了Verilog HDL本身是一门硬件描述语言，其最终目的是描述硬件电路，也就是初学者在设计的时候缺少硬件意识。本书正是针对初学者这一常犯的毛病，每个例子都给出了综合的结果，不但用硬件描述语言向读者描述硬件电路，更重要的是将描述的结果直观地展现给读者，让初学者培养硬件意识，每写一句程序都清楚地知道其描述的硬件电路。HDL不仅定义了语法，而且对每个语法结构都定义了清晰的模拟、仿真语义。因此，用这种语言编写的模型能够使用Verilog仿真器进行验证。仿真给整个设计带来了很大的方便，一些在硬件实现中很难查找的逻辑和时序错误往往可以很容易通过仿真定位并排

除。因此本书在每个实例后面都给出了主要信号的仿真结果，使读者能够更加清晰地掌握整个设计的逻辑和时序。

本书内容丰富、实用性很强，可以作为高等院校通信电子类高年级本科生、研究生的教材或教学参考书，同时对于通信、雷达、信号处理等各个领域的硬件设计工程师来说，也是一本具有实用价值和指导意义的技术参考书。

本书共分9章，前两章主要介绍了 Verilog HDL 的基本语法和基本逻辑电路及时序电路的描述方法。第3章详细介绍了如何用 Model Sim 仿真软件仿真已经设计好的系统，包括激励文件的编写、功能仿真和布线后仿真。第4章重点介绍了如何写出能够硬件实现的 Verilog HDL 程序，阐述了硬件可综合性的重要性，并给出了一些工程建议。第5章～第9章主要介绍了 Verilog HDL 在通信和信号处理中的应用。需要强调的是，这些实例都已经应用到实际的软件无线电收发信机中，而且这些实例涵盖的范围非常广泛，包括基带信号处理、编解码、调制/解调、扩频/解扩以及 FIR 滤波器等方面。

需要说明的是，为了保持电路图原样，书中部分文字符号和图形符号并未按国家标准做统一处理，这点请读者注意。

本书由姜宇柏和黄志强编写，书中包含着作者多年从事 EDA 开发工程项目的经验总结。在编写本书的过程中，李晓凯、张海风、渠丰沛、渠丽娜、杜平、吴鹏、王涛、王辉、夏钦东、张博、尤晓丽、姜海亭、姜雪华、蒋建新、李玉红、程显奎、曹霖、潘天保、刘磊、赵海波完成了全书的文字整理和校对工作，并完成了部分章节的编写工作，在此表示衷心的感谢！由于 Verilog HDL 和 EDA 技术发展迅速，同时限于作者的理论水平和实际开发经验，书中难免存在一些不足之处或者错误，恳望广大读者和相关专家批评指正。

作　者

目　录

第 1 章　组合逻辑的 Verilog HDL 实现

在科学技术和电子技术不断发展的今天，集成电路的设计正朝着速度快、性能高、容量大、体积小和微功耗的方向发展，这种发展必将导致集成电路的设计规模日益增大，复杂程度日益提高。在这种情况下，沿用了几十年的传统硬件电路设计方法已不能满足需要，它已经远远落后于当今科学技术的发展。

目前，随着电子设计自动化（EDA）技术的不断普及和广泛应用，传统的“固定功能模块 + 连线”的设计方法正在逐步退出历史舞台，而基于芯片的设计方法正在成为电子系统设计的主流。在这种新的设计方法中，可编程逻辑器件和硬件描述语言成为两大主要开发工具，其中 Verilog HDL 是硬件描述语言中的主流语言之一，目前获得了广泛的应用。

Verilog HDL 是一种硬件描述语言，它可以对实际电路进行不同级别的抽象描述，这些抽象的级别和它们对应的模型类型共有以下 5 种：

1）系统级（System Level）：用高级语言结构设计模块的外部性能的模型。

2）算法级（Algorithm Level）：用高级语言结构设计算法的模型。

3）RTL（Register Transfer Level，寄存器传输级）：大多数硬件工程师工作在 RTL，RTL 模型是描述数据在寄存器之间流动和如何处理这些数据的模型。

4）门级（Gate Level）：描述逻辑门以及逻辑门之间的连接模型。

5）开关级（Switch Level）：描述器件中晶体管和储存节点以及它们之间连接的模型。

Verilog HDL 行为描述语言作为一种结构化和过程性的语言，其语法结构非常适合于算法级和 RTL 的模型设计。这种行为描述语言具有以下几个特点：

1）描述程序结构灵活，可描述顺序执行和并行执行的程序结构。

2）可以用不同的方式明确地控制过程的启动时间，可以用延迟表达式或事件表达式来控制过程的启动时间。

3）通过命名的事件来触发其他过程里的激活行为或停止行为。

4）提供了条件、if - else、case、循环程序结构。

5）提供了可带参数且非零延续时间的任务（Task）程序结构。

6）提供了可定义新的操作符的函数（Function）结构。

7）提供了用于建立表达式的算术运算符、逻辑运算符、位运算符。

8）Verilog HDL 作为一种结构化的语言，也非常适合于门级和开关级的模型设计。

Verilog HDL 的构造性语句可以精确地建立信号的模型，这样可以更加有效地利用硬件资源，也加强了程序的可控性，更好地反映硬件工程师的硬件设计原型。这是因为在 Verilog HDL 中，提供了延迟和输出强度的原语来建立精确程度很高的信号模型。信号值可以有不同的强度，可以通过设定宽范围的模糊值来降低不确定条件的影响。

设计工程师在不同的设计阶段可采用不同的抽象级：首先在行为级描述各功能块，以降低描述难度，提高仿真速度，在综合前将各功能模块进行 RTL 描述，用于综合的库中的大

多数单元采用结构级描述，对于不同的设计阶段，工程师可以采用比较方便的描述方式来描述功能模块。

下面讨论逻辑门的 Verilog HDL 描述，有了最基本的逻辑门的描述就可以对纯逻辑电路进行描述。

作为一种标准化的硬件描述语言，Verilog HDL 在进行具体的设计描述时需要遵循一定的开发流程，具体的操作流程如下所示：

1）进行设计要求的具体定义，主要包括性能指标和技术指标两大类；

2）确定具体的设计功能，划分功能模块；

3）采用 Verilog HDL 进行设计描述；

4）Verilog HDL 程序编译和仿真；

5）综合、优化和布局布线；

6）布局布线后的仿真操作；

7）设计实现，主要包括芯片的物理版图设计或者可编程逻辑器件的编程。

1.1 基本逻辑器件的 Verilog HDL 门级描述

在数字电路中，按其完成的逻辑功能的不同特点，可划分为组合逻辑电路和时序逻辑电路两大类。所谓组合逻辑电路，是指该电路在任意时刻的稳定状态仅取决于该时刻的输入信号，而与输入信号作用前电路所处的状态无关。时序逻辑电路是指它的输出不仅取决于当前输入，而且也取决于过去的输入序列，即过去输入序列不同，则在同一当前输入的情况下，输出也可能不同。本节主要讲述组合逻辑的 Verilog HDL 描述。

组合逻辑电路由许多逻辑门和开关组成。在基于芯片的设计方法中，要硬件实现一个组合逻辑门电路，首先要将逻辑门和开关等用某种硬件描述语言描述出来。在数字电路中，基本门电路是构成所有数字逻辑电路的基本单位，因此在电路设计中有很重要的地位。正像所有的数字电路书籍先介绍基本门电路一样，这里也要先讨论基本门电路的 Verilog HDL 描述。Verilog HDL 中有关门类型的关键字共有 26 个之多，这里主要讨论一下几种最为常用的门电路：

1）多输入门：and，nand，or，nor，xor，xnor；

2）多输出门：buf，not；

3）三态门：bufif0，bufif1，notif0，notif1；

4）上拉、下拉电阻：pullup，pulldown；

5）MOS 开关：cmos，nmos，pmos，rcmos，rnmos，rpmos；

6）双向开关：tran，tranif0，tranif1，rtran，rtranif0，rtranif1。

门级逻辑设计描述中一般使用具体的门实例语句实现，其用法很类似，简单的门实例语句的实例化语句格式为

gate _ type [instance _ name] (term1，term2，...，termN)；

在上述实例化语句格式中，instance _ name 是可选的，gate _ type 为前面列出的某种门类型，各 term 用于表示与门的输入/输出端口相连的线网或寄存器，同一门类型的多个实例能够在一个结构形式中定义。其语法如下：

```
gate _ type
[instance _ name1](term11,term12,...,term1N),
[instance _ name2](term21,term22,...,term2N),
                    .
                    .
                    .
[instance _ nameM](termM1,termM2,...,termMN);
```

1.1.1 内置的多输入门的 Verilog 描述

内置的多输入门只有单个输出，1 个或多个输入。多输入门实例化语句的语法如下：

multiple _ input _ gate _ type [instance _ name] (OutputA, Input1, Input2, ..., InputN);

第一个端口是输出，其他端口是输入，具体的例化方法如例 1-1 所示。

【例 1-1】 用 Verilog HDL 描述的与非门模块。

与非逻辑是与逻辑和非逻辑的结合，其逻辑函数表达式为 $F = \overline{A \cdot B}$，其真值表如表 1-1 所示。

表 1-1 与非逻辑真值表

I_1	I_0	O
0	0	1
0	1	1
1	0	1
1	1	0

与非门的 Verilog HDL 描述如下：

```
module S2 _ D1 ( S1 _ 1, S1 _ 2, LED );      // 模块名及端口定义，范围至 endmodule
  input S1 _ 1, S1 _ 2;                      // 输入端口定义
  output LED;                                // 输出端口定义
  nand U1 ( LED, S1 _ 1, S1 _ 2 );           // 门级描述语句，实现与非门的功能
endmodule                                    // 模块结束
```

例 1-1 是与非门的 Verilog HDL 描述，在这个 Verilog HDL 结构描述的模块中，S2 _ D1 定义了模块名，设计上层模块时可以用这个名（S2 _ D1）调用这个模块；module、input、output、endmodule 等都是关键字；nand 表示与非门；//……表示注释部分，注释只是为了方便程序员理解程序，对编译结果不起作用。完成程序以后，可以通过下面几个步骤来检查程序的正确性：

1）需要有测试激励信号输入到被测模块；

2）记录被测模块的输出信号；

3）把用功能和行为描述的 Verilog 模块转换为门级电路互连的电路结构（综合）；

4）对已经转换为门级电路结构的逻辑进行测试（门级电路仿真）；

5）对布局布线后的电路结构进行测试（布局布线后仿真）。

例 1-1 的综合结果如图 1-1 所示，其仿真结果如图 1-2 所示。

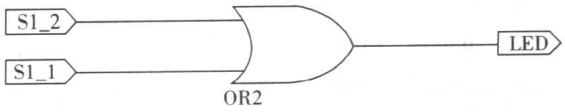

图 1-1　与非门模块的综合结果

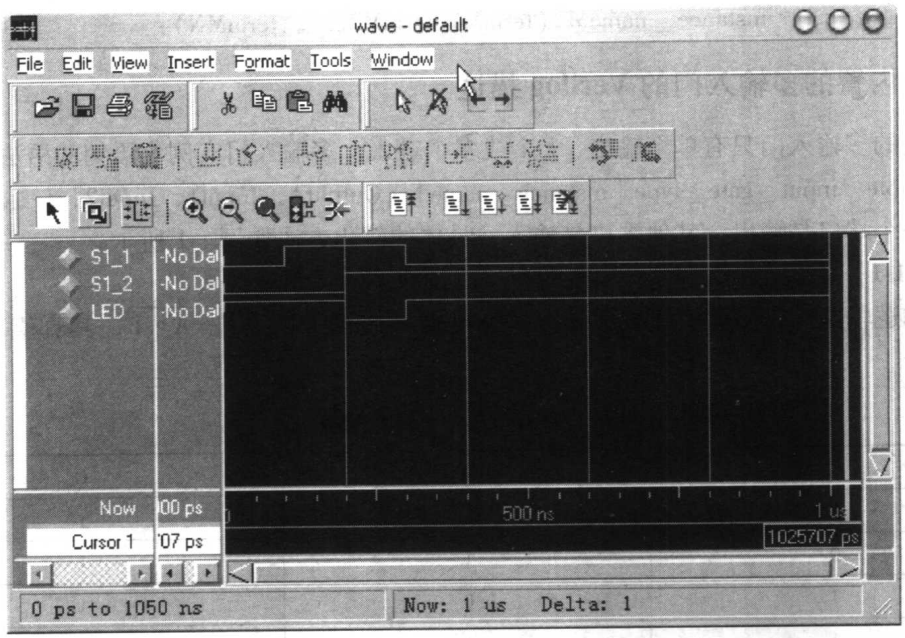

图 1-2　与非门模块的仿真结果

从上面的仿真图形可以看出，当输入端分别为 0、0 时，输出为 1；输入为 1、0 时，输出为 1；当输出为 1、1 时，输出为 0。这完全符合与非门的逻辑。

【例 1-2】　异或门例化程序。

```
'timescale 1ns / 1ps
module S21 (
    Out1,
    In1,
    In2,
    Adder_1_out,
    Adder_1_in_1,
    Adder_1_in_2,
    Adder_1_in_3,
    Adder_1_in_4,
    xorer_1_out,
    xorer_1_in,
    xorer_1_in,
    xorer_1_in,
```

```
    xorer_2_out,
    xorer_2_in,
    xorer_2_in,
    xorer_3_out,
    xorer_3_in,
    xorer_3_in,
    xorer_3_in,
    xorer_3_in
);
input In1;
input In2;
input Adder_1_in_1;
input Adder_1_in_2;
input Adder_1_in_3;
input Adder_1_in_4;
input [2:0]xorer_1_in;
input [1:0]xorer_2_in;
input [3:0]xorer_3_in;
output Adder_1_out;
output xorer_1_out;
output xorer_2_out;
output xorer_3_out;
output Out1;
and Adder(Out1, In1, In2);
and Adder_1(Adder_1_out,Adder_1_in_1, Adder_1_in_2,Adder_1_in_3, Adder_1_in_4);
xor(xorer_1_out, xorer_1_in[0], xorer_1_in[1], xorer_1_in[2]);
xor(xorer_2_out, xorer_2_in[0],xorer_2_in[1]);
xor(xorer_3_out, xorer_3_in[2],xorer_3_in[1],xorer_3_in[0],xorer_3_in[3]);
endmodule
```

例 1-2 中第 1 个门实例语句是单元名为 Adder、输出为 Out1 并带有两个输入 In1 和 In2 的两输入与门。第 2 个门实例语句是四输入与门，单元名为 Adder_1，输出为 Adder_1_out，4 个输入为 Adder_1_in_1、Adder_1_in_2、Adder_1_in_3 和 Adder_1_in_4。第 3 个门实例语句是异或门的具体实例，没有单元名。它的输出是 xorer_1_out，三个输入分别为 xorer_1_in [0]、xorer_1_in [1] 和 xorer_1_in [2]。同时，这一个实例语句中还有两个相同类型的单元。

例 1-2 的综合结果如图 1-3 所示，其仿真结果如图 1-4 所示。从综合的结果可以看出，上面的 Verilog 语句准确无误地描述了所要实现的硬件电路。

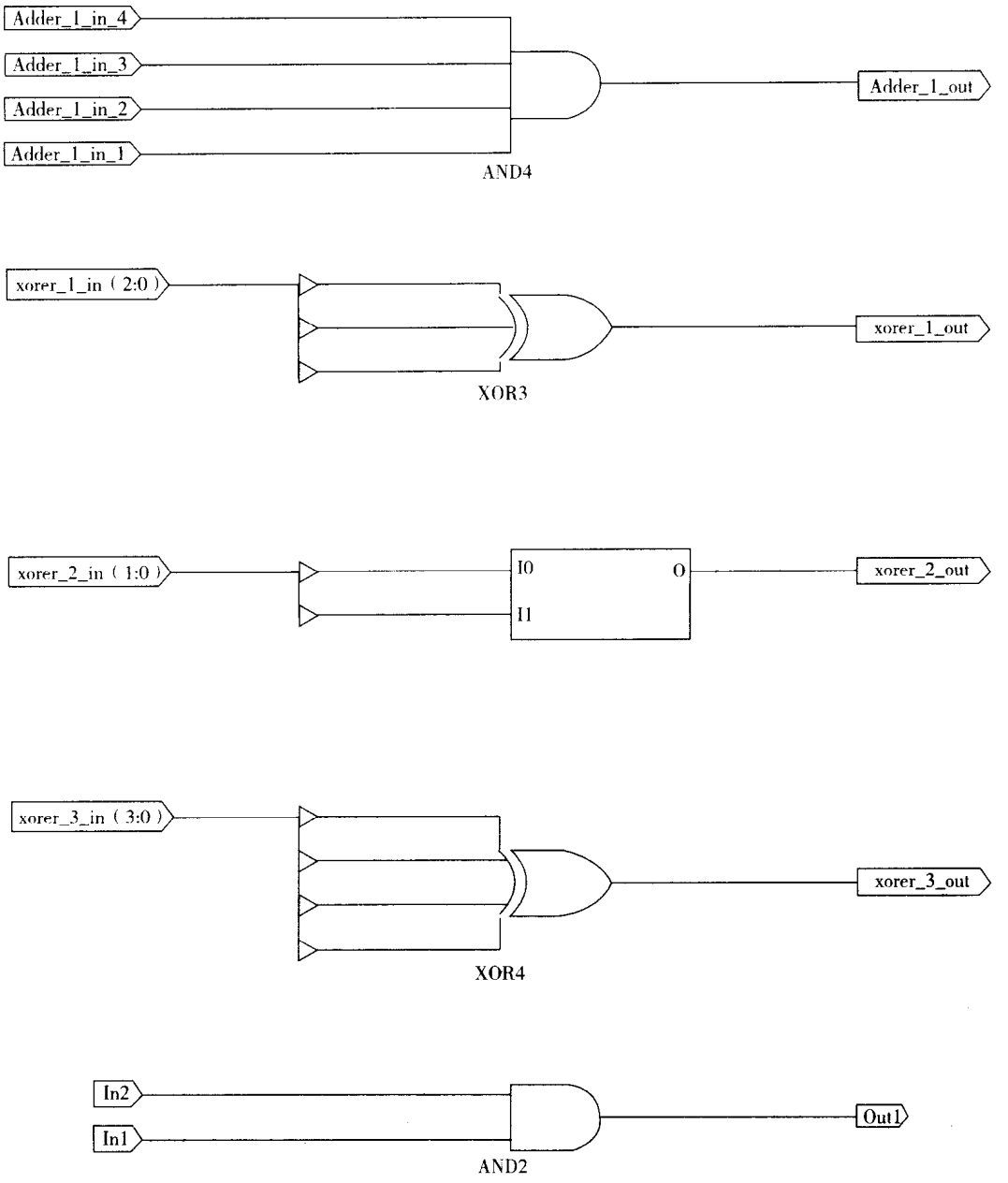

图 1-3　与门和异或门例化程序的综合结果

1.1.2　多输出门的 Verilog 描述

多输出门，顾名思义，是指只有单个输入，一个或多个输出的逻辑门，这些门的实例语句的基本语法如下：

multiple _ output _ gate［实例名］（Out1，Out2，...OutN，InputA）；

前面的几个端口为输出端口，最后的端口是输入端口。这种门的例化与前面的内置的多输入门类似，如例 1-3 所示。图 1-4 所示为异或门例化程序的仿真结果。

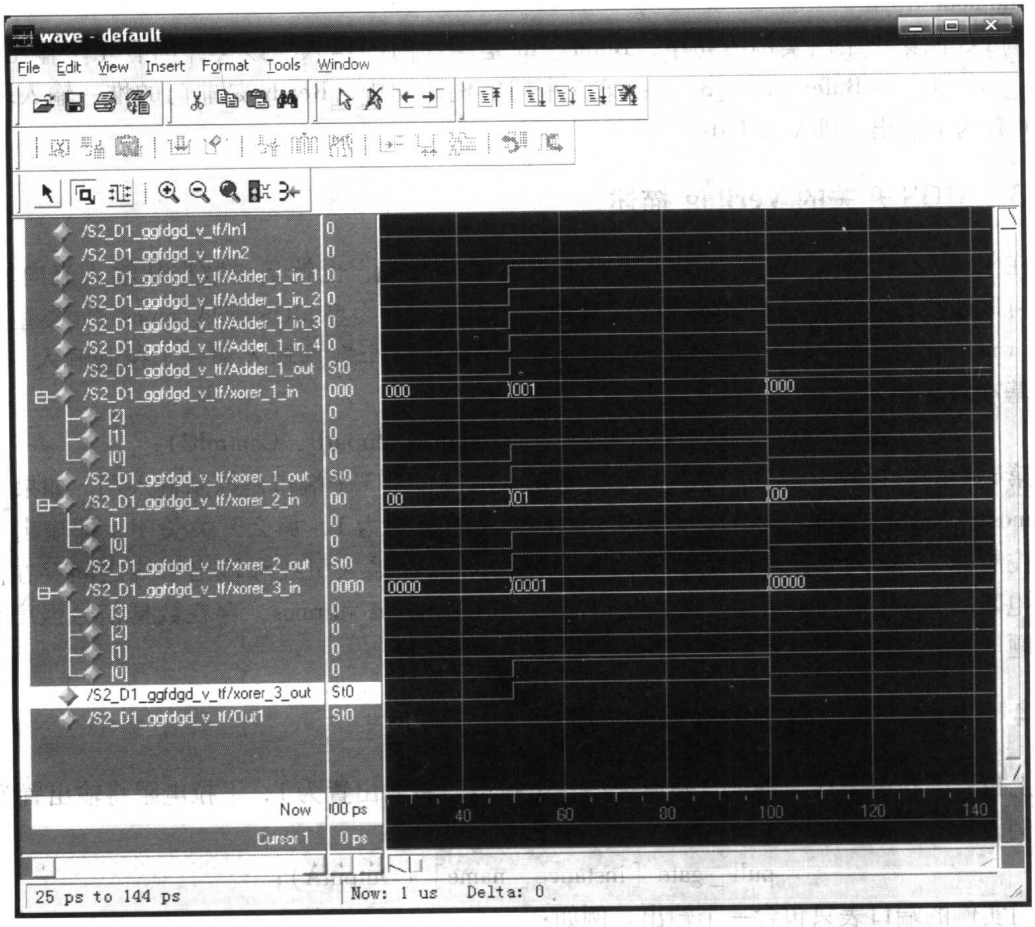

图 1-4　异或门例化程序的仿真结果

【例 1-3】　多输出门 buf。

```
module S21 (
Bufer _ in,
Bufer _ out,
A.
B,
Ready
);
output[3:0]Bufer _ out;
input Bufer _ in;
wire A;
wire B;
output Ready;
buf Bufer(Bufer _ out[0],Bufer _ out[1],Bufer _ out[2],Bufer _ out [3],Bufer _ in);
not N1(A,B,Ready);
```