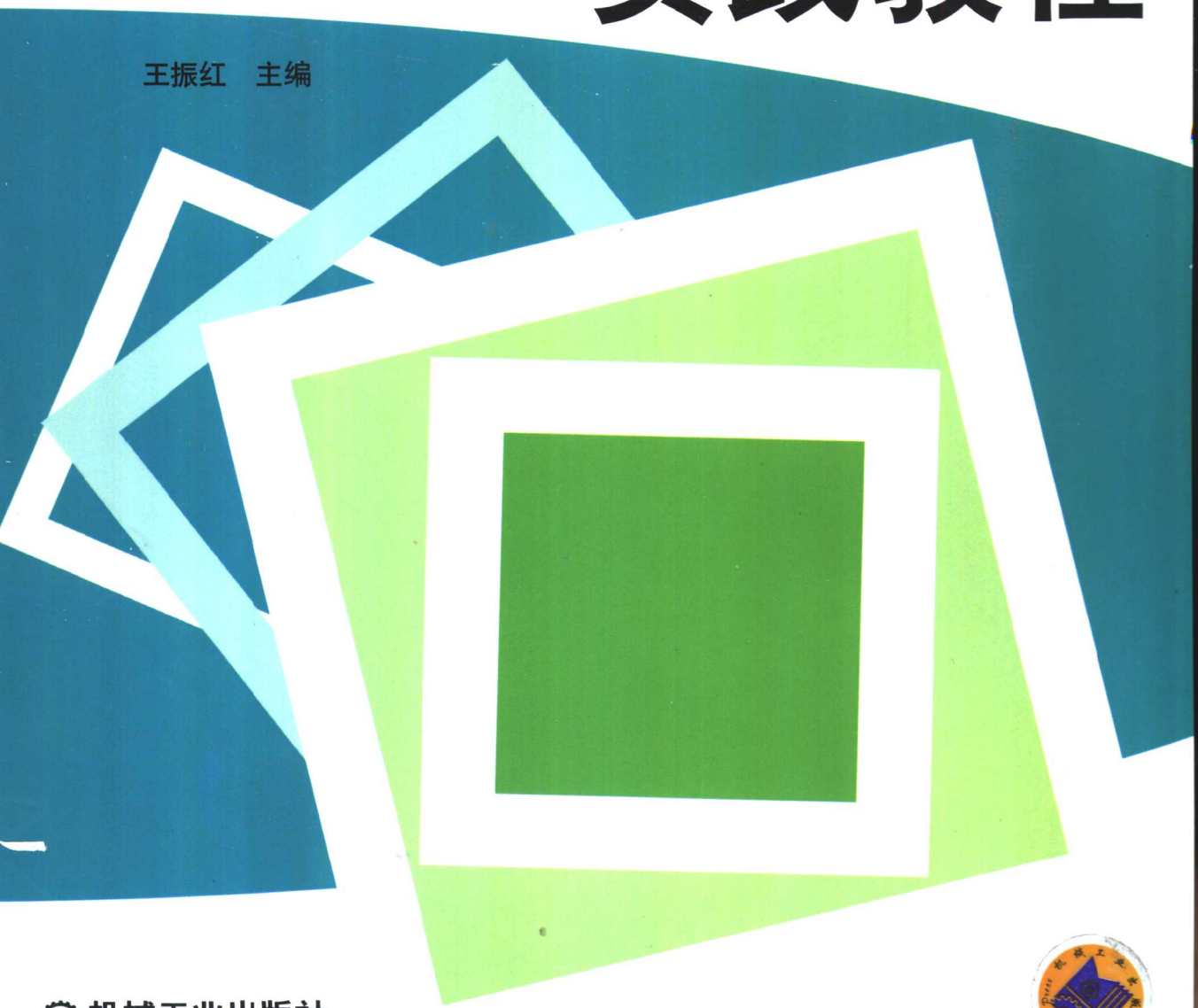
VHDL

数字电路设计与应用实践教程

王振红　主编



机械工业出版社
CHINA MACHINE PRESS

VHDL 数字电路设计与应用实践教程

王振红　　主编

机械工业出版社

本书分为上下两篇。上篇与清华大学阎石主编的《数字电子技术基础》（第4版）同步，内容包括门电路、组合逻辑电路、触发器、时序逻辑电路及存储器，对其中的各种功能芯片以及一些例题，讲解了基于VHDL及可编程逻辑器件的实现方法。下篇与电子课程设计同步，有许多新课题，也有些设计题目选自以往的电子课程设计，但设计方法是不相同的，设计人员可以体会到采用VHDL及可编程逻辑器件设计数字电子电路系统的优越性。

本书可作为大专院校电类学生学习VHDL及可编程逻辑器件的实训教科书，也可供有关工程技术人员参考使用。

图书在版编目（CIP）数据

VHDL数字电路设计与应用实践教程/王振红主编．—北京：机械工业出版社，2003.6

ISBN 7-111-12115-5

Ⅰ.V… Ⅱ.王… Ⅲ.①数字电路－电路设计②硬件描述语言，VHDL－程序设计 Ⅳ.TN79

中国版本图书馆CIP数据核字（2003）第034876号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）

策　　划：胡毓坚

责任编辑：刘　青

责任印制：闫　焱

北京京丰印刷厂印刷·新华书店北京发行所发行

2003年6月第1版·第1次印刷

787mm×1092mm 1/16·12.75印张·314千字

0 001—5 000册

定价：19.00元

序

统计资料表明，当前电子工业在世界范围内已发展成为最大的产业，超过了钢铁和石油工业。超大规模集成电路（VLSI）的飞速发展及电子产品设计上的革命，使电子工业产品尤其是家电产品每半年左右就要更新换代，竞争非常激烈。本书的目的旨在向高校电子信息类本科生推荐这种新的设计方法。

采用国际标准化 VHDL 语言的行为级描述方法正在取代传统的电子线路自下而上的设计方法。本书不同于其他 VHDL 语言的教材，没有讲 VHDL 语言和语法，而是在数字电子技术基础上，把 VHDL 语言的介绍融合到不同的多个抽象层次的设计中，即从门级实现到多个层次的抽象数字设计的建模技术。书中列举了大量设计实例，由简单到复杂，循序渐进，帮助学生学会使用 VHDL 语言，对设计数字电子系统有较系统的理解。有些设计课题还给出多种描述方法，使初学者便于在比较中学习和掌握 VHDL 语言的设计技巧，有些题目则给出了设计方法的提示和程序。大部分题目的设计内容可以扩展，使学生有比较大的思维空间，对提高和发挥学生的创造能力、综合能力和设计能力是很有帮助的，对学好和掌握 VHDL 语言也会有直接的帮助。

本书非常适合电子工程、计算机和自动化等专业的学生使用，也可作为数字电子技术的后续课——VHDL 数字电路设计及相关课程的实验指导书或课程设计教材，并且非常适合作为自学和电子技术比赛的培训教材。

北京工业大学电子信息与控制工程学院　陈文楷

前　言

现在，电子技术的发展非常迅猛，高新技术日新月异。特别是专用集成电路（ASIC）设计技术的日趋进步和完善，推动了数字电路系统的设计和发展，使它从单纯的 ASIC 设计走向了系统设计和单片系统设计。传统的电子技术设计方法，即从单元电路入手到整体电路的设计、“固定功能集成电路+连线”的设计、自下而上的设计，已不能够满足市场的需要。根据系统的功能和行为要求，利用计算机辅助设计自上而下地逐层完成相应的描述，并与大规模可编程器件相结合，使设计出的电路系统速度更快、体积更小、重量更轻、功耗更小、稳定性更高，大大提高了产品的竞争能力。

电子设计自动化（EDA）工具给电子设计带来了巨大变革，特别是硬件描述语言的出现和发展，解决了用传统的电路原理设计大系统工程时的诸多不便，成为电子电路设计人员的最得力助手。其实，早在 20 世纪 80 年代后期，各个 ASIC 研制和生产厂商为了缩短产品开发周期，提高产品在市场上的竞争力，就相继开发了用于各自目的的硬件描述语言，如 ABEL、AHDL 等。但是由于没有统一的标准，这些语言的普及受到了限制。1987 年 12 月，IEEE 对美国国防部开发的超高速集成电路硬件描述语言（Very High Speed Integrate Circuit Hardware Description Language,VHDL）进行了标准化的工作，得到广大用户的一致欢迎。自此以后，VHDL 成了数字电路系统设计的“世界语”。各个 CAD 厂商都努力使自己的电子设计软件与 VHDL 兼容，各高等院校纷纷开设了 VHDL 设计课程，国内也有越来越多的设计人员开始学习和使用 VHDL 进行电路系统的设计。

近年来，可编程逻辑器件的开发生产和销售规模以惊人的速度增长。发展集成电路事业是我国制定的新世纪的重要发展目标，也是经济全球化新形势下的科技挑战。编写本书的目的，是通过大量的设计实例，由浅入深、由简到繁地宣传和推广 VHDL，以提高电子设计领域人员的设计能力。

本书分为上下两篇，上篇是基于 VHDL 的功能芯片设计，下篇是数字电路系统设计。所用的设计手段，包括 ALTERA 公司的 MAX+PLUSII软件工具、VHDL 编程语言、EPM7128SLC84-15 和 EPF10K10LC84-4 可编程器件，本书不做介绍。

本书的所有课题，有些是北方工业大学参加全国大学生电子竞赛训练的题目，有些是毕业设计或课程设计的题目。每个题目从编程、编译、仿真、布局布线和适配，直至配置/下载和硬件测试，都运用了 VHDL 设计方法。

参加本书编写和程序调试的有王仲、赵新建。北方工业大学信息工程学院院长张常年教授担任本书的主审，在认真审阅的同时提出了许多宝贵意见。赵红怡、曹淑琴、刘红、范锦宏、王玉花、张东彦、康晓麓、赵徐森、刘淑敏、周燕平、吴晓林等对本书的编写工作给予了很多关心和支持。在此对他们表示衷心的感谢。

由于作者水平有限，书中难免存在错误和不妥之处，敬请读者批评指正。读者的反馈信息可通过电子邮件发送至：WZH_writer@sohu.com。

作　者

目　录

下 篇

上　篇

第1章　门　电　路

本章介绍与非门、或非门、异或门、反向器、总线缓冲器的 VHDL 设计方法和程序。

1.1　与非门电路

1. 二输入与非门电路

二输入与非门的逻辑方程为 $Y = \overline{AB}$ 。
二输入与非门的电路符号如图 1-1 所示。
二输入与非门的真值表见表 1-1。

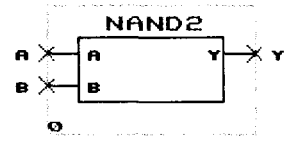


图 1-1　二输入与非门

表 1-1　二输入与非门的真值表

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

源程序：

（1）方法一

```
library ieee;
use ieee.std_logic_1164.all;

entity nand2 is
     port(a,b:in std_logic;
          y:out std_logic);
end nand2;

architecture nand2_1 of nand2 is
   begin
   y<=a nand b;
end nand2_1;
```

（2）方法二

```
library ieee;
use ieee.std_logic_1164.all;

entity nand2 is
        port(a,b:in std_logic;
             y:out std_logic);
 end nand2;

 architecture nand2_2 of nand2 is
  begin
  p1:process(a,b)
      variable comb:std_logic_vector(1 downto 0);
  begin
      comb:=a&b;
       case comb is
         when "00" =>y<='1';
         when "01" =>y<='1';
         when "10" =>y<='1';
         when "11" =>y<='0';
         when others=>y<='X';
       end case;
      end process p1;
   end nand2_2;
```

2. 四输入与非门电路

四输入与非门的电路逻辑方程为 $Y = \overline{ABCD}$ 。
四输入与非门的电路符号如图 1-2 所示。

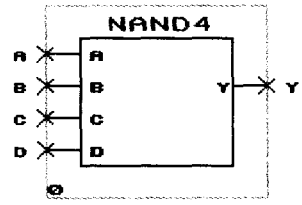


图 1-2　四输入与非门

源程序：

（1）方法一

```
library ieee;
use ieee.std_logic_1164.all;

entity nand4 is
   port(a,b,c,d:in std_logic;
        y:out std_logic);
end nand4;

architecture nand4_1 of nand4 is
   begin
   y<=not(a and b and c and d);
end nand4_1;
```

（2）方法二

```
library ieee;
use ieee.std_logic_1164.all;

entity nand4 is
   port(a,b,c,d:in std_logic;
        y:out std_logic);
end nand4;

architecture nand4_2 of nand4 is
   begin
   p1:process(a,b,c,d)
      variable tmp:std_logic_vector(3 downto 0);
   begin
     tmp:=a&b&c&d;
     case tmp is
          when "0000"=>y<='1';
          when "0001"=>y<='1';
          when "0010"=>y<='1';
          when "0011"=>y<='1';
          when "0100"=>y<='1';
          when "0101"=>y<='1';
          when "0110"=>y<='1';
          when "0111"=>y<='1';
          when "1000"=>y<='1';
          when "1001"=>y<='1';
          when "1010"=>y<='1';
          when "1011"=>y<='1';
          when "1100"=>y<='1';
          when "1101"=>y<='1';
          when "1110"=>y<='1';
          when "1111"=>y<='0';
          when others=>y<='X';
       end case;
    end process;
end nand4_2;
```

（3）方法三

```
library ieee;
use ieee.std_logic_1164.all;

entity nand4 is
   port(a,b,c,d:in std_logic;
        y:out std_logic);
end nand4;

architecture nand4_3 of nand4 is
```

```
    component nand2
    port(a,b:in std_logic;
         y:out std_logic);
    end component;
    component or2
    port(a,b:in std_logic;
         y:out std_logic);
    end component;
signal yy1,yy2:std_logic;
   begin
      u1:nand2 port map(a,b,yy1);
      u2:nand2 port map(c,d,yy2);
      u3:or2 port map(yy1,yy2,y);
end nand4_3;
```

1.2 二输入或非门电路

二输入或非门的逻辑方程为 $Y=\overline{A+B}$。

二输入或非门的电路符号如图 1-3 所示。

二输入或非门的真值表见表 1-2。

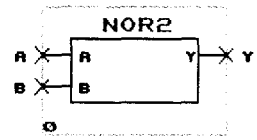


图 1-3 二输入或非门

表 1-2 二输入或非门的真值表

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

源程序：

（1）方法一

```
library ieee;
use ieee.std_logic_1164.all;

entity nor2 is
      port(a,b:in std_logic;
           y:out std_logic);
end nor2;

architecture nor2_1 of nor2 is
begin
   y<=a nor b;
end nor2_1;
```

（2）方法二

```
library ieee;
use ieee.std_logic_1164.all;

entity nor2 is
        port(a,b:in std_logic;
             y:out std_logic);
end nor2;

architecture nor2_2 of nor2 is
begin
   p1:process(a,b)
      variable comb:std_logic_vector(1 downto 0);
   begin
      comb:=a&b;
      case comb is
          when "00"=>y<='1';
          when "01"=>y<='0';
          when "10"=>y<='0';
          when "11"=>y<='0';
          when others=>y<='X';
      end case;
    end process p1;
 end nor2_2;
```

1.3 二输入异或门电路

二输入异或门逻辑方程为 Y = A ⊕ B 。

二输入异或门的电路符号如图 1-4 所示。

二输入异或门的真值表见表 1-3。

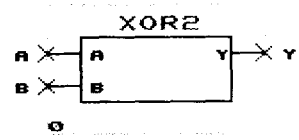


图 1-4　二输入异或门

表 1-3　二输入异或门的真值表

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

源程序：

（1）方法一

```
library ieee;
```

```
use ieee.std_logic_1164.all;

entity xor2 is
      port(a,b:in std_logic;
           y:out std_logic);
end xor2;

architecture xor2_1 of xor2 is
begin
  y<=a xor b;
end xor2_1;
```

（2）方法二

```
library ieee;
use ieee.std_logic_1164.all;

entity xor2 is
      port(a,b:in std_logic;
           y:out std_logic);
end xor2;

architecture xor2_2 of xor2 is
begin
  p:process(a,b)
    variable comb:std_logic_vector(1 downto 0);
  begin
    comb:=a&b;
    case comb is
         when "00"=>y<='0';
         when "01"=>y<='1';
         when "10"=>y<='1';
         when "11"=>y<='0';
         when others=>y<='X';
    end case;
  end process;
end xor2_2;
```

1.4 反向器门电路

反向器的逻辑方程为 $Y = \overline{A}$ 。

反向器的逻辑符号如图 1-5 所示。

反向器的真值表见表 1-4。

图 1-5 反向器

表 1-4　反向器的真值表

A	Y
0	1
1	0

源程序：

（1）方法一

```
library ieee;
use ieee.std_logic_1164.all;

entity inv is
        port(a:in std_logic;
             y:out std_logic);
end inv;

architecture inv_1 of inv is
begin
   y<=not a;
end inv_1;
```

（2）方法二

```
library ieee;
use ieee.std_logic_1164.all;

entity inv is
        port(a:in std_logic;
              y:out std_logic);
end inv;

architecture inv_2 of inv is
begin
   p:process
    begin
     if a='1' then
      y<='0';
      else
      y<='1';
     end if;
   end process p;
end inv_2;
```

1.5 三态门

三态门的电路符号如图 1-6 所示。

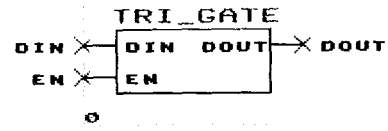


图 1-6 三态门

源程序：

```
library ieee;
use ieee.std_logic_1164.all;

entity tri_gate is
port(din,en:in std_logic;
     dout:out std_logic);
end tri_gate;

architecture arch of tri_gate is
  begin
  process(din,en)
   begin
    if en='1' then
     dout<=din;
    else
     dout<='Z';
    end if;
  end process;
end arch;
```

1.6 单向总线缓冲器

单向总线缓冲器的管脚图如图 1-7 所示。

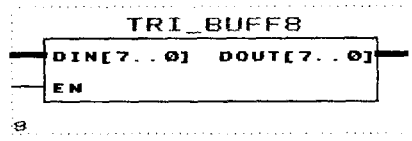


图 1-7 单向总线缓冲器管脚图

源程序：

```
library ieee;
use ieee.std_logic_1164.all;

entity tri_buff8 is
      port(din:in std_logic_vector(7 downto 0);
           dout:out std_logic_vector(7 downto 0);
           en:in std_logic);
end tri_buff8;

architecture behav of tri_buff8 is
begin
  p:process(en,din)
```

```
    begin
      if en='1' then
        dout<=din;
      else
        dout<="ZZZZZZZZ";
      end if;
  end process;
end behav;
```

1.7 双向总线缓冲器

双向总线缓冲器的管脚图如图 1-8 所示。

源程序：

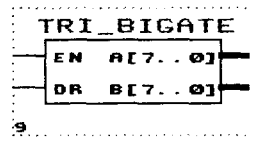


图 1-8 双向总线缓冲器管脚图

```
library ieee;
use ieee.std_logic_1164.all;

entity tri_bigate is
      port(a,b:inout std_logic_vector(7 downto 0);
           en:in std_logic;
           dr:in std_logic);
end tri_bigate;

architecture rtl of tri_bigate is
      signal aout,bout:std_logic_vector(7 downto 0);
begin
     process(a,dr,en)
      begin
       if (en='0') and (dr='1') then
          bout<=a;
       else
          bout<="ZZZZZZZZ";
       end if;
       b<=bout;
     end process;
     process(b,dr,en)
      begin
       if (en='0') and (dr='0') then
          aout<=b;
       else
          aout<="ZZZZZZZZ";
       end if;
      a<=aout;
     end process;
end rtl;
```

第 2 章　组合逻辑电路

本章介绍了常用的编码器、译码器、数据选择器、数值比较器、加法器等器件的编程方法。

2.1　监视交通信号灯工作状态的逻辑电路

取红、黄、绿三盏灯的状态为输入状态，分别用 R、A、G 表示，并规定灯亮时为 1，不亮时为 0。取故障信号为输出变量，用 Z 表示，并规定正常状态下为 0，发生故障时为 1。状态表见表 2-1，管脚图如图 2-1 所示。

表 2-1　交通信号灯工作状态表

R	A	G	Z
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

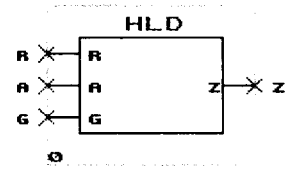


图 2-1　管脚图

源程序：

```
library ieee;
use ieee.std_logic_1164.all;

entity hld is
   port(r,a,g:in std_logic;
        z:out std_logic);
end hld;

architecture rtl of hld is
begin
     process(r,a,g)
     variable comb:std_logic_vector(2 downto 0);
     begin
       comb:=r&a&g;
       case comb is
         when"000"=>z<='1';
```