

工科高等院校电路系列课程教材

数字电路EDA 技术入门

延 明 张亦华 编著



北京邮电大学出版社
www.buptpress.com

数字电路 EDA 技术入门

延 明 张亦华 编 著

北京邮电大学出版社

·北 京·

内 容 简 介

本书是学习数字电路电子设计自动化(EDA)技术的入门工具书。书中通过大量实例介绍了用 VHDL 和电路图输入法设计组合电路、时序电路、数字系统的基本方法和技巧。书中提供的实例均在 MAX + plus II 或 Quartus II 软件平台上进行过编译、仿真,用可编程逻辑器件 EPM7128S 或 FLEX10K 进行过下载、实际测量验证;其中大部分实例都是进行数字电路设计时常用的电路,均可直接被调用,亦可作为进行二次电路开发设计的参考。

本书可作为大学本科和专科院校通信、电子工程类专业的 EDA 实验教材,也可供从事 EDA 工作的技术人员作程序开发的参考书。

本书附有光盘,内容包括学习 VHDL 的一些资料与工具,供读者参考、研习、应用。

图书在版编目(CIP)数据

数字电路 EDA 技术入门/延明,张亦华编著. —北京:北京邮电大学出版社,2005

ISBN 7-5635-1207-1

I. 数... II. ①延... ②张... III. ①数字电路—电路设计:计算机辅助设计②硬件描述语言,VHDL—程序设计 IV. TN790.2

中国版本图书馆 CIP 数据核字(2005)第 161298 号

出 版 者:北京邮电大学出版社(北京市海淀区西土城路 10 号) 邮编:100876

发行部电话:(010)62282185 62283578(传真)

电子信箱:publish@bupt.edu.cn

经 销:各地新华书店

印 刷:北京通州皇家印刷厂

开 本:787 mm × 1 092 mm 1/16

印 张:14.25

字 数:371 千字

印 数:1—3 000 册

版 次:2006 年 1 月第 1 版 2006 年 1 月第 1 次印刷

ISBN 7-5635-1207-1/TN·434

定价:23.00 元(附光盘)

·如有印装质量问题,请与北京邮电大学出版社发行部联系·

前 言

电子设计自动化(EDA)工具给电子设计带来了巨大的变革,特别是可编程逻辑器件和硬件描述语言的出现和发展,解决了用传统的方法设计大系统工程时的诸多不便,成为电子电路设计人员最得力的助手。

电子设计的飞速发展,推动了数字电路系统的设计和发展。利用计算机辅助设计自上而下地逐层完成相应的描述,并与大规模可编程逻辑器件相结合,可以使设计出的电路系统速度快、体积小、功耗小、稳定性高。

本书用 VHDL 编写的实例是在多年教学、科研的基础上开发整理出来的,所有的实例都在 MAX + plus II 或 Quartus II 软件平台上进行过编译、仿真,同时用可编程逻辑器件 EPM7128S 或 FLEX10K 等进行过下载及实际测量验证。

学习 VHDL 应遵循边学边练的原则,其语法规则只有在实践中才能完全领悟、掌握。本书用 VHDL 设计了各种基本组合电路、时序电路、数字系统(或综合电路),并对传统的电路图输入法进行了概括介绍。本书(含光盘)以大量实例介绍了设计组合电路和时序电路的基本方法,供读者参考、研习、应用。

本书的出版得益于北京邮电大学电子工程学院电路中心各个时期从事 EDA 教学工作的老师们的教学成果,没有他们的宝贵经验,也就没有本书的顺利出版。本书在准备、编写过程中得到了学校各级领导、教务处、电子工程学院、电路中心广大教师和工程技术人员的热情支持和帮助,在此向他们表达诚挚的谢意,特别向安德宁、徐惠民、白中英、赵廷瑞、赵振纲教授,以及陈凌霄、袁东明、史晓东、魏学军、张咏梅、赵文深、张晓磊、高英、孙丹丹、任维政、高惠平、柴淑玲、赵同刚、崔岩松、龚新华、姬红强、陈玉波、曹静、王勤、肖冰、郭莉、袁宝库等老师的无私支持和帮助表示衷心的感谢。

本书的编写是我们从事 EDA 教学工作的一个粗浅体会,真诚的希望广大读者能对书中存在的问题提出宝贵意见,恳请同行不吝赐教。

作者

2005 年 10 月

目 录

第 1 章 用 VHDL 设计数字逻辑电路初步

1.1 VHDL 程序的基本模型结构	1
1.2 描述多个电路的基本模型结构	2
1.3 VHDL 的描述风格	6
1.3.1 行为描述方式	6
1.3.2 数据流描述	7
1.3.3 结构描述	8

第 2 章 用 VHDL 设计组合逻辑电路

2.1 用 VHDL 设计组合逻辑电路的几种方法	10
2.1.1 用 VHDL 的逻辑表达式设计组合电路	10
2.1.2 用 VHDL 的算术表达式设计组合逻辑电路	11
2.1.3 利用真值表设计组合逻辑电路	12
2.2 基本门电路	14
2.2.1 二输入同或门电路	14
2.2.2 三态门控制电路	15
2.2.3 单向总线缓冲器	16
2.2.4 用 oc 门实现线与功能	18
2.3 组合逻辑电路	19
2.3.1 编码器	19
2.3.2 优先级编码器	20
2.3.3 码制变换译码器	22
2.3.4 二-十进制 BCD 译码器	24
2.3.5 6 位二进制码-十进制 BCD 码译码器	25
2.3.6 可控代码转换器	27
2.3.7 变量译码器	28
2.3.8 显示译码器	31
2.3.9 多路数据选择器	32
2.3.10 半加器	34
2.3.11 全加器	36
2.3.12 多位数值比较器	38
2.3.13 奇校验电路	39
2.3.14 求补运算器	41

第3章 用 VHDL 设计时序电路

3.1 用 VHDL 描述时钟信号的几种方法	42
3.1.1 时钟信号出现在敏感信号表中	42
3.1.2 时钟信号不出现在敏感信号表中	45
3.2 用 VHDL 描述时序电路中复位信号的方法	46
3.2.1 用 VHDL 描述时序电路中同步复位信号的方法	46
3.2.2 用 VHDL 描述时序电路中异步复位信号的方法	48
3.3 触发器	50
3.3.1 RS 触发器	50
3.3.2 带异步复位置位功能的 D 触发器	51
3.3.3 T 触发器	52
3.3.4 带异步复位置位功能的 JK 触发器	54
3.3.5 8 位锁存器	55
3.4 用 VHDL 设计计数器	56
3.4.1 4 位二进制可逆计数器	56
3.4.2 二进制异步计数器	59
3.4.3 中规模同步计数器 74LS169	61
3.4.4 BCD 码 24 进同步计数器	63
3.4.5 BCD 码 24 进异步计数器	64
3.4.6 初始值可控计数器	66
3.4.7 模值可控计数器	67
3.5 分频器	69
3.5.1 用 VHDL 设计分频器	69
3.5.2 用 VHDL 设计 2^n 分频器	71
3.5.3 2 分频器	72
3.5.4 奇数分频器	73
3.5.5 数控分频器	76
3.6 用 VHDL 描述几种常用的同步时序电路	77
3.6.1 寄存器	77
3.6.2 4 位环形计数器	78
3.6.3 扭环型计数器	81
3.6.4 序列信号发生器	83
3.6.5 m 序列信号发生器	84
3.6.6 脉冲分配器	86
3.6.7 串入串出移位寄存器	88
3.6.8 串入串出双向移位寄存器	90
3.6.9 串入并出移位寄存器	92
3.6.10 并入串出移位寄存器	94

第4章 用电路图输入法设计数字电路

实例 4-1	TFFE 触发器	96
实例 4-2	冒险电路	97
实例 4-3	74169 计数器的应用	98
实例 4-4	与门阵列控制器	99
实例 4-5	参数型宏功能与门	100
实例 4-6	序列发生器	101

第5章 资源调用与特色电路

5.1	资源调用	104
实例 5-1	不用调库令调用自制器件	104
实例 5-2	使用调库、调包令调用程序包中的自制器件	105
实例 5-3	使用调库、调包令调用程序包中定义的函数	107
实例 5-4	使用调库、调包命令调用软件包中定义的子程序	108
实例 5-5	为支持不同数据类型进行运算,调用库系统程序包	110
实例 5-6	在结构体内定义一个子程序(过程)	111
实例 5-7	调用 Altera 公司的库元件 DFF(D 触发器)和 74151b(选择器)	112
5.2	特色电路	112
实例 5-8	计数器型防抖动电路(1)	112
实例 5-9	计数器型防抖动电路(2)	114
实例 5-10	采样型防抖动微分电路	117
实例 5-11	积分分频器	118
实例 5-12	4×4 键盘输入电路	120
实例 5-13	串行偶校验器	126

第6章 用 VHDL 设计数字系统

6.1	函数发生器	128
6.2	正弦波发生器	136
6.3	简易电子琴	148
6.4	带数字显示的电容测量仪	162
6.5	单次脉冲发生器及单次脉冲测试仪	169
6.6	LED 写字板	177
6.7	LCD 字符显示器	179
6.8	乒乓游戏机	189
6.9	自动售货机	199
6.10	三层电梯控制器	207

参考文献	219
------------	-----

第 1 章 用 VHDL 设计数字逻辑电路初步

本书实例均在 MAX + plus II 软件平台上进行过编译、仿真验证。因 MAX + plus II 软件要求源程序文件的名字与实体名必须一致,为了使同一个 VHDL 源程序文件能适应各个 EDA 开发软件的使用要求,本书源程序文件的命名均与实体名一致。

1.1 VHDL 程序的基本模型结构

VHDL 程序的基本结构就是一个设计实体。一个完整的设计实体包括库(Library)、程序包(Package)、配置(Configuration)、实体(Entity)和结构体(Architecture)5 个部分,后 4 项是待编译的源设计单元。

库里存放已编译的实体、结构体、程序包和配置;程序包存放各设计模块都能共享的数据类型、常数和子程序等;配置是用于选取库中所需单元来组成系统设计的不同版本;实体是描述设计实体的外部接口信号;结构体用于描述系统内部的电路结构和行为。

无论是简单的还是复杂的数字电路,都可以用一个设计实体来描述。一个设计实体最基本的组成部分是实体和结构体。下面通过一个例子来说明 VHDL 语言最基本的结构。

1. 设计要求

用 VHDL 设计一个二输入或非门电路,令该二输入或非门电路的输入信号为 a1、a2,输出信号为 y。

2. 源程序

```
entity or2 is
    port (a1,a2 : in bit;
          y      : out bit);
end or2;
-- 以上部分为实体部分。
```

-- 以下部分为结构体部分。

```
architecture or_2 of or2 is
begin
    y<= a1 nor a2;
end or_2;
```

3. 程序说明

(1) 这是一个比较简单的程序,它只有两部分,即实体和结构体,也就是说再简单的设计,这两部分也必不可少。

(2) 从 entity 到 end or2 是实体部分,主要描述了器件对外的端口,即端口的名称和数据类型。

① or2 是实体名。实体中说明了二输入或非门有两个输入端 a1 和 a2, 一个输出端 y。

② bit 是输入信号的数据类型, 即“二值逻辑”数据类型, 这种类型的数据只有'0'和'1'两种逻辑。bit 是 VHDL 本身定义的数据类型, 使用 bit 数据类型可以省略调库语句。

(3) 从 architecture 到 end or_2 是结构体部分, or_2 是结构体名, 结构体主要用于描述元件内部各个逻辑器件和功能部件的连接关系; nor 是 VHDL 的一个运算符, 表示“或非”操作; “<=”是赋值运算符。结构体部分通过表达式“y<= a1 nor a2”表明了输入信号 a1 和 a2“或非”操作后的结果传送到输出端 y。这样一个二输入的或非门不论是从输入、输出的端口方面, 还是内部特性都得到了充分的说明。VHDL 的编程软件可以使这样的描述经过转换得到硬件电路的结构。

(4) VHDL 的每条语句是以英文分号“;”结束。

(5) VHDL 的每条注释语句是以两个连续英文短划线“--”开始。注释语句不会被编译。

1.2 描述多个电路的基本模型结构

从上面的例子可以看到, 一个设计实体只描述了一个门电路, 下面介绍如何在一个设计实体中描述多个电路。

1. 设计要求

用 VHDL 设计一个电路, 要求电路具有如下逻辑功能:

(1) 分频器, 能够输出一个 2 分频和 4 分频的信号。令该分频器的时钟信号为 clk, 输出信号为 q2、q4。

(2) 二输入异或门电路, 令该二输入异或门电路的输入信号为 a1、a2, 输出信号为 y。

(3) 二选一数据选择器, 令该二选一数据选择器的输入信号为 b1、b2, 选择信号为 sel, 输出信号为 f。

2. 源程序 1

library ieee; -- 调用 ieee 库语句。

use ieee.std_logic_1164.all; -- 调用程序包语句。

use ieee.std_logic_signed.all;

entity xitong1 is

```
    port(clk          : in std_logic; -- 分频器的端口描述。
          q2 , q4      : out std_logic;
          a1 , a2      : in std_logic; -- 异或门的端口描述。
          y            : out std_logic;
          b1 , b2 , sel : in std_logic; -- 数据选择器的端口描述。
          f            : out std_logic);
```

end ;

architecture timer of xitong1 is -- 开始描述结构体。

```
    signal tout : std_logic_vector ( 3 downto 0 );
```

```

begin
p1 :
  process ( a1 , a2 ) -- 用一个进程语句描述异或门。
    variable com : std_logic_vector ( 1 downto 0 );
  begin
    com := a1 & a2 ;
    case com 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;
p2 :
  process (clk) -- 用一个进程语句描述分频器。
  begin
    if (clk'event and clk = '1') then
      if tout = "1111" then
        tout <= ( others => '0' );
      else
        tout <= tout + 1;
      end if ;
    end if;
    q2 <= tout(0);
    q4 <= tout(1);
  end process p2 ;
p3 :
  process ( b1 , b2 , sel ) -- 用一个进程语句描述选择器。
  begin
    if ( sel = '1' ) then
      f <= b1;
    else
      f <= b2;
    end if ;
  end process p3;
end timer;

```

3. 程序说明

- (1) library ieee 是调用 IEEE 库。
- (2) use ieee.std_logic_1164.all; 是调用 IEEE 库中的程序包: use ieee.std_logic_signed.all。

(3) 程序中输入、输出信号的数据类型用的是枚举类型 std_logic。在 IEEE 库 std_logic_1164 程序包中,所定义的枚举类型 std_logic 有 9 种取值。

(4) 在 VHDL 语言中可利用多个结构体来描述多个独立电路,但是在编译时,必须用语言指明一个结构体供软件系统编译使用。若是用 MAX + plus II 软件进行编译,则该软件将自动选择排列到最后的结构体作为当前结构体进行编译。

(5) 本例采用的是一个在结构体中有多个进程语句(process)的方法。进程语句是一种并行处理语句,进程和进程之间是并行运行的,可通过全局信号进行进程之间的通信,但进程中的所有语句都是顺序执行。

(6) 进程 p1 描述了一个二输入异或门电路;进程 p2 描述了一个分频器;进程 p3 描述了一个二选一数据选择器。

(7) 一个进程语句可以描述一个独立的电路,而一个电路也可以用多个进程语句来描述。

下面介绍用两个进程语句来完成分频器的描述,并将其输出信号作为二输入异或门电路的输入信号。

4. 源程序 2

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

entity xitong2 is
    port (clk          : in std_logic;
          q2,q4        : out std_logic;
          y            : out std_logic;
          b1 , b2 , sel : in std_logic;
          f            : out std_logic );
end;

architecture timer of xitong2 is
    signal  tout : std_logic_vector ( 3 downto 0 );
    signal cp1,cp2 : std_logic;
    signal  a1,a2 : std_logic;
begin
    p1 :
        process ( a1 , a2 ) --p1 进程用于描述一个异或门。
            variable com : std_logic_vector ( 1 downto 0 );
        begin
            a1 <= cp1;
            a2 <= cp2;
            com := a1 & a2;
            case com 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;

p2:
process(clk) -- p2 进程用于描述一个分频器。
begin
    if ( clk'event and clk = '1' ) then
        if tout = "1111" then
            tout <= ( others => '0' );
        else
            tout <= tout + 1;
        end if ;
    end if ;
    cp1 <= tout(0);
end process p2;

p3 :
process(cp1) -- p3 进程用于描述一个 2 分频电路。
begin
    if ( cp1'event and cp1 = '1' ) then
        cp2 <= not cp2;
    end if ;
    q4 <= cp2;
    q2 <= cp1;
end process;

p4 :
process ( b1 , b2 , sel ) -- p4 进程用于描述一个选择器。
begin
    if ( sel = '1' ) then
        f <= b1;
    else
        f <= b2;
    end if ;
end process;

end timer;

```

可以看出用两个进程 p2、p3 完成了分频器的功能,其输出的 cp1、cp2 的信号作为异或门电路 a1、a2 的输入。

1.3 VHDL 的描述风格

在 VHDL 中,有三种描述方式(或者说建模方式),即行为描述方式、数据流描述方式、结构描述方式。VHDL 可以通过这三种描述方式,从不同的侧面来描述数字电路和系统,但在实际应用中,为了兼顾整个设计的功能、资源和性能的因素,通常混合使用这三种描述方式。

1.3.1 行为描述方式

1. 设计要求

设计一个四人表决器,设表决信号分别为 a1、a2、a3、a4。信号值为'1'表示赞成,信号值为'0'表示反对。设表决结果为 y,如多数赞成,y 值为'1';反之,y 值为'0'。

表 1.1 四人表决器的真值表

a1	a2	a3	a4	y	a1	a2	a3	a4	y
0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0	1	0
0	0	1	0	0	1	0	1	0	0
0	0	1	1	0	1	0	1	1	1
0	1	0	0	0	1	1	0	0	0
0	1	0	1	0	1	1	0	1	1
0	1	1	0	0	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1

2. 源程序

```
entity vote4_1 is
    port ( a1,a2,a3,a4 : in bit;
           y           : out bit );
end ;

architecture behave of vote4_1 is
begin
    process
        variable comb : bit_vector (3 downto 0);
    begin
        comb := a4 & a3 & a2 & a1;
        case comb is
            when "0000" => y <= '0';
            when "0001" => y <= '0';
            when "0010" => y <= '0';
            when "0011" => y <= '0';
            when "0100" => y <= '0';
            when "0101" => y <= '0';
```

```

when "0110" => y <= '0';
when "0111" => y <= '1';
when "1000" => y <= '0';
when "1001" => y <= '0';
when "1010" => y <= '0';
when "1011" => y <= '1';
when "1100" => y <= '0';
when "1101" => y <= '1';
when "1110" => y <= '1';
when "1111" => y <= '1';
when others => null;

end case;

end process;

end;

```

3. 程序说明

- (1) 用一个进程语句来描述电路输入、输出之间的关系,不涉及具体电路的结构。
- (2) 从四人表决器的功能仿真波形图 1.1 可以看出,当 $a_4a_3a_2a_1 = "1011"$ 时,表示赞成成为多数,输出 $y = '1'$ 。

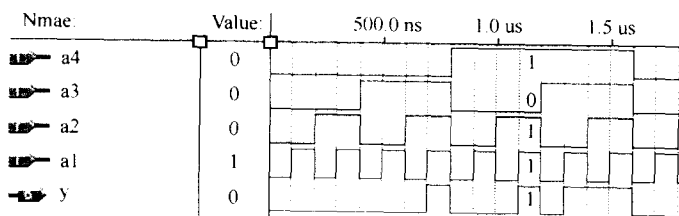


图 1.1 四人表决器仿真波形

1.3.2 数据流描述

1. 设计要求

设计一个四人表决器,设表决信号分别为 a_1 、 a_2 、 a_3 、 a_4 。信号值为 '1',表示赞成;信号值为 '0'表示反对。设表决结果为 y ,如多数赞成, y 值为 '1';反之, y 值为 '0'。

逻辑表达式

$$y = a_1a_2a_3 + a_1a_2a_4 + a_1a_3a_4 + a_2a_3a_4$$

2. 源程序

```

entity vote4_2 is
    port (a1,a2,a3,a4 : in bit;
          y           : out bit );
end;

```

```

architecture behave of vote4_2 is

```

```
begin
```

```
    y <= (a1 and a2 and a3) or (a1 and a2 and a4) or (a1 and a3 and a4) or (a2 and a3 and a4);
end;
```

3. 程序说明

(1) 数据流描述是一种非结构化并行语句的描述,它包含一定的结构信息,又隐含某种逻辑功能行为。通常用并行赋值语句来描述。

(2) 在设计中只要有布尔表达式就很容易将它转换成用 VHDL 描述的表达式,即将布尔表达式中代表的逻辑运算符转换成 VHDL 中的逻辑运算符。

例如:本例中的或“+”换成 or;等号“=”换成赋值运算符“<=”。

1.3.3 结构描述

1. 设计要求

设计一个四人表决器,设表决信号分别为 a1、a2、a3、a4。信号值为‘1’,表示赞成;信号值为‘0’,表示反对。设表决结果为 y,如多数赞成,y 值为‘1’;反之,y 值为‘0’。

2. 顶层源程序

```
entity vote4_3 is
```

```
    port (a1 , a2 , a3 , a4 : in bit;
          y                : out bit);
end;
```

```
architecture behave of vote4_3 is
```

```
    component and3_1
```

```
        port( a,b,c : in bit;
              y1    : out bit);
    end component;
```

```
    component or4_1
```

```
        port( a,b,c,d : in bit;
              y2      : out bit);
    end component;
```

```
    signal temp1 , temp2 , temp3 , temp4 : bit;
```

```
begin
```

```
    u1 : and3_1 port map (a1,a2,a3,temp1);
    u2 : and3_1 port map (a1,a2,a4,temp2);
    u3 : and3_1 port map (a1,a3,a4,temp3);
    u4 : and3_1 port map (a2,a3,a4,temp4);
    u5 : or4_1 port map (temp1,temp2,temp3,temp4,y);
```

```
end;
```

3. 程序说明

(1) 结构描述方式,就是描述组成设计实体的各电路的逻辑结构及它们的电路连接结构。

其描述风格更接近实际硬件系统。

(2) 在结构描述方式中,元件例化语句是基本描述语句。元件例化语句由两部分组成,第一部分是调用一个已设计的实体(元件);第二部分是此元件与当前设计实体的端口映射说明。

(3) 用 component 语句调用已有的元件。

(4) 映射的方法有两种,一种是位置映射;另一种是名称映射。

位置映射:指正在设计中的模块端口的各信号位置排列顺序和被调用的下一层模块的各信号位置排列顺序一一对应。

名称映射:指正在设计中的模块端口的各信号名与被调用的下一层元件端口的各信号名称相同。

本例采用的是位置映射,例如,下层四输入或门电路中有四个输入端 a、b、c、d,一个输出端 y2;在上层模块映射的位置也应是四个输入端 temp1、temp2、temp3、temp4 和一个输出端 y。

(5) 由上述例子可以看出,VHDL 可采用不同语句来描述同一个电路,设计者可根据实际需要来进行电路方案优选。

4. 被调用元件的源程序

(1) 被调用元件 and3_1 的源程序

```
entity and3_1 is
    port (a , b , c : in bit;
          y1      : out bit );
end;
```

```
architecture and3 of and3_1 is
begin
```

```
    y1 <= a and b and c;
end;
```

(2) 被调用元件 or4_1 的源程序

```
entity or4_1 is
    port (a , b , c , d : in bit;
          y2            : out bit );
end;
```

```
architecture or4 of or4_1 is
begin
```

```
    y2 <= a or b or c or d;
end;
```

(3) 被调用元件程序文件的存放

被调用元件的程序均和顶层的源程序存放在同一个文件夹里。

第 2 章 用 VHDL 设计组合逻辑电路

用 VHDL 设计组合逻辑电路,可借助组合逻辑电路的逻辑表达式、算术表达式和真值表来设计。在描述方式上可采用行为描述方式、数据流描述方式和结构描述方式的任何一种,也可以混合使用。

下面用几个实例来说明怎样借助逻辑表达式、算术表达式和真值表设计组合逻辑电路。

2.1 用 VHDL 设计组合逻辑电路的几种方法

2.1.1 用 VHDL 的逻辑表达式设计组合电路

1. 设计要求

用 VHDL 设计一个组合逻辑电路 $y = ab + bc + ac$ 。

2. 源程序

```
entity zuhel is
    port (a , b , c : in bit;
          y          : out bit);
end;

architecture behave of zuhel is
begin
    y <= ( a and b ) or ( b and c ) or ( a and c );
end;
```

3. 程序说明

(1) 本程序采用数据流的描述方法。

(2) “ $y <= (a \text{ and } b) \text{ or } (b \text{ and } c) \text{ or } (a \text{ and } c);$ ”是并行赋值语句。用 VHDL 的逻辑表达式设计一个用布尔方程表达的组组合逻辑电路的方法是,布尔方程中的信号名保留,只要用 VHDL 的相应逻辑运算符代换布尔方程中相应的布尔代数运算符即可。

(3) 输入和输出信号都是“BIT”类型,所以无需调用“IEEE”库。

(4) 功能仿真波形如图 2.1 所示。

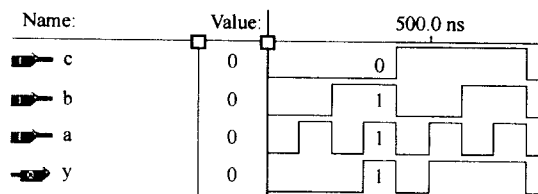


图 2.1 $y = ab + bc + ac$ 组合电路的仿真波形