VHDL
硬件描述语言与数字逻辑电路设计

修订版

——电子工程师必备知识

侯伯亨　顾 新　编著

西安电子科技大学出版社

VHDL 硬件描述语言与数字逻辑电路设计

——电子工程师必备知识

（修订版）

侯伯亨　顾新　编著

西安电子科技大学出版社

2000

内 容 简 介

本书系统地介绍了一种硬件描述语言，即 VHDL 语言设计数字逻辑电路和数字系统的新方法。这是电子电路设计方法上一次革命性的变化，也是迈向 21 世纪的电子工程师所必须掌握的专门知识。本书共分 12 章，第 1 章～第 8 章主要介绍 VHDL 语言的基本知识和使用 VHDL 语言设计简单逻辑电路的基本方法；第 9 章和第 10 章分别以定时器和接口电路设计为例，详述了用 VHDL 语言设计复杂电路的步骤和过程；第 11 章简单介绍了 VHDL 语言 93 版和 87 版的主要区别；第 12 章介绍了 MAX＋plus Ⅱ 的使用说明。

本书以数字逻辑电路设计为主线，用对比手法来说明数字逻辑电路的电原理图和 VHDL 语言程序之间的对应关系，并列举了众多的实例。另外，还对设计中的有关技术，如仿真、综合等作了相应说明。本书简明扼要，易读易懂。它可作为大学本科和研究生的教科书，也可以作为一般从事电子电路设计工程师的自学参考书。

VHDL 硬件描述语言与数字逻辑电路设计
——电子工程师必备知识
（修订版）

侯伯亨　顾新　编著

责任编辑　徐德源
出版发行　西安电子科技大学出版社
（西安市太白南路 2 号）
邮　　编　710071
电　　话　(029)8227828
经　　销　新华书店
印　　刷　陕西省富平印刷有限责任公司
版　　次　1997 年 9 月第 1 版
1999 年 1 月第 2 版
2000 年 5 月第 4 次印刷
开　　本　787 毫米×1092 毫米　1/16　印张 20.75
字　　数　485 千字
印　　数　16 001～22 000 册
定　　价　20.80 元
ISBN 7－5606－0534－6/TP・0264

前 言

随着电子技术的发展，当前数字系统的设计正朝着速度快、容量大、体积小、重量轻的方向发展。推动该潮流迅猛发展的引擎就是日趋进步和完善的 ASIC 设计技术。目前数字系统的设计可以直接面向用户需求，根据系统的行为和功能要求，自上至下地逐层完成相应的描述、综合、优化、仿真与验证，直到生成器件。上述设计过程除了系统行为和功能描述以外，其余所有的设计过程几乎都可以用计算机来自动地完成，也就是说做到了电子设计自动化(EDA)。这样做可以大大地缩短系统的设计周期，以适应当今品种多、批量小的电子市场的需求，提高产品的竞争能力。

电子设计自动化(EDA)的关键技术之一是要求用形式化方法来描述数字系统的硬件电路，即要用所谓硬件描述语言来描述硬件电路。所以硬件描述语言及相关的仿真、综合等技术的研究是当今电子设计自动化领域的一个重要课题。

硬件描述语言的发展至今已有几十年的历史，并已成功地应用到系统的仿真、验证和设计综合等方面。到本世纪 80 年代后期，已出现了上百种的硬件描述语言，它们对设计自动化起到了促进和推动作用。但是，它们大多各自针对特定设计领域，没有统一的标准，从而使一般用户难以使用。广大用户所期盼的是一种面向设计的多层次、多领域且得到一致认同的标准的硬件描述语言。80 年代后期由美国国防部开发的 VHDL 语言(VHSIC Hardware Description Language)恰好满足了上述这样的要求，并在 1987 年 12 月由 IEEE 标准化(定为 IEEE std 1076—1987 标准，1993 年进一步修订，被定为 ANSI/IEEE std 1076—1993 标准)。它的出现为电子设计自动化(EDA)的普及和推广奠定了坚实的基础。据 1991 年有关统计资料表明，VHDL 语言业已被广大设计者所接受，据称已有 90%的设计者使用或即将使用 VHDL 语言来设计数字系统。另外，众多的 CAD 厂商也纷纷使自己新开发的电子设计软件与 VHDL 语言兼容。由此可见，使用 VHDL 语言来设计数字系统是电子设计技术的大势所趋。

作者编写此书的目的就在于向广大电子设计人员介绍 VHDL 语言的基本知识和使用它来设计数字系统硬件电路的方法，从而使读者摆脱传统的人工设计方法的框框，使数字系统设计的水平上升到一个新的阶段。

本书共分 10 章，第 1 章～第 8 章主要介绍 VHDL 语言的基本知识和使用 VHDL 语言设计简单逻辑电路的基本方法。第 9 章和第 10 章分别以定时器和虚拟处理器电路设计为例，详述了如何用 VHDL 语言设计复杂逻辑电路的步骤和过程。全书采用对照说明的方法来叙述逻辑电原理图和 VHDL 语言描述的对应关系，并且安排了大量的程序实例。读者只要通读全书，就可以基本掌握用 VHDL 语言设计一般逻辑电路的方法。由于受条件和环境限制，本书对仿真和综合只作了概略性的介绍，因为这些工具的具体使用方法随各厂商不同而有较大差别。另外，有些 VHDL 语言的程序实例取材于不同版本的资料，本书在编写时虽作了一些统一，但某些书写格式上还会存在个别的差异，敬请读者谅解。

本书在编写过程中引用了诸多学者和专家的著作和论文中的研究成果，在这里向他们表示衷心的感谢。同时，也向一贯热情支持和关心作者的西安电子科技大学出版社的领导和编辑及工作人员表示深深的谢意。

由于作者水平有限，错误和不当之处在所难免，敬请各位读者不吝赐教。

编者

1997 年 2 月 21 日元霄节于西安

第 1 章

数字系统硬件设计概述

自计算机诞生以来，数字系统设计历来存在两个分枝，即系统硬件设计和系统软件设计。同样，设计人员也因工作性质不同，被分成两群：硬件设计人员和软件设计人员。他们各自从事各自的工作，很少涉足对方的领域。特别是软件设计人员更是如此。但是，随着计算机技术的发展和硬件描述语言 HDL(Hardware Description Language)的出现，这种界线已经被打破。数字系统的硬件构成及其行为完全可以用 HDL 语言来描述和仿真。这样，软件设计人员也同样可以借助 HDL 语言，设计出符合要求的硬件系统。不仅如此，利用 HDL 语言来设计系统硬件与利用传统方法设计系统硬件相比，还带来了许多突出的优点。它是硬件设计领域的一次变革，对系统的硬件设计将产生巨大的影响。在本章将详细介绍这种硬件设计方法的变化。

1.1 传统的系统硬件设计方法

在计算机辅助电子系统设计出现以前，人们一直采用传统的硬件电路设计方法来设计系统的硬件。这种硬件设计方法主要有以下几个主要特征。

1.1.1 采用自下至上(Bottom Up)的设计方法

自下至上的硬件电路设计方法的主要步骤是：根据系统对硬件的要求，详细编制技术规格书，并画出系统控制流图；然后根据技术规格书和系统控制流图，对系统的功能进行细化，合理地划分功能模块，并画出系统的功能框图；接着就是进行各功能模块的细化和电路设计；各功能模块电路设计、调试完成后，将各功能模块的硬件电路连接起来再进行系统的调试，最后完成整个系统的硬件设计。

自下至上的设计方法在各功能模块的电路设计中的体现大概最能说明问题。下面以一个六进制计数器设计为例作一说明。

要设计一个六进制计数器，其方案是多种多样的，但是摆在设计者面前的一个首要问题是，如何选择现有的逻辑元、器件构成六进制计数器。那么，设计六进制计数器将首先从选择逻辑元、器件开始。

第一步，选择逻辑元、器件。由数字电路的基本知识可知，可以用与非门，或非门，D 触发器，JK 触发器等基本逻辑元、器件来构成一个计数器。设计者根据电路尽可能简单，

价格合理，购买和使用方便及各自的习惯来选择构成六进制计数器的元、器件。本例中我们选择 JK 触发器和 D 触发器作为构成六进制计数器的主要元、器件。

第二步，进行电路设计。假设六进制计数器采用约翰逊计数器。3 个触发器连接应该产生 8 种状态，现在只使用 6 个状态，将其中的 010 和 101 两种状态禁止掉。这样六进制计数器的状态转移图如图 1－1 所示。

从这个状态转移图可以看到，在计数过程中计数器中的 3 个触发器的状态是这样转移的：首先 3 个触发器状态均为 0，即 $Q_2Q_1Q_0=000$，以后每来一个计数脉冲，其状态变化情况为 000→001→011→111→110→100→000→001→……。

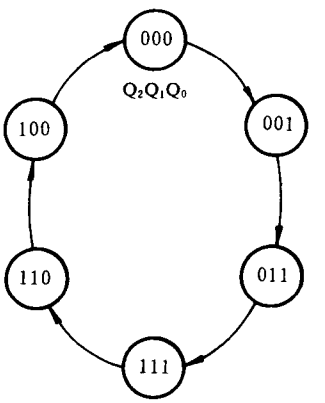


图 1－1　六进制计数器状态转移图

在知道六进制计数器的状态变化规律以后，就可以列出每个触发器的前一个状态和后一个状态变化的状态表，如表 1－1 所示。从表中可以发现，Q_2 当前的输出是 Q_1 前一个状态的输出，而 Q_1 当前的输出就是 Q_0 前一个状态的输出。这样，如 Q_2 和 Q_1 采用 D 触发器，只要将 Q_0 输出与 D_1 触发器的 d 输入端相连接，将 D_1 输出(Q_1)与 D_2 触发器的 d 输入端相连接就行了。Q_0 输出关系复杂一些，就必须选用 JK 触发器，并且利用 Q_1，Q_2 输出作为约束条件，经组合逻辑电路作为 Q_0 的 J 和 K 的输入。Q_2，Q_1 输出和 Q_0 的 J，K 输入关系如表 1－2 所示。从表 1－2 就很容易写出以 Q_2，Q_1 为输入，J，K 为输出的两个真值表。该真值表实际上就是或非门的真值表和与门的真值表。那么，将 Q_2，Q_1 分别连到或非门的输入端，将或非门的输出连到 Q_0 的 J 输入端；再将 Q_2，Q_1 分别连接到与门的输入端，将与门的输出端与 Q_0 的 K 输入端相连。这样，一个六进制计数器的硬件电路设计就完成了，如图 1－2 所示。当然，触发器的时钟端应和计数脉冲端相连接，系统复位信号应和触发器的置“0”端相连接，这样就可以保证实际电路的正常工作。

表 1－1　触发器状态变化表

触发器状态 / 计数脉冲	Q_2		Q_1		Q_0	
	前一状态	当前状态	前一状态	当前状态	前一状态	当前状态
1	0	0	0	0	0	1
2	0	0	0	1	1	1
3	0	1	1	1	1	1
4	1	1	1	1	1	0
5	1	1	1	0	0	0
6	1	0	0	0	0	0

表 1－2　Q_2，Q_1 输出和 Q_0 的 J，K 输入关系表

触发器状态 / 计数脉冲	Q_2	Q_1	Q_0			
	前一状态	前一状态	J	K	前一状态	当前状态
1	0	0	1	0	0	1
2	0	0	1	0	1	1
3	0	1	0	0	1	1
4	1	1	0	1	1	0
5	1	1	0	1	0	0
6	1	0	0	0	0	0

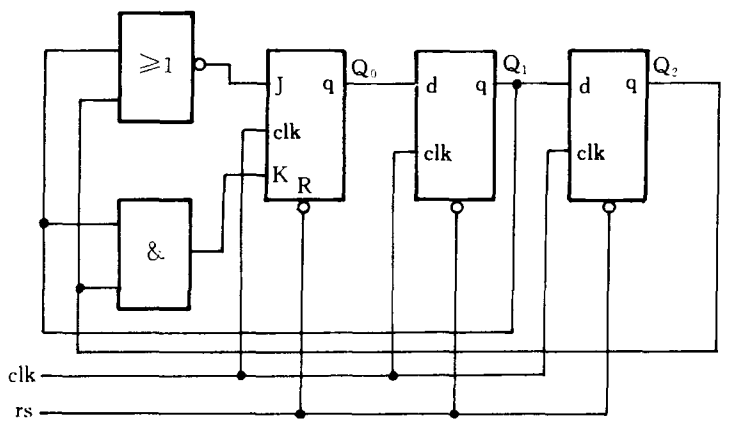


图 1－2　六进制约翰逊计数器原理图

与六进制计数器模块设计一样，系统的其它模块也按此方法进行设计。在所有硬件模块设计完成以后，再将各模块连接起来，进行调试，如有问题则进行局部修改，直至整个系统调试完毕为止。

从上述设计过程我们可以看到，系统硬件的设计是从选择具体元、器件开始的，并用这些元、器件进行逻辑电路设计，完成系统各独立功能模块设计，然后再将各功能模块连接起来，完成整个系统的硬件设计。上述过程从最底层开始设计，直至到最高层设计完毕，故将这种设计方法称为自下至上的设计方法。

1.1.2　采用通用的逻辑元、器件

在传统的硬件电路设计中，设计者总是根据系统的具体需要，选择市场上能买到的逻辑元、器件，来构成所要求的逻辑电路，从而完成系统的硬件设计。尽管随着微处理器的出现，在由微处理器及其相应硬件构成的系统中，许多系统的硬件功能可以用软件功能来实现，从而在较大程度上简化了系统硬件电路的设计，但是，这种选择通用的元、器件来构成系统硬件电路的方法并未改变。

1.1.3 在系统硬件设计的后期进行仿真和调试

在传统的系统硬件设计方法中，仿真和调试通常只能在后期完成系统硬件设计以后，才能进行。因为进行仿真和调试的仪器一般为系统仿真器、逻辑分析仪和示波器等。因此只有在硬件系统已经构成以后才能使用。系统设计时存在的问题只有在后期才能较容易发现。这样，传统的硬件设计方法对系统设计人员有较高的要求。一旦考虑不周，系统设计存在较大缺陷，那么就有可能要重新设计系统，使得设计周期也大大增加。

1.1.4 主要设计文件是电原理图

在用传统的硬件设计方法对系统进行设计并调试完毕后，所形成的硬件设计文件，主要是由若干张电原理图构成的文件。在电原理图中详细标注了各逻辑元、器件的名称和互相间的信号连接关系。该文件是用户使用和维护系统的依据。对于小系统，这种电原理图只要几十张至几百张就行了。但是，如果系统比较大，硬件比较复杂，那么这种电原理图可能要有几千张、几万张，甚至几十万张。如此多的电原理图给归档、阅读、修改和使用都带来了极大的不方便。

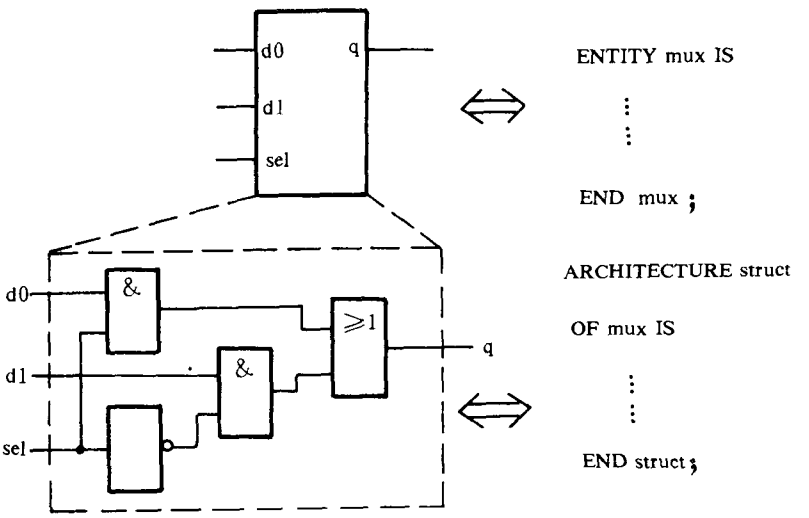
传统的硬件电路设计方法已经沿用几十年，是目前广大电子工程师所熟悉和掌握的一种方法。但是，随着计算机技术、大规模集成电路技术的发展，这种传统的设计方法已大大落后于当今技术的发展。一种崭新的，采用硬件描述语言的硬件电路设计方法已经兴起，它的出现将给硬件电路设计带来一次重大的变革。

1.2 利用硬件描述语言(HDL)的硬件电路设计方法

在硬件电路设计中采用计算机辅助设计技术(CAD)，一般来说到 80 年代才得到了普及和应用。在一开始，仅仅是利用计算机软件来实现印刷板的布线，以后慢慢地才实现了插件板级规模的电子电路的设计和仿真。在我国所使用的工具中，最有代表性的设计工具是 Tango 和早期的 ORCAD。它们的出现，使得电子电路设计和印刷板布线工艺实现了自动化。但是，就其设计方法而言，仍是自下至上的设计方法，利用已有的逻辑元、器件来构成硬件电路。

随着大规模专用集成电路(ASIC)的开发和研制，为了提高开发的效率，增加已有开发成果的可继承性以及缩短开发时间，各 ASIC 研制和生产厂家相继开发了用于各自目的的硬件描述语言。其中最有代表性的是美国国防部开发的 VHDL 语言(VHSIC Hardware Description Language)，Viewlogic 公司开发的 Verilog HDL 以及日本电子工业振兴协会开发的 UDL/I 语言。

所谓硬件描述语言，就是可以描述硬件电路的功能，信号连接关系及定时关系的语言。它能比电原理图更有效地表示硬件电路的特性。例如，一个二选一的选择器的电原理图如图 1 - 3(*a*)所示，而用 VHDL 语言描述的二选一的选择器如图 1 - 3(*b*)所示。利用硬件描述语言编程来表示逻辑器件及系统硬件的功能和行为，这是该设计方法的一个重要特征。



(*a*)

```
ENTITY mux IS
PORT(d0, d1, sel: IN BIT;
                  q: OUT BIT);
END mux;
ARCHITECTURE connect OF mux IS
BEGIN
  calc: PROCESS(d0, d1, sel)
    VARIABLE tmp1, tmp2, tmp3: BIT;
  BEGIN
    tmp1:=d0 AND sel;
    tmp2:=d1 AND (NOT sel);
    tmp3:=tmp1 OR tmp2;
    q<=tmp3;
  END PROCESS;
END connect;
```

(*b*)

图 1-3　二选一选择器描述

(*a*) 电原理图表示；(*b*) 用 VHDL 语言描述

利用 HDL 语言设计系统硬件的方法，归纳起来有以下几个特点。

1.2.1　采用自上至下(Top Down)的设计方法

所谓自上至下的设计方法，就是从系统总体要求出发，自上至下地逐步将设计内容细化，最后完成系统硬件的整体设计。在利用 HDL 的硬件设计方法中，设计者将自上至下分

成 3 个层次对系统硬件进行设计。

第一层次是行为描述。所谓行为描述，实质上就是对整个系统的数学模型的描述。一般来说，对系统进行行为描述的目的是试图在系统设计的初始阶段，通过对系统行为描述的仿真来发现设计中存在的问题。在行为描述阶段，并不真正考虑其实际的操作和算法用什么方法来实现。考虑更多的是系统的结构及其工作过程是否能达到系统设计规格书的要求。下面仍以六进制计数器为例，说明一下如何用 VHDL 语言，以行为方式来描述它的工作特性，其实例如例 1 - 1 所示。

【例 1 - 1】

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY counter IS
PORT(
      clk: IN STD_LOGIC;
       rs: IN STD_LOGIC;
      count_out: OUT STD_LOGIC_VECTOR(2 DOWNTO 0));
END counter;
ARCHITECTURE behav OF counter IS
   SIGNAL next_count: STD_LOGIC_VECTOR(2 DOWNTO 0);
BEGIN
   count_proc: PROCESS(rs, clk)
BEGIN
  IF rs='0' THEN
     mext_count<="000";
  ELSIF (clk'EVENT AND clk='1') THEN
     CASE next_count IS
     WHEN "000"=>next_count<="001";
     WHEN "001"=>next_count<="011";
     WHEN "011"=>next_count<="111";
     WHEN "111"=>next_count<="110";
     WHEN "110"=>next_count<="100";
     WHEN "100"=>next_count<="000";
     WHEN OTHERS=>next_count<="XXX";
    END CASE;
  END IF;
   count_out<=next_count AFTER 10 ns;
  END PROCESS;
END behav;
```

从例 1 - 1 可以看出，该段 VHDL 语言程序勾画出了六进制计数器的输入输出引脚和内部计数过程的计数状态变化时序和关系。这实际上是计数器工作模型的描述。当该程序仿真通过以后，说明六进制计数器模型是正确的。在此基础上再改写该程序，使其语句表达式易于用逻辑元件来实现。这是第二层次所要做的工作。

第二层次是 RTL 方式描述。这一层次称为寄存器传输描述(又称数据流描述)。如前所述，用行为方式描述的系统结构的程序，其抽象程度高，是很难直接映射到具体逻辑元件结构的硬件实现的。要想得到硬件的具体实现，必须将行为方式描述的 VHDL 语言程序改写为 RTL 方式描述的 VHDL 语言程序。也就是说，系统采用 RTL 方式描述，才能导出系统的逻辑表达式，才能进行逻辑综合。当然，这里所说的"可以"进行逻辑综合是有条件的，它是针对某一特定的逻辑综合工具而言的。与例 1 - 1 行为方式描述所等价的六进制计数器的 RTL 描述，如例 1 - 2 所示。

【例 1 - 2】

```
LIBRARY IEEE;
USE IEEE. STD_LDGIC_1164. ALL;
USE WORK. NEW. ALL;
ENTITY counter IS
PORT(clk, rs: IN STD_LOGIC;
      q1, q2, q3: OUT STD_LOGIC);
END counter;
ARCHITECTURE rtl OF counter IS
COMPONENT dff
  PORT(d, rs, clk: IN STD_LOGIC;
               q: OUT STD_LOGIC);
END COMPONENT;
COMPONENT djk
PORT(j,k,rs,clk: IN STD_LOGIC;
             q: OUT STD_LOGIC);
END COMPONENT;
COMPONENT and2
  PORT(a,b: IN STD_LOGIC;
         c: OUT STD_LOGIC);
END COMPONENT;
COMPONENT nor2
  PORT(a,b: IN STD_LOGIC;
         c: OUT STD_LOGIC);
END COMPONENT;
SIGNAL jin, kin, q1_out, q2_out, q3_out:
       STD_LOGIC;
BEGIN
  u1: nor2
    PORT MAP(q3_out, q2_out, jin);
  u2: and2
    PORT MAP(q3_out, q2_out, kin);
  u3: djk
    PORT MAP(jin, kin, rs, clk, q1_out);
  u4: dff
```

```
    PORT MAP(q1_out, rs, clk, q2_out);
  u5: dff
    PORT MAP(q2_out, rs, clk, q3_out);
  q1<=q1_out;
  q2<=q2_out;
  q3<=q3_out;
END rtl;
```

在该例中，JK 触发器、D 触发器、与门和或非门都已在库 WORK. NEW. ALL 中定义了，这里可以直接引用。例中的构造体直接描述了它们之间的连接关系。与例 1 - 1 相比，例 1 - 2 更趋于实际电路的描述。

在把行为方式描述的程序改写为 RTL 方式描述的程序时，编程人员必须深入了解逻辑综合工具的详细说明和具体规定，这样才能编写出合格的 RTL 方式描述的程序。

在完成编写 RTL 方式的描述程序以后，再用仿真工具对 RTL 方式描述的程序进行仿真。如果通过这一步仿真，那么就可以利用逻辑综合工具进行综合了。

第三层次是逻辑综合。逻辑综合这一阶段是利用逻辑综合工具，将 RTL 方式描述的程序转换成用基本逻辑元件表示的文件(门级网络表)。此时，如果需要，可以将逻辑综合结果，以逻辑原理图方式输出。也就是说，逻辑综合的结果相当于在人工设计硬件电路时，根据系统要求画出了系统的逻辑电原理图。此后对逻辑综合结果在门电路级上再进行仿真，并检查定时关系。如果一切都正常，那么系统的硬件设计就基本结束。如果在 3 个层次的某个层次上发现有问题，都应返回上一层，寻找和修改相应的错误，然后再向下继续未完的工作。

由逻辑综合工具产生门级网络表后，在最终完成硬件设计时，还可以有两种选择。第一种是由自动布线程序将网络表转换成相应的 ASIC 芯片的制造工艺，做出 ASIC 芯片。第二种是将网络表转换成 FPGA(现场可编程门阵列)的编程码点，利用 FPGA 完成硬件电路设计。

在用 HDL 语言设计系统硬件时，无论是设计一个局部电路，还是设计由多块插件板组成的复杂系统，上述自上至下的 3 个层次的设计步骤是必不可少的。利用自上至下设计系统硬件的过程如图 1 - 4 所示。

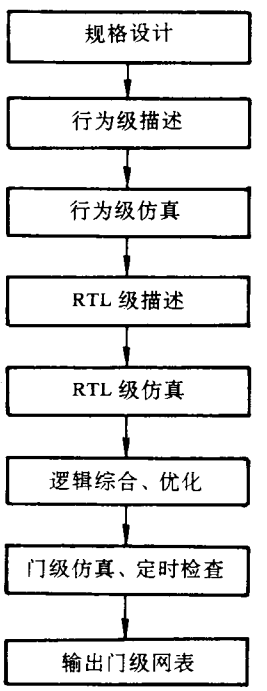


图 1 - 4　自上至下设计系统硬件的过程

由自上至下的设计过程可知，从总体行为设计开始到最终逻辑综合，形成网络表为止，每一步都要进行仿真检查，这样有利于尽早发现系统设计中存在的问题，从而可以大大缩短系统硬件的设计周期。这是用 HDL 语言设计系统硬件的最突出的优点之一。

1.2.2 系统中可大量采用 ASIC 芯片

由于目前众多的制造 ASIC 芯片的厂家，它们的工具软件都可支持 HDL 语言的编程，因此，硬件设计人员在设计硬件电路时，无须受只能使用通用元、器件的限制，而可以根据硬件电路设计需要，设计自用的 ASIC 芯片或可编程逻辑器件。这样最终会使系统电路设计更趋合理，体积也可大为缩小。

1.2.3 采用系统早期仿真

从自上至下的设计过程可以看到，在系统设计过程中要进行三级仿真，即行为层次仿真、RTL 层次仿真和门级层次仿真。也就是说进行系统数学模型的仿真、系统数据流的仿真和系统门电路电原理的仿真。这 3 级仿真贯穿系统硬件设计的全过程，从而可以在系统设计早期发现设计中存在的问题。与自下至上设计的后期仿真相比可大大缩短系统的设计周期，节约大量的人力和物力。

1.2.4 降低了硬件电路设计难度

在采用传统的硬件电路设计方法时，往往要求设计者在设计电路前应写出该电路的逻辑表达式或真值表(或时序电路的状态表)。这一工作是相当困难和繁杂的，特别是在系统比较复杂时更是如此。例如，在设计六进制计数器时，必须编写输入和输出的真值表和状态表。根据表中的关系，写出逻辑表达式，并用相应的逻辑元件来实现。

在用 HDL 语言设计硬件电路时，就可以使设计者免除编写逻辑表达式或真值表之苦。如图 1-1 和例 1-1 所示，只要知道六进制计数器的 6 个计数状态就行了，而无须写出相关电路的逻辑表达式。这样使硬件电路的设计难度有了大幅度的下降，从而也缩短了硬件电路的设计周期。据有关资料估计，仅此一项可使设计周期大约缩短 1/3～1/2。

1.2.5 主要设计文件是用 HDL 语言编写的源程序

在传统的硬件电路设计中，最后形成的主要文件是电原理图，而采用 HDL 语言设计系统硬件电路时，主要的设计文件是用 HDL 语言编写的源程序。如果需要也可以转换成电原理图形式输出。用 HDL 语言的源程序作为归档文件有很多好处。其一是资料量小，便于保存。其二是可继承性好。当设计其它硬件电路时，可以使用文件中的某些库、进程和过程等描述某些局部硬件电路的程序。其三是阅读方便。阅读程序比阅读电原理图要更容易一些。阅读者很容易在程序中看出某一硬件电路的工作原理和逻辑关系。而阅读电原理图，推知其工作原理却需要较多的硬件知识和经验，而且看起来也不那么一目了然。

1.3 利用 VHDL 语言设计硬件电路的优点

当前各 ASIC 芯片制造商都相继开发了用于各自目的的 HDL 语言，但是大多都未标准化和通用化。唯一已被公认的是美国国防部开发的 VHDL 语言，它已成为 IEEE STD-1076标准。另外，从近期 HDL 语言发展的动态来看，许多公司研制的硬件电路设计工具也都逐渐向 VHDL 语言靠拢，使得它们的硬件电路设计工具也能支持 VHDL 语言。

Viewlogic 公司开发的 Verilog_HDL 语言目前使用也较广泛，据资料介绍近期也有可能成为另一种标准的 HDL 语言。但是，从整体来看 Verilog_HDL 语言不如 VHDL 语言。

VHDL 语言和其它 HDL 语言相比到底有哪些特色呢？下面作一简要说明。

1.3.1 设计技术齐全、方法灵活、支持广泛

VHDL 语言可以支持自上至下(Top Down)和基于库(Library-Based)的设计方法，而且还支持同步电路、异步电路、FPGA 以及其它随机电路的设计。其范围之广是其它 HDL 语言所不能比拟的。例如，SFL 语言和 UDL/ I 语言，它们只能描述同步电路。另外，由于 VHDL 语言早在 1987 年 12 月已作为 IEEE_STD_1076 标准公开发布。因此，目前大多数 EDA 工具几乎在不同程度上都支持 VHDL 语言。这样给 VHDL 语言进一步推广和应用创造了良好的环境。

1.3.2 系统硬件描述能力强

如前所述，VHDL 语言具有多层次描述系统硬件功能的能力，可以从系统的数学模型直到门级电路。另外，高层次的行为描述可以与低层次的 RTL 描述和结构描述混合使用。例如，在 PC 机扩展槽上要设计一块接口卡，该接口卡的硬件设计应满足主机的接口要求。此时，主机部分功能可以用行为方式描述，而接口卡可以采用 RTL 方式描述。在系统仿真时就可以验证接口卡的工作是否正确。这样，在接口卡设计出来以前就可以知道接口卡的工作是否满足系统要求。

VHDL 语言能进行系统级的硬件描述，这是它的一个最突出的优点。其它 HDL 语言，如 UDL/ I 、Verilog 等只能进行 IC 级、PCB 级描述，而不能对系统级的硬件很好地进行描述。

再如，VHDL 语言可以自定义数据类型，这样也给编程人员带来了较大的自由和方便。

1.3.3 VHDL 语言可以与工艺无关编程

在用 VHDL 语言设计系统硬件时，没有嵌入与工艺有关的信息。当然，这样的信息是可以用 VHDL 语言来编写的。与大多数 HDL 语言的不同之处是，当门级或门级以上层次的描述通过仿真检验以后，再用相应的工具将设计映射成不同的工艺(如 MOS、CMOS 等)。这样，在工艺更新时，就无须修改原设计程序，只要改变相应的映射工具就行了。由此可见，无论修改电路还是修改工艺相互之间不会产生什么不良影响。

1.3.4 VHDL 语言标准、规范，易于共享和复用

由于 VHDL 语言已作为一种 IEEE 的工业标准，这样，设计成果便于复用和交流，反过来就能更进一步推动 VHDL 语言的推广及完善。另外，VHDL 语言的语法比较严格，其风格类似于 Ada 语言，给阅读和使用都带来了极大的好处。

第 2 章

VHDL 语言程序的基本结构

一个完整的 VHDL 语言程序通常包含实体(Entity)、构造体(Architecture)、配置(Configuration)、包集合(Package)和库(Library)5 个部分。前 4 种是可分别编译的源设计单元。实体用于描述所设计的系统的外部接口信号；构造体用于描述系统内部的结构和行为；包集合存放各设计模块都能共享的数据类型、常数和子程序等；配置用于从库中选取所需单元来组成系统设计的不同版本；库存放已经编译的实体、构造体、包集合和配置。库可由用户生成或由 ASIC 芯片制造商提供，以便于在设计中为大家所共享。本章将对上述 VHDL 设计的主要构成作一详细介绍。

2.1 VHDL 语言设计的基本单元及其构成

所谓VHDL 语言设计的基本单元(Design Entity)，就是 VHDL 语言的一个基本设计实体。一个基本设计单元，简单的可以是一个与门(AND Gate)，复杂点的可以是一个微处理器或一个系统。但是，不管是简单的数字电路，还是复杂的数字电路，其基本构成是一致的。它们都由实体说明(Entity Declaration)和构造体 (Architecture Body) 两部分构成。如前所述，实体说明部分规定了设计单元的输入输出接口信号或引脚，而构造体部分定义了设计单元的具体构造和操作(行为)。图 2 - 1 示出了作为一个设计单元的二选一电路的 VHDL 描述。由图 2 - 1 可以看出，实体说明是二选一器件外部引脚的定义；而构造体则描述了二选一器件的逻辑电路和逻辑关系。

```
ENTITY mux IS
GENERIC(m:TIME := 1 ns);
PORT(d0, d1, sel: IN BIT;
                 q: OUT BIT);
END mux;
ARCHITECTURE connect OF mux IS
SIGNAL tmp: BIT;
BEGIN
  cale: PROCESS(d0, d1, sel)
    VARIABLE tmp1, tmp2, tmp3: BIT;
  BEGIN
```

```
    tmp1:=d0 AND sel;
    tmp2:=d1 AND (NOT sel);
    tmp3:=tmp1 OR tmp2;
    tmp<=tmp3;
      q<=tmp AFTER m;
  END PROCESS;
END connect;
```

图 2-1　一个基本设计单元的构成

下面以二选一器件描述为例，说明一下这两部分的具体书写规定。

2.1.1　实体说明

任何一个基本设计单元的实体说明都具有如下的结构：

ENTITY　实体名 IS
[类属参数说明];
[端口说明];
END 实体名;

一个基本设计单元的实体说明以“ENTITY 实体名 IS”开始至“END 实体名”结束。例如在图 2-1 中从“ENTITY mux IS”开始，至“END mux”结束。这里大写字母表示实体说明的框架，即每个实体说明都应这样书写，是不可缺少和省略的部分。小写字母是设计者添写的部分，随设计单元不同而不同。实际上，对 VHDL 而言，大写或小写都一视同仁，不加区分。这里仅仅是为了阅读方便而加以区分的。

1. 类属参数说明

类属参数说明必须放在端口说明之前，用于指定参数，例如图 2-1 中的 GENERIC (m:TIME:=1 ns)。该语句指定了构造体内 m 的值为1 ns。这样语句

q<=tmp AFIER m;

在这个例子中，GENERIC 利用类属参数为 tmp 建立一个延迟值。

2. 端口说明

端口说明是对基本设计实体(单元)与外部接口的描述，也可以说是对外部引脚信号的名称，数据类型和输入、输出方向的描述。其一般书写格式如下：

PORT(端口名{，端口名}：方向　数据类型名；
　　⋮
　　端口名{，端口名}：方向　数据类型名)；

1) 端口名

端口名是赋予每个外部引脚的名称，通常用一个或几个英文字母，或者用英文字母加数字命名之。例如图 2-1 中的外部引脚为 d0，d1，sel，q。

2) 端口方向

端口方向用来定义外部引脚的信号方向是输入还是输出。例如，图 2-1 中的 d0，d1，sel 为输入引脚，故用方向说明符“IN”说明之，而 q 则为输出引脚，用方向说明符“OUT”说明之。