第一章

名词解释：

(1)EDA

(2)VHDL

(3)FPGA

(4) ASIC

(5)CPLD

填空

303 页例 9-4

1-1 EDA 技术与 ASIC 设计和 FPGA 开发有什么关系?

答：利用 EDA 技术进行电子系统设计的最后目标是完成专用集成电路 ASIC 的设计和实现；FPGA 和 CPLD 是实现 这一途径的主流器件。FPGA 和 CPLD 通常也被称为可编程专用 IC，或可编程 ASIC。FPGA 和 CPLD 的应用是 EDA 技术 有机融合软硬件电子设计技术、SoC（片上系统）和 ASIC 设计，以及对自动设计与自动实现最典型的诠释。

1-2 与软件描述语言相比，VHDL 有什么特点? P6

答：编译器将软件程序翻译成基于某种特定 CPU 的机器代码，这种代码仅限于这种 CPU 而不能移植，并且机器 代码不代表硬件结构，更不能改变 CPU 的硬件结构，只能被动地为其特定的硬件电路结构所利用。 综合器将 VHDL 程序转化的目标是底层的电路结构网表文件，这种满足 VHDL 设计程序功能描述的电路结构，不依赖于任何特定硬 件环境：具有相对独立性。综合器在将 VHDL(硬件描述语言)表达的电路功能转化成具体的电路结构网表过程中，具 有明显的能动性和创造性，它不是机械的一一对应式的“翻译”，而是根据设计库、工艺库以及预先设置的各类约 束条件，选择最优的方式完成电路结构的设计。

l-3 什么是综合?有哪些类型?综合在电子设计自动化中的地位是什么? 什么是综合?

答：在电子设计领域中综合的概念可以表示为：将用行为和功能层次表达的电子系统转换为低层 次的便于具体实现的模块组合装配的过程。

有哪些类型?

答：(1)从自然语言转换到 VHDL 语言算法表示，即 自然语言综合。(2)从算法表示转换到寄存器 传输级(RegisterTransport Level，RTL)，即从行为域到结构域的综合，即行为综合。(3)从 RTL 级表示转换到逻 辑门(包括触发器)的表示，即逻辑综合。(4)从逻辑门表示转换到版图表示(ASIC 设计)，或转换到 FPGA 的配置网表 文件，可称为版图综合或结构综合。

综合在电子设计自动化中的地位是什么?

答：是核心地位（见图 1-3)。综合器具有更复杂的工作环境，综合器 在接受 VHDL 程序并准备对其综合前，必须获得与最终实现设计电路硬件特征相关的 工艺库信息，以及获得优化综 合的诸多约束条件信息；根据工艺库和约束条件信息，将 VHDL 程序转化成电路实现的相关信息。

1-4 在 EDA 技术中，自顶向下的设计方法的重要意义是什么? P7~10

答：在 EDA 技术应用中，自顶向下的设计方法，就是在整个设计流程中各设计环节逐步求精的过程。

1-5 IP 在 EDA 技术的应用和发展中的意义是什么? P11~12

答：IP 核具有规范的接口协议，良好的可移植与可测试 性，为系统开发提供了可靠的保

证。

第二章

2-1 叙述 EDA 的 FPGA/CPLD 设计流程。 P13~16

答：1.设计输入(原理图/HDL 文本编辑)；2.综合；3.适 配；4.时序仿真与功能仿真；5.编程下载；6.硬件测试。

2-2 IP 是什么?IP 与 EDA 技术的关系是什么? P24~26 IP 是什么?

答：IP 是知识产权核或知识产权模块，用于 ASIC 或 FPGA/CPLD 中的预先设计好的电路功能模块。 I

P 与 EDA 技术的关系是什么?

答：IP 在 EDA 技术开发中具有 十分重要的地位；与 EDA 技术的关系分有软 IP、 固 IP、硬 IP：软 IP 是用 VHDL 等硬件描述语言描述的功能块，并不涉及用什么具体电路元件实现这些功能；软 IP 通常是以硬件描述语言 HDL 源文件的形式出现。固 IP 是完成了综合的功能块，具有较大的设计深度，以网表文件 的形式提交客户使用。硬 IP 提供设计的最终阶段产品：掩模。

2-3 叙述 ASIC 的设计方法。 P18~19

答：ASIC 设计方法,按版图结构及制造方法分有半定制(Semi-custom)和全定制(Full-custom)两种实现方法。 全定制方法是一种基于晶体管级的，手工设计版图的制造方法。 半定制法是一种约束性设计方式，约束的目的是简化设计，缩短设计周期，降低设计成本，提高设计正确率。 半定制法按逻辑实现的方式不同，可再分为门阵列法、标准单元法和可编程逻辑器件法。

2-4 FPGA/CPLD 在 ASIC 设计中有什么用途? P16,18

答：FPGA/CPLD 在 ASIC 设计中，属于可编程 ASIC 的逻辑器件；使设计效率大为提高，上市的时间大为缩短。

2-5 简述在基于 FPGA/CPLD 的 EDA 设计流程中所涉及的 EDA 工具，及其在整个流程中的作用。 P19~23

答：基于 FPGA/CPLD 的 EDA 设计流程中所涉及的 EDA 工具有：设计输入编辑器（作用：接受不同的设计输 入表达方式，如原理图输入方式、状态图输入方式、波形输入方式以及 HDL 的文本输入方式。)；HDL 综合器（作用： HDL 综合器根据工艺库和约束条件信息，将设计输入编辑器提供的信息转化为目标器件硬件结构细节的信息，并在 数字电路设计技术、化简优化算法以及计算机软件等复杂结体进行优化处理)；仿真器（作用：行为模型的表达、 电子系统的建模、逻辑电路的验证及门级系统的测试)；适配器（作用：完成目标系统在器件上的布局和布线)；下 载器（作用：把设计结果信息下载到对应的实际器件，实现硬件设计)。

第三章

3-1 OLMC（输出逻辑宏单元）有何功能?说明 GAL 是怎样实现可编程组合电路与时序电路的。 P34~36

OLMC 有何功能?

答：OLMC 单元设有多种组态，可 配置成专用组合输出、专用输入、组合输出双向口、寄存器 输出、寄存器输出双向口等。

说明 GAL 是怎样实现可编程组合电路与时序电路的?

答：GAL（通用阵列逻辑器件）是通过对其中的 OLMC （输出逻辑宏单元）的编程和三种模式配置（寄存器模式、复合模式、简单模式)，实现组合电路与时序电路设计的。



3-2 什么是基于乘积项的可编程逻辑结构? P33~34，40

答：GAL、CPLD 之类都是基于乘积项的可编程结构； 即包含有可编程与阵列和固定的或阵列的 PAL（可编程阵 列逻辑）器件构成。

3-3 什么是基于查找表的可编程逻辑结构？ P40~41

答：FPGA（现场可编程门阵列）是基于查找表的可编程逻辑结构。

3-4 FPGA 系列器件中的 LAB 有何作用?

答：FPGA（Cyclone/Cyclone II）系列器件主要由逻辑阵列块 LAB、嵌入式存储器块（EAB）、I/O 单元、嵌入 式硬件乘法器和 PLL 等模块构成：其中 LAB（逻辑阵列块）由一系列相邻的 LE（逻辑单元）构成的；FPGA 可编程 资源主要来自逻辑阵列块 LAB。

3-5 与传统的测试技术相比，边界扫描技术有何优点？ P47~50

答：使用 BST（边界扫描测试）规范测试，不必使用物理探 针，可在器件正常工作时在系统捕获测量的功能数 据。克服传统的外探针测试法和“针床”夹具测试法来无法对 IC 内部节点无法测试的难题。

3-6 解释编程与配置这两个概念。 P58

答：编程：基于电可擦除存储单元的 EEPROM 或 Flash 技 术。CPLD 一股使用此技术进行编程。CPLD 被编程后改 变了电可擦除存储单元中的信息，掉电后可保存。电可擦除编程工艺的优点是编程后信息不会因掉电而丢失，但编 程次数有限，编程的速度不快。

配置：基于 SRAM 查找表的编程单元。编程信息是保 存在 SRAM 中的，SRAM 在掉电后编程信息立即丢失，在 下次上电后，还需要重新载入编程信息。大部分 FPGA 采用该种编程工艺。该类器件的编程一般称为配置。对于 SRAM 型 FPGA 来说，配置次数无限，且速度快；在加电时可随时更改逻辑；下载信息的保密性也不如电可擦除的编程。

3-7 请参阅相关资料，并回答问题：

按本章给出的归类方式，将基 于乘积项的可编程逻辑结构的 PLD 器件归类为 CPLD；将基于查找表的可编程逻辑结构的 PLD 器什归类为 FPGA，那么，

APEX 系列属于什么类型 PLD 器件?

MAX II 系列又属于什么类型的 PLD 器件?为什么? P54~56

答：APEX(Advanced Logic Element Matrix)系列属于 FPGA 类型 PLD 器件；编程信息存于 SRAM 中。MAX II 系列属于 CPLD 类型的 PLD 器件；编程信息存于 EEPROM 中。

第四章

4-1：画出与下例实体描述对应的原理图符号元件：

```
ENTITY buf3s IS
-- 实体 1：三态缓冲器
-- 输入端
    -- 使能端
    -- 输出端
PORT (input : IN STD_LOGIC
          enable : IN STD_LOGIC
          output : OUT STD_LOGIC )
END buf3x
ENTITY mux21 IS
--实体 2： 2 选 1 多路选择器
PORT (in0, in1, sel : IN STD_LOGIC;
```

output : OUT STD_LOGIC);

4-1.答案

4-2. 图 3-30 所示的是 4 选 1 多路选择器，试分别用 IF_THEN 语句和 CASE 语句的表达方式写出此电路的 VHDL 程序。选择控制的信号 s1 和 s0 的数据类型为 STD_LOGIC_VECTOR；当 s1='0'，s0='0'；s1='0'，s0='1'；s1='1'，s0='0' 和 s1='1'，s0='1' 分别执行 y<=a、y<=b、y<=c、y<=d。

4-2.答案

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY MUX41 IS
PORT(s:IN STD_LOGIC_VECTOR(1 DOWNTO 0); --输入选择信号
      a,b,c,d:IN STD_LOGIC; --输入信号
      y:OUT STD_LOGIC);--输出端
END ENTITY;
ARCHITECTURE ART OF MUX41 IS
BEGIN
PROCESS(s)
BEGIN
IF (S="00") THEN y<=a;
ELSIF (S="01") TH EN y<=b;
ELSIF (S="10") TH EN y<=c;
ELSIF (S="11") TH EN y<=d;
ELSE y<=NULL; END IF;
EDN PROCESS; END ART;
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY MUX41 IS
PORT(s:IN STD_LOGIC_VECTOR(1 DOWNTO 0); --输入选择信号
      a,b,c,d:IN STD_LOGIC; --输入信号
      y:OUT STD_LOGIC);--输出端
END MUX41;
ARCHITECTURE ART OF MUX41 IS
BEGIN
PROCESS(s)
BEGIN
CASE s IS
WHEN “00” => y<=a;
WHEN “01” => y<=b;
WHEN “10” => y<=c;
WHEN “11” => y<=d;
WHEN OTHERS =>NULL;
END CASE; END PROCESS;
END ART;
```

4-3. 图 3-31 所示的是双 2 选 1 多路选择器构成的电路 MUXK，对于其中 MUX21A，当

s='0'和'1'时，分别有 y<='a'　和 y<='b'。试在一个结构体中用两个进程来表达此电路，每个进程中用 CASE 语句描述一个 2 选 1 多路选择器 MUX21A。

4-3.答案

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY MUX221 IS
PORT(a1,a2,a3:IN STD_LOGIC_VECTOR(1 DOWNTO 0); --输入信号
 s0,s1:IN STD_LOGIC;
outy:OUT STD_LOGIC);--输出端
 END ENTITY;
ARCHITECTURE ONE OF MUX221 IS
SIGNAL tmp : STD_LOGIC;
BEGIN
PR01:PROCESS(s0) BEGIN
IF s0=”0” THEN tmp<=a2;
ELSE tmp<=a3;
END IF;
END PROCESS; PR02:PROCESS(s1) BEGIN
IF s1=”0” THEN outy<=a1;
 ELSE outy<=tmp;
END IF;
END PROCESS;
END ARCHITECTURE ONE;
END CASE;
```

4-4.下图是一个含有上升沿触发的 D 触发器的时序电路，试写出此电路的 VHDL 设计文件。

4-4.答案

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY MULTI IS  PORT(CL:IN STD_LOGIC; --输入选择信号
CLK0:IN STD_LOGIC; --输入信号
OUT1:OUT STD_LOGIC);--输出端
END ENTITY;
ARCHITECTURE ONE OF MULTI IS
SIGNAL Q : STD_LOGIC;
BEGIN
PR01:
PROCESS(CLK0)
BEGIN
IF CLK ‘EVENT AND CLK=’1’ THEN Q<=NOT(CL OR Q);ELSE
END IF;
END PROCESS;
PR02:
PROCESS(CLK0)
```

```
BEGIN
OUT1<=Q;
END PROCESS;
END ARCHITECTURE ONE;
END PROCESS;
```

4-5.给出 1 位全减器的 VHDL 描述。

要求： (1) 首先设计 1 位半减器，然后用例化语句将它们连接起来，图 3-32 中 h_suber 是半减器，diff 是输出差， s_out 是借位输出，sub_in 是借位输入。

(2) 以 1 位全减器为基本硬件，构成串行借位的 8 位减法器，要求用例化语句来完成此项设计(减法运算是 x - y - sun_in = diffr)

4-5.答案

底层文件 1：or2a.VHD 实现或门操作

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY or2a IS
PORT(a,b:IN STD_LOGIC;
        c:OUT STD_LOGIC);
END ENTITY or2a;
ARCHITECTURE one OF or2a IS
BEGIN c <= a OR b;
END ARCHITECTURE one;
```

底层文件 2：h_subber.VHD 实现一位半减器

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY h_subber IS
PORT(x,y:IN STD_LOGIC;
diff,s_out::OUT STD_LOGIC);
END ENTITY h_subber;
ARCHITECTURE ONE OF h_subber IS
SIGNAL xyz: STD_LOGIC_VECTOR(1 DOWNTO 0);
BEGIN
xyz <= x & y;
PROCESS(xyz)
BEGIN
CASE xyz IS
WHEN "00" => diff<='0';s_out<='0';
WHEN "01" => diff<='1';s_out<='1';
WHEN "10" => diff<='1';s_out<='0';
WHEN "11" => diff<='0';s_out<='0';
WHEN OTHERS => NULL;
END CASE; END PROCESS;
END ARCHITECTURE ONE;
```

顶层文件：f_subber.VHD 实现一位全减器

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY f_subber IS
PORT(x,y,sub_in:IN STD_LOGIC;
diffr,sub_out:OUT STD_LOGIC);
END ENTITY f_subber;
ARCHITECTURE ONE OF f_subber IS
COMPONENT h_subber
PORT(x,y:IN STD_LOGIC;
diff,S_out:OUT STD_LOGIC);
END COMPONENT;
COMPONENT or2a
PORT(a,b:IN STD_LOGIC;
c:OUT STD_LOGIC);
END COMPONENT;
SIGNAL d,e,f: STD_LOGIC;
BEGIN
u1: h_subber PORT MAP(x=>x,y=>y,diff=>d,s_out=>e);
u2: h_subber PORT MAP(x=>d,y=>sub_in,diff=>diffr,s_out=>f);
u3: or2a PORT MAP(a=>f,b=>e,c=>sub_out);
END ARCHITECTURE ONE;
END ARCHITECTURE ART;
```

4-6.根据下图，写出顶层文件 MX3256.VHD 的 VHDL 设计文件。

4-6.答案

MAX3256 顶层文件

```
 LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY MAX3256 IS
PORT (INA,INB,INCK: IN STD_LOGIC;
INC: IN STD_LOGIC;
E,OUT:OUT STD_LOGIC);
END ENTITY MAX3256;
ARCHITECTURE ONE OF MAX3256 IS
COMPONENT LK35 --调用 LK35 声明语句
PORT(A1,A2:IN STD_LOGIC;
CLK:IN STD_LOGIC;
Q1,Q2:OUT STD_LOGIC);
END COMPONENT;
COMPONENT D --调用 D 触发器声明语句
 PORT(D,C:IN STD_LOGIC;
CLK:IN STD_LOGIC;
```

```
 Q:OUT STD_LOGIC);
END COMPONENT;
COMPONENT MUX21--调用二选一选择器声明语句
PORT(B,A:IN STD_LOGIC;
S:IN STD_LOGIC;
C:OUT STD_LOGIC);
END COMPONENT;
SIGNAL AA,BB,CC,DD: STD_LOGIC;
BEGIN
u1: LK35 PORT MAP(A1=>INA,A2=>INB,CLK=INCK,
Q1=>AA,Q2=>BB);
u2: D PORT MAP(D=>BB;CLK=>INCK,C=>INC,Q=>CC);
u3: LK35 PORT MAP (A1=>BB,A2=>CC,CLK=INCK,
Q1=>DD,Q2=>OUT1)；
u4: MUX21 PORT MAP (B=>AA,A=>DD,S=>BB,C=>E);
END ARCHITECTURE ONE;
```

设计含有异步清零和计数使能的 16 位二进制加减可控计数器。

4-7.答案：

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY CNT16 IS
PORT(CLK,RST,EN:IN STD_LOGIC;
             CHOOSE:IN BIT;
             SETDATA:BUFFER INTEGER RANCE 65535 DOWNTO 0;
             COUT: BUFFER INTEGER RANCE 65535 DOWNTO 0);
END CNT16;
ARCHITECTURE ONE OF CNT16 IS
BEGIN
      PROCESS(CLK,RST,SDATA)
          VARIABLE QI:STD_LOGIC_VECTOR(65535 DOWNTO 0);
BEGIN IF RST='1' THEN --计数器异步复位
                                QI:=(OTHERS=>'0');
ELSIF SET=’1’ THEN--计数器一步置位
QI:=SETDATA;
ELSIF CLK'EVENT AND CLK='1' THEN --检测时钟上升沿
                   IF EN=’1’ THEN －检测是否允许计数
IF CHOOSE=’1’ THEN --选择加法计数
QI:=QI+1; --计数器加一
ELSE QI=QI-1; --计数器加一
            END IF;
END IF;
END IF;
     COUT<=QI;--将计数值向端口输出
```

END PROCESS;
END ONE;
第五章
5-1 归纳利用 Quartus II 进行 VHDL 文本输入设计的流程：从文件输入一直到 SignalTap II 测试。P95~P115
答：1 建立工作库文件夹和编辑设计文件；2 创建工程；3 编译前设置；4 全程编译；5 时序仿真；6 引 脚锁定；7 配置文件下载；8 打开 SignalTap II 编辑窗口；9 调入 SignalTap II 的待测信号；10 SignalTap II 参数设置；11 SignalTap II 参数设置文件存盘；12 带有 SignalTap II 测试信息的编译下载；13 启动 SignalTap II 进行采样与分析；14 SignalTap II 的其他设置和控制方法。
第六章
6-1 什么是固有延时?什么是惯性延时?P150~151
答：固有延时(Inertial Delay)也称为惯性延时，固有延时的主要物理机制是分布电容效应。
6-2 δ 是什么?在 VHDL 中，δ 有什么用处?P152 δ 是什么?
答：在 VHDL 仿真和综合器中，默认的固有延时量（它在 数学上是一个无穷小量），被称为 δ 延时。 在 VHDL 中，δ 有什么用处?答：在 VHDL 信号赋值中未给出固 有延时情况下，VHDL 仿真器和综合器将自动为 系统中的信号赋值配置一足够小而又能满足逻辑排序的延时量 δ；使并行语句和顺序语句中的并列赋值逻辑得以正 确执行。
6-4 说明信号和变量的功能特点，以及应用上的异同点。P128~P129
答：变量：变量是一个局部量，只能在进程和子程序中使用。变量不能将信息带出对它做出定义的当前结构。 变量的赋值是一种理想化的数据传输，是立即发生的，不存在任何延时行为。变量的主要作用是在进程中作为临时 的数据存储单元。 信号：信号是描述硬件系统的基本数据对象，其性质类似于连接线；可作为设计实体中并行语句模块间的 信息交流通道。信号不但可以容纳当前值，也可以保持历史值；与触发器的记忆功能有很好的对应关系。
6-5 在 VHDL 设计中，给时序电路清零(复位)有两种力方法，它们是什么?
解：设 Q 定义成信号，一种方法：Q<=“000„000”； 其中“000„000”反映出信号 Q 的位宽度。第二种方 法：Q<=(OTHERS=>‘0’)；其中 OTHERS=>‘0’不需要给出信号 Q 的位宽度，即可对 Q 清零。
6-6 哪一种复位方法必须将复位信号放在敏感信号表中?给出这两种电路的 VHDL 描述。
解：边沿触发复位信号要将复位信号放在进程的敏感信号表中。
（1） 边沿触发复位信号

```
,,,,,,,,,,,,,,,,,-
ARCHITECTURE bhv 0F DFF3 IS
SIGNAL QQ:STD_LOGIC;
BEGIN PROCESS(RST)
 BEGIN
IF RST’EVENT AND RST=‘1' THEN
QQ<=(OTHERS=>‘0’);
                        END IF;
            END PROCESS;
            Q1<=QQ;
      END;
     ,,,,,,,,,,,,,,,,,
```

（2） 电平触发复位信号

,,,,,,,,,,,,,,,,.

```
ARCHITECTURE bhv 0F DFF3 IS
SIGNAL QQ:STD_LOGIC;
BEGIN
        PROCESS(CLK)
        BEGIN
                    IF RST=‘1' THEN
                         QQ<=(OTHERS=>‘0’);
                   END IF;
       END PROCESS;
       Q1<=QQ;
END;
```

,,,,,,,,,,,,,,,,

6-7 什么是重载函数?重载算符有何用处?如何调用重载算符函数?

答：（1）什么是重载函数? 根据操作对象变换处理功能。

（2）重载算符有何用处? 用于两个不同类型的操作数据自动转换成同种数据类型，并进行运算处理。

（3）如何调用重载算符函数?采用隐式方式调用，无需 事先声明。

6-8 判断下面三个程序中是否有错误，若有则指出错误所在，并给出完整程序。

程序 1:

```
Signal A,EN : std_logic;
```

,,,,,,,,,,,,,

```
Process(A, EN)
Variable B: std_log ic;
Begin
if EN=l then   B<=A;
end if;
--将“B<=A”改成“B:=A”
end process;
```

程序 2:

```
Architecture one of sample is
variable a，b，c:integer;
begin c<=a+b;
--将“c<=a+b”改成“c:=a+b”
end;
```

程序 3:

```
library ieee;
use ieee.std_logic_1164.all;
entity mux21 is
PORT(a,b:in std_logic; sel:in std_loglc;c:out std_logle;);
--将“;)”改成“)”
end sam2;
--将“sam2”改成“entity mux21”
```

```
architecture one of mux2l is
begin
      --增加“process(a,b,sel) begin ”
if sel= '0' then c:=a; else c:=b; end if;
--应改成“if sel= '0' then c<=a; else c<=b; end if;”
--增加“end process;”
end two;
--将“two”改成“architecture one”
```



7-2 LPM_ROM、LPM_RAM、LPM_FIFO 等模块与 FPGA 中嵌入的 EAB、ESB、M4K 有怎样的联系　　？

答：ACEXlK 系列为 EAB；APEX20K 系列为 ESB；　Cyclone 系列为 M4K