第5章 中小规模可编程设计实验

5.1 GAL 原理

GAL 的基本结构和工作原理：

GAL 是美国 LATTICE 半导体公司生产的可编程逻辑器件的专用商标。GAL 器件采用高性能的 CMOS 工艺，可以电擦除，具有浮栅结构。这种先进的工艺使 GAL 器件具有可以重新配置的逻辑功能、类同双极型器件的高性能（主要指速度）以及大为降低了的功耗。器件可擦除重写至少 100 次，数据可保持 20 年以上。

GAL 器件有很多种，主要有 16V8、20V8、22V10、39V18 等，本教程以 GAL20V8 为例介绍。

GAL20V8 是普通型 GAL 器件，具有 24 只引脚，与门阵列规模（乘积项×输入项）为 64×40；OLMC 数（最大输出量）为 8。

其功能框图见图 5.1.1。

其逻辑电路图见图 5.1.2。

图 5.1.2 示出了 GAL20V8 有输入缓冲器（左面 10 个缓冲器），输出三态缓冲器（右面 8 个缓冲器），输出反馈/输入缓冲器和 IMUX/输入缓冲器（中间 10 个缓冲器），8 个输出逻辑宏单元 OLMC（或门阵列包含其中），2 个输入多路开关 IMUX，与门阵列由 8×8 个与门构成，共形成 64 个乘积项。每个与门有 40 个输入端，除了 10 个引脚（2～11）固定作为输入外，还可能有其他 10 个引脚被配置成输入模式。因此，GAL20V8 最多可有 20 个引脚作为输入脚，而输出脚最多为 8 个，这就是芯片型号中两个数字的含义。

1）输出逻辑宏单元 OLMC

OLMC（Output Logic Macro Cell）的内部结构见图 5.1.3。

每个 OLMC 中有 1 个 XOR（n）端，用来控制输出信号的极性。每个 OLMC 中有 4 个多路开关，其中，PTMUX 用于控制第一乘积项；TSMUX 用作选择输出三态缓冲器的选通信号；FMUX 决定了反馈信号的来源；OMUX 则用于选择输出信号是组合的还是寄存（存储）的。这些多路开关的状态取决于结构控制字中 AC0、AC1（n）的值（OLMC（15）和 OLMC（22）除外，见注），详见表 5.1.1。

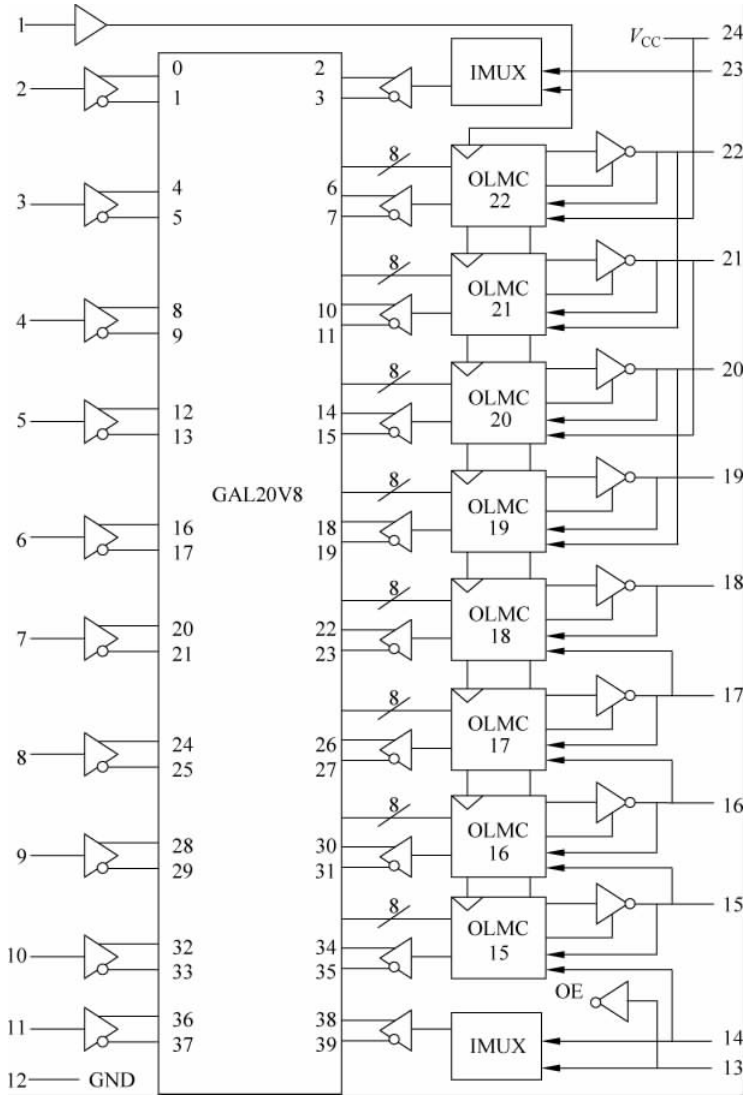

图 5.1.1 GAL20V8 功能框图

表 5.1.1 OLMC 的配置控制表

SYN	AC0	AC1(*n*)	XOR(*n*)	配置功能	输出极性	备 注
1	0	1	—	输入模式	—	1 脚和 13 脚为数据输入,三态门不通
1	0	0	0	所有输出是组合的	低有效	1 脚和 13 脚为数据输入,三态门总是选通
1	0	0	1		高有效	
1	1	1	0	所有输出是组合的	低有效	1 脚和 13 脚为数据输入,三态门选通信号是第一乘积项
1	1	1	1		高有效	
0	1	1	0	组合输出	低有效	1 脚 = CK,13 脚 = $\overline{OE}$,至少另有一个 OLMC 是寄存的(存储的)
0	1	1	1		高有效	
0	1	0	0	寄存输出	低有效	1 脚=CK,13 脚=$\overline{OE}$
0	1	0	1		高有效	

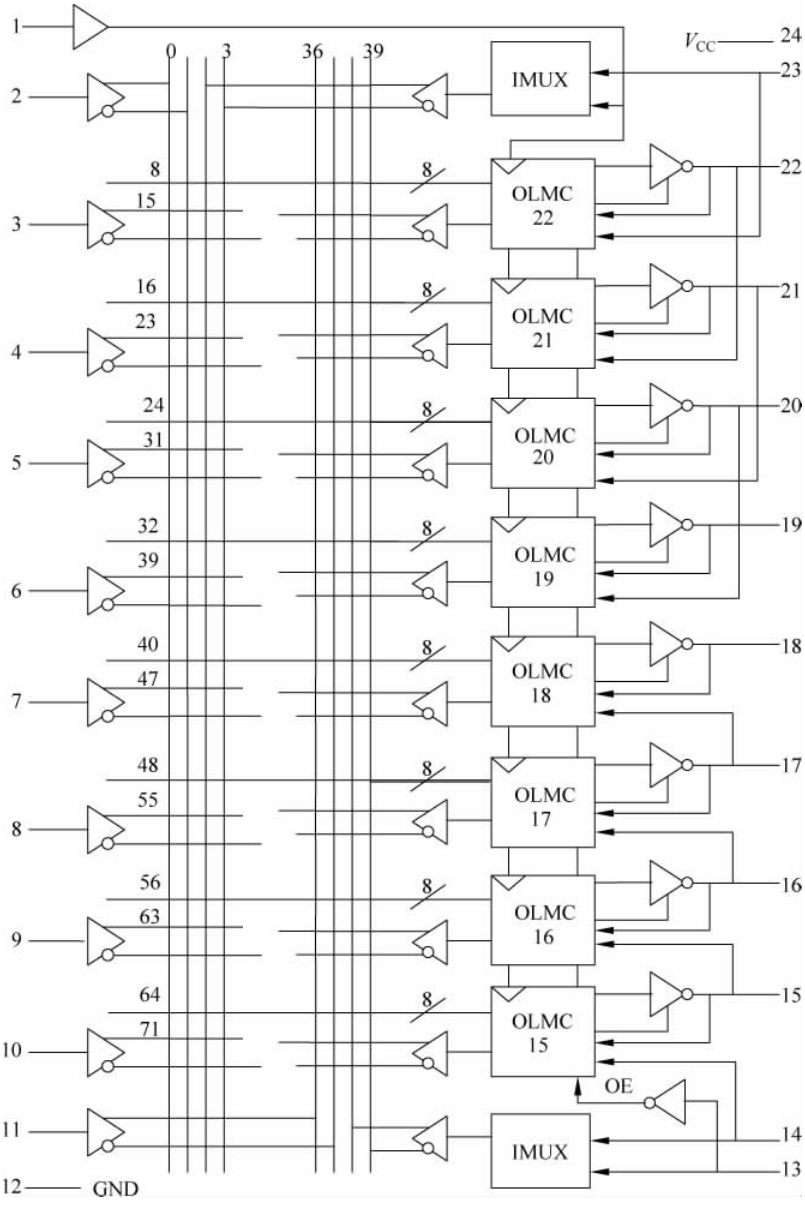

图 5.1.2 GAL20V8 逻辑电路图

注：当 $n=15$ 和 $n=22$ 时，用$\overline{\text{SYN}}$代替 AC1(m)，用 SYN 代替 AC0。SYN 决定 GAL 器件是否具有寄存输出能力。

2) 行地址映射

行地址 0～39 对应于与门阵列，每行包含 64 位。行地址 40 是电子标签字。行地址 41～59 由制造商保留，用户不能使用。行地址 60 是结构和输出极性控制字，共 82 位(见图 5.1.4)。行地址 61 仅包含一位，用于加密。行地址 62 为制造商保留。行地址 63 也只包含一位，用于整体擦除。

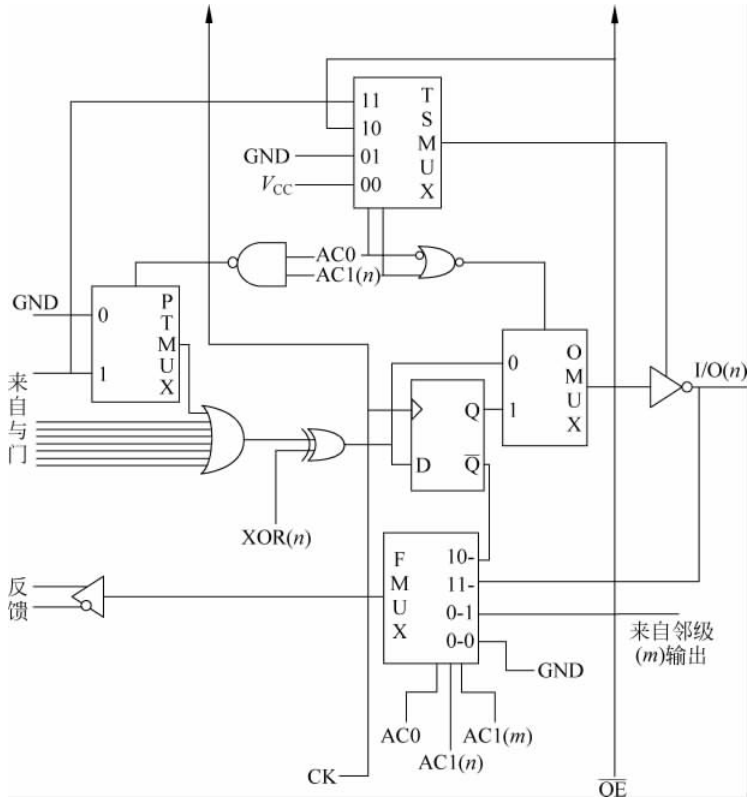

图 5.1.3 OLMC(*n*)的内部结构图

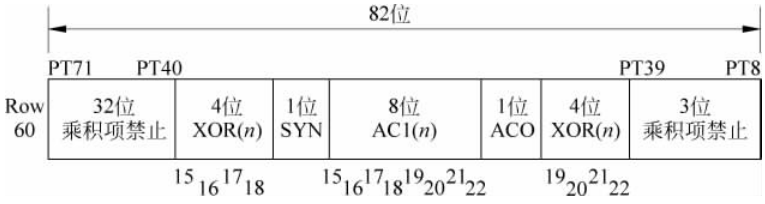

图 5.1.4 GAL20V8 结构控制字

5.2 TPC 多用编程器

TPC 多用编程器是一种多功能的编程器，它能完成对 EPROM、EEPROM 和 GAL 的编程。对 GAL 的编程采用标准的 JEDEC 文件格式，从而可采用 FM、ABEL、CUPL 等软件进行 GAL 逻辑设计，可在屏幕上手动编辑 GAL 芯片的熔丝图。

5.2.1 系统软件说明

1）系统软件构成

PLD 软件盘上包括如下文件：

PLD.EXE：固化软件，可完成对 EPROM、GAL 的各种操作。

PLDHELP.DOC：在线帮助文本文件，提供再现帮助信息。

ROMTYPE.DOC：部分 EPROM 芯片 ID 与生产厂家名称、编程电压对照表。

FM.EXE：逻辑设计编译软件，用于完成由逻辑表达式到 JEDEC 格式文件的转换。

2）PLD.EXE 的使用方法

在 DOS 提示符下，输入 PLD 并按 Enter 键，即可启动。

PLD 运行后，首先显示 PLD 软件版本号，然后在屏幕上端显示主菜单，屏幕下端显示提示信息。

主菜单内容如下：

```
UVEPROM EEPROM GAL File Test Dos Inst Quit
```

利用光标控制键“↑”、“↓”、“→”和“←”可任意选择所需要的功能，也可以用菜单命令中每一项的第一个大写字母来选择该项功能。

UVEPROM：不需要掌握。

EEPROM：不需要掌握。

GAL：对 GAL 芯片的操作。

File：不需要掌握。

Test：不需要掌握。

Dos：不退出 PLD 软件环境，进入 DOS 操作，用 EXIT 返回。

Inst：不需要掌握。

Quit：退出系统。

提示信息栏安排如下：

F1——HELP　　F2——Exec　　Esc——Exit

F1——当处于主菜单选择状态下时，首先选择某一项功能，再按 F1 键即可得到该功能的在线帮助信息。

F2——当选择了某一项操作以后，在屏幕提示下还会让用户输入文件名、地址或其他信息，这些信息输入完后，按 F2 键系统就会执行该功能操作。所以在本系统中 F2 键为“执行”键，EPROM 或 GAL 的读写、文件的转换等最终按 F2 键才能执行。

ESC——使用 ESC 键可以退出某项操作。

3）GAL 编程

进入 PLD 主菜单后，选择 GAL 项并按 Enter 键可进入 GAL 子菜单。

对 GAL 的操作共有下列 9 种：

（1）TYPE——类型设定

可供选择的类型有 GAL20V8/A/B/C/D，GAL22V10 和 GAL16V8/A/B/C/D 请注意它们的写入方法和编程电压完全不同，必须正确选择。

（2）LOAD——装入 JEDEC 文件

该命令可以将标准的 JEDEC 文件读到内存，以供编辑修改和编程操作，文件装入后，自动进入编程状态，关于编辑方法请阅读 EDIT 部分。输入文件名时，对于后缀为 JED 的文件可以省略后缀。

(3) SAVE——存入 JEDEC 文件

该命令是 LOAD 的逆操作,是将内存中的熔丝信息按 JEDEC 标准存入文件中,同样文件默认后缀为 JED。

(4) READ——读芯片内部熔丝图到内存中

该命令可将未加密的 GAL 芯片内容读入到内存中,并自动进入到编辑状态。

(5) EDIT——熔丝图编辑

该命令可用于对芯片内部结构的手工编辑,可编辑的内容有:

① 芯片电子标签(Elec Signaturc)

按字母键 E 后可输入电子标签,内容可以是任何字符,但长度不应该超过 8 个,输入以 Enter 键结束。

② 芯片的结构控制位

芯片的结构控制位有 SYN、AC0、AC1、XOR(极性控制),PT(积禁止项)和与阵列项。对于这些熔丝,统一用"X"号代表"0",用"—"号代表 1,按 X 键、—键或空格键可以修改熔丝状态;组合键 Alt+—和 Alt+X 可以修改一行的熔丝状态,但不含 PT 项。

对于 SYN、AC0、AC1、XOR 可分别按 S、0、1、P 键后再修改,其他熔丝可用"↑"、"↓"、"→"、"←"、PgUp、PgDn 键将光标移动到相应位置后,进行修改。

注意:*屏幕每次显示一个输出引脚的熔丝图,共有 8 个,引脚号为 22~15。*

一般来说,JEDEC 文件中不含 GAL 的电子标签信息,因此编程前,可以用 EDIT 命令写入电子标签后再完成固化。

(6) Programming——编程

编程操作是将内存中的熔丝图写到 GAL 芯片中,同时给出校验信息,如果需要加密可接着使用 Mask 功能操作。

(7) VERIFY——校验

该命令读出未加密的 GAL 芯片内容并与内存熔丝内容相比较,且报告校验结果。

(8) MASK——写加密位

该命令用于对 GAL 芯片的保密位进行操作,完成 Mask 操作后,将不能读 GAL 芯片的熔丝图。

(9) ERASE——全片擦除操作

该命令用于片擦除。在一般情况下,用户无须使用这项功能,因为每次进行编程操作时,都首先自动进行擦除操作。

GAL 编程的一般步骤:

(1) 使用 Load 命令装入 JEDEC 文件到内存。

(2) 编辑熔丝图,一般仅需编辑电子标签字。

(3) 使用 Programming 命令将熔丝图固化到 GAL 芯片中。

(4) 如果必要可用 Mask 命令写 GAL 芯片的保密位。

5.2.2 FM 使用说明

FM 是 GAL 逻辑设计软件 FAST-MAP 的缩写,该软件可以完成对 GAL20V8、

GAL16V8 的逻辑机设计，其使用步骤如下：

(1) 用文本编辑程序编辑产生后缀为 PLD 的逻辑设计源程序文件。

(2) 使用 FM 编译源程序文件产生 JEDEC 文件，其后缀为 JED。

(3) 运行 PLD. EXE 将 JEDEC 文件固化到 GAL 芯片中。

1) FM 的语法规则

GAL 逻辑设计源程序的后缀应为 PLD，可以用任何编辑软件形成。

下面是一个源程序的例子：

```
PLD20V8              ; GAL 型号标志
BASIC GATE           ; 标题行
WJP Mar.28 2001      ; 姓名,日期
BGATES               ; 电子标签
NC  A  B  C  D  E  F  G  H  I  J  GND
NC  N  M  L  Q6 Q5 Q4 Q3 Q2 Q1 K
VCC                  ; 引脚表
Q1 = /A              ; 输出逻辑表达式
Q2 = B*C
Q3 = D + E + F
Q4 = /G + /H + /L
Q5 = /I*/J*/K
Q6 = M*/N + /M*N

DESCRIPTION          ; 说明部分
This is a sample of basic gates
END
```

其内部结构应符合下列规定：

(1) 第 1 行为 GAL 型号标志，必须始于第一行第一列，必须以大写字母 PLD 开头。

(2) 第 2 行、第 3 行为标题，包括设计者姓名、日期等，可缺省。

(3) 第 4 行为电子标签。

(4) 从第 5 行开始为引脚表，是器件引脚的定义。引脚名最多可用 8 个字符，之间用空格分隔，引脚名必须按引脚号的次序排列，可占用多行。其中，电源用 V_{CC} 表示，地用 GND 表示，不用的引脚用 NC 表示，引脚可随意命名，但不能重名。

(5) 引脚表下面为输出逻辑表达式。逻辑表达式可含有“*”、“/”、“+”三种逻辑运算符。由于 GAL20V8 硬件结构的限制，表达式中的“+”不应多于 8 个(有三态控制的应小于 8 个)，参加“与”运算的引脚不应多于 20 个。式中不得有任何括号，FM 不对表达式进行任何化简。

每一个输出引脚可以也只能用“=”、“:=”、“.OE=”三种符号和表达式进行连接。其中，“=”表明等式右边的表达式直接决定输出的状态；“:=”表明仅当时钟脉冲的上升沿到来时，引脚输出才会锁定到等式右边的表达式所给出的值上面；“.OE=”表明仅当等式右边的表达式(此时表达式只能由“与”、“非”两种运算组成，不能有“或”运算)为真时，左边引脚的输出才能为有效电平，否则就会保持高阻状态，形成三态控制。

注意：如果对某个引脚进行了三态输出控制，那么全部输出引脚的输出表达式中最多只能有 7 个或项，并且需列出全部输出引脚的三态控制逻辑，无须三态控制的引脚用 V_{CC} 作

为控制逻辑。

GAL 允许在输出引脚前加非运算符，进行负逻辑设计。其关系详见表 5.2.1。

表 5.2.1 输出极性表

引脚表 采用的极性	逻辑方程 采用的极性	输出	
		有效电平	反馈项极性
$\overline{X}$	$\overline{X}$	H	$\overline{X}$
$\overline{X}$	X	L	$\overline{X}$
X	$\overline{X}$	L	X
X	X	H	X

值得注意的是由于 GAL 结构上的特殊性，有时某些合法的表达式会产生编译错误，这时应通过阅读 GAL 器件手册或调换某些引脚来解决。

(6) 说明部分以关键字 DESCRIPTION 开始，后面可以接任何文字，最后以 END 结束，FM 将这一部分理解为注释，对逻辑设计无本质影响。

(7) 源程序中要注意下列几点：

① 每一个语句行均可加注释，注释前必须加分号；

② 一个 PLD 文件的最大长度为 200 行；

③ 最后一行必须以 Enter 键结束；

④ 大小写字母区别对待。

2) FM 的使用方法

(1) 在 DOS 提示符下，输入 FM 并按 Enter 键。

(2) 输入逻辑设计文件名，可省略 PLD 后缀，按 Enter 键后 FM 将自动检查源文件中逻辑表达式的合法性，检查通过后将进行下一步操作。

(3) FM 显示下列操作菜单：

```
Fast Map Menu———Current Source File—> 20V8.PLD
1) Crcate Document File (source plus pinout)
2) Crcate Fuse Plot File (buman readable fuse map)
3) Crcate Jedec File (programmer fuse map)
4) Get a new Source File
5) Exit from fuse map
Please enter number corresponding to desired operation
```

其中：

第 1 项为产生一个以 LST 为后缀的列表文件；

第 2 项为建立以 PLT 为后缀的熔丝图文件；

第 3 项为建立以 JED 为后缀的 JEDEC 文件；

第 4 项为读入一个新的源文件；

第 5 项为退出 FM 操作。

列表文件和熔丝图文件可供核对参考，JED 文件将用来完成编程。如果源程序中含有语法错误或存在结构冲突问题，FM 将指出。出错时可用 Ctrl+Break 组合键退出。

5.3 GAL 实现的基本逻辑器件

5.3.1 GAL 实现的基本门电路

1. 实验目的

(1) 了解 GAL 的工作原理。
(2) 掌握 GAL 的应用方法。
(3) 学会用 GAL 实现基本门电路器件。

2. 实验任务与步骤

(1) 用 GAL20V8 实现基本门电路。
(2) 按图 5.3.1 进行设计,绘出框图和引脚分配图。

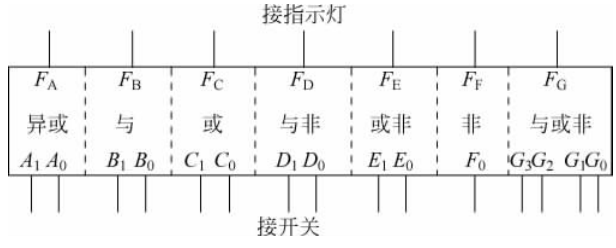

图 5.3.1 基本门电路示意图

(3) 正确形成源程序文件、列表文件、JEDEC 文件。
(4) 正确编辑熔丝图,并把熔丝图写入 GAL 芯片。
(5) 检验 GAL 芯片的功能。

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;
(3) 示波器一台;
(4) 万用表一只;
(5) 集成电路: GAL20V8 一片。

4. 预习要求

(1) 做好实验预习,重点复习本实验原理和有关 GAL 书籍、资料。
(2) 熟练掌握 GAL 的编程方法。
(3) 掌握熔丝图的编辑和 GAL 的制作。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 画好实验用表格。

(6) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程,包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.3.2 GAL 实现的常用触发器电路

1. 实验目的

(1) 了解 GAL 的工作原理。
(2) 掌握 GAL 的应用方法。
(3) 学会用 GAL 实现触发器器件。

2. 实验任务与步骤

(1) 用 GAL20V8 实现常用触发器电路。
(2) 按图 5.3.2 进行设计,绘出框图和引脚分配图。

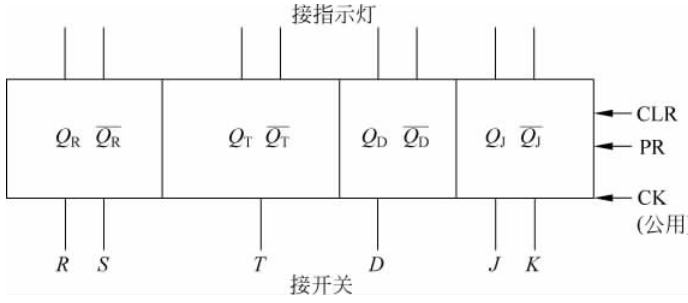

图 5.3.2 常用触发器电路示意图

(3) 正确形成源程序文件、列表文件、JEDEC 文件。
(4) 正确编辑熔丝图,并把熔丝图写入 GAL 芯片。
(5) 分别列出 RS 锁存器、D 触发器、T 触发器和 JK 触发器的真值表并检验 GAL 芯片的功能。

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;

(3) 示波器一台；

(4) 万用表一只；

(5) 集成电路：GAL20V8 一片。

4. 预习要求

(1) 做好实验预习，重点复习本实验原理和有关 GAL 书籍、资料。

(2) 熟练掌握 GAL 编程方法。

(3) 掌握熔丝图的编辑和 GAL 的制作。

(4) 画好实验验证用逻辑布线图或物理布线图。

(5) 画好实验用表格。

(6) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。

(2) 实验任务及要求。

(3) 逻辑设计过程，包括化简的步骤。

(4) 画好实验验证用逻辑布线图或物理布线图。

(5) 按要求填写各实验表格。

(6) 整理、分析实验数据和结果，列表比较实验任务的理论分析值和实验结果值。

(7) 写出心得体会。

5.4 GAL 组合逻辑设计

5.4.1 GAL 实现的优先权编码器

1. 实验目的

(1) 了解 GAL 的工作原理。

(2) 掌握 GAL 的应用方法。

(3) 掌握用 GAL 设计优先权编码器电路的方法。

(4) 熟悉 GAL 编程的过程。

2. 实验任务与步骤

(1) 用 GAL 实现一个十六位优先权编码器。

(2) 按图 5.4.1 进行设计，绘出逻辑图和引脚分配图。

(3) 使十六位输入的优先级别为从 I_0 至 I_{15} 由低至高，即当 I_{15} 为有效时，则屏蔽其他输入。

(4) 正确形成源程序文件、列表文件、JEDEC 文件。

(5) 正确编辑熔丝图，并把熔丝图写入 GAL 芯片。

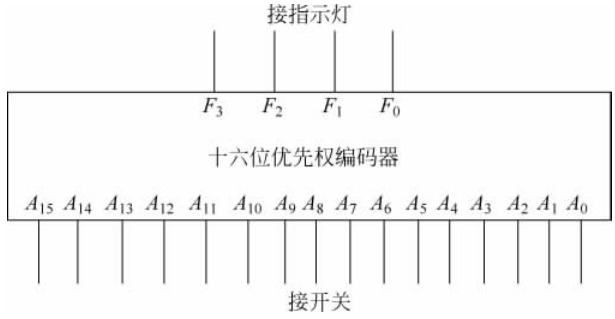

图 5.4.1 十六位优先权编码器电路示意图

(6) 测量 GAL 器件状态,测试其是否能完成设计要求。

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;
(3) 示波器一台;
(4) 万用表一只;
(5) 集成电路: GAL20V8 一片。

4. 预习要求

(1) 做好实验预习,重点复习本实验原理和有关 GAL 书籍、资料。
(2) 熟练掌握 GAL 编程方法。
(3) 掌握熔丝图的编辑和 GAL 的制作。
(4) 掌握用 GAL 设计优先权编码器电路的方法。
(5) 熟悉 GAL 编程的过程。
(6) 画好实验验证用逻辑布线图或物理布线图。
(7) 画好实验用表格。
(8) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程,包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.4.2 GAL 实现的 3-8 译码器

1. 实验目的

(1) 了解 GAL 的工作原理。
(2) 掌握 GAL 的应用方法。
(3) 掌握用 GAL 设计 3-8 译码器电路的方法。
(4) 熟悉 GAL 编程的过程。

2. 实验任务与步骤

(1) 用 GAL 实现一个 3-8 译码器。
(2) 按图 5.4.2 进行设计,绘出逻辑图和引脚分配图。
(3) 正确形成源程序文件、列表文件、JEDEC 文件。
(4) 正确编辑熔丝图,并把熔丝图写入 GAL 芯片。
(5) 测量 GAL 器件状态,测试其是否能完成设计要求。

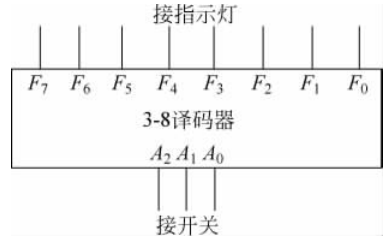

图 5.4.2 3-8 译码器电路示意图

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;
(3) 示波器一台;
(4) 万用表一只;
(5) 集成电路:GAL20V8 一片。

4. 预习要求

(1) 做好实验预习,重点复习本实验原理和有关 GAL 书籍、资料。
(2) 熟练掌握 GAL 编程方法。
(3) 掌握熔丝图的编辑和 GAL 的制作。
(4) 掌握用 GAL 设计 3-8 译码器电路的方法。
(5) 熟悉 GAL 编程的过程。
(6) 画好实验验证用逻辑布线图或物理布线图。
(7) 画好实验用表格。
(8) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程,包括化简的步骤。

(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.4.3 GAL 实现的多路转换器

1. 实验目的

(1) 了解 GAL 的工作原理。
(2) 掌握 GAL 的应用方法。
(3) 掌握用 GAL 设计多路转换器电路的方法。
(4) 熟悉 GAL 编程的过程。

2. 实验任务与步骤

(1) 用 GAL 实现一个四位 4-1 多路转换器。
(2) 按图 5.4.3 进行设计,绘出逻辑图和引脚分配图。

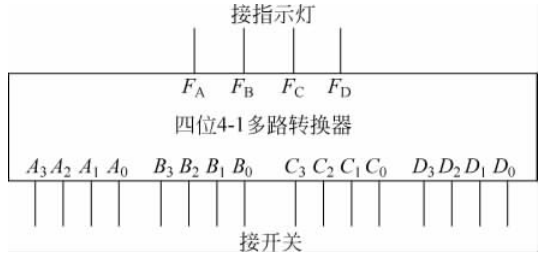

图 5.4.3 四位 4-1 多路转换器电路示意图

(3) 正确形成源程序文件、列表文件、JEDEC 文件。
(4) 正确编辑熔丝图,并把熔丝图写入 GAL 芯片。
(5) 测量 GAL 器件状态,测试其是否能完成设计要求。

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;
(3) 示波器一台;
(4) 万用表一只;
(5) 集成电路: GAL20V8 一片。

4. 预习要求

(1) 做好实验预习,重点复习本实验原理和有关 GAL 书籍、资料。
(2) 熟练掌握 GAL 编程方法。

(3) 掌握熔丝图的编辑和 GAL 的制作。
(4) 掌握用 GAL 设计多路转换器电路的方法。
(5) 熟悉 GAL 编程的过程。
(6) 画好实验验证用逻辑布线图或物理布线图。
(7) 画好实验用表格。
(8) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程,包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.5 GAL 时序逻辑设计

5.5.1 GAL 实现的八位双向通用移位寄存器

1. 实验目的

(1) 掌握 GAL 的工作原理。
(2) 掌握 GAL 的应用方法。
(3) 掌握用 GAL 设计移位寄存器逻辑电路的方法。
(4) 熟悉 GAL 编程的过程。

2. 实验任务与步骤

(1) 用 GAL 实现一个八位双向通用移位寄存器。
(2) 按图 5.5.1 进行设计,绘出逻辑图和引脚分配图。

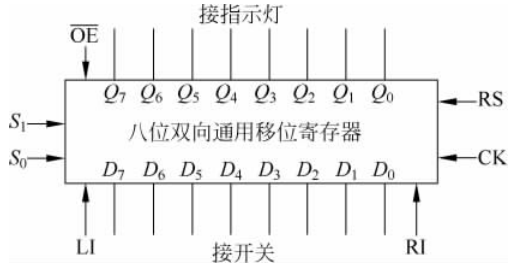

图 5.5.1 八位双向通用移位寄存器电路示意图

(3) 要求具有：①数据左移(数据由右面串行输入)；②数据右移(数据由左面串行输入)；③数据并行装入；④数据并行输出；⑤高阻抗输出。见表 5.5.1。

表 5.5.1 八位双向通用移位寄存器功能表

$\overline{OE}$	S_1	S_0	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_0	说 明
1	×	×	Z	Z	Z	Z	Z	Z	Z	Z	高阻输出
0	0	0	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_0	保持
0	0	1	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_0	RI	左移
0	1	0	LI	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	右移
0	1	1	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	并行装入

(4) 正确形成源程序文件、列表文件、JEDEC 文件。

(5) 正确编辑熔丝图，并把熔丝图写入 GAL 芯片。

(6) 测量 GAL 器件状态，测试其是否能完成设计要求。

3. 实验器材

(1) 数字逻辑实验箱一台；

(2) 直流稳压电源一台；

(3) 示波器一台；

(4) 万用表一只；

(5) 集成电路：GAL20V8 一片。

4. 预习要求

(1) 做好实验预习，重点复习本实验原理和有关 GAL 书籍、资料。

(2) 熟练掌握 GAL 编程方法。

(3) 掌握熔丝图的编辑和 GAL 的制作。

(4) 掌握用 GAL 设计移位寄存器逻辑电路的方法。

(5) 熟悉 GAL 编程的过程。

(6) 画好实验验证用逻辑布线图或物理布线图。

(7) 画好实验用表格。

(8) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。

(2) 实验任务及要求。

(3) 逻辑设计过程，包括化简的步骤。

(4) 画好实验验证用逻辑布线图或物理布线图。

(5) 按要求填写各实验表格。

(6) 整理、分析实验数据和结果，列表比较实验任务的理论分析值和实验结果值。

(7) 写出心得体会。

5.5.2 GAL 实现的四位可逆计数器

1. 实验目的

(1) 掌握 GAL 的工作原理。
(2) 掌握 GAL 的应用方法。
(3) 掌握用 GAL 设计可逆计数器逻辑电路的方法。
(4) 熟悉 GAL 编程的过程。

2. 实验任务与步骤

(1) 用 GAL 实现一个四位可逆计数器。
(2) 按图 5.5.2 进行设计,绘出逻辑图和引脚分配图。
(3) 要求具有清除和置数功能,高阻抗输出,见表 5.5.2。
(4) 正确形成源程序文件、列表文件、JEDEC 文件。
(5) 正确编辑熔丝图,并把熔丝图写入 GAL 芯片。
(6) 测量 GAL 器件状态,测试其是否能完成设计要求。

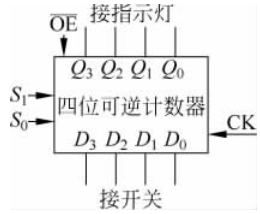

图 5.5.2 四位可逆计数器电路示意图

表 5.5.2 四位可逆计数器功能表

控制		状态				功能
S_1	S_0	Q_3	Q_2	Q_1	Q_0	
0	0	0	0	0	0	清零
0	1	D_3	D_2	D_1	D_0	置数
1	0					加 1
1	1					减 1

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;
(3) 示波器一台;
(4) 万用表一只;
(5) 集成电路: GAL20V8 一片。

4. 预习要求

(1) 做好实验预习,重点复习本实验原理和有关 GAL 书籍、资料。
(2) 熟练掌握 GAL 编程方法。
(3) 掌握熔丝图的编辑和 GAL 的制作。
(4) 掌握用 GAL 设计可逆计数器逻辑电路的方法。
(5) 熟悉 GAL 编程的过程。

(6) 画好实验验证用逻辑布线图或物理布线图。

(7) 画好实验用表格。

(8) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。

(2) 实验任务及要求。

(3) 逻辑设计过程,包括化简的步骤。

(4) 画好实验验证用逻辑布线图或物理布线图。

(5) 按要求填写各实验表格。

(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。

(7) 写出心得体会。

5.6 GAL 应用电路设计

5.6.1 GAL 实现的滚动移位器

1. 实验目的

(1) 掌握 GAL 的工作原理。

(2) 掌握 GAL 的应用方法。

(3) 掌握用 GAL 设计滚动移位器电路的方法。

(4) 熟悉 GAL 编程的过程。

2. 实验任务与步骤

(1) 用 GAL 实现一个滚动移位器。

(2) 按图 5.6.1 进行设计,绘出逻辑图和引脚分配图。

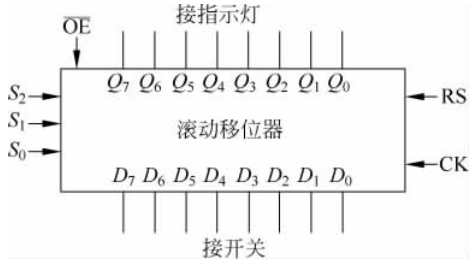

图 5.6.1 滚动移位器电路示意图

(3) $S_2S_1S_0$ 为移位次数选择控制位,$D_7 \sim D_0$ 为数据输入,$Q_7 \sim Q_0$ 为滚转后的数据输出,RS 是复位控制端,RS=1 表示系统复位。功能见表 5.6.1。

(4) 正确形成源程序文件、列表文件、JEDEC 文件。

(5) 正确编辑熔丝图,并把熔丝图写入 GAL 芯片。

(6) 检验 GAL 芯片的功能。

表 5.6.1 滚动移位器功能表

S_2	S_1	S_0	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_0
0	0	0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	1	D_6	D_5	D_4	D_3	D_2	D_1	D_0	D_7
0	1	0	D_5	D_4	D_3	D_2	D_1	D_0	D_7	D_6
0	1	1	D_4	D_3	D_2	D_1	D_0	D_7	D_6	D_5
1	0	0	D_3	D_2	D_1	D_0	D_7	D_6	D_5	D_4
1	0	1	D_2	D_1	D_0	D_7	D_6	D_5	D_4	D_3
1	1	0	D_1	D_0	D_7	D_6	D_5	D_4	D_3	D_2
1	1	1	D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1

3. 实验器材

(1) 数字逻辑实验箱一台；
(2) 直流稳压电源一台；
(3) 示波器一台；
(4) 万用表一只；
(5) 集成电路：GAL20V8 一片。

4. 预习要求

(1) 做好实验预习，重点复习本实验原理和有关 GAL 书籍、资料。
(2) 熟练掌握 GAL 编程方法。
(3) 掌握熔丝图的编辑和 GAL 的制作。
(4) 掌握用 GAL 设计滚动移位器电路的方法。
(5) 熟悉 GAL 编程的过程。
(6) 画好实验验证用逻辑布线图或物理布线图。
(7) 画好实验用表格。
(8) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程，包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果，列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.6.2 GAL 实现的四位比较器

1. 实验目的

(1) 掌握 GAL 的工作原理。
(2) 掌握 GAL 的应用方法。
(3) 掌握用 GAL 设计四位比较器电路的方法。
(4) 熟悉 GAL 编程的过程。

2. 实验任务与步骤

(1) 用 GAL 实现一个四位比较器。
(2) 按图 5.6.2 进行设计,绘出逻辑图和引脚分配图。
(3) 功能见表 5.6.2。
(4) 正确形成源程序文件、列表文件、JEDEC 文件。
(5) 正确编辑熔丝图,并把熔丝图写入 GAL 芯片。
(6) 检验 GAL 芯片的功能。

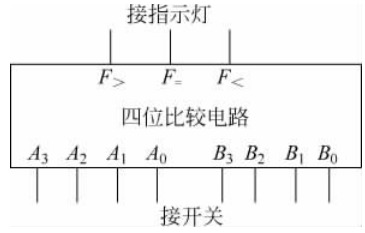

图 5.6.2 四位比较电路示意图

表 5.6.2 四位比较器的部分真值表

比较输入				输出		
A_3 B_3	A_2 B_2	A_1 B_1	A_0 B_0	$A>B$	$A<B$	$A=B$
$A_3>B_3$	d	d	d	1	0	0
$A_3<B_3$	d	d	d	0	1	0
$A_3=B_3$	$A_2>B_2$	d	d	1	0	0
$A_3=B_3$	$A_2<B_2$	d	d	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1>B_1$	d	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1<B_1$	d	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0>B_0$	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0<B_0$	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	0	0	1

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;
(3) 示波器一台;
(4) 万用表一只;
(5) 集成电路: GAL20V8 一片。

4. 预习要求

(1) 做好实验预习,重点复习本实验原理和有关 GAL 书籍、资料。
(2) 熟练掌握 GAL 编程方法。
(3) 掌握熔丝图的编辑和 GAL 的制作。
(4) 掌握用 GAL 设计四位比较器电路的方法。
(5) 熟悉 GAL 编程的过程。
(6) 画好实验验证用逻辑布线图或物理布线图。
(7) 画好实验用表格。
(8) 做好预习报告。

5. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程,包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.7 Synario 软件的安装和 ISP 器件下载

5.7.1 Synario 软件的安装

Synario 是由 DATA I/O 公司开发的一个通用电子设计软件,在 Windwos 平台上运行。它支持 ABEL-HDL 语言、VHDL 语言、原理图三种电子设计方式以及这些设计方式的混合使用,是可编程器件设计的优秀工具之一。通过与器件公司合作,研制适配软件(接口软件),Synario 支持许多公司(如 Altera、AMD、Lattice、Philips、Xilinx 等)的可编程器件。器件公司发行支持该公司器件设计的 Synario 版本。Synario 软件装在一张光盘上,启动光盘上的 CDSETUP 程序(或类似程序),在该程序的安装指导下,可一步步将 Synario 正确安装。

5.7.2 ISP 器件下载

ISP 是 In System Program(在系统可编程)的缩写。ISP 器件的出现实现了先安装器件在电路板上,然后对其编程的愿望。这样在不更改系统硬件的条件下,能够对系统进行修改、升级。ISP 器件的在系统编程(下载)是通过 JTAG 接口实现的。JTAG 是 Joint Test Action Group 的简称。JTAG 接口标准原是为采用边界扫描法测试芯片和电路板制定的标准。ISP 主要是使用 JTAG 接口中的 TDI(Test Data Input)、TDO(Test Data Output)、

TMS(Test Mode Select)、TCK(Test Clock)信号,采用了与 JTAG 类似的方法。对 ISP 器件来说,TDI、TCK、TMS 是输入信号,TDO 是输出信号。由于在一块系统板上可能有多个 ISP 器件,为了使用一个下载插座对它们编程,同在 JTAG 接口中一样,这些 ISP 器件在系统板上也连接成"链"的形式,见图 5.7.1。

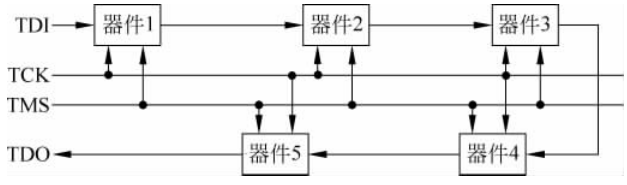

图 5.7.1 ISP 器件下载链

为了对 ISP 器件下载,在 PC 上运行的下载程序借用 PC 的打印机并行端口。PC 的打印机端口与一条下载电缆连接,下载电缆的另一端接下载插座,ISP 器件通过 ISP 器件以 TDI、TDO、TMS、TCK 等信号与下载插座相连。下载程序通过打印机数据端口向下载插座发送数据,通过打印机状态端口从下载插座接收数据。这样,在 PC 运行的下载程序就能将标准 JEDEC 文件中的数据下载到 ISP 器件中,从而实现对 ISP 器件的在系统编程。

D2H+实验箱采用两套下载系统,一套用于 Vantice 公司的 ISP(MACH)器件,一套用于 Lattice 公司的 ISP 器件。采用两套系统,主要是防止干扰。每套下载系统有一条下载电缆,一个下载插座,一个 44 脚 PLCC 插座,连接时要正确连接,注意不要接错。

MACH 下载插座的信号如下:

引脚 1 TCK	引脚 2 NC	引脚 3 TMS	引脚 4 GND	引脚 5 TDI
引脚 6 V_{CC}或 NC	引脚 7 TDO	引脚 8 GND	引脚 9 NC	引脚 10 NC

Lattice 下载插座的信号如下:

引脚 1 SCLK	引脚 2 GND	引脚 3 MODE	引脚 4 NC	引脚 5 isoEN
引脚 6 SDI	引脚 7 SDO	引脚 8 V_{CC}		

下载前,首先用下载电缆将 PC 的打印机端口和实验箱上的下载插座连接好,将 ISP 器件插入相应 44 脚 PLCC 插座,打开实验箱电源。Lattice 公司 ISP 器件的下载程序集成在 ISP Synario 中。Vantice 公司的器件下载软件是一个单独的软件,名为 machpro。使用 machpro 下载前,首先应将光盘上 maehpro 子目录下的所有文件复制到硬盘上一个独立的子目录中。

D2H+数字电路实验系统目前支持 Lattice 公司和 Vantice(AMD)公司的 PLCC 封装的 44 引脚 ISP 器件下载。

5.7.3 下载软件的使用

5.7.3.1 Vantice ISP 器件下载

在 Windows 环境下,单击 machpro 图标启动下载程序,进入下载程序界面。这是一个标准 Windows 界面,熟悉 Windows 使用方法的用户很容易掌握。这里介绍 ISP 器件下载的主要步骤。

(1) 选择 File 菜单中的 New 子菜单条目,打开一个新的下载器材链窗口,生成一个新 ISP 下载器材链。

(2) 在链文件窗口下,选择 Edit 菜单中的子菜单条目 Add Device,向下载器件链(此时为空链)中加一个新 ISP 器件。

(3) 屏幕上出现 JTAG Part Properties 对话框,用于定义新 ISP 器件下载过程中有关的特性。

① 在 Part(器件型号)对话框中选择新器件的型号,如 M4-64/32 44 PIN PLCC。

② 在 JTAG Operation 对话框中选择所需的操作。进行下载操作时,选择 P=Erase, Program&Verify,Device w/JEDEC File 选项,即进行擦除、编程、编程后与 JEDEC 文件校验。

③ 单击 Get File 选项,选择欲下载的文件。文件名出现在 JEDEC File for 对话框中。或者直接在 JEDEC File for 对话框内输入所需的 JEDEC 文件名。

④ 单击 Get File 选项,选择输出结果的文件名。文件名出现在 Output Result 对话框中。或者直接在 Output Result 对话框内输入所需的 JEDEC 文件名。此文件用于记录下载过程中出现的现象。如果不进行这一步,结果将记录在 logfile.out 文件中。

⑤ 在 State of IO pins while 对话框中选择 z=Tri-state 选项。这是下载时最常用的 IO 引脚状态选择,它指出下载时 IO 引脚处于三态(高阻)状态。

⑥ 最后单击 OK 按钮,结束新 ISP 下载器件特性的定义,回到 MACHPRO 主屏幕。

(4) 这时可以看到,一个新的 ISP 器件加到了下载链中。单击 GO 按钮,则对链中的所有 ISP 器件(这里只有一个)开始编程,即对它们下载。

5.7.3.2 Lattice ISP 器件下载

(1) 在 ISP Synario 主屏幕上,双击 ISP Down Load System 选项,将调用 ISP Chain Download 软件,进入 ISP Chain Download 界面。

(2) 单击 Configuration 子菜单中的 Port Assignment 条目,选择下载电缆所在的并行口地址。

(3) 单击 Configuration 菜单下的 Scan Board 条目,下载软件对下载链中的器件自动扫描,确定下载链中有多少器件,每个器件是何种型号。若连接无误,将会弹出目标器件列表。

(4) 在器件列表中的 File 栏内,选择下载到各器件中的相应 JEDEC 文件。

(5) 在 Option 栏内选择 Program & Verify,即编程和校验操作。

(6) 单击 Command 子菜单中的 Run Operation 条目,即启动下载操作。

5.7.3.3 Synario 软件的安装和 ISP 器件下载方法二

(1) 首先打开 alu.syn 项目文件,单击选择窗口左侧列表框的 ALU1(alu1.ab1)项,再双击窗口右侧列表框的 Compile Logic 选项,如果程序无语法错误,则编译完毕。

(2) 单击选择窗口左侧列表框的器件名 ispLSI1016E-80LJ44,再双击窗口右侧列表框的 Compile Design 进行编译设计。

(3) 单击菜单栏 Tools/Compile Design Environment 命令进入编译设计环境,单击菜单栏 Assign/Pin Locations 命令打开引脚锁定窗口,锁定引脚的方法是单击选定 Unassigned 列表框中的引脚名使其高亮显示,再双击窗口右侧引脚图中欲定义的引脚即可

完成引脚定义，可在 Assigned Pins 列表框中查看已定义的引脚。引脚锁定后再选择菜单栏 Tools/Compile 进行编译，生成 alu.jed 文件。

(4) 单击菜单栏 Tools/ispDCD 命令打开下载窗口，在下载窗口的菜单栏选择 Command/Turbo Download/Run Turbo Download 命令将生成的 JEDEC 文件下载至 ISP LSI1016E 芯片中。如果该菜单命令项呈屏蔽状态，说明下载电缆没有被检测到，请检查后再试。

5.8 ISP 可编程器件综合实验

5.8.1 简单电子琴

1. 实验目的

(1) 掌握较复杂逻辑的设计、调试。
(2) 进一步掌握用 ABEL 语言设计数字逻辑电路的方法。
(3) 熟悉 Synario 软件的使用方法。
(4) 熟悉 ISP 器件的使用。
(5) 了解音调的初步知识。

2. 实验内容

用 ABEL 语言设计一个电子琴。使用 D2H＋实验箱上的 8 个电平开关作为琴键。电平开关输出为高电平时相当于琴键按下，电平开关输出为低电平时相当于琴键松开。电子琴共有 C 调的 8 个音：1、2、3、4、5、6、7 和 i。

在 Synario 中，将设计好的程序输入、编译、连接，生成 JEDEC 格式的文件。

将 JEDEC 格式的文件下载到器件中。

在数字电路实验箱上对设计进行调试。调试时用实验箱上的小喇叭作发声装置。

3. 实验器材

(1) 数字逻辑实验箱一台；
(2) 直流稳压电源一台；
(3) 示波器一台；
(4) 万用表一只；
(5) PLCC 封装的 ISP 1016 或者 M4-64/32 一片。

4. 实验提示

C 调的音符与频率的关系如表 5.8.1 所示。

表 5.8.1 音符与频率的关系对照表

音符	1	2	3	4	5	6	7	i
频率/Hz	262	294	330	349	392	440	494	523

只要向 D2H＋实验箱上的喇叭输出某一频率的方波，喇叭就发出相应音调的声音。将实验箱喇叭区域的开关 J_1 置为开路，从“输入”插孔向驱动喇叭的三极管基极送控制信号，则控制喇叭按希望的频率发声。

设计一个多模计数器，对实验箱上的某一时钟（例如 100kHz）进行分频，产生 8 种希望的频率。注意驱动喇叭的方波占空比应是 50％，以增大音量。

根据开关电平输出确定 9 种状态（包括不发声状态）之间的转换。

ISP 1016 或者 M4-64/32 的引脚图见附录。

5. 实验步骤

1）设计思想

本项目由两个模块组成。

顶层模块 piano 主要由一个多模计数器和二分频计数器组成，二分频计数器输出送往喇叭驱动级，控制喇叭发出相应音乐。为了提高音量，输出信号的占空比为 50％，由于多模计数器输出占空比不能满足要求，因此多模计数器的输出又接了一级占空比为 50％的二分频器。在某一音频下，多模计数器的模应为：

$$M = 100\,000/(2f)$$

其中，M 代表计数器的模，f 代表音频的频率。例如，当按下“2”键时，为使二分频计数器的输出频率为 294Hz 的方波，计数器的模应为 100 000/(294×2)＝170。

底层模块 value 用来判断当前按键状态是否有效。所谓按键状态有效是指有一个键按下，且同一时刻只有一个键按下。本实验中用电平开关代表按键，即任一时刻只能有一个开关置为 1 状态。按键状态有效时，right＝1；按键状态无效时，right＝0。

2）新建项目

打开 Synario 软件，选择 File 菜单中的 New Project 选项，弹出 Create New Project 对话框。在对话框的 Directories 一栏中，选择一个目录（最好是为该项目专门建立的目录），存放该项目中的所有文件。然后在对话框的 Project FileName 一栏中输入项目文件名 piano. syn，并按“确定”按钮关闭对话框。

3）选择器件

双击 Sources in Project 窗口中 Virtual Device 项，弹出 Choose Device 对话框，选择器件 ispLSI 1016-60 PLCC44。

4）建立源文件

选择 Source 菜单中的 New，选择 ABEL-HDL Module，按 OK 按钮。在弹出对话框的 Module Name 栏中输入模块名，按 OK 按钮，进入 Synario Text Editor，开始输入源文件。本项目由顶层模块 piano 和底层模块 value 组成。

5）顶层模块 piano 的 ABEL 语言源文件

```
MODULE piano
DECLARATIONS
    //lower module declaration
value         interface(d1…d7,di_>right):
value_0       functional_block  value;
    //input
```

```
clock                pin 11:                    //时钟输入 100kHz
d1,d2,d3,d4          pin 9,40,36,3;             //琴键
d5,d6,d7,di          pin 29,10,16,43;
    //output
mu                   pin 38 istype'reg';        //音频输出
    //node
mu2                  node;
mu0,mu1              node;
q0…q7                node istype'reg'           //多模计数器
q = [q7…q0];
d = [di,d7…d1];

EQUATIONS
Value_0.[di,d7…d1] = d;
    //多模计数器,模 191、模 170、模 151、模 143、模 128、模 114、模 101、模 97
q.clk = cIock;
q := (q + 1) &  !mu2  & Value_0.right;
mu0 = (q == 190)&(d1 == 1) # (q == 169)&(d2 == 1) # (q == 150)&(d3 == 1)
    # (q == 142)&(d4 == 1);
mu1 = (q == 127)&(d5 == 1) # (q == 113)&(d6 == 1) # (q == 100)&(d7 == 1)
    # (q == 96)&(di == 1);
mu2 = mu0 # mul;
mu.clk = mu2;                                   //二分频计数器
mu = !mu;
END
```

6）底层模块 value 的 ABEL 语言源文件

```
MODULE value
DECLARATIONS
    //input
d1…d7,di         pin;                           //琴键 1,2,3,4,5,6,7,i
    //output
right            pin;                           //为 1 表示当前按键有效
    //node
right0,fight1    node;
d = [d1…d7,di];

EQUATIONS
right0 = (d == ^b00000001)} # (d == ^b00000010) # (d == ^b00000100)
     # (d == ^b00001000);
right1 = (d == ^b00010000) # (d == ^b00100000) # (d == ^b01000000)
     # (d == ^b10000000);
right = right0 # right1;                        //为 1 表示当前按键有效
END
```

7）编译源文件

选择 Source in Project 窗口中的 ABEL 语言源文件，双击 Process For Current Source 窗口中的 Compile Logic 和 Reduce Logic，如果在两命令前出现对号，则表示编译通过。否则根据错误提示修改源文件。

8) 生成 JED 文件

选择 Source in Project 窗口中的 ispLSI 1016-60 PLCC44，依次执行 Process For Current Source 窗口中的 Link Design 和 Fit Design，如果在 JEDEC File 前出现一对号，则表示 piano.jed 文件已生成。否则根据错误提示进行修改。

9) 下载 JED 文件到 ispLSI 1016 芯片

用 Lattice 编程电缆将 D2H＋数字电路实验箱与主机连接，将 ispLSI 1016 芯片插入对应 PLCC 插座，接通电源。选择 Source in Project 窗口中的 ispLSI 1016-60 PLCC44，双击 Process For Current Source 窗口中的 isp Download System，则打开 ISP Daisy Download 软件。执行 Configration 菜单中的 Scan Board，单击窗口中的 browse 按钮，选择 piano.jed。选择 Operation 中的 Program & Verify，执行 Command 菜单中的 Run Operation，有关信息在 Messages 窗口中显示。

10) 调试片内程序

根据 piano.ab1 文件中的管引脚定义连线，即：ispLSI 1016 芯片的引脚 11 接实验箱的 100kHz 时钟输出端；引脚 38 接实验箱喇叭“输入”端：引脚 9、40、36、3、29、10、16、43 分别接实验箱的逻辑开关输出 K1、K2、K3、K4、K5、K6、K7、K8。K1、K2、K3、K4、K5、K6、K7、K8 分别代表音符 1、2、3、4、5、6、7、i。当逻辑开关输出只有一个为高电平时，喇叭应发出相应的音乐。

6. 预习要求

(1) 做好实验预习，重点复习本实验原理和有关 ISP 书籍、资料。
(2) 熟练掌握 ISP 过程和编程方法。
(3) 进一步掌握用 ABEL 语言设计数字逻辑电路的方法。
(4) 熟悉 Synario 软件的使用方法。
(5) 了解音调的初步知识。
(6) 做好预习报告。

7. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程，包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果，列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.8.2 简易频率计

1. 实验目的

(1) 掌握较复杂逻辑的设计、调试。

(2) 掌握用 ABEL 语言设计数字逻辑电路。

(3) 熟悉 Synario 软件的使用方法。

(4) 熟悉 ISP 器件的使用。

(5) 了解频率计的初步知识。

2. 实验内容

设计一个简易频率计,用于测量 1MHz 以下数字脉冲信号的频率。闸门只有 1s 一档。测量结果在数码管上显示出来。不测信号脉宽。用一片 ISP 芯片实现此设计,并在实验箱上完成调试。建议设计用 ABEL 语言编写。

3. 实验器材

(1) 数字逻辑实验箱一台;

(2) 直流稳压电源一台;

(3) 示波器一台;

(4) 万用表一只;

(5) PLCC 封装的 ISP 1016 或者 M4-64/32 一片。

4. 实验提示

频率计的基本工作原理如下:首先产生一系列准确闸门信号,例如 1ms、0.1s 和 1s 等。然后用这些闸门信号控制一个计数器对被测脉冲信号进行计数,最后将结果显示出来。如果闸门信号是 1s,那么 1s 内计数的结果就是被测信号的频率。如果闸门信号是 1ms,那么计数结果是被测信号频率的千分之一,或者说结果是以 kHz 为单位的频率值。

频率计中,最原始的时基信号的准确度一定要高。建议用实验箱上的 100kHz 时钟信号作原始时基信号。

1s 的闸门信号,由 100kHz 时钟经 5 次 10 分频后,再经 2 分频产生。这样产生的闸门信号脉宽是 1s,占空比是 50%。在 2s 的时间内,1s 用于计数,1s 用于显示结果。

用于被测信号计数的计数器应采用十进制。测得的结果可直接送到实验箱上的 6 个数码管显示。每次对被测信号计数前,计数器应被清零。

5. 实验步骤

1) 设计思想

本项目由两个模块组成。

底层模块 gate 为顶层模块提供闸门信号(即 2Hz 的方波)和频率计数器的清零信号。当顶层频率计数器重新计数前,由该清零信号复位,以便从零计数。

顶层模块 frequenc 在使能状态下,当底层模块 gate 输出的闸门信号为 1 时,对被测信号进行十进制计数;当闸门信号为 0 时,显示计数结果。

2) 新建项目

打开 Synario 软件,选择 File 菜单中的 New Project 选项,弹出 Create New Project 对话框。在对话框的 Directories 一栏中,选择一个目录(最好是为该项目专门建立的目录),

存放该项目中的所有文件。然后在对话框的 Project File Name 一栏中输入项目文件名 frequenc. syn,并按“确定”按钮关闭对话框。

3) 选择器件

双击 Sources in Project 窗口中 Virtual Device,弹出 Choose Device 对话框,选择器件 ispLSI 1016-60 PLCC44。

4) 建立源文件

选择 Source 菜单中的 New,选择 ABEL-HDL Module,按 OK 按钮。在弹出对话框的 Module Name 栏中输入模块名,按 OK 按钮,进入 Synario Text Editor,开始输入源文件。本项目由顶层模块 frequenc 和底层模块 gate 组成。

5) 顶层模块 frequenc 的 ABEL 语言源文件

```
MODULE frequenc
DECLARATIONS
      //lower module declaration
gate  interface([begin,clock] ->[gate,clear]);
gare_1  functional_block  gate;
    //input
clock          pin 11;                          //时基信号 100kHz
insignal       pin 30;                          //被测信号 1～1MHz
begin          pin 10;                          //频率计使能高电平有效
//ouput
cnt00…cnt03    pin 40,22,27,20 istype 'reg';    //频率计个位
cnt10…cnt13    pin 19,17,18,16 istype 'reg';    //频率计十位
cnt20…cnt23    pin 15,28,31,26 istype 'reg';    //频率计百位
cnt30…cnt33    pin 25,43,42,44 istype 'reg';    //频率计千位
cnt40…cnt43    pin 41,6,39,38  istype 'reg';    //频率计万位
cnt50…cnt53    pin 37,5,8,3    istype 'reg';    //频率计十万位
cnt0 = [cnt03…cnt00];
cnt1 = [cnt13…cnt10];
cnt2 = [cnt23…cnt20];
cnt3 = [cnt33…cnt30];
cnt4 = [cnt43…cnt40];
cnt5 = [cnt53…cnt50];
      //node
  c0,c1,c2,c3,c4  node;                          //频率计各进位位
EQUATIONS
gate_1.begin = begin;
gate_1.clock = clock;
[cnt0,cnt1,cnt2,cnt3,cnt4,cnt5].clk = insignal & gate_1.gate;
      //闸门信号为 1 时,计数
[cnt0,cnt1,cnt2,cnt3,cnt4,cnt5].ar = begin # gate_1.clear;
      //个位十进制计数器
cnt0 := (cnt0 + 1) & !c0;
c0 = (cnt03 == 1)&(cnt00 == 1);
    //十位十进制计数器,只有 c0 为 1 时,才允许加 1,否则保持原值
cnt1 := (cnt1 + 1) & !((cnt13 == 1)&(cnt10 == 1))&c0 # cnt1&!c0;
c1 = (cnt13 == 1)&(cnt10 == 1)&c0;
    //百位十进制计数器,只有 c1 为 1 时,才允许加 1,否则保持原值
```

```
cnt2 := (cnt2 + 1)&!((cnt23 == 1)&(cnt20 == 1))&c1 # cnt2 & !c1;
c2 = (cnt23 == 1)&(cnt20 == 1)&c1;
    //千位十进制计数器,只有 c2 为 1 时,才允许加 1,否则保持原值
cnt3 := (cnt3 + 1)&!((cnt33 == 1)&(cnt30 == 1))& c2 # cnt3 & !c2;
c3 = (cnt33 == 1)&(cnt30 == 1)& c2;
    //万位十进制计数器,只有 c3 为 1 时,才允许加 1,否则保持原值
cnt4 := (cnt4 + 1)&!((cnt43 == 1)&(cnt40 == 1)) & c3 # cnt4 & !c3;
c4 = (cnt43 == 1)&(cnt40 == 1)& c3;
    //十万位十进制计数器,只有 c4 为 1 时,才允许加 1,否则保持原值
cnt5 := (cnt5 + 1)&!((cnt53 == I)&(cnt50 == 1))& c4 # cnt5 & !c4;
END
```

6) 底层模块 gate 的 ABEL 语言源文件

```
MODULE gate
DECLARATIONS
    //input
begin          pin;                          //频率计使能
clock          pin;                          //时基信号 100kHz
//output
gate           pin istype 'reg';             //闸门信号脉宽为 1s 的方波
clear          pin;                          //顶层频率计数器清零信号
//nnde
q0…q17      node istype 'reg';               //100K 计数器
q = [q17…q0];
c    node;
EQUATIONS
    //100K 计数器
q.clk = clock;
q.ar = begin;
c = (q == 100000 - 1);
q := (q + 1)& !c;
gate.clk = c:                                //闸门信号 脉宽为 1s 的方波
gate := !gate;
gate.ar = begin;
clear = (q == 100000 - 2) & !gate;           //为顶层频率计数器重新计数前提供清零信号
```

7) 编译源文件

选择 Source in Project 窗口中的 ABEL 语言源文件,双击 Process For Current Source 窗口中的 Compile Logic 和 Reduce Logic,如果在两命令前出现对号,则表示编译通过。否则根据错误提示修改源文件。

8) 生成 JED 文件

选择 Source in Project 窗口中的 ispLSI 1016-60 PLCC44,依次执行 Process For Current Source 窗口中的 Link Design 和 Fit Design 选项,如果在 JEDEC File 前出现一对号,则表示 frequenc.jed 已生成。否则根据错误提示进行修改。

9) 下载 JED 文件到 ispLSI 1016 芯片

用 Lattice 编程电缆将 D2H+数字电路实验箱与主机连接,将 ispLSI 1016 芯片插入对应 PLCC 插座,接通电源。选择 Source in Project 窗口中的 ispLSI 1016-60 PLCC44 选项,

双击 Process For Current Source 窗口中的 isp Download System 选项，则打开 ISP Daisy Download 软件。执行 Configration 菜单中的 scan Board，单击窗口中的 browse 按钮，选择 frequenc.jed。选择 Operation 选项中的 Program & Verify，执行 Command 菜单中的 Run Operation 选项，有关信息在 Messages 窗口中显示。

根据 frequenc.ab1 文件中的管引脚定义连线，即：ispLSI 1016 芯片的引脚 11 接实验箱 100kHz 时钟输出端；引脚 30 接被测信号(可接实验箱的可调时钟输出端 1～100kHz)；引脚 10 接一逻辑开关输出，当为低电平时，测量被测信号频率并在数码管上显示；引脚 20、27、22、40 分别接数码管 LD1 的 D、C、B、A 端，引脚 16、18、17、19，引脚 26、31、28、15，引脚 44、42、43、25，引脚 38、39、6、41 和引脚 3、8、5、37 分别接数码管 LD2、LD3、LD4、LD5 和 LD6 的 D、C、B、A 端。

6. 预习要求

(1) 做好实验预习，重点复习本实验原理和有关 ISP 书籍、资料。
(2) 熟练掌握 ISP 过程和编程方法。
(3) 进一步掌握用 ABEL 语言设计数字逻辑电路的方法。
(4) 熟悉 Synario 软件的使用方法。
(5) 了解频率计的初步知识。
(6) 做好预习报告。

7. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程，包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果，列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.8.3 交通灯实验

1. 实验目的

(1) 掌握状态机的设计、调试。
(2) 掌握用 ABEL 语言设计状态机的方法。
(3) 熟悉 Synario 软件的使用方法。
(4) 熟悉 ISP 器件的使用。

2. 实验内容

以实验箱上的 4 个红色电平指示灯、4 个绿色电平指示灯和 4 个黄色电平指示灯模仿

路口的东、西、南、北 4 个方向的红、绿、黄交通灯。控制这些指示灯，使它们按下列规律亮、灭：

(1) 初始状态为 4 个方向的红灯全亮。时间 1s。

(2) 东、西方向绿灯亮，南、北方向红灯亮。东、西方向通车。时间 5s。

(3) 东、西方向黄灯闪烁，南、北方向红灯亮。时间 2s。

(4) 东、西方向红灯亮，南、北方向绿灯亮。南、北方向通车。时间 5s。

(5) 东、西方向红灯亮，南、北方向黄灯闪烁。时间 2s。

(6) 返回第(2)步，继续运行。

(7) 如果发生紧急事件，例如救护车、警车通过，则按下单脉冲按钮，使得东、西、南、北 4 个方向的红灯亮。紧急事件结束后，松开单脉冲按钮，恢复到被打断的状态继续运行。

3. 实验器材

(1) 数字逻辑实验箱一台；

(2) 直流稳压电源一台；

(3) 示波器一台；

(4) 万用表一只；

(5) PLCC 封装的 ISP 1016 或者 M4-64/32 一片。

4. 实验提示

(1) 这是一个典型的时序状态机，一共有 6 个大的状态。

(2) 黄灯闪烁可通过连续亮 0.2s、灭 0.2s 实现。

(3) 将实验箱上的可调频率时钟为 1kHz，作为设计中的初始时钟，以减少需要的分频数。

(4) 紧急事件发生时，要注意保存必要的信息，以备紧急事件结束后，恢复到原状态继续运行使用。

5. 实验步骤

(以 Lattice 公司 ispLSI 1016-60 PLCC44 芯片为例)

1) 设计思想

本项目由一个模块 tralight 组成，由时序状态机实现，共 6 个状态，用 s0，s1，…，s5 表示。

本项目有 4 个计数器：0.2s、1s、2s 和 5s 计数器。0.2s 计数器除用作黄灯闪烁外，还作为 1s、2s 和 5s 计数器的计数时钟使用。

刚加电或复位后，状态机为 s0 状态，1s 计数器开始计数；1s 后，状态机要为 s1，1s 计数器复位停止计数，5s 计数器开始计数；5s 后，状态机变为 s2，5s 计数器停止计数，2s 计数器开始计数，同时为黄灯提供闪烁信号；2s 后，状态机变为 s3，5s 计数器开始计数，2s 计数器停止计数；5s 后，状态机变为 s4，5s 计数器停止计数，2s 计数器计数，同时为黄灯提供闪烁信号；2s 后，状态机回到 s0。在 s0～s4 任何状态时，当按下“紧急情况控制”键时，状态机变为 s5，所有计数器停止计数，且上一状态被保存在一组寄存器中；再按一次“紧急情况控制”

键后，状态机恢复上一状态，计数器也恢复正常工作。

2）新建项目

打开 Synario 软件，选择 File 菜单中的 New Project，弹出 Create New Project 对话框。在对话框的 Directories 一栏中，选择一个目录（最好是为该项目专门建立的目录），存放该项目中的所有文件。然后在对话框的 Project File Name 一栏中输入项目文件名 tralight.syn，并按“确定”按钮关闭对话框。

3）选择器件

双击 Sources in Project 窗口中的 Virtual Device，选择器件 ispLSI 1016-60 PLCC44。

4）建立源文件

选择 Source 菜单中的 New，选择 ABEL-HDL Module，按 OK 按钮。在弹出对话框的 Module Name 栏中输入模块名，按 OK 按钮，进入 Synario Text Editor，开始输入源文件。本项目由一个模块 tralight 组成。

5）模块 tralight 的 ABEL 语言源文件

```
MODULE tralight
declarations
    //Input
clk       pin 29;                              //时钟 1000Hz
rt        pin 6;                               //复位输入，低电平有效
control   pin 11;                              //紧急情况控制
    //Output
q0r,q0y,q0g,d0r,d0y,d0g pin 15,40,18,19,44,22;
        //东西方向红、黄、绿灯
q1r,q1y,q1g,d1r,d1y,d1g pin 38,16,17,42,20,21;
        //南北方向红、黄、绿灯
q = [q0r,q0y,q0g,q1r,q1y,q1g];
d = [d0r,d0y,d0g,d1r,d1y,d1g];
    //Node
ct1     node istype 'reg';                     //紧急情况控制标志
cnt0…cnt2     node istype 'reg';               //状态机寄存器
cnt = [cnt2…cnt0];
ecnt0…ecnt2     node istype 'reg';             //记忆状态机寄存器
ecnt = [ecnt2…ecnt0];
t10…t12     node istype 'reg';                 //1s 计数器
t20…t23     node istype 'reg';                 //2s 计数器
t50…t54     node istype 'reg';                 //5s 计数器
flash0…flash7     node istype 'reg';           //0.2s 计数器
flashc     node istype'reg';                   //黄灯闪烁控制
t1 = [t12…t10];
t2 = [t23…t20];
t5 = [t54…t50];
flash = f[flash7…flash0];
s0 = 0; s1 = 1; s2 = 2; s3 = 3; s4 = 4; s5 = 5: //状态机状态定义
equations
cnt.clk = clk;                                 //状态寄存器
ecnt.clk = clk;                                //状态保存寄存器
flash.clk = clk;                               //0.2s 计数器
```

```
flash.ar = (flash == 200 - 1) # !rt;
flashc.clk = (flash == 200 - 1);
flashc := !flashc;                          //黄灯闪烁控制寄存器
flashc.ar = !rt;
t1.clk = (flash == 200 - 1);                //1s 计数器
t2.clk = (flash == 200 - 1);                //2s 计数器
t5.clk = (flash == 200 - 1);                //5s 计数器
d = q;
ct1.clk = control;
ct1 := !ct1;
ct1.ar = !rt;
state_diagram  [cnt2…cnt0];
state s0:                                   //状态 0: 东西南北红灯亮
q0r = 1; q0y = 0; q0g = 0;
q1r = 1; q1y = 0; q1g = 0;
flash := flash + 1:                         //1s 计数器时钟
t1.ar = !rt;
t2.ar = 1;                                  //2s 计数器复位
t5.ar = 1;                                  //5s 计数器复位
t1 := t1 + 1;                               //1s 计数器计数
if(rt == 0)
then    s0
else{if(ct1 == 1)
    then s5 with{ecnt := 0; }
    else{if(t1 == 5)
    then s1
    else s0; }
    }
state s1:                                   //状态 1: 东西绿灯亮,南北红灯亮
  q0r = 0; q0y = 0; q0g = 1;
  q1r = 1; q1y = 0; q1g = 0;
  flash := flash + 1:                       //5s 计数器时钟
  t1.ar = 1;                                //1s 计数器复位
  t2.ar = 1;                                //2s 计数器复位
  t5.ar = 0:                                //5s 计数器计数
  t5 := t5 + 1;
  if(rt == 0)
  then s0
  else{if(ct1 == 1)
    then s5 with{ecnt := 1; }
    else{
    if(t5 == 25)
    then s2
    else s1; }
}
state s2:                                   //状态 2: 东西黄灯亮,南北红灯亮
  q0r = 0; q0y = flashc; q0g = 0;
  q1r = 1; q1y = 0; q0g = 0;
  flash := flash + 1;                       //2s 计数器时钟
  t1.ar = 1;                                //1s 计数器复位
  t2.ar = 0;                                //2s 计数器计数
```

```
  t5.ar = 1;                                  //5s 计数器复位
  t2 := t2 + 1;
  if(rt == 0)
  then    s0
  else{ if(ct1 == 1)
    then s5 with{ecnt := 2; }
    else{
    if(t2 == 10)
    then s3
    else s2; }
}
state s3:                                     //状态 3: 东西红灯亮,南北绿灯亮
    q0r = 1; q0y = 0; q0g = 0;
    q1r = 0; q1y = 0; q1g = 1;
    flash := flash + 1;                       //5s 计数器时钟
    t1.ar = 1;                                //1s 计数器复位
    t2.ar = 1;                                //2s 计数器复位
    t5.ar = 0;                                //5s 计数器计数
    t5 := t5 + 1;
    if(rt == 0)
    then    s0
    else{ if(ct1 == 1)
     then s5 with{ecnt := 3; }
     else{
          if(t5 == 25)
          then s4
          else s3; }
    }
state s4:                                     //状态 4: 东西红灯亮,南北黄灯亮
    q0r = 1; q0y = 0; q0g = 0;
    q1r = 0; q1y = flashc; q1g = 0;
    flash: = flash + 1;                       //2s 计数器时钟
    t1.ar = 1;                                //1s 计数器复位
    t2.ar = 0;                                //2s 计数器计数
    t5.ar = 1;                                //5s 计数器复位
    t2 := t2 + 1;
    if(rt == 0)
    then s0
    else{ if(ct1 == 1)
     then s5 with{ecnt := 4; }
     else{
          if(t2 == 10)
          then s1
          else s4; }
    }
state s5:                                     //状态 5: 紧急情况
    q0r = 1; q0y = 0; q0g = 0;
    q1r = 1; q1y = 0; q1g = 0;
    ecnt := ecnt;
    flash := flash;
    if(rt == 0)
```

```
    then s0
    else{if(ct1 == 1)
     then    s5
     else{case  (ecnt == 0): s0;
                (ecnt == 1): s1;
                (ecnt == 2): s2;
                (ecnt == 3): s3;
                (ecnt == 4): s4;
          endcase}
     }
END
```

6) 编译源文件

选择 Source in Project 窗口中的 ABEL 语言源文件，双击 Process For Current Source 窗口中的 Compile Logic 和 Reduce Logic，如果在两命令前出现对号，则表示编译通过。否则根据错误提示修改源文件。

7) 生成 JED 文件

选择 Source in Project 窗口中的 ispLSI 1016-60 PLCC44，依次执行 Process For Current Source 窗口中的 Link Design 和 Fit Design，如果在 JEDEC File 前出现一对号，则表示 tralight.jed 文件已生成。否则根据错误提示进行修改。

8) 下载 JED 文件到 ispLSI 1016 芯片

用 Lattice 编程电缆将 D2H＋数字电路实验箱与主机连接。将 ispLSI 1016 芯片插入对应 PLCC 插座，接通电源。选择 Source in Proiect 窗口中的 ispLSI 1016-60 PLCC44，双击 Process For Current Source 窗口中的 isp Download System，则打开 ISP Daisy Download 软件。执行 Configration 菜单中的 Scan Board，单击窗口中的 browse 按钮，选择 tralight.jed。选择 Operation 中的 Program & Verify，执行 Command 菜单中的 Run Operation，有关信息在 Messages 窗口中显示。

9) 调试片内程序

根据 tralight.ab1 文件中的管引脚定义连线。即：ispLSI 1016 芯片的引脚 29 接实验箱 1kHz 可调时钟输出端；引脚 6 接一逻辑开关，当为低电平时复位，4 个红灯亮；引脚 11 接一宽单脉冲，用于紧急事件控制；引脚 15、19，引脚 40、44 和引脚 18、22 分别接左边的两个红灯、黄灯和绿灯，表示东西方向；引脚 38、42，引脚 16、20 和引脚 17、21 分别接右边的两个红灯、黄灯和绿灯，表示南北方向。

6. 预习要求

(1) 做好实验预习，重点复习本实验原理和有关 ISP 书籍、资料。

(2) 熟练掌握 ISP 过程和编程方法。

(3) 进一步掌握用 ABEL 语言设计数字逻辑电路的方法。

(4) 熟悉 Synario 软件的使用方法。

(5) 了解交通灯的初步知识。

(6) 做好预习报告。

7. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程,包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。

5.8.4 电子钟

1. 实验目的

(1) 掌握较复杂逻辑的设计、调试。
(2) 学习用原理图设计数字逻辑电路的方法。
(3) 学习数字电路模块层次设计。
(4) 学习 Synario 软件的使用方法。
(5) 熟悉 ISP 器件的使用。

2. 实验内容

(1) 设计并用 ISP 1016 或者 M4-64/32 实现一个电子钟。电子钟具有下述功能:
① 实验箱上的 6 个数码管显示时、分、秒。
② 能使电子钟复位(清零)。
③ 能启动或者停止电子钟运行。
④ 在电子钟停止运行状态下,能够修改时、分、秒的值。
⑤ 具有报时功能,整点时喇叭鸣叫。
(2) 要求整个设计分为若干模块。顶层模块用原理图设计,底层模块用 ABEL 语言设计。
(3) 在 D2H+实验箱上调试设计。

3. 实验器材

(1) 数字逻辑实验箱一台;
(2) 直流稳压电源一台;
(3) 示波器一台;
(4) 万用表一只;
(5) PLCC 封装的 ISP 1016 或者 M4-64/32 一片。

4. 实验步骤

(以 Lattice 公司 ispLSI 1016-60 PLCC44 芯片为例)

(1) 设计思想

本项目由一个顶层模块和多个底层模块组成。

顶层模块 clock 由原理图实现,包含时、分、秒计数器和时钟发生器、整点响铃发声模块、译码器等 6 个底层模块(其中 cnt60 用了两次)及若干门电路,并定义了信号与引脚的对应关系。

底层模块 clk-ring,对输入时钟 1000 分频,产生 1Hz 的脉冲提供给模块 cnt60,并为模块 ring 提供整点响铃脉冲。

底层模块 cnt60(两个)、cnt24 分别用作秒、分、时计数器,为了实现暂停与预置数的功能,其时钟输入加入了若干控制信号。为了使进位信号的高电平脉宽尽量窄,则进位信号的复位端取其自身。

底层模块 Encode24 是一个 2-4 译码器,用于选择需预置数的计数器,输入 00 表示选择秒计数器,01 表示选择分计数器,10 表示选择小时计数器。

底层模块 ring,内有一个计数器,当小时计数器(cnt24)的时钟信号到来时,该计数器开始计数,并有整点响铃脉冲输出; 3s 后,计数器复位,响铃脉冲不再输出。

(2) 新建项目

打开 Synario 软件,选择 File 菜单中的 New Project 选项,弹出 Create New Project 对话框。在对话框的 Directories 一栏中,选择一个目录(最好是为该项目专门建立的目录),存放该项目中的所有文件。然后在对话框的 Project File Name 一栏中输入项目文件名 clock. syn,并按“确定”按钮关闭对话框。

(3) 选择器件

双击 Sources in Project 窗口中 Virtual Device,选择器件 ispLSI 1016-60 PLCC44。

(4) 建立源文件

本项目由顶层模块 clock(由原理图实现)和底层模块 clk-ring,cnt24,cnt60,encode24,ring 组成。选择 Source 菜单中的 New 选项,选择 Schematic,按 OK 按钮。在弹出的对话框中输入文件名 clock. sch,按“确定”按钮,进入 Schematic Editor 界面,开始编辑原理图,之后存盘退出。选择 Source 菜单中的 New 选项,选择 ABEL-HDL Module,按 OK 按钮。在弹出对话框的 Module Name 栏中输入模块名,按 OK 按钮,进入 Synario Text Editor 界面,开始编辑底层 ABEL 语言源文件。

(5) 顶层模块 clock 的原理图如图 5.8.1 所示。

(6) 底层模块 clk-ring 的 ABEL 语言源文件

```
MODULE clk_ring
    //提供频率为 1Hz 的脉冲及整点响铃脉冲
DECLARATIONS
    //input
clk             pin;                    //时钟输入 1kHz
    //output
clock           pin;                    //输出频率为 1Hz 的脉冲
ring            pin;                    //整点响铃脉冲
    //node
q0…q9     node istype 'leg';            //1000 分频器
q = [q9…q0];
```

```
EQUATIONS
ring = q0;
q.clk = clk;
q := (q + 1)&!(q == 1000 - 1);
clock = (q == 1000 - 1);
END
```

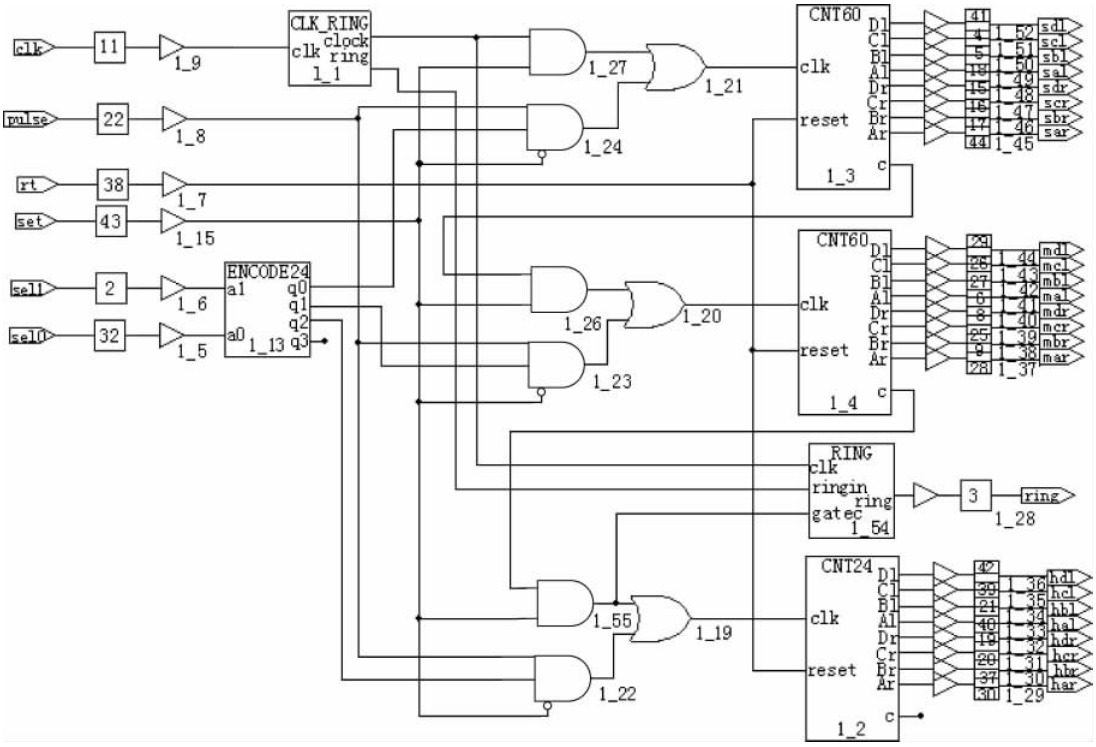

图 5.8.1　电子钟原理图

（7）底层模块 cnt24 的 ABEL 语言源文件

```
MODULE cnt24
    //二十四进制计数器
DECLARATlONS
    //input
clk                pin;                    //时钟输入
reset              pin;                    //复位信号低电平有效
    //output
DI,CI,BI,AI        pin istype 'reg';       //计数器高位输出
Dr,Cr,Br,Ar        pin istype 'reg';       //计数器低位输出
q0 = [Dr,Cr,Br,Ar];
qI = [DI,CI,BI,AI];
c     pin istype'reg';                     //进位信号
EQUATIONS
q0.clk = cIk;
q1.clk = clk;
q0.ar = !reset;
```

```
q1.ar = !reset;
q0 := (q0 + I)&!(q0 == 9)&!((q1 == 2)&(q0 == 3));
q1 := ((q1 + 1)&(q0 == 9)#q1&!(q0 == 9))&!((q1 == 2)&(q0 >= ));
c.clk = clk;
c := (q0 == 3)&(q1 == 2);
c.ar = c
END
```

(8) 底层模块 cnt60 的 ABEL 语言源文件

```
MODULE ent60
    //六十进制计数器
DECLARATIONS
    //input
clk              pin;                    //时钟输入
reset            pin;                    //复位信号低电平有效
//output
DI,CI,BI,AI      pin istype 'reg';       //计数器高位输出
Dr,Cr,Br,Ar      pin istype'reg';        //计数器低位输出
q0 = [Dr,Cr,Br,Ar];
qI = [DI,CI,BI,AI];
c                pin istype 'reg';       //进位信号
EQUATIONS
Q0.clk = clk;
q1.clk = clk;
q0.ar = !reset;
q1.ar = !reset;
q0 := (q0 + 1)&!(q0 == 9);
q1 := ((q1 + 1)&(q0 == 9)#q1&!(q0 == 9))&!((q1 == 5)&(q0 == 9));
c.clk = clk;
c := (q0 == 9)&(q1 == 5);
c.ar = c;
END
```

(9) 底层模块 encode24 的 ABEL 语言源文件

```
MODULE encode2 - 4
    //2 - 4 译码器
DECLARATIONS
    //input
a0,a1    pin;
    //output
q0…q3    pin;
EQUATIONS
q0 = !a1&!a0;
q1 = !a1&a0;
q2 = a1&!a0;
q3 = a1&a0;
END
```

(10) 底层模块 ring 的 ABEL 语言源文件

```
MODULE ring
    //提供整点响铃信号
DECLARATIONS
    //input
clk     pin;                                    //时钟输入 1Hz
gatec,ringin     pin;                           //响铃控制信号,响铃脉冲输入
    //output
ring     pin;                                   //响铃脉冲输出
    //node
q0…q1     node istype 'reg';                    //响铃时间计数器
q = [q1…q0];
EQUATIONS
q.clk = clk;
q := (q + 1)&!(q == 3) # q&(q == 3);
q.ar = gatec;                                   //每次到整点时,计数器先清零,然后计数
      //要求信号 gatec 很窄
ring = ringin &(q < 3);                         //响铃时间小于 3s 时,有脉冲输出
END
```

(11) 编译源文件

选择 Source in Project 窗口中的源文件,双击 Process For Current Source 窗口中的 Compile Logic 和 Reduce Logic 选项,如果在两命令前出现对号,则表示编译通过。否则根据错误提示修改源文件。

(12) 生成 JED 文件

选择 Source in Project 窗口中的 ispLSI 1016-60 PLCC44,依次执行 Process For Current Source 窗口中的 Link Design 和 Fit Design 选项,如果在 JEDEC File 前出现一对号,则表示 clock.jed 文件已生成。否则根据错误提示进行修改。

(13) 下载 JED 文件到 ispLSI 1016 芯片

用 Lattice 编程电缆将 D2H+数字电路实验箱与主机连接,将 ispLSI 1016 芯片插入对应 PLCC 插座,接通电源。选择 Source in Project 窗口中的 ispLSI 1016-60 PLCC44,双击 Process For Current Source 窗口中的 isp Download System,则打开 ISP Daisy Download 软件。执行 Configration 菜单中的 Scan Board,单击窗口中的 browse 按钮,选择 clock.ied。选择 Operation 中的 Program & Verify,执行 Command 菜单中的 Run Operation,有关信息在 Messages 窗口中显示。

(14) 调试片内程序

根据 clock.sch 中的管引脚定义连线,即:ispLSI 1016 芯片的引脚 11 接实验箱 1kHz 可调时钟输出端;引脚 38 接一逻辑开关,为复位信号,低电平有效:引脚 43、2、32 分别接三个逻辑开关,用于暂停时钟及选择需修改的时、分、秒计数器;引脚 22 接一宽单脉冲,为修改时、分、秒计数器提供计数脉冲;引脚 15、16、17、44 分别接数码管 LD1 的 D、C、B、A 端;引脚 41、4、5、18,引脚 8、25、9、28,引脚 29、26、27、6,引脚 19、20、37、30 和引脚 42、39、21、40 分别接数码管 LD2、LD3、LD4、LD5 和 LD6 的 D、C、B、A 端;引脚 3 接喇叭的"输入"端,整点时响 3s。

5. 预习要求

(1) 做好实验预习,重点复习本实验原理和有关 ISP 书籍、资料。
(2) 熟练掌握 ISP 过程和编程方法。
(3) 进一步掌握用 ABEL 语言设计数字逻辑电路的方法。
(4) 熟悉 Synario 软件的使用方法。
(5) 了解电子钟的初步知识。
(6) 做好预习报告。

6. 实验报告内容及要求

(1) 实验目的。
(2) 实验任务及要求。
(3) 逻辑设计过程,包括化简的步骤。
(4) 画好实验验证用逻辑布线图或物理布线图。
(5) 按要求填写各实验表格。
(6) 整理、分析实验数据和结果,列表比较实验任务的理论分析值和实验结果值。
(7) 写出心得体会。