“计算机接口技术”实验指导书

（试用版）

南开大学计算机与控制工程学院

2015 年 2 月

目　　录

第一章　基础实验设计与案例

1.1　实验 1　无条件输出端口的构成与地址译码……3
1.2　实验 2　手动数字量输入与无条件输入端口的构成……5
1.3　实验 3　可编程接口芯片 8255A 的使用……7
1.4　实验 4　4×4 小键盘的使用……9
1.5　实验 5　七段数码管的使用……11
1.6　实验 6　D/A 转换器的使用……13
1.7　实验 7　A/D 转换器的使用……16
1.8　实验 8　可编程计数器/定时器 8254 的使用……18

第二章　开放性实验设计与案例

2.1　案例 1 USB 设备的 I/O 扩展……20
2.2　案例 2 USB 数据采集系统……20
2.3　案例 3 USB 无线控制系统……20

第一章 基础实验设计与案例

实验 1——无条件输出端口的构成与地址译码

1.1 实验目的

① 掌握无条件输出端口的构成以及如何对其进行写操作；

② 学会利用 QuartusII 设计地址译码电路的构建方法；

1.2 相关背景及说明

无条件输出端口和无条件输入端口是构建接口电路的基础。从原理上看，无条件输出端口是由寄存器、地址译码器和负与逻辑构成的。利用 FPGA 芯片可构成一个典型的 8 位无条件输出端口。无条件输出端口模块处实际是一个进行了相关连接的 8 位寄存器(74LS273)，如图 1.1 所示。

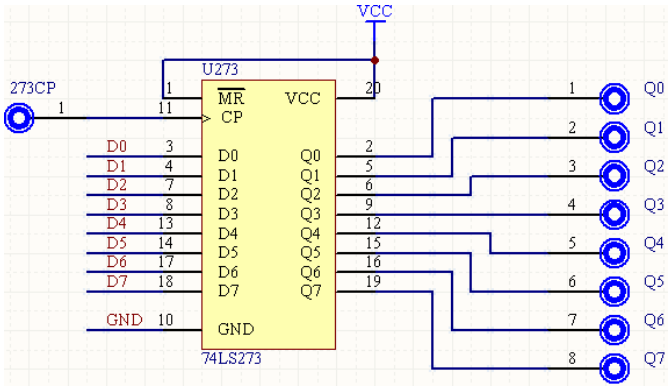

图 1.1　74LS273 的连接

图中，74LS273 的数据输入端需要和实验箱提供的系统数据总线的低 8 位（XD7～XD0）相连(组成无条件输出端口时通常要求这样连)，输出端分别接到 8 个插孔(标注为 Q7～Q0)，打入脉冲输入端也安排了一个插孔（标注为 273CP)。为了方便观看写入到该输出端口的值，可将 74LS273 的输出接到 LED 显示模块的输入。注意，QuartusII 中的 74273 是由数字逻辑生成的电路，其引脚与此图有细微差别。

实验板 LED 显示模块的电路如图 1.2 所示。

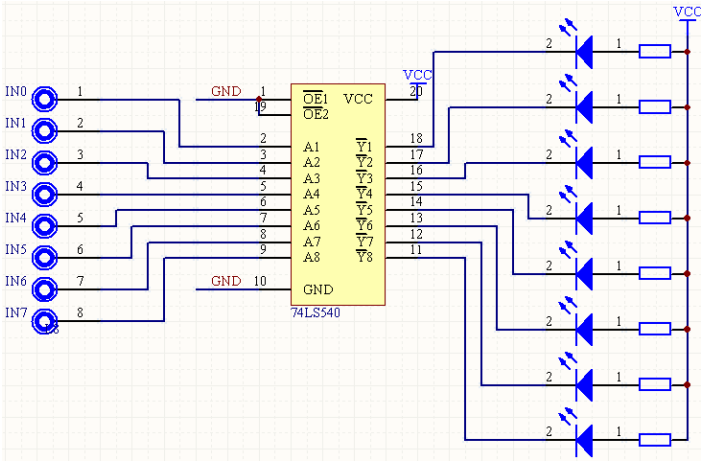

图 1.2　LED 显示模块的电路连接

图中，74LS540 为八反相缓冲器/驱动器，参数之一 I_{OL}（低电平输出电流，即输出为低时的灌入电流）为 24mA，足可以点亮发光二极管 LED。在其 8 个输入端均安排了插孔，从低位到高位依次标注为 IN0～IN7。注意，由于一般实验板中的 LED 灯均已配置 74LS540 或其他类似功能的驱动电路芯片，故而不需要再在 FPGA 板中设计实现 74LS540 的相关电路了。

在 74LS273 的输出与 74LS540 的输入对应相连的情况下，当 74LS273 的某位为 1，则对应的 LED 亮，为 1，则对应的 LED 灭（因为 74LS540 为反相逻辑）。

构建无条件输出端口除寄存器外，还需要地址译码器。本次实验的地址译码电路可以自行设定。构成无条件输出端口的关键是产生寄存器的打入脉冲，实际上它是 IO 写信号（IOW#）和地址译码信号的负与。写入时刻是负与逻辑输出负脉冲的后沿。

建议利用 74LS138，74LS273 以及 IOW 信号构成译码电路。**注意由于实验开发环境将 windows 的资源配置进行了重新分配，实验中我们使用的 I/O 地址范围为 0x3000H-0x30FFH。**因此其中地址译码的实际连线部分可以用 A1－A7（A0 不要用），但程序指令必须符合该范围。

1.3 实验内容

① 构建无条件输出端口，自主设计译码电路，由 FPGA 芯片实现。编一程序，给该无条件输出端口输出不同的数，将 74LS273 的输出接 LED 显示模块,用循环方法让 8 个 LED 呈现规律性变化，变化规律自行设计。

② 改变地址译码电路，体会改变后的地址与改变前的地址在程序中的不同之处。

1.4 实验报告

① 画出实验①和②中的地址译码电路。

② 说明实验板的线路接法。

③ 写出实现输出数据至端口的程序段。

④ 写出让 LED 呈规律性变化的程序段。

实验 2——手动数字量输入与无条件输入端口的构成

2.1 实验目的

① 了解什么是数字量输入,手动数字量输入有什么用途；

② 学会无条件输入端口的构成以及对其进行读操作。

2.2 相关背景及说明

数字量输入是指输入信号是数字量即数字信号。为使实验者对数字量输入有个直观了解，实验板提供了 8 位手动数字量输入模块。该模块由 8 个开关和一个排电阻（此为 A 型排电阻，所有电阻被封装在一起，而它们的一端被连在一起，作为公共端）组成，具体电路如图 2.1 所示。

开关拨到上方表示 1，否则表示 0（在实验板上，每个开关的接点 3 位于上方，而接点 1 位于下方）。该模块的输出 DD7～DD0（安排有插孔）可作为无条件输入端口的输入；也可直接接 LED 显示模块，即由手拨开关控制 LED 的亮灭；还可以接七段数码显示模块，以控制数码显示。在实际应用中手动数字量输入通常用来控制一些设备或装置的动作，或进行分步操作。

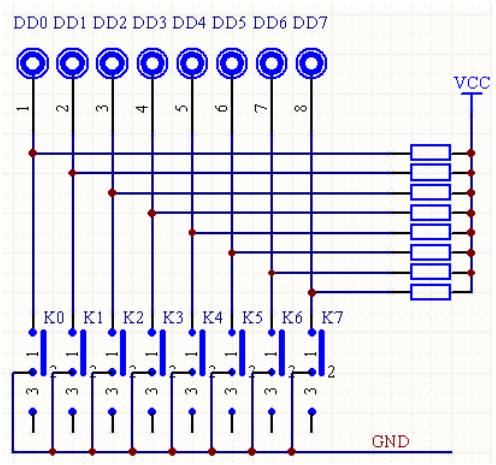

图 2.1　手动数字量输入模块的电路图

无条件输入端口通常也是接口的组成部分。从原理上看，无条件输入端口是由三态门、地址译码器和负与逻辑构成的。利用 FPGA 板可构成一个典型的 8 位无条件输入端口。无条件输入端口电路实际是一个进行了相关连接的八三态门（74LS244），如图 2.2 所示。

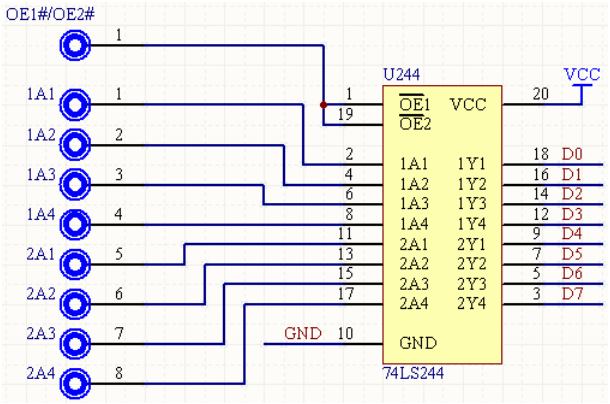

图 2.2　无条件输入模块的电路图

74LS244 内部的 8 个三态门被分成两组。1A1～1A4 分别是第一组四个三态门的输入，1Y1～1Y4 分别是第一组四个三态门的输出，2A1～2A4 是第二组的输入，2Y1～2Y4 是第二组的输出。两组三态门分别设置了控制信号（OE1#和 OE2#），低电平有效，即控制信号为低电平时，对应的一组三态门为开放状态，否则为关闭状态。

构成无条件输入端口的关键是产生三态门的控制信号，实际上它是 IO 读信号（IOR#）

和地址译码信号的负与。

2.3 实验内容

① 将手动数字量输入模块的输出接 LED 显示模块，拨动开关，观察 LED 的亮灭；
② 构建无条件输入端口，输入数据由手动数字量输入模块提供，编一程序，读该端口并显示在屏幕上；
③ 设计电路并编写循环程序，在循环体中，先读入无条件输入端口的数值，经过处理后再向无条件输出端口输出，并接 LED 显示模块，要求输入数据与输出数据呈现不同规律。

2.4 实验报告

① 画出实验③中的地址译码电路。
② 说明实验板的线路接法。
③ 写出实验③的循环程序。

实验 3——可编程接口芯片 8255A 的使用

3.1 实验目的

① 加深对 8255A 可编程特性的理解。

② 学会 8255A 的使用以及与其他模块的连接。

3.2 相关背景及说明

可编程输入输出接口芯片 8255A 几乎在所有微机接口技术的书中都会有介绍，这是因为它最能体现在 CPU 与外设之间的接口作用，并且仍然有着广泛应用。本实验系统也将其列入其中，图 3.1 是 8255A 连接示意图。三个端口共 24 个引脚都安排了插孔（分别标注为 PA7～PA0、PB7～PB0 和 PC7～PC0）。

由于方式 1 和方式 2 涉及中断，而在 Windows 环境下，中断处理属于设备驱动程序，因此本实验仅仅围绕方式 0 展开。

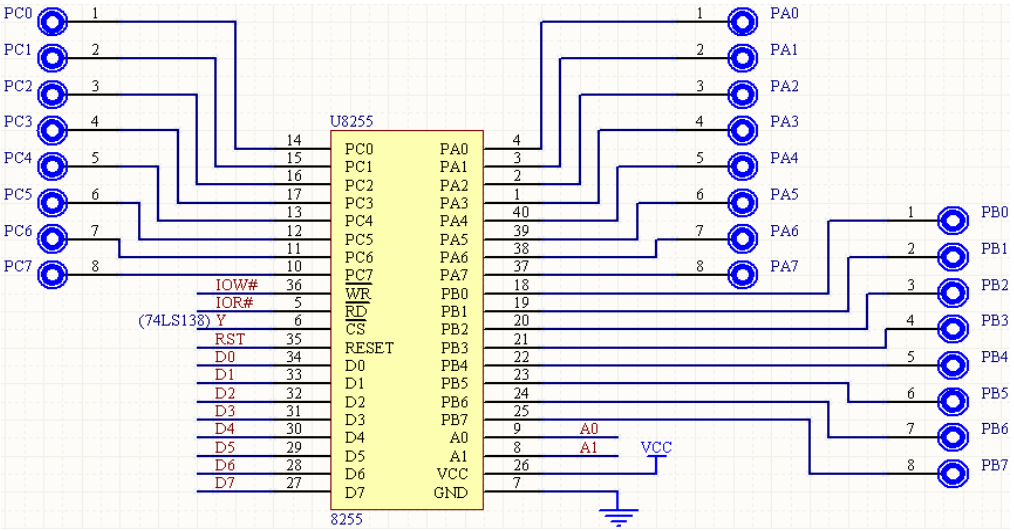

图 3.1　8255A 连接示意图

3.3 实验内容

① 将手动数字量输入模块的输出（K7～K0）接 8255A 的 A 口（PA7～PA0）。编程时，先将 A 口设置成方式 0 输入，B 口设置成方式 0 输出（C 口未用，方式自定，以下实验未用的端口均自行设定），然后安排一个循环结构，读 A 口，并用二进制形式显示读入值。在程序运行期间，拨动开关，观察显示是否与开关状态一致。

说明：printf 函数不支持二进制形式显示，要二进制显示，可利用_itoa 函数。以下是该函数用法举例：

```
int temp;
char buff[9];
temp=15;
itoa(temp, buff, 2 );    // 将 temp 中的值转换成二进制串，放到 buff 中
printf( "%08s\n", buff); // 以 8 位字符串形式显示 buff 内容，不够 8 位时前面补 0
```

② 在实验①基础上，将 B 口的 8 个引脚（插孔）和 LED 显示模块相连。读 A 口，将读入值输出至 B 口。在程序运行期间，拨动开关，观察 LED 的变化。

③ 去掉实验①和②的连线，将 A 口（PA7～PA0）和 B 口（PB7～PB0）连接起来，并同时和 LED 显示模块相连。编程时，将 A 口设置成方式 0 输出、B 口设置成方式 0 输入，通过循环给 A 口输出不同的数，读 B 口，并显示（进位制自定）。此期间，观察 LED

的变化。

④ 不改变实验③的连线，将 A 口和 B 口的设置反过来，即将 A 口设置成方式 0 输入、B 口设置成方式 0 输出，通过循环给 B 口输出不同的数，读 A 口，并显示（进位制自定）。

⑤ 去掉 8255A 的所有连线，将 C 口（PC7～PC0）和 LED 显示模块连接起来。编程时，将 C 口设置成方式 0 输出，通过循环，向 C 口输出，使 LED 呈现规律性变化（变化规律自行设计）。

⑥ 保留实验⑤的连线，编程时，仍将 C 口设置成方式 0 输出，但要求通过置位/复位操作，使 LED 呈现规律性变化（变化规律同实验⑤或重新设计）。

说明：8255A 还可以和无条件输入模块以及无条件输出模块进行组合实验。将 8255A 准备置成输出的端口和无条件输入模块相连，先对该端口进行写操作，再对无条件输入端口进行读操作。该实验可进一步理解 8255A 的输出是锁存的。将 8255A 准备置成输入的端口和无条件输出模块相连，先对无条件输出进行写操作，再对 8255A 的输入端口进行读操作。该实验可进一步理解利用 8255A 可输入寄存器中的内容，起三态门的作用，确切说，起无条件输入端口的作用。

在后续实验中，还会用到 8255A。

3.4 实验报告

① 画出实验中的地址译码电路。

② 说明实验板的线路接法。

③ 分别写出实验③和实验④循环体程序段。

④ 分别写出实验⑤和实验⑥循环体程序段。

实验 4——4×4 小键盘的使用

4.1 实验目的

① 认识键盘矩阵。

② 学会用行扫描法和行反转法识别按键。

4.2 相关背景及说明

在单片机等小应用系统以及智能设备中，经常使用简单的键盘进行输入操作。这些键盘上的每一个按键或开关实际是一个一位二进制数字量。当按键或开关个数比较少时，可直接将其构成输入端口。但是，如果个数比较多时，仍然按通常 8 个构成一个字节型输入端口的做法，则将占用较多的 I/O 端口。这时，一般将按键排列成矩阵。4×4 键盘矩阵的典型电路图如图 4.1 所示。

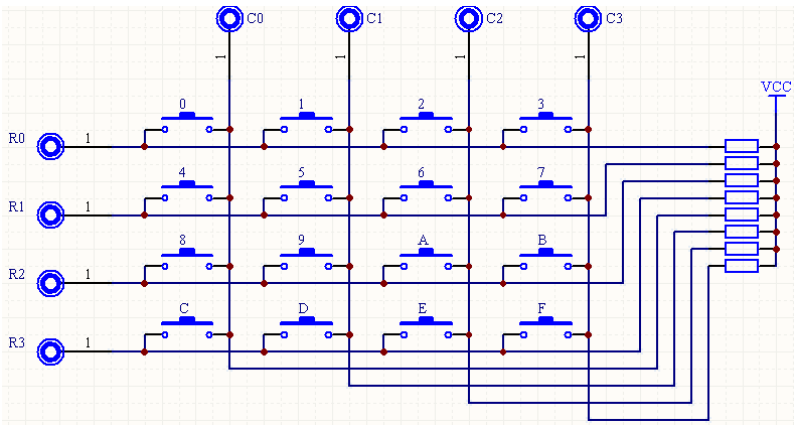

图 4.1　4×4 键盘矩阵模块电路图

在图 4.1 中，有 4 根行线（水平线）和 4 根列线（竖线），行线和列线相交位置上都接有一个按键。这样就构成了一个 4×4 键盘矩阵。不难想象，多行多列可构成更大规模的键盘矩阵。图的右边有一个排电阻，公共端接 VCC，每个电阻的另一端与一根行线或列线相连。这种电阻接法是与识别按键的方法有关。每根行线和列线都安排了一个引脚。

为了识别键盘上的闭合键，通常可以采用两种方法：行扫描法和行反转法。

行扫描法识别按键的原理如下：先使第 0 行接地，其余行为高电平，然后读取列线的电位。如果有某列线为低电平，则表示第 0 行与该列线相交的位置上的键被按下；如果没有任何一条列线为低电平，则说明第 0 行没有任何键按下。此后，再将第 1 行接地，然后检测是否有变为低电平的列线。如此往下一行一行地扫描。在扫描过程中，当发现某一行有键闭合（即被按下）时，便可结束扫描，通过组合行线和列线即可识别此刻按下的是哪一键。

实际操作时，先快速检查有无键按下，然后再确定具体位置。参看图 4.1，检查有无键按下通过向行线输出全 0 读列线实现。如果列线输入端口的读入值为 0x×f（假定列线接到输入端口的低 4 位，所以只需考虑低 4 位，高 4 位是要去掉的，这里用×表示），则表明无键按下，否则，表明有键按下。如果发现有键按下，则进行下一步，即判断哪一个键被按下。为此，输出端口依次输出行扫描码。所谓行扫描码，是指欲扫描的的一行对应的位为 0 而其它行对应的位都为 1。这里共需 4 个行扫描码，其十六进制值分别是×E、×D、×B、×7（×表示任意，这里假定行线接到输出端口的低 4 位）。每当输出一个行扫描码后，读一次列线输入端口。若读入值为 0x×f，则继续输出下一个行扫描码；若读入值为非 0x×f，则找到

了闭合键。拿此次读入值（反映了闭合键所在的列）及行扫描码（反映了闭合键所在的行）进行后续处理（如直接组合在一起）即可形成闭合键的编码。

行反转法要求与之配合的行列线的数据端口能改变输入输出方式。先将行端口设置为输出口，列端口设置为输入口。通过行端口输出一个（低 4 位）全 0 的字节数据，然后读入列线值。如果此时有键按下，则必定会使某列线值为 0。接着，程序改变对两个端口的设置：使行端口变成输入口，而列端口变成输出口。接下来，将刚才读得的列值从列端口输出，从行端口读取行线的输入值。显然，闭合键所在的行线值为 0。这样，当一个键被按下时，必定可以读到一对唯一的行值和列值。

不难看出，行反转法比行扫描法方便并且发现按键所需的时间少（指平均发现按键的时间）。但是行反转法要求行列线的数据端口能改变输入输出方式。8255A 是可编程的，恰好有这种功能，并且可以取多种组合，例如，A 口作为行数据端口，B 口作为列数据端口，或者 C 高 4 位口作为行数据端口，C 低 4 位口作为列数据端口等。

需要指出，一般的键在按下和释放时会有抖动，抖动的持续时间与键本身的特性以及操作员的操作有关，不过通常为数十毫秒。抖动过后按键才会稳定在接通或断开状态。消除抖动的影响可用硬件方法也可用软件方法。软件方法是：发现有键按下时，延时一段时间后再调用键盘扫描程序。如果两次所得的结果相同，则表明按键处于稳定状态，因此，所得结果正确反映了键盘的当前状态。延时的时间可通过试验确定。

显然，上面所介绍的键盘矩阵，只有执行键盘扫描程序，才会发现是否有键按下。所以要周期性地执行键盘扫描程序。如果想减少程序在键盘上的开销，可采用定时中断的方法，在中断服务程序中执行键盘扫描程序，也可以采用具有中断功能的可编程键盘接口芯片（如 Intel 8279 等）。由于受 Windows 环境下中断处理不便的限制，本实验仅要求采用循环扫描的方法，目的是掌握识别键盘矩阵中闭合键的基本方法。

4.3 实验内容

① 用 8255A 作接口，用行扫描法识别闭合键。要求程序编成循环结构，不按键时无输出，按下某个键，将其对应的字符显示出来，并且只显示一次。

② 了解行反转法编程要点。

4.4 实验报告

① 画出实验中的地址译码电路。

② 说明实验板的线路接法。

③ 写出实验①的程序。

实验 5——七段数码管的使用

5.1 实验目的

① 认识多位七段数码管。

② 学会用软件方法进行段选和位选（称为动态显示）。

5.2 相关背景及说明

在单片机等小型应用系统以及仪器仪表中，经常使用七段数码管或十六段数码管，以前者居多。七段数码管由七条发光线段组成，它们排列成“日”字形，各段依次记为 a、b、c、d、e、f、g，另外还有一个小数点，记为 dp，如图 5. 1（a）所示。每条发光线段（或点）是一个发光二极管，可以排列成共阴极或共阳极形式，如图 5. 1（b）（c）所示。

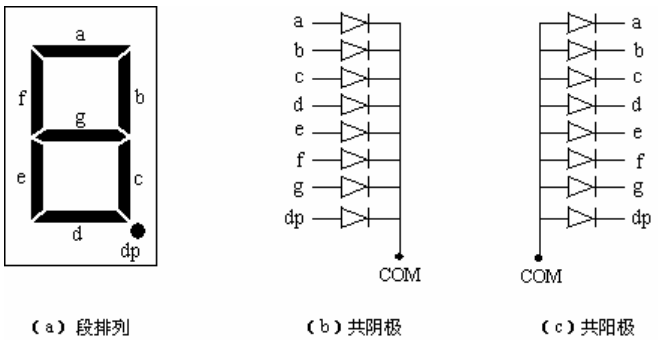

图 5.1　七段数码管结构示意图

一个七段数码管可显示一位十进制数码，而通常的显示一般需要若干位。为了减少电路板上的连线，一些制造商将若干个原本独立的七段数码管封装在一起并在内部进行了一些连接。图 5.2 是一种 4 位结构，图 5.3 表示了其内部连接。

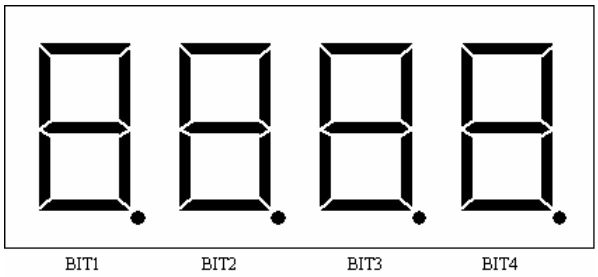

图 5.2　4 位七段数码管外形图

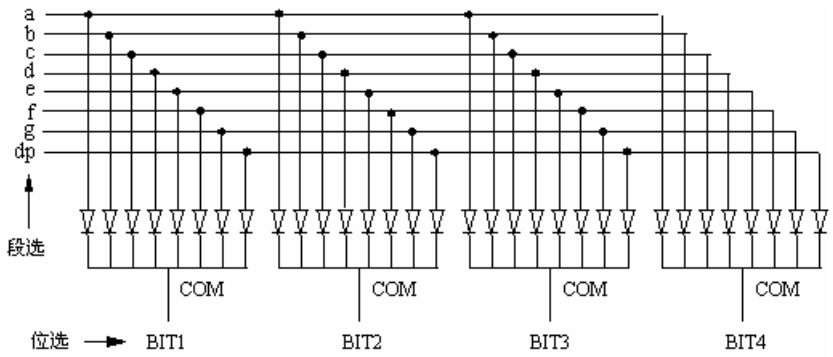

图 5.3　4 位七段数码管的内部连线

这种数码管在内部将各位相同位置的段连在一起，并引出，称为段选信号；4 个公共端分别引出，以进行位选择，称为位选信号。对于这种数码管，通常采用软件方法进行段选和位选。基本做法是，每次选择一位，在送位选信号的同时，提供该位的段选信号，于是该位被“点亮”，接着进行下一位。由于存在“余辉”，所以只要按一定周期，对各位依次进行操作，就会看到 4 个数码管同时被“点亮”。这种显示称为**动态显示**。显然，动态显示软件开销较大（需要周期性对各位操作，一旦周期打乱或者周期时间过长，显示将变得不稳定或闪烁）。与此相对应，还有另一种方法，称为**静态显示**。对于静态显示，每位数码管是独立的。需要为每个数码管配备一个 BCD-七段译码器/驱动器（为了方便编程，送往数码管显示的数据一般采用 BCD 码），为每两位（十进制）数码显示配备一个字节型输出端口用于 BCD 码的输出和锁存（寄存），因此硬件开销比较大。静态显示的优点是编程方便，因为一次输出将长久起作用，不需要象动态显示那样——周期性对各位操作。在多数小型应用系统中，从降低硬件成本考虑，一般采用动态显示。

5.3 实验内容

① 手动位选和段选：将手动数字量输入模块的引脚接七段数码管显示模块的相应引脚，拨动用于位选和段选的开关，观察数码管的显示内容。

② 程序位选和段选：用 8255A 作接口（端口使用自行确定，最好和键盘矩阵所用端口不要重复），通过程序在数码管上显示一个 4 位十进制数（数据自定）。

③ 和 4×4 键盘矩阵结合：在 4×4 键盘矩阵上按一个键（0～9），将该键对应的编号（0～9）在最低位数码管上显示出来（原有的显示被去掉）。要求编成循环结构，上述操作可重复进行。按非数字键，程序退出。

④（选做）在实验③的基础上改进：在 4×4 键盘矩阵上按一个键（0～9），将原有的显示向左移一位，新输入的按键编号（0～9）在最低位数码管上显示出来；按“B”键原有显示向右移一位，最高位补 0 或不显示；按其他键程序退出。

5.4 实验报告

① 画出实验中的地址译码电路。

② 说明实验板的线路接法。

③ 分别写出实验③和实验④循环体程序段。

实验 6——D/A 转换器的使用

6.1 实验目的

了解 D/A 转换器的外部特性，学会使用 D/A 转换器。

6.2 相关背景及说明

D/A 转换器，即将数字量变成模拟量的转换器，也是计算机本身及应用系统中的常用电路。例如，在声卡中就用到 D/A 转换器，将数字信号变成音频电压信号。有了 D/A 转换器，计算机就可以实现各种控制，如对生产过程进行控制。D/A 转换器也早已集成化，种类、型号也非常多。本实验系统中安排了一片 DAC0832。这是用 CMOS 工艺制成的 8 位精度的 D/A 转换芯片，它所接收的待转换的数字量为二进制 8 位（TTL 电平）。转换成的模拟量是电流，电流建立时间为 1μs，若要变成与数字量对应的电压，需要外接一个运算放大器（简称运放）。

为了理解 DAC0832 的连接，有必要看一下它的内部结构框图（图 6.1）。

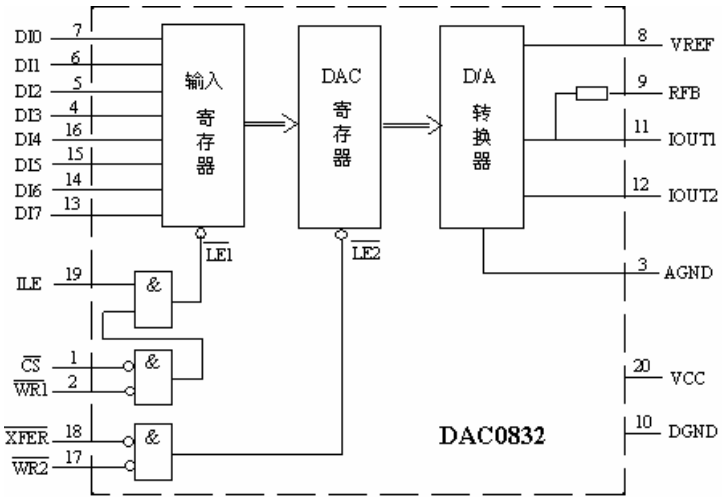

图 6.1　DAC0832 内部结构框图

DAC0832 由四个部分所组成：输入寄存器、DAC 寄存器、D/A 转换器和输入控制逻辑。这是一种两级数据缓冲结构。当数据进到第二级，便进行 D/A 转换。之所以这样安排，是考虑到一种特殊需要：一个系统中使用多片 D/A 芯片，要求各芯片同时开始进行 D/A 转换，时间上严格保持一致（可称为同步转换）。这时，把第二级控制信号都连在一起，需转换时先让各待转换数据依次（按程序的执行顺序）写入对应 D/A 芯片的第一级寄存器，然后发一个命令，使各 D/A 芯片的第一级寄存器的内容同时进入到第二级，因而同时开始 D/A 转换。

图中内部信号 LE1#和 LE2#为锁存允许（Latch Enable）。为高时，寄存器为跟随状态，即输出随输入的变化而变化；变低的瞬间，输入端的状态被锁存到寄存器中。

ILE、CS#和 WR1#为第一级缓冲控制。ILE（Input Latch Enable）为输入锁存允许，高有效，一般接高。CS#为芯片的片选，低有效。WR1#为写信号 1，低有效。下面是第一级缓冲控制的时序图（图 6.2）。

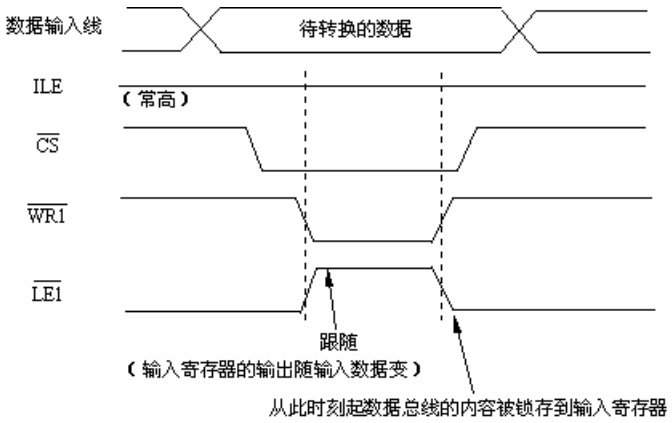

图 6.2　第一级缓冲控制的时序图

图中已表示出信号的配合及作用。可将该时序关系简述为：当 CS#和 WR1#同时有效时，待转换的数据被锁存到输入寄存器。

XFER#和 WR2#为第二级缓冲控制。XFER#为传输控制信号（Transfer Control Signal），低有效。WR2#为写信号 2，低有效。LE2#的产生逻辑和 LE1#基本相同。所以，其时序关系可简述为：当 XFER#和 WR2#同时有效时，输入寄存器中的内容被锁存到 DAC 寄存器，随即开始 D/A 转换。

实验板中 D/A 转换模块的电路连接如图 6.3 所示。

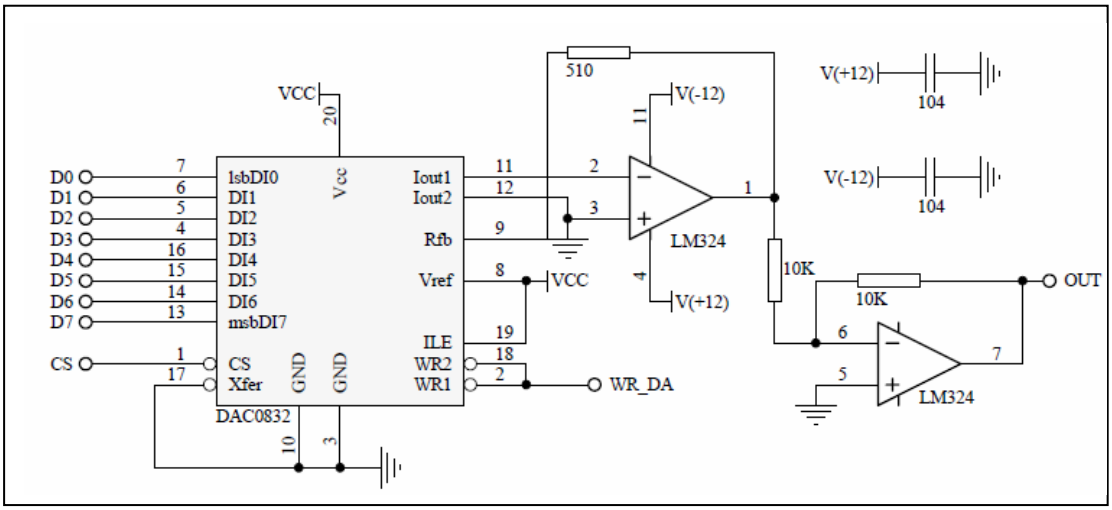

图 6.3 D/A 转换模块的电路连接图

图中，左上角是基准电压（VREF）产生电路。其原理和 A/D 转换所需的基准电压产生电路相似，但有两点不同：一是供电电压这里是－12V,二是稳压二极管的极性变了。因此这里产生的基准电压是负值。

图中 LM324 是通用运算放大器。按照图中的接法，该运算放大器的输出的电压值与 D/A 转换数据成正比，具体如下式所示：

$$V_{OUT} = -\frac{D}{2^8}VREF$$

这里，D 是待转换的数字量，除以 2 的 8 次方是因为转换的数字量为二进制 8 位。

6.3 实验内容

① 从[0,255]区间分别取数输入到 D/A 转换器中，用万用表测 D/A 转换器的输出。并根据测量结果画出输出电压和转换数据之间的转换关系。

6.4 实验报告

① 画出实验中的地址译码电路。
② 根据测量结果画出输出电压和转换数据之间的转换关系曲线。

实验 7——A/D 转换器的使用

7.1 实验目的

① 了解 A/D 转换器的外部特性，学会 A/D 转换器与用户总线接口电路的构成。

② 学会对多通道 A/D 转换器编程，掌握查询和定时两种方法。

7.2 相关背景及说明

A/D 转换器，即将模拟量变成数字量的转换器，是计算机本身及应用系统中常用的电路。有了它，可以将各种模拟量采集到计算机中（对于非电模拟信号，需要先通过传感器或变送器变成电信号），进行数据处理、统计或决策，有时再通过 D/A 转换器和执行结构，实现对生产过程或被控对象的自动控制。A/D 转换器早已被集成化，并有多种类型。本实验系统中安排了一片 ADC0809，虽然简单，但颇具代表性。该转换器采用 CMOS 工艺制成，可分时转换 8 路（或者说成 8 个通道）模拟信号，其内部含有 8 选 1 模拟开关，转换时间为 100 μs，转换精度为 8 位，即转换结果为 8 位二进制数字量。实验板中对 ADC0809 已进行了一些连接，并在一些引脚处安排了插孔，如图 3.1 所示。

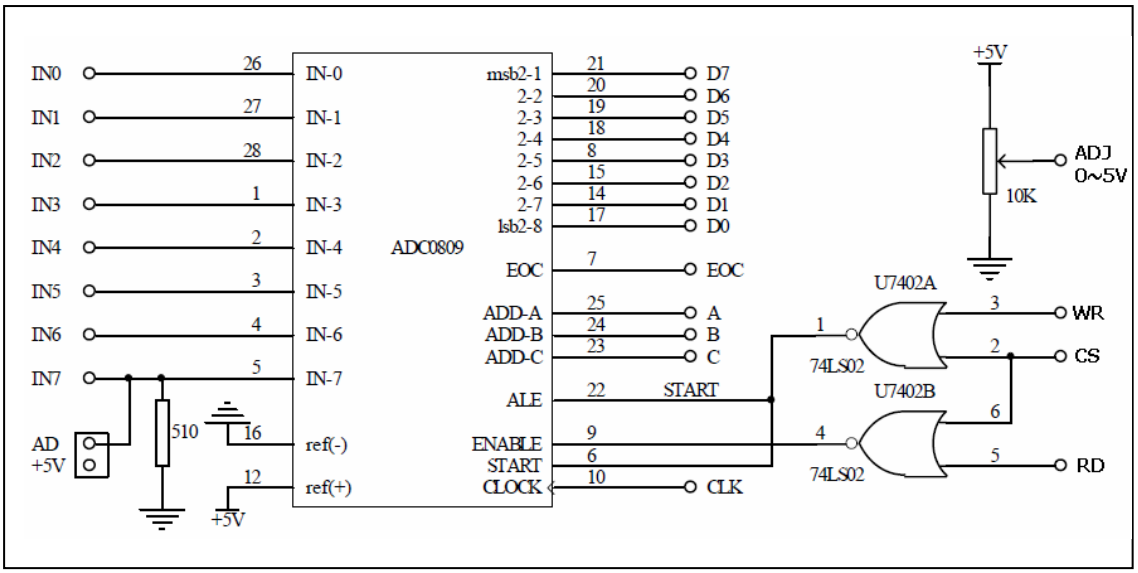

图 3.1　ADC0809 连接示意图

图中，IN0～IN7 为模拟量输入。ADD C、ADD B 和 ADD A 为通道地址输入。内部的译码器对通道地址译码，选中对应的一路模拟量去进行 A/D 转换（ADD C、ADD B 和 ADD A 为 000 时，选中 IN0，为 001 时，选中 IN1,其他组合依此类推）。

CLK 为时钟信号（A/D 转换需要在 CLK 的作用下用若干个 CLK 周期才得以完成），其频率（f_C）的典型值为 640KHz,最大值为 1280KHz,最小值为 10KHz。请从实验箱选取符合要求的时钟频率。

ALE 为地址锁存允许信号，当它从低到高的跳变时将通道地址的输入锁存。

START 为启动信号，正脉冲有效，它的上升沿将内部寄存器复位（为转换作准备），下降沿启动 A/D 转换。

ADC0809 对 START 和 ALE 的脉冲宽度（高电平持续时间）的最小值（前者称为 T_{WS}，后者称为 T_{WALE}）有要求，并且要求相同：典型值为 100ns，最大值为 200ns。这就是说，这两个信号的脉冲宽度不能小于 100 ns，至少不能小于 200ns。

EOC 为转换结束标志。在转换期间 EOC 为低电平，一旦转换完毕，则变成高电平。EOC 可作为查询信号，了解转换是否结束；也可以作为中断请求信号，在转换结束时由低到高

的跳变提出中断请求。

EOC 和 START 的时序关系如图 3.2 所示。

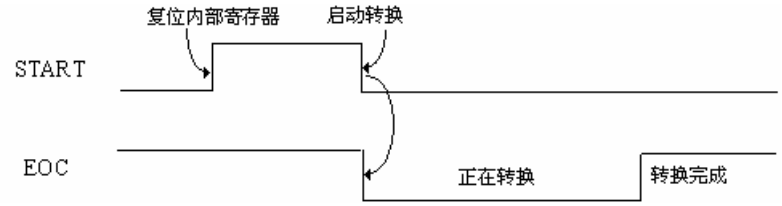

图 3.2　EOC 和 START 的时序关系

D7～D0 为转换数据输出，内部有三态门，所以属于三态输出。

OE 为输出允许，高电平有效，有效时打开内部输出三态门，使转换结果（8 位二进制数）出现在对外的数据线上。

在图 3.1 的右下方是基准电压（VREF+）产生电路。根据 A/D 转换原理，模拟输入信号 Vin 与转换结果 B（二进制数）有下面的关系：

$$B = \frac{V_{in} - V_{ref-}}{V_{ref+} - V_{ref-}} \times 2^n$$

对 A/D 转换器的编程可采用查询法或定时法。对一个通道进行 A/D 转换，两种方法的流程分别见图 3.3 中的（a）和（b）。

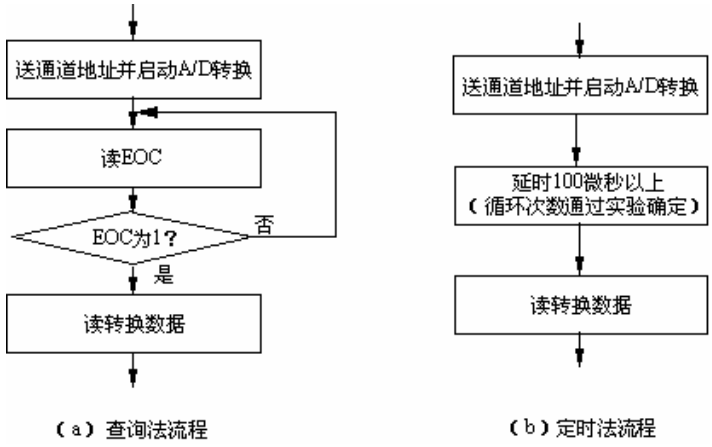

图 3.3　查询法流程和定时法流程

7.3 实验内容

① 构建 ADC0809 与 CPU 总线间的接口电路。

② 用查询法和定时法分别采集 8 个通道的 A/D 转换读数（又称采样值），并转换成对应的电压，将结果显示出来。(各通道的输入电压自定义,可接地或者 Vcc)

③选取若干个数进行 D/A 转换，再通过 ADC0809 的某个通道进行循环采集和转换,并将结果显示出来。

7.4 实验报告

① 画出地址译码电路（注明端口地址安排）。

② 写出为完成实验②及实验③所编写的程序段。

实验 8——可编程计数器/定时器 8254 的使用

8.1 实验目的

① 加深对 8254 计数功能和定时功能的理解。

② 加深对 8254 可编程特性以及前四种工作方式的理解。

③ 学会用 8254 构建一些基本应用电路。

8.2 相关背景及说明

可编程计数器/定时器 8253/8254 几乎在所有微机接口技术的书中都会有介绍，这是因为计算机本身以及多数应用系统中都要用到计数功能和定时功能。众所周知，8253 内含 3 个独立的 16 位计数器，每个计数器有 6 种工作方式，而 8254 是 8253 的改进型。8254 的引脚、工作方式与 8253 完全相同。改进主要在两个方面：一是计数频率更高，二是 8254 多了一个读回命令，该命令可将选择的计数器的状态字和（或）当前计数值一起锁存，供 CPU 读取。在目前的情况下，8254 用的可能更多一些，因为利用读回命令可知道初值是否写入了计数器。因此，本实验系统对于计数/定时功能的实验安排的是 8254。图 3.1 是实验板上 8254 的引脚示意图。

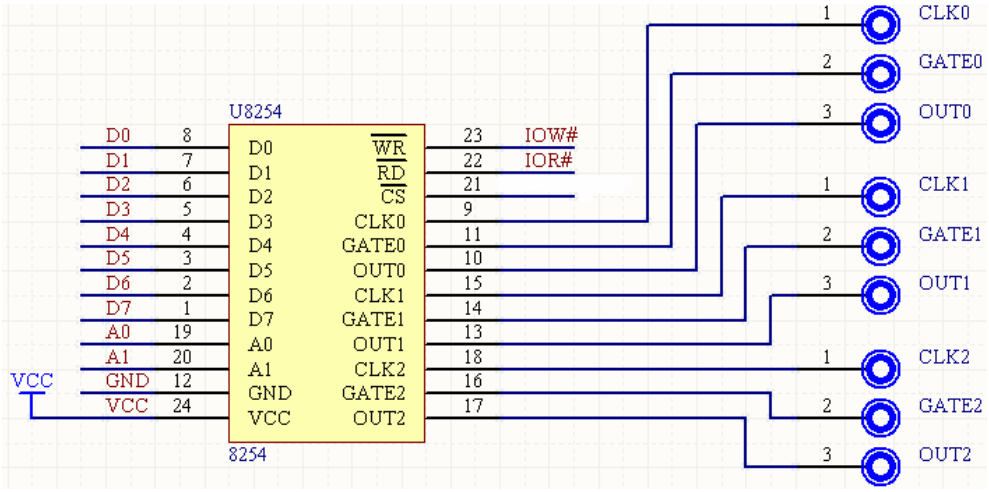

图 3.1　8254 引脚示意图

如果需要为某个计数器提供计数脉冲，可从实验箱的时钟源单元引出。

8.3 实验内容

① 了解计数脉冲来时的计数规律。

先给计数器 0 提供计数脉冲（频率不限），门控置成允许，然后分别按以下要求对其进行设置，用循环方法（循环 200 次）读取计数器 0 的值，观察计数值的变化：

1）方式 2，初值为 8；

2）方式 2，初值为 9；

3）方式 3，初值为 8；

4）方式 3，初值为 9；

5）方式 3，初值为 1000；

② 构建脉冲计数器，记录脉冲数。

先进行连接：将 3 个计数器的门控均置成允许，3 个计数器的计数脉冲输入端分别接入不同频率的时钟信号。然后编程，在对 3 个计数器均置成方式 0 并置初值为 65535（或 0）之后，延时同样一段时间后，读取 3 个计数器的值，计算出每个计数器所记录的脉冲个数。

说明：在程序中**延时可利用 clock（）**，该函数返回程序从开始执行到调用处处理机（CPU）所用的时间，除以 CLK_TCK 即得秒数。注意本实验系统实际测试 CLK_TCK 为 18.2。注意，该函数在 time.h 中定义，因此需要引用此头文件。下面是延时 10 秒钟的程序段：

```
clk1=clock(); // clk1、clk2 定义成 int 型即可
do
{ clk2=clock();}
while ((clk2-clk1)<182);
```

为了去除初始化顺序及需要一个计数脉冲才能将初值写入的影响，可进行两次延时，即先延时少许时间（如 1 秒），读取并记住此刻各计数器的计数值，然后，再延时较长时间（如 10 秒），到时再读取各计数器的计数值，拿上次值减本次值即可得后一个时间段内的脉冲个数。

注意:由于本实验系统所用的脉冲频率差别较大，因此这个实验中延时时间选择要尽量的小，以免超出计时量程造成错误。

③ 构建可编程定时信号发生器，并记录一个时间段内其发出脉冲的个数。

将计数器 0 和计数器 1 串接，组成 32 位计数器（这样时间调节范围可大些）。自行选择计数器 0 的计数脉冲以及计数器 0 和计数器 1 的初值，最终使计数器 1 输出周期为 0.1s 的方波。再参照实验②用计数器 2 记录 3 秒内计数器 1 发出的脉冲的个数。

8.4 实验报告

① 说明做实验①时所观察到的计数规律。分 5 种情况，对第 5 种仅说明值的变化范围，而其他几种情况，列出计数过程中所有出现过的值（每种出现过的值只需列一个）。

② 列出做实验②时所记录的数据，并解释不同计数器之间数据差异的原因。

③ 写出实验③的程序，列出可编程定时信号发生器在 3 秒钟内周期为 0.1s 时实际发出的脉冲数。

第二章 开放性实验设计与案例

本阶段实验主要是利用 USB2.0 应用开发平台，并结合前期实验所用的实验箱与FPGA 板，进行自主设计实验。同学们可以多人自由组合，结合课堂所讲授的 USB 原理，共同完成一个实验案例。

典型 USB 实验案例如下：

2.1 案例 1---USB 设备的 I/O 扩展

利用 USB 设备中转换的各种信号线，实现通过计算机 USB 总线与外部 I/O 设备之间进行数据交换。提示：需要利用接口实验箱上的 8255 芯片来实现 I/O 数据位的扩展，可以接入小键盘及 LED 灯等输入输出设备。

2.2 案例 2---USB 数据采集系统

利用 USB 设备中转换的各种信号线，以及接口实验箱上的 ADC0809 和 DAC0832 芯片来实现数据采集系统。可自主设计应用场景，使其具有一定的实际应用价值。

2.3 案例 3---USB 无线控制系统

研究开发出一套基于 USB 无线控制系统，利用无线电发射与接收模块实现对计算机的远程控制，同时实现计算机对功能设备的控制，实现无线电的双向控制。提示：需要额外添加射频接收芯片，比如 RX3310A 以及配套的发射电路和编解码电路相配合来实现无线遥控和数据传输等功能。