第六章 习题解答

1. IO 接口的一般结构包括哪些部分？它们各具有什么功能？

解答：接口电路主要由控制命令逻辑电路、状态设置和存储电路、数据存储和缓冲电路3部分组成

控制命令逻辑电路一般由命令寄存器和控制执行逻辑组成，这一部分是接口电路的“中央处理器”，用来完成全部接口操作的控制。

状态设置和存储电路主要由一组状态寄存器构成，中央处理器和外设就是根据状态寄存器的内容进行协调动作的。

数据存储和缓冲电路也是一组寄存器，用于暂存中央处理器和外设之间传送的数据，以完成速度匹配工作。

2. IO 接口的数据通信方式分为几类？各有什么优缺点？各自主要的应用场景有哪些？

解答：

只有两种：串行传送和并行传送。

并行数据的每一位都对应独立的传输线路，所以数据传送速度快，但线路多，一般只用于较短距离的数据传送。

串行数据传送主要用于远程终端或经过公共电话网的计算机之间的通信。远距离数据传送采用串行方式比较经济。但串行数据传送比并行数据传送控制复杂，要求收发双方遵从统一的通讯协议。

3. IO 接口的通信控制方式有哪些？分别具有什么特点？

解答：

查询、中断和DMA 3种方式控制接口的传送操作。

查询方式是中央处理器在数据传送之前通过接口的状态设置存储电路询问外设，待外设允许传送数据后才传送数据的操作方式。在查询方式下，中央处理器需要花费较多的时间去不断地“询问”外设，外设的接口电路处于被动状态。

中断方式是在外设要与中央处理器传送数据时，外设向中央处理器发出请求，中央处理器响应后再传送数据的操作方式。在中断方式下，中央处理器不必查询外设，而由接口在外设的输出数据发送完毕或接收数据准备好时通知中央处理器，中央处理器再发送或接收数据。

DMA方式是数据不经过中央处理器在存储器和外设之间直接传送的操作方式。DMA方式是这3种方式中效率最高的一种传送方式，DMA方式控制接口也最复杂，需要专用的DMA控制器。

4. IO 接口寻址方式分为几类？各有什么优缺点？

解答：

有两种不同的I/O接口寻址方式：一种是标准的I/O寻址，另一种是存储器映象I/O寻址(memory mapped I/O)。

存储器映象 I/O 寻址方式的优点是：

（1）CPU 对外设的操作可使用全部的存储器操作指令，故指令多，使用方便，如可对外设中的数据(存于外设的寄存器中)进行算术和逻辑运算，进行循环或移位等；

（2）存储器和外设的地址分布图是同一个；

（3）不需要专门的输入/输出指令。

其缺点是：

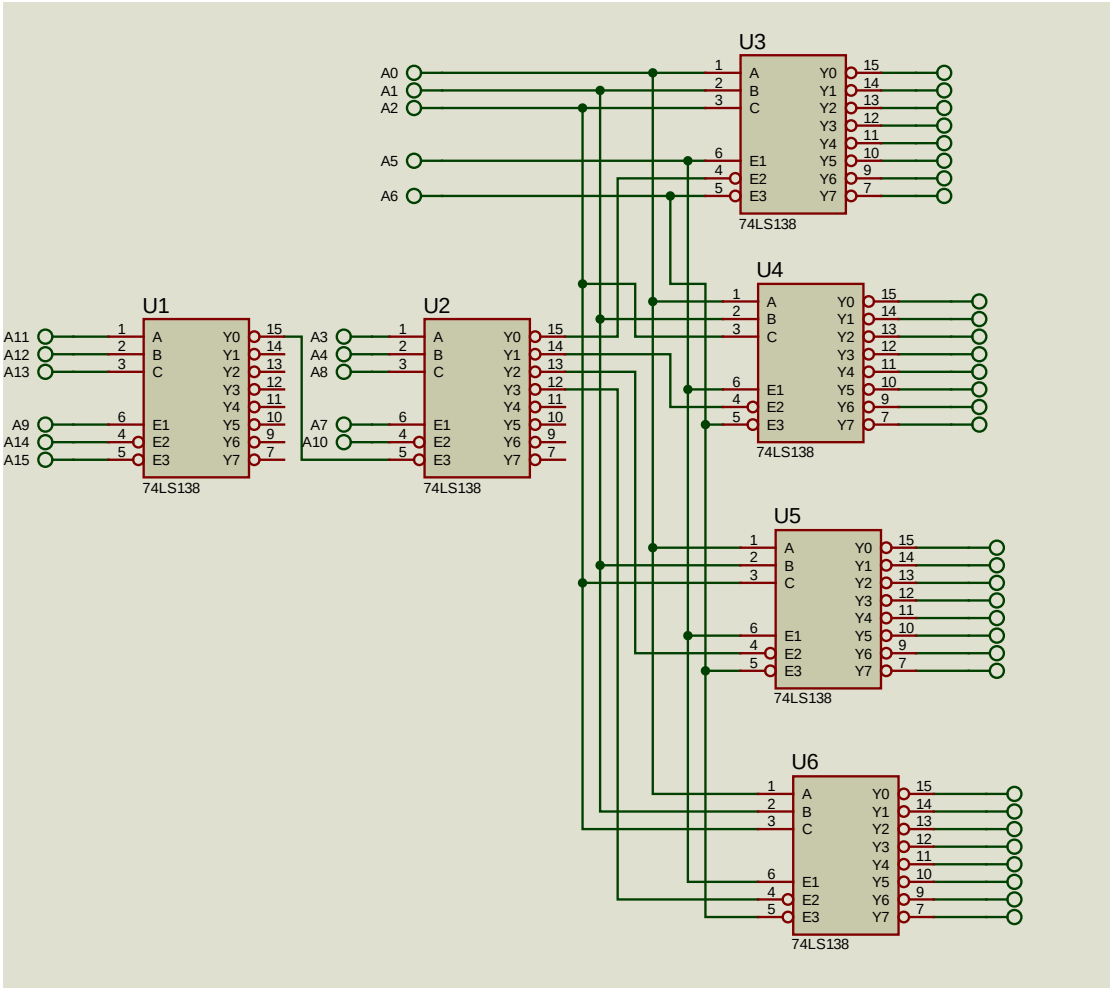
（1）外设占用了内存单元，使内存容量减小；

（2）存储器操作指令通常要比I/O指令的字节多，故加长了I/O操作的时间。

标准I/O寻址方式优缺点正好相反。

5. 设某16 位地址总线的接口要求端口地址范围为0x2A0H～0x2BF，试仅用138 译码器设计端口译码电路，并写出各输出端的地址。

解答：16位地址总线，表示提供了A0~A15地址信号，端口地址范围为0x2A0H～0x2BF，由此可知A15~A5固定为：0000 0010 101；A4~A0可以任意变化，因此需要将A15~A5参与译码，形成138的使能信号，而A4~A0形成各个具体端口地址的选择信号，共32个不同的端口地址，每片138可以提供8个不同的端口地址，由此可知需要4片138，而这4片138需要4个不同的使能信号，因此又需要一片138，这片138共6个输入端，2个输入端为地址选择，剩余的4个输入端可以连接高位地址信号。电路原理图如下图所示。

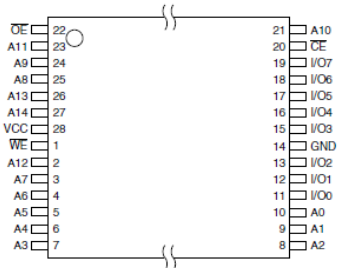


6. 通过查阅存储芯片AT29C256 的数据手册，说明该存储芯片的容量，地址线、数据线宽度以及读写操作时序。

解答：

Pin Name	Function
A0 - A14	Addresses
$\overline{CE}$	Chip Enable
$\overline{OE}$	Output Enable
$\overline{WE}$	Write Enable
I/O0 - I/O7	Data Inputs/Outputs
NC	No Connect
DC	Don't Connect

Type 1



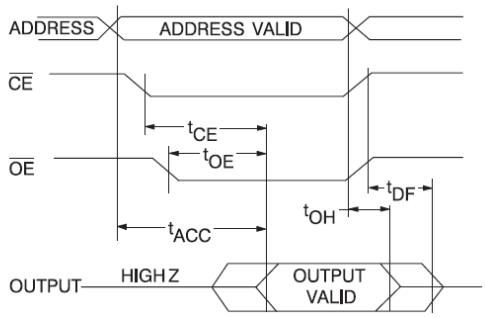
地址线：15位

数据线：8位

容量：32KB

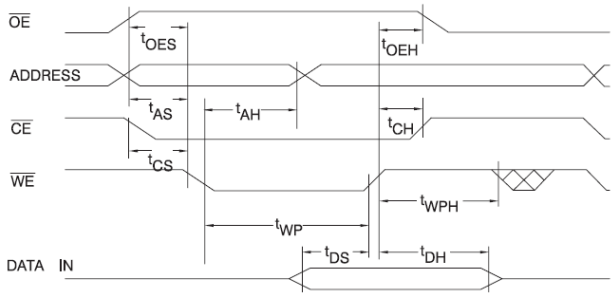
读时序图：

Read Waveforms(1)(2)(3)(4)

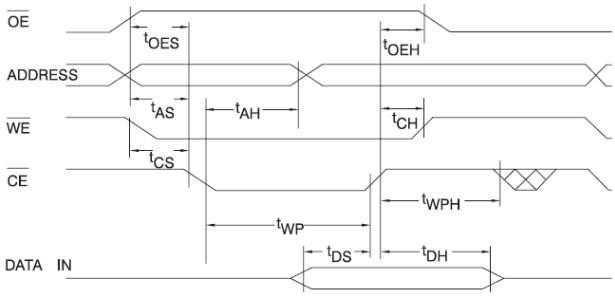


写时序图：

WE Controlled



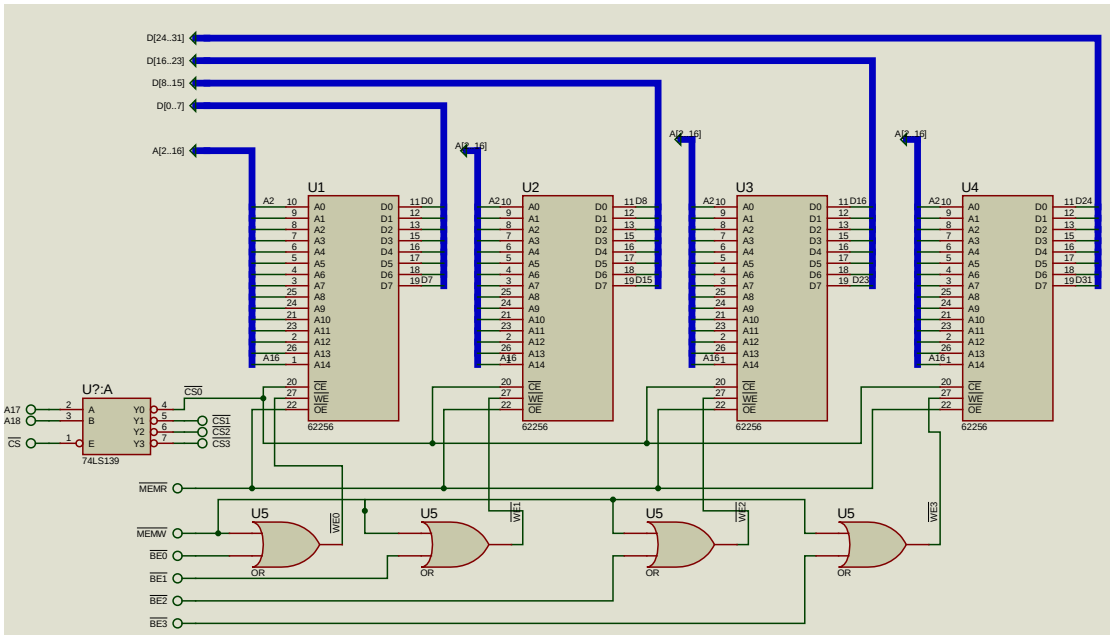
CE Controlled



7. 试为一个32 位的微处理器设计一个存储容量为128K*32b 的SRAM 存储器，要求采用SRAM 62256 存储芯片，并且该存储器需能支持字节、半字以及字类型的数据访问。

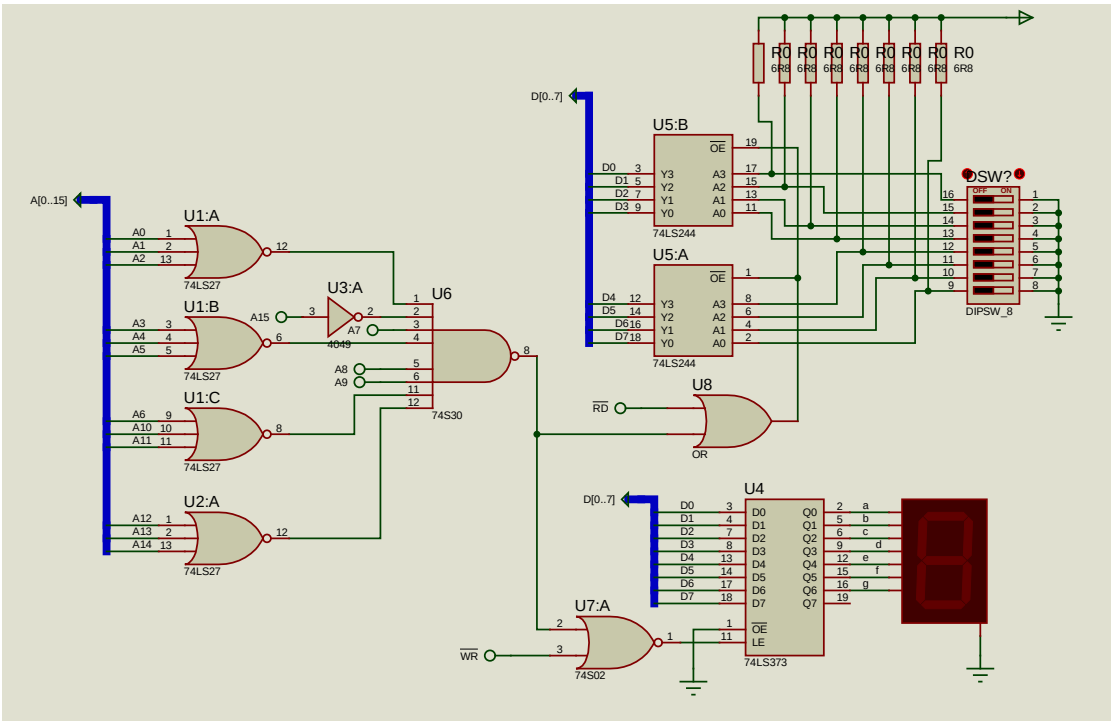
解答：128K*32b的SRAM存储器共需32位数据线（D0~D31），19根地址线（A0~A18），由于要支持字节、半字以及字类型的数据访问，因此提供字节使能信号BE0，BE1，BE2，BE3，从而

A0~A1不再提供。SRAM 62256为32K*8位的SRAM芯片，因此共需=4*4=16片芯片，其中4片构成一组进行字长扩展，4组进行字数扩展。下图画出了采用139译码之后CS0连接的其中一组的电路连接图，这里采用WE信号实现不同字节访问的控制。另外三组除了CE信号分别连接CS1，CS2，CS3，其余信号的连接完全相同。



8. 采用某8 位独立开关输入十六进制字符（0~9，a~f）的ASCII 码，并将该ASCII 码表示的十六进制字符通过一位七段显示器显示出来。试针对8 位数据总线和16 位地址总线的计算机系统设计接口电路和控制程序。要求输入输出端口地址均为0x380，控制程序Xilinx C 语言。地址译码可任选一种方式。

解答：译码电路采用逻辑芯片完成译码的接口电路如下图所示

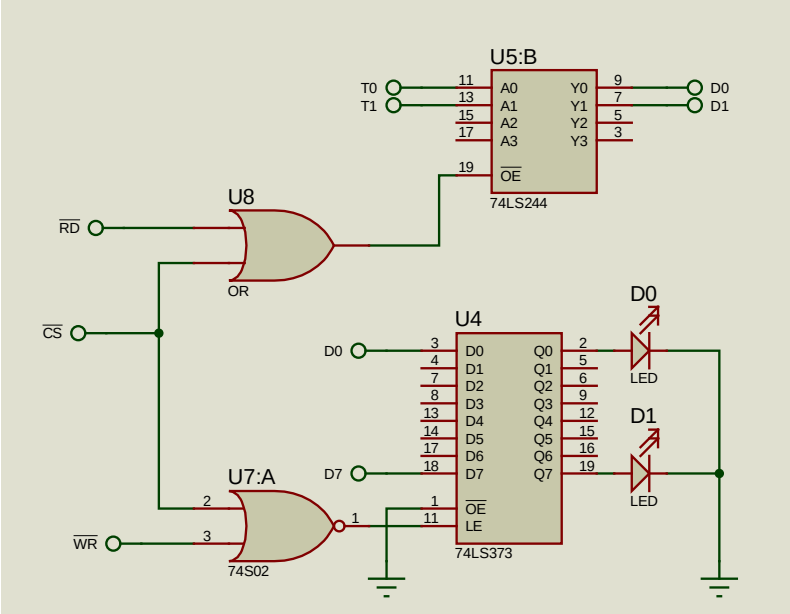


控制程序：
原理：通过输入指令输入的为ASCII码，首先需转换为16进制数字，然后再根据16进制数字查找7段数码管显示的字符表得到16进制数字对应的段码，再将段码通过输出指令进行输出。采用Xilinx C语言实现，主要限定输入输出指令分别采用：Xil_In8(PORT)，Xil_Out8(PORT, DATA).

```
char segcode[17]= {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f,0x77,
0x7c,0x49,0x1e,0x79,0x71,0x40};//“0~9,a~f,-”的段码,共阴极型7段数码管的段码
void main(){
    char asciicode,hexdigit;
    int temp;

    while(1){
        asciicode=Xil_In8(0x380);
        if((asciicode>0x2f)&(asciicode<0x3a))
            hexdigit=asciicode-0x30;//数字0~9
            else if ((asciicode>0x40)&(asciicode<0x47))
                hexdigit=asciicode-0x37;//大写A~F
                else if ((asciicode>0x60)&(asciicode<0x67))
                    hexdigit=asciicode-0x57;//小写a~f
                    else hexdigit=17;//显示横线
        Xil_Out8(0x380,segcode[hexdigit]);//查表并输出
        for(temp=0;temp<0x1000;temp++);//延时
        }
    return;
  }
```

9. 试针对8 位数据总线和16 位地址总线的计算机系统设计一监视2 台设备状态的接口电路和监控程序：若发现某一设备状态异常(由低电平变为高电平)，则发出报警信号(指示灯亮)，一旦状态恢复正常，则将其报警信号撤除。
解答：电路原理图如下图所示，其中T0，T1分别表示监控设备0和设备1的状态线，led灯D0，D1分别表示监控设备0和设备1的报警状态。假设CS译码地址为0x380.

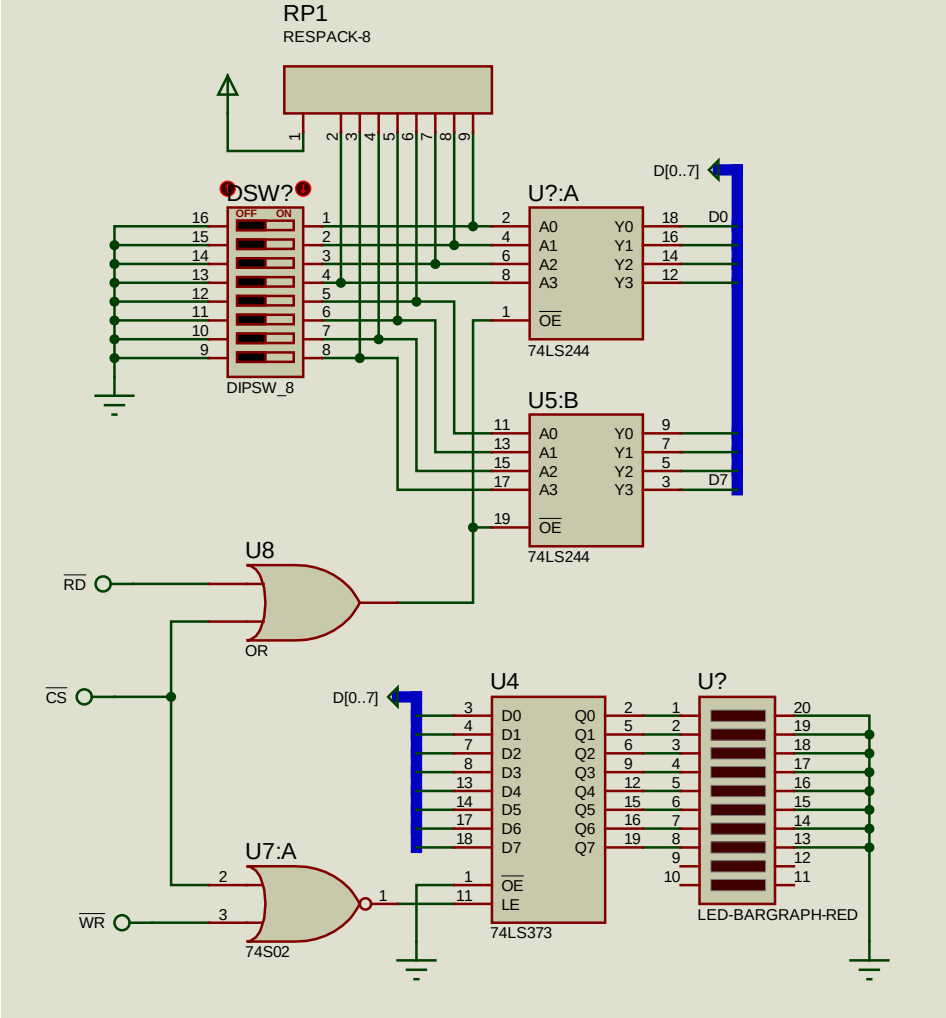


C语言程序源码如下：

```
void main(){
    char state,alarm=0;
    while(1){
        state=Xil_In8(0x380);
        if(state&0x1==0x1)
            alarm=|0x1;//T0为高电平，预输出D0为高电平，其余位不变
            else alarm=&0xfe;// T0为低电平，预输出D0为低电平，其余位不变
        if(state&0x2==0x2)
            alarm=|0x80; //T1为高电平，预输出D7为高电平，其余位不变
            else alarm=&0xef; // T1为低电平，预输出D7为低电平，其余位不变
        Xil_Out8(0x380,alarm);//合并输出
    }
return;
}
```

10. 试针对8 位数据总线和16 位地址总线的计算机系统设计接口电路和控制程序，用8 个理想开关输入二进制数，8 只发光二极管显示二进制数。设输入的二进制数为原码，输出的二进制数为补码。

解答：电路原理图如下：CS可以由任意译码电路产生。假定译码地址为0x380. 控制程序如下：



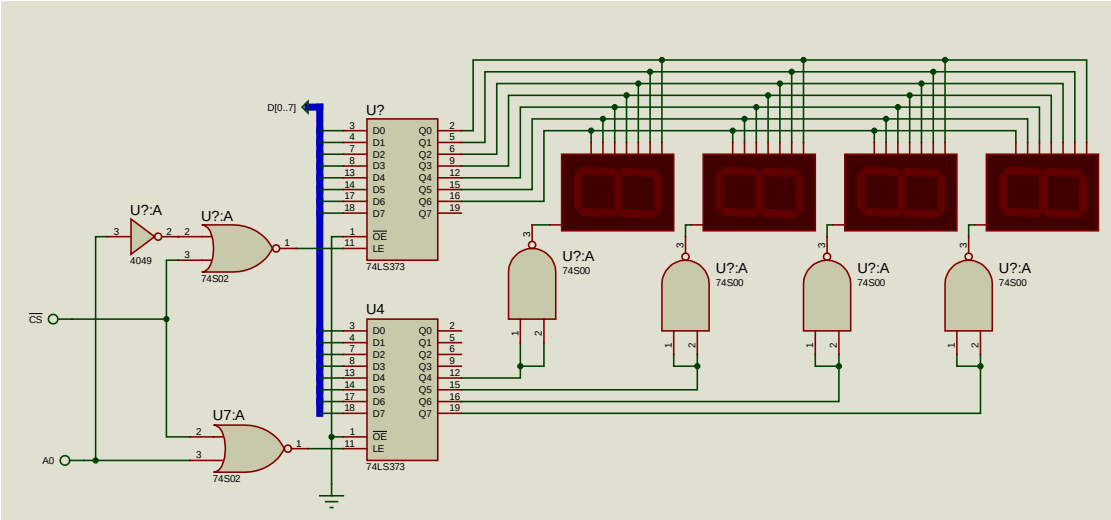

```
void main(){
    char indata,outdata;
    while(1){
        indata=Xil_In8(0x380);
        if(indata&0x80==0x80){
            outdata=indata&0x7f;
            outdata=~outdata+1;//对负数求补码
            }
            else outdata=indata;//正数的补码不变
        Xil_Out8(0x380,outdata);
        }
    return;
}
```

11. 试针对8 位数据总线和32 位地址总线的计算机系统设计接口电路和控制程序，要求用4个7 段数码管显示数字1，2，3，4，且仅占用两个端口地址0x80000000，0x80000001.

解答：

4个7段数码管仅两个端口地址，表明需采用动态显示方式，一个端口输出段码，一个端口输出位码，电路原理图如下，这里的CS来自地址A1~A31以及WE信号的译码，译码电路忽略，

没有画出。




```
char segcode[4]= {0x3f,0x6,0x5b,0x4f};
int addrport=0x80000000;
int dataport=0x80000001;
void main(){
    char position=0x10;
    int i;
    Xil_Out8(dataport,0x0);
    Xil_Out8(addrport,0x00);
    while (1) {
        for(i=0;i<4;i++){
            Xil_Out8(dataport,segcode[i]);
            Xil_Out8(addrport,position);
            delay(2.5ms);
            position=position<<1;
            }
        position=0x10;
    }
return;
}
```

12. 修改图6-49 7 段数码管控制电路原理图利用GPIO 控制器实现题8，9，10，11 的接口电路和控制程序。

解答：

题8，9，10，11都需要两个端口地址，一个GPIO控制器可以提供两个IO通道，因此可以分别采用其中的一个通道控制一个端口。而且所有这些题目的端口数据宽度都小于或等于8位，因此可以将GPIO通道的数据宽度都设置为8位，这里假定所有具有输入、输出端口的电路，统一采用GPIO0_IO作为输入通道，GPIO2_IO作为输出通道，IO数据线的连接顺序不变。控制程序需要首先设置各个通道数据的传输方向，然后才能进行数据输入输出，假定GPIO控制器的基地址为XPAR_GPIO_0_BASEADDR，则GPIO通道0的数据端口地址为XPAR_GPIO_0_BASEADDR，控制端口地址为XPAR_GPIO_0_BASEADDR+4，通道2的数据端口地

址为XPAR_GPIO_0_BASEADDR+8，控制端口地址为XPAR_GPIO_0_BASEADDR+0xC。
题8的控制程序为：

```
char segcode[17]= {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f,0x77,
0x7c,0x49,0x1e,0x79,0x71,0x40};//“0~9,a~f,-”的段码,共阴极型7段数码管的段码
void main(){
    char asciicode,hexdigit;
    int temp;
    Xil_Out8(XPAR_GPIO_0_BASEADDR+4,0xff);
    Xil_Out8(XPAR_GPIO_0_BASEADDR+0xC,0x0);
    while(1){
        asciicode=Xil_In8(XPAR_GPIO_0_BASEADDR);
        if((asciicode>0x2f)&(asciicode<0x3a))
            hexdigit=asciicode-0x30;//数字0~9
            else if ((asciicode>0x40)&(asciicode<0x47))
                hexdigit=asciicode-0x37;//大写A~F
                else if ((asciicode>0x60)&(asciicode<0x67))
                    hexdigit=asciicode-0x57;//小写a~f
                    else hexdigit=17;//显示横线
        Xil_Out8(XPAR_GPIO_0_BASEADDR+8,segcode[hexdigit]);//查表并输出
        for(temp=0;temp<0x1000;temp++);//延时
        }
    return;
  }
```

题9的控制程序为：

```
void main(){
    char state,alarm=0;
    Xil_Out8(XPAR_GPIO_0_BASEADDR+4,0xff);
    Xil_Out8(XPAR_GPIO_0_BASEADDR+0xC,0x0);
    while(1){
        state=Xil_In8(XPAR_GPIO_0_BASEADDR);
        if(state&0x1==0x1)
            alarm=|0x1;//T0为高电平，预输出D0为高电平，其余位不变
            else alarm=&0xfe;// T0为低电平，预输出D0为低电平，其余位不变
        if(state&0x2==0x2)
            alarm=|0x80; //T1为高电平，预输出D7为高电平，其余位不变
            else alarm=&0xef; // T1为低电平，预输出D7为低电平，其余位不变
        Xil_Out8(XPAR_GPIO_0_BASEADDR+8,alarm);//合并输出
    }
return;
}
```

题10的控制程序为：

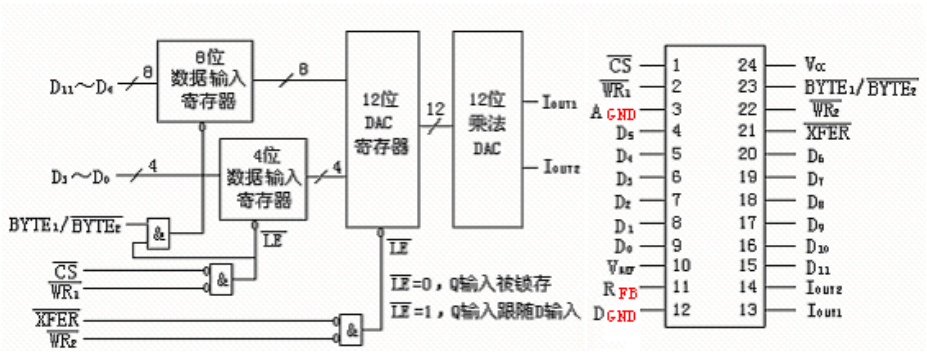
```
void main(){
    char indata,outdata;
    Xil_Out8(XPAR_GPIO_0_BASEADDR+4,0xff);
```

```
    Xil_Out8(XPAR_GPIO_0_BASEADDR+0xC,0x0);
    while(1){
        indata=Xil_In8(XPAR_GPIO_0_BASEADDR);
        if(indata&0x80==0x80){
            outdata=indata&0x7f;
            outdata=~outdata+1;//对负数求补码
            }
            else outdata=indata;//正数的补码不变
        Xil_Out8(XPAR_GPIO_0_BASEADDR+8,outdata);
        }
    return;
}
```

题11的控制程序为：

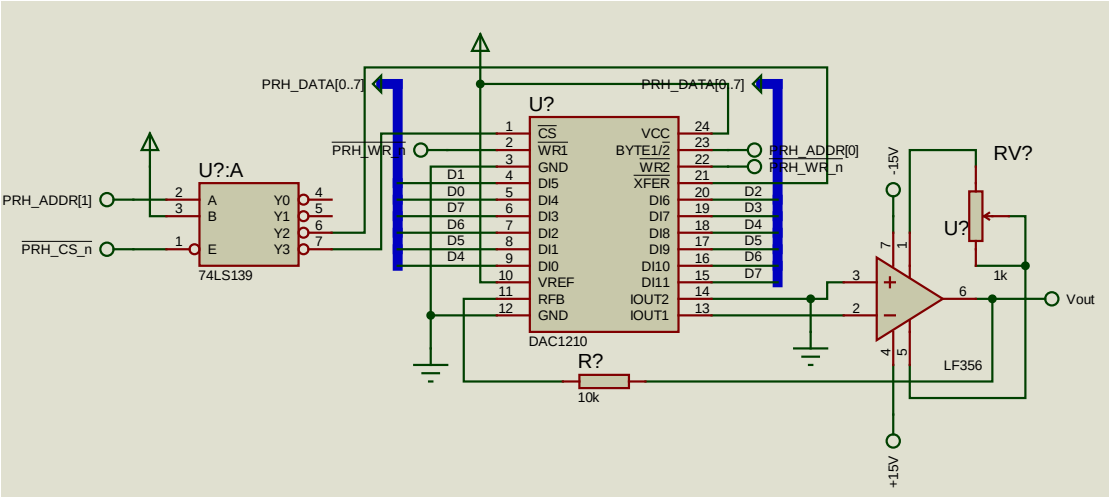
```
char segcode[4]= {0x3f,0x6,0x5b,0x4f};
void main(){
    char position=0x10;
    int i;
    Xil_Out8(XPAR_GPIO_0_BASEADDR+4,0x0);
    Xil_Out8(XPAR_GPIO_0_BASEADDR+0xC,0x0);
    Xil_Out8(XPAR_GPIO_0_BASEADDR,0x0);
    Xil_Out8(XPAR_GPIO_0_BASEADDR+8,0x00);
    while (1) {
        for(i=0;i<4;i++){
            Xil_Out8(XPAR_GPIO_0_BASEADDR,segcode[i]);
            Xil_Out8(XPAR_GPIO_0_BASEADDR+8,position);
            delay(50ms);
            position=position<<1;
            }
        positon=0x10;
    }
return;
}
```

13. 12 位D/A 接口芯片DAC1210 其内部结构如图6- 61 所示。画出DAC1210 与8 位数据线外设控制器之间的接口电路图，采用Xilinx C 语言写出输出周期性锯齿波的程序。



解答：

由于要求采用 8 位 EPC 进行接口设计，因此 EPC 必须提供 1 位地址信号区分低 4 位数据和高 8 位数据的输入，并且由于高 8 位有效输入时，低 4 位也会有效，因此必须先输入高 8 位锁存后，再输入低 4 位锁存在 12 位 DAC 寄存器的输入端，然后再通过一次总线操作将 12 位数据一次送入 12 位乘法 DAC，因此 EPC 共需提供至少 3 个不同的端口地址给 DAC，即需提供 2 位地址信号，供外部译码产生 3 个不同的端口。电路原理图如下图所示。



假定 EPC 的基地址为：XPAR_AXI_EPC_0_PRH0_BASEADDR，那么 12 位 DAC 寄存器的端口地址为 XPAR_AXI_EPC_0_PRH0_BASEADDR+0x3

或 XPAR_AXI_EPC_0_PRH0_BASEADDR+0x4

低 4 位数据输入寄存器端口地址为 XPAR_AXI_EPC_0_PRH0_BASEADDR；高 8 位数据输入寄存器端口地址为 XPAR_AXI_EPC_0_PRH0_BASEADDR+0x1;

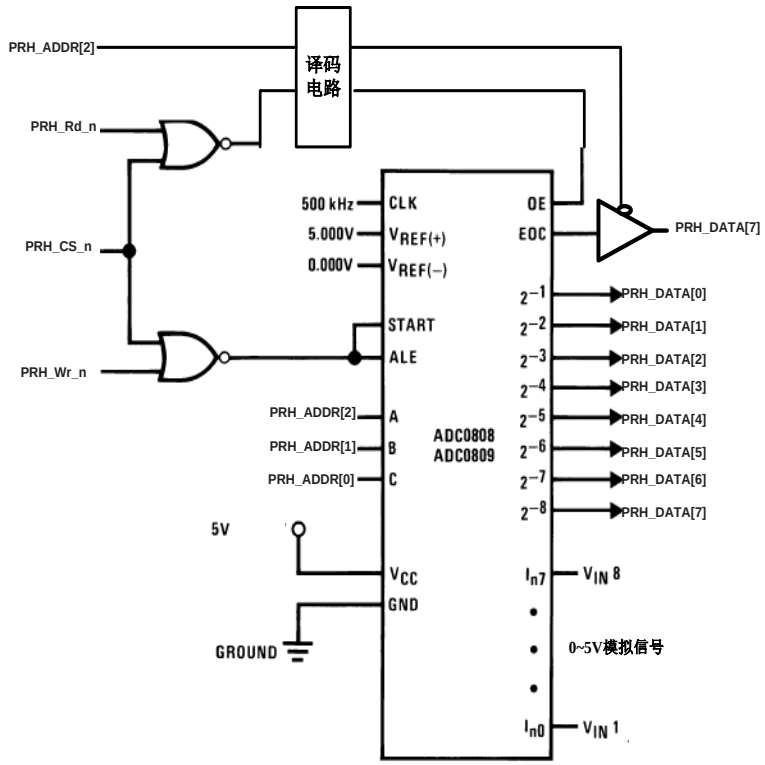
控制程序段为：

```
void main（）{
    short outdata=0;
    char high8,low4;
    while(1){
        high8=(char)(outdata>>8);
        low4=(char)(outdata);
        Xil_Out8(XPAR_AXI_EPC_0_PRH0_BASEADDR+0x1,high8);
        Xil_Out8(XPAR_AXI_EPC_0_PRH0_BASEADDR,low4);
        Xil_Out8(XPAR_AXI_EPC_0_PRH0_BASEADDR+0x3,0x0);
        outdata=+0x10;
        }
    return;
}
```

14. 请针对ADC0808 基于外设控制器用查询方式设计一数据采集接口电路，并编写对8 路模拟量循环采样一遍数据的程序，采集数据存入字节数组BUFF 中。

解答：

电路原理图如下：EPC采用大端字节序，而ADC0808采用小端字节序，EPC配置为支持3位地址线，8位数据线，因此ADDR[2]实质为最低位地址，DATA[7]为最低位数据位。控制程序如下




```
#include<stdio.h>
#include"xparameters.h"
#include"xil_io.h"
Intmain()
{
    unsignedchar data[8],status;
    int i;
    for(i=0;i<8;i++){
    Xil_Out8(XPAR_AXI_EPC_0_PRH0_BASEADDR+i,0x0);//启动通道I ad转换
    Do
    status=Xil_In8(XPAR_AXI_EPC_0_PRH0_BASEADDR+0x01);//查询转换结束信号
    while((status&0x1)!=0x1);
    data[i]=Xil_In8(XPAR_AXI_EPC_0_PRH0_BASEADDR+0x00);//读取转换结果
    }//完成8通道采样
    return 0;
}
```