**[1.5s]简易计算机系统设计**

**[1.5s]**本课程力图以“培养学生现代数字系统设计能力”为目标，贯彻以CPU设计为核心，以层次化、模块化设计方法为抓手的组织思路，培养学生设计与实现数字系统的能力。

**[1.5s]**本设计要求在进行了多个单元实验后，综合利用所学的理论知识，结合在单元实验中所积累的成果（包括已经设计好的功能部件和调试方法），设计出一个简易计算机系统。

**[1.5s]**按给定的数据格式和指令系统，使用EDA工具设计一台用硬连线逻辑控制的简易计算机系统；

要求灵活运用各方面知识，使得所设计的计算机系统具有较佳的性能；

对所做设计的性能指标进行分析，整理出设计报告。

**[1.5s]**数据格式

数据字采用8位二进制定点补码表示，其中最高位（第7位）为符号位，小数点可视为最左或最右，其数值表示范围分别为：-1≤X＜1或-128≤X ≤ 127。

**[1.5s]**寻址方式

指令的高4位为操作码，低4位分别用2位表示目的寄存器和源寄存器的编号或表示寻址方式。

**[1.5s]**在寄存器直接寻址中，指令中包含操作数的地址，指令的低4位分别对应目的寄存器R1和源寄存器R2的编号，源寄存器往往提供指令执行所需的操作数，而目的寄存器在某些指令中也提供操作数并保存执行的结果。

本模型机有三个寄存器A、B、C，其中寄存器A的编号是00，寄存器B的编号是01，寄存器C的编号是10.

**[1.5s]**比如指令MOV C，B，该指令的功能是将寄存器B中的数据写入寄存器C。指令编码为1100 10 01，高4位1100表示这是一条传输指令，低4位10 01表示目的寄存器为寄存器C，源寄存器是寄存器B，设指令执行前寄存器C中存放的数据是34H(H表示该数是十六进制数)，寄存器B中存放的数据是15H，指令执行完后寄存器C中的数据变为15H。

**[1.5s]**若指令的R1或R2中有一个为“11”，表示寻址方式为寄存器间接寻址，此时寄存器C中的数据是RAM的存储单元地址。这种寻址方式需要先读取寄存器C中的数据，再将该数据加载至RAM的地址端口，读取RAM对应单元的数据获得操作数。

**[1.5s]**例如MOV B，M，该指令的功能是从RAM的指定存储单元读取数据写入寄存器B。指令编码为1100 0111，高4位1100表示这是一条传输指令，低4位01 11表示目的寄存器是寄存器B，采用间接寻址方式访问RAM获得操作数。设指令执行前寄存器B中存放的数据是26H，寄存器C中存放的数据是34H，RAM中第34H存储单元中的数据是15H，指令执行完后，寄存器B中的数据变为15H。

**[1.5s]**本模型机要求实现此十六条指令。指令系统表中第一列是指令的汇编符号，第二列是指令的功能，第三列是指令的二进制编码，高4位是指令的操作码，用以区分各条指令，低4位表示寄目的寄存器R1和源寄存器R2的编号，表示操作数的来源。

前三条是MOV指令，实现数据传输，接下来为了便于区分，第一条MOV指令叫MOVA指令，第二条MOV指令叫MOVB指令，第三条MOV指令叫MOVC指令。MOVA指令将寄存器R2中的数据写入寄存器R1，MOVB指令将寄存器R2中的数据写入RAM存储单元，写入RAM的哪个存储单元呢？寄存器C中的数据就是写入的存储单元地址。MOVC指令是从RAM中取数据写入寄存器R1，具体为寄存器C提供RAM的存储单元地址，读取RAM中该存储单元的数据写入寄存器R1。

指令ADD、SUB、AND、NOT是四条逻辑运算指令，分别实现加、减、与和非运算，ADD、SUB、AND三条指令都是从寄存器R1和寄存器R2中读取操作数，分别进行加、减、与运算，再将运算结果写入寄存器R1。NOT指令是将寄存器R1中的数据取反后，再写回寄存器R1。另外，ADD和SUB指令影响状态位CF和ZF，当加法和减法的和（或差）为0时，ZF为1，否则ZF为0，当加法和减法产生进位（或借位）时，CF为1，否则CF=0。

指令RSR和RSL是两条移位指令，RSR指令是将寄存器R1中存放的数据循环右移一位写回寄存器R1，寄存器R1原来存放的是D7D6D5D4D3D2D1D0，执行完RSR后，寄存器R1中的数据变为D0D7D6D5D4D3D2D1且D0写入CF。RSL指令是将寄存器R1中存放的数据循环左移一位写回寄存器R1，寄存器R1原来存放的是D7D6D5D4D3D2D1D0，执行完RSL后，寄存器R1中的数据变为D6D5D4D3D2D1D0D7且D7写入CF。

指令JMP是无条件跳转指令，是一条双字节指令，第一字节为操作码，第二字节为跳转的转移地址，执行完这条指令，程序跳转至转移地址执行。

指令JZ和JC是有条件跳转指令，同样是双字节指令。JZ是结果为0时(即ZF=1)，程序跳转至转移地址执行，结果不为0（即ZF=0），程序顺序执行。JC是结果有进位时（即CF=1），程序跳转至转移地址执行，结果没有进位（即CF=0），程序顺序执行。

指令IN是输入指令，将输入设备的数据写入寄存器R1。

指令OUT是输出指令，将寄存器R2中的数据输出至输出设备。

指令NOP是空操作指令，该指令不执行任何操作。

指令HALT是停机指令，执行完这条指令，模型机进入停机状态，指令HALT后面即使还有其他指令，模型机也不再执行。

**[1.5s]**模型机中的数据在控制信号的控制下进行流动

**[1.5s]**控制信号决定了数据流动的时间和方向

**[1.5s]**数据通路是数据流动的路径

右图为模型机的结构框架，接下来一一介绍各个模块的作用：

**[1.5s]**这根宽数据线称为总线BUS，它由8根导线组成，同时传输8位数据，实现模块间的数据交互。

**[1.5s]**指令计数器PC保存接下来要执行指令在RAM中的存放地址，控制信号LD PC为1，其装载跳转地址，IN PC为1，PC自加1。

**[1.5s]**选择器选择RAM的地址来源，其有3个输入，每个输入8位，分别接指令计数器PC和通用寄存器组的A口和B口，由MADD选择其中一个传递至RAM的地址端口。

**[1.5s]**RAM用来存放指令和数据，它的大小是256×8位，即有256个存储单元，每个单元存放8位数据，所以该RAM有一个8位的地址端口，一个8位的数据端口，数据端口是双向的，DL为1,时钟上升沿将数据从数据端口读出传至总线BUS；XL为1,时钟上升沿将总线上的数据通过数据端口写入RAM。

**[1.5s]**指令寄存器IR用于存放当前正在执行的指令，信号LD IR控制在时钟下降沿将一条新的指令写入指令寄存器IR。

**[1.5s]**SM用来区分当前是取指周期还是执行周期。SM为0，取指周期，SM为1，执行周期。

**[1.5s]**指令译码器根据指令寄存器IR传过来的8位数据，解析是哪条指令，并产生该条指令执行所需的控制信号。

**[1.5s]**通用寄存器组用来保存操作数和中间计算结果，它内部有3个8位寄存器A、B、C，RAA1、RAA0二位控制信号选择A、B、C中的一个作为源寄存器，源寄存器的数据从A口输出，RWBA1、RWBA0二位控制信号选择A、B、C中的一个作为目的寄存器，目的寄存器的数据通过B口输出，WE是写控制信号，在RWBA1、RWBA0的配合下，时钟下降沿将总线上的数据写入对应的目的寄存器。

**[1.5s]**函数发生器ALU主要实现算术逻辑运算，但在执行MOV、RSR、RSL、OUT指令时它提供数据通路，将输入不做任何运算输出出去，MOV和OUT指令是将输入端A的数据输出，RSR和RSL指令是将输入端B的数据输出。ALU的具体运算由控制信号S3S2S1S0以及M决定，执行ADD、SUB指令影响状态位CF和ZF。CF和ZF决定JC和JZ的执行，故CF和ZF的 值要用寄存器存储起来，以便执行JC和JZ时能访问到。

**[1.5s]**移位逻辑主要实现循环右移和循环右移，这两条指令也影响状态位CF。但在执行MOV、ADD、SUB、AND、NOT和OUT指令时提供数据通路，将输入直传至输出，具体执行直传还是移位由控制信号F BUS、FL BUS、FR BUS决定。但要注意的是，当F BUS、FL BUS、FR BUS都为0时，移位逻辑的输出高阻态，释放总线。

**[1.5s]**存储器相当于一栋大楼，大楼里有很多房间，每一个房间都有一个号码，号码编号从0开始，一直到某一个最大值。房间在存储器中叫存储单元，房间号码叫存储单元的地址。

每一个存储单元存放着一条指令或一字节数据。

**[1.5s]**程序在RAM中是顺序存放的，如0号存储单元存放第一条指令，1号存储单元存放第二条指令，依次存放。

**[1.5s]**从存储器的哪一个单元取指令，由指令计数器PC来决定。

**[1.5s]**当语句按顺序执行时，意味着指令要从存储器中按顺序取出来。先取出第一条，再取第二条，再取第三条，等等。这时指令计数器PC从0开始按顺序进行加1计数，先是0，然后是1，再然后是2，等等。计数值作为指令存放的地址送到存储器的地址端口，供取指令时使用。

**[1.5s]**当执行跳转指令时，指令计数器PC中的计数值失效，跳转指令的转移地址从存储器中取出来，然后再送到指令计数器PC中，计数器从新的值开始继续计数。

**[1.5s]**当从存储器中取出数据，注意是取数据，不是取指令，或者向存储器存放运算结果时，存储单元的地址由通用寄存器组提供。

**[1.5s]**模型机中的控制信号及其功能：

指令计数器PC的控制信号IN PC为1时，PC进行自加1操作；LD PC为1时，将总线BUS上的数据写入PC。

指令寄存器IR的控制信号LD IR为1时，允许将总线BUS上的数据写入指令寄存器IR

通用寄存器组的控制信号WE为0时，允许将总线BUS上的数据写入通用寄存器组，RAA1、RAA0是选择源寄存器，RWBA1、RWBA0是选择目的寄存器

移位逻辑的控制信号F BUS为1，将输入的数据直接传至输出，FL BUS为1时，输入的数据循环左移一位传至输出且D7送至CF，FR BUS为1时，输入的数据循环右移一位传至输出且D0送至CF。

选择器的控制信号MADD选择存储器RAM的地址来源，MADD=0时，选择指令计数器PC，MADD=1时，选择通用寄存器的A口，MADD=2时，选择通用寄存器的B口

RAM的控制信号DL为1,读存储器RAM，XL为1，写存储器RAM

函数发生器ALU的控制信号M为1时，表示ALU进行算术逻辑运算，M=0时，ALU进行数据传输。S3-S0控制ALU执行不同的运算。

**[1.5s]**每条指令的执行都有取指令和执行指令两个过程，所有指令的取指令都是相同的，不同的是执行指令。

**[1.5s]**取指令的过程为：

指令计数器PC中的内容经选择器送至RAM的地址端口；在DL和地址的共同作用下，指令在时钟的上升沿从RAM中读出送至总线BUS；在LD IR信号的作用下，BUS上的指令在时钟的下降沿载入到指令寄存器IR；同时指令计数器PC自加1，SM变为1。

**[1.5s]**每条指令执行都需要两个时钟周期完成，第一个周期取指，第二个周期执行。SM=0，表示当前为取指周期。

**[1.5s]**MADD为00，PC中的数据00000000通过选择器传至RAM的地址端口。

**[1.5s]**DL为1,XL为0，F BUS，FL BUS，FR BUS都为0，时钟上升沿，00H地址单元中的指令11000001读至总线BUS。

**[1.5s]**LD IR、WE、LN PC为1、LD PC为0，时钟下降沿，将总线BUS上的指令写入指令寄存器IR，同时PC自加1，SM变为1，准备执行该指令。

**[1.5s]**寄存器之间的传输指令MOV A,B

**[1.5s]**其执行过程为：将编码01对应的寄存器B的数据经A口读出，经ALU、移位逻辑传送至总线BUS；由WE控制在时钟下降沿将BUS上的数据写入编码00对应的寄存器A中。

**[1.5s]**SM等于1，执行MOV A,B指令，具体的执行由控制信号决定。

**[1.5s]**MOV A,B的指令编码是1100 0001，RAA1、RAA0为01，寄存器B中的数据通过A口输出，S3-S0为1100，M为0，ALU将输入端A的数据传至输出，F BUS为1、FL BUS和FR BUS为0，DL和XL都为0，移位逻辑将输入传至总线BUS。

**[1.5s]**WE、LD IR、LD PC、IN PC都为0，RWBA1，RWBA0为00，时钟下降沿，将总线上的数据写入寄存器A，至此指令执行完成，同时SM变为0，准备取下一条指令。

**[1.5s]**寄存器数据传输至RAM的指令MOV M,B

**[1.5s]**其执行过程为：编码11对应的寄存器C中的数据34H从通用寄存器B口读出，经选择器到达RAM的地址端口；编码01对应的寄存器B中的数据26H从通用寄存器组的A口读出，在经ALU、移位逻辑和总线BUS送至RAM的数据端，在XL信号和地址的共同作用下，在时钟上升沿总线BUS上的数据26H写入RAM的34H存储单元。

**[1.5s]**SM等于1，执行MOV M,B，具体的执行由控制信号决定。

**[1.5s]**MOV M,B的指令编码是1100 1101，RWBA1、RWBA0为11，寄存器C中的数据34H从通用寄存器B口读出，MADD为10，通过选择器到达存储器RAM的地址端口。RAA1、RAA0为01，寄存器B中的数据26H从通用寄存器组A口读出，S3-S0为1100，M=0，ALU将输入端A的数据传至输出，F BUS为1、FL BUS和FR BUS为0，移位逻辑将输入传至总线BUS。

**[1.5s]**XL、WE为1、DL、LD IR、LD PC、IN PC都为0，时钟上升沿，将总线上的数据26H写入RAM的34H存储单元，至此指令执行完成。

**[1.5s]**时钟下降沿，SM变为0，准备取下一条指令。

**[1.5s]**RAM数据传输至寄存器的指令MOV A,M。

**[1.5s]**其执行过程为：编码11对应的寄存器C中的数据34H从通用寄存器A口读出，经选择器到达RAM的地址端口；在DL信号和地址共同作用下，时钟上升沿将34H内存单元中的数据12H读出，传送至总线BUS，在WE控制下，时钟下降沿将BUS上的数据12H写入编码00对应的寄存器A中。

**[1.5s]**SM等于1，执行MOV A，M，具体的执行由控制信号决定。

**[1.5s]**MOV A,M的指令编码是1100 0011，RAA1、RAA0为11，寄存器C中的数据34H从通用寄存器A口读出，MADD为01，通过选择器到达存储器RAM的地址端口。

**[1.5s]**DL为1，XL为0，F BUS、FL BUS和FR BUS都为0，时钟上升沿将34H存储单元中的数据12H读至总线BUS。

**[1.5s]**WE、LD IR、LD PC、IN PC都为0，RWBA1、RWBA0为00，时钟下降沿，将总线上的数据12H写入寄存器A，至此指令执行完成，同时SM变为0，准备取下一条指令。

**[1.5s]**算术逻辑指令ADD、SUB、AND的执行过程基本相同，不同的是在ALU中实现的运算不同，所以可以统一将它们写成CCC的形式。

**[1.5s]**以CCC A,B为例，介绍算术逻辑类指令的执行过程：将编码01对应的寄存器B中的数据从通用寄存器组A口读出；同时将编码00对应的寄存器A中的数据从通用寄存器组B口读出；在S3-S0和M的控制下，两个操作数在ALU中进行运算，结果经移位逻辑传输至总线BUS；在WE控制下，时钟下降沿将BUS上的运算结果写入寄存器A中。ADD、SUB指令影响状态位CF和ZF。

**[1.5s]**SM等于1，执行ADD A,B，具体的执行由控制信号决定。

**[1.5s]**ADD A,B的指令编码是1001 0001，RAA1、RAA0为01，寄存器B中的数据从通用寄存器组A口读出，RWBA1、RWBA0为00，寄存器A中的数据从通用寄存器组B口读出，S3-S0为1001，M为1，ALU将寄存器A和寄存器B中的数据相加，如果和的低八位为0，则ZF等于1，否则为0，如果产生进位，则CF等于1，否则为0，同时输出相加的结果，F BUS为1、FL BUS和FR BUS为0，DL和XL都为0，移位逻辑将相加的和输出至总线BUS。

**[1.5s]**WE、LD IR、LD PC、IN PC都为0，RWBA1，RWBA0为00，时钟下降沿，将总线上的相加结果写入寄存器A，至此指令执行完成，同时SM变为0，准备取下一条指令。

**[1.5s]**NOT指令NOT B

**[1.5s]**其执行过程为：编码01对应的寄存器B中的数据从通用寄存器B口读出；在ALU中求反后经移位逻辑送入总线BUS；在WE控制下，时钟下降沿将BUS上求反的结果写入寄存器B。

**[1.5s]**SM等于1，执行NOT B，具体的执行由控制信号决定。

**[1.5s]**NOT B的指令编码是0101 0100，RWBA1、RWBA0为01，寄存器B中的数据从通用寄存器组B口读出，S3-S0为0101，M为1，ALU将输入端口B的数据取反后输出，F BUS为1、FL BUS和FR BUS为0，DL和XL都为0，移位逻辑将输入传至总线BUS，

**[1.5s]**WE、LD IR、LD PC、IN PC都为0，RWBA1，RWBA0为01，时钟下降沿，将总线上的数据写入寄存器B，至此指令执行完成，同时SM变为0，准备取下一条指令。

**[1.5s]**无条件跳转指令JMP 34H，其指令编码是00110000 00110100，其为双字节指令，取指周期取出了00110000存入指令寄存器IR，同时PC自加1，指向存放34H的地址单元。

**[1.5s]**其执行过程为：在DL信号控制下，时钟上升沿，从RAM中PC对应单元读出34H送至总线BUS，LD PC为1，时钟下降沿将总线BUS上的34H装载到PC中。

**[1.5s]**有条件跳转指令JZ 34H。

**[1.5s]**若ZF=1，则DL=1，LD PC=1，时钟上升沿从RAM中PC对应单元读出跳转地址34H，在时钟下降沿装载到PC中。若ZF=0，则IN PC=1，时钟下降沿PC自加1。

**[1.5s]**有条件跳转指令JC 34H

**[1.5s]**若CF=1，则DL=1，LD PC=1，时钟上升沿从RAM中PC对应单元读出跳转地址34H，在时钟下降沿装载到PC中。若CF=0，则IN PC=1，时钟下降沿PC自加1。

**[1.5s]**JMP与JZ、JC条件成立时的执行过程相同。JMP(JZ,JC) 34H的指令编码是0011 00CC 00110100，为双字节指令，存储在RAM的00H和01H两个存储单元中。

**[1.5s]**PC的值为0000 0001, MADD为00，PC的值经选择器传至RAM的地址端口。

**[1.5s]**DL为1，XL为0，F BUS、FL BUS和FR BUS都为0，时钟上升沿将01H单元中的0011 0100读取传至总线BUS。

**[1.5s]**WE、LD PC都为1，LD IR、IN PC为0，时钟下降沿，总线上的34H写入PC，至此指令执行完成。同时SM变为0，准备取下一条指令。

**[1.5s]**SM等于1，执行JZ(JC) 34H，此时条件不成立，DL，XL都为0，RAM不执行读写操作。

**[1.5s]**LD PC为0，IN PC为1，时钟下降沿，PC自加1，变为0000 0010，同时SM变为0，准备取下一条指令。

**[1.5s]**每条指令都需要两个周期完成，在控制信号的作用下，完成取指和执行操作。ADD A,B取指令时的控制信号为：SM为0，IN PC为1，LD PC为0, MADD为00, XL为0，DL为1, LD IR为1。

**[1.5s]**执行指令时的控制信号为：SM为1，RAA1 RAA0为01, RWBA1 RWBA0为00, WE为0，M为0，S3S2S1S0为1001，F BUS为1, FL BUS和FR BUS都为0。

**[1.5s]**控制信号产生逻辑中每个控制信号如何生成呢？前面分析了各条指令的执行过程，总结每个控制信号在各条指令取指和执行时所需的值，就能得到每个控制信号的逻辑表达式。以移位逻辑的F BUS为例说明，我们知道MOVA、MOVB、ADD、SUB、AND、NOT、OUT指令都需要将数据经移位逻辑直传至总线BUS，这时F BUS为1。

**[1.5s]**所以F BUS是MOVA、MOVB、ADD、SUB、AND0、NOT0和OUT0的逻辑和。