**temp4_r1_c8**

计算机组织与结构

实验报告

学号： 09017421 姓名： 胡世杰

学号： 09017423 姓名： 杨彬

东南大学计算机科学与工程学院

二019年3月

**实验一 寄存器组成实验**

**一、实验目的**

（1）熟悉D触发器的功能及使用方法。

（2）掌握寄存器组的逻辑组成及使用方法。

**二、实验内容**

（1）掌握Quartus II的使用方法，能够进行数字电路的设计及仿真。

（2）验证Quartus II所提供D触发器的功能及使用方法。

（3）设计具有1个读端口、1个写端口的寄存器组，并进行读写操作仿真/验证。

**三、实验原理及设计方案**

寄存器组将多个寄存器组织在一起，按地址进行操作，以简化控制复杂程度。

**本实验的寄存器组要求:**

1. 包含8个8位寄存器，具有1个读端口、1个写端口。操作地址均为3位，数据均为8位。
2. 读写端口分离，读操作的相关引脚有地址raddr[2..0]、数据输出q[7..0]，写操作的相关引脚有地址waddr[2..0]、数据输入data[7..0]、写使能wen。
3. 省略读使能信号可以简化控制，即数据输出不受限制。

寄存器组通过写地址waddr[2..0]、写使能wen信号来实现触发器的写入控制，通过读地址raddr[2..0]信号来控制触发器的数据输出选择。其组成原理如图2所示。



图2 寄存器组的组成

**设计方案:**

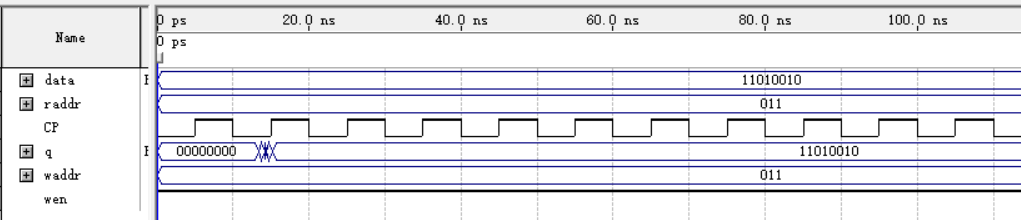
1. 使用8个8位D触发器lpm\_dff作为寄存器，一个参数化译码器lpm\_decode和一个参数化选择器lpm\_mux分别作为读端口和写端口，通过设置使操作地址为3位，数据为8位。
2. 数据输入以单独的INPUT进行输入，配合写端口的八位输出控制八个触发器的使能端控制数据的输入，读端口的八位输入连接八个触发器的输出以选择数据输出。

3、在参数设定时省略读使能信号。

设计电路如下：

#### 实验结果

功能仿真波形图如下：



#### **实验小结**

复习加强了D触发器的功能及使用方法，掌握了寄存器组的逻辑组成及使用方法，从理论到实践，加深了课本所学的知识，第一次使用参数化的元件，体会到Quartus II的方便实用。能将此次实验内容封装为模块，以便下次调用。

**实验二 运算器组成实验**

**一、实验目的**

（1）熟悉加/减法器的功能及使用方法。

（2）掌握算术逻辑部件(ALU)的功能及其逻辑组成。

（3）加深对运算器工作原理的理解。

**二、实验内容**

（1）验证Quartus II所提供加/减法器的功能及使用方法。

（2）设计具有加法、减法、逻辑与、逻辑非4种功能的ALU，并进行功能仿真/验证。

**三、实验原理及设计方案**

运算器是计算机硬件进行数据加工的重要部件，包括核心部件算术逻辑部件ALU，以及存放操作数和运算结果的寄存器/锁存器、存放运算结果状态的触发器等器件。根据给定控制信号的不同，运算器可以实现不同的运算功能。

**1、ALU的逻辑组成**

ALU能够实现多种算术运算、逻辑运算功能，其功能种类由指令系统决定。

ALU的外部引脚通常由两组数据入端A和B、一组数据出端F、一组功能选择信号SEL、几个结果状态信号组成，数据宽度通常有8位、16位等。由于F＝function(A, B)，故ALU又常称为函数功能发生器，其功能选择信号SEL决定了所实现的功能。

本实验所要求的ALU数据宽度为8位、具有4种算术及逻辑运算功能，其功能选择引脚记为SEL[1..0]。该ALU实现的具体功能如表1所示，其中，算术运算可以实现有符号数、无符号数的运算功能。

表1 ALU功能表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能选择 | | 实现功能 | | |
| SEL[1] SEL[0] | | 操作 | 助记符 | 功能函数 |
| 0 | 0 | 加法 | ADD | F＝A＋B |
| 0 | 1 | 减法 | SUB | F＝A－B |
| 1 | 0 | 逻辑与 | AND | F＝A·B |
| 1 | 1 | 逻辑非 | NOT | F＝ |

ALU的加法、减法功能可由Quartus II提供的lpm\_add\_sub模块实现，逻辑与、逻辑非功能可由Quartus II提供的lpm\_and、lpm\_inv模块实现。Quartus II中，lpm\_开头的模块均为参数化模块，使用时可以自定义其功能参数，如lpm\_add\_sub是一种参数化加/减法器，可以定义其实现功能、数据宽度、结果状态等。

ALU引脚与内部模块引脚之间的关联，可以通过门电路、数据选择器、编码/译码器等实现。如各功能模块的数据输出，与ALU数据出端F的连接，可通过多路数据选择器实现。相应地，Quartus II提供了参数化的数据选择器lpm\_mux、译码器lpm\_decode。

为了降低硬件实现成本，比较操作通常通过减法操作、逻辑操作的组合实现，例如，判断X＜Y可通过X－Y、判断结果是否有借位或是否为负数实现，判断X＝Y可通过X－Y、判断结果是否为零实现。由于减法与逻辑判断是2个操作，因此，运算器应提供算术运算结果的状态，如零标志、进位/借位标志、溢出标志、结果符号标志。

本实验所要求ALU的逻辑结构如图3所示，控制形成电路负责产生各功能模块的控制信号，本实验中与门、非门无需控制是特例；状态形成电路负责产生4个状态标志，分别是零标志ZF、进位/借位标志CF、溢出标志OF、结果符号标志SF。



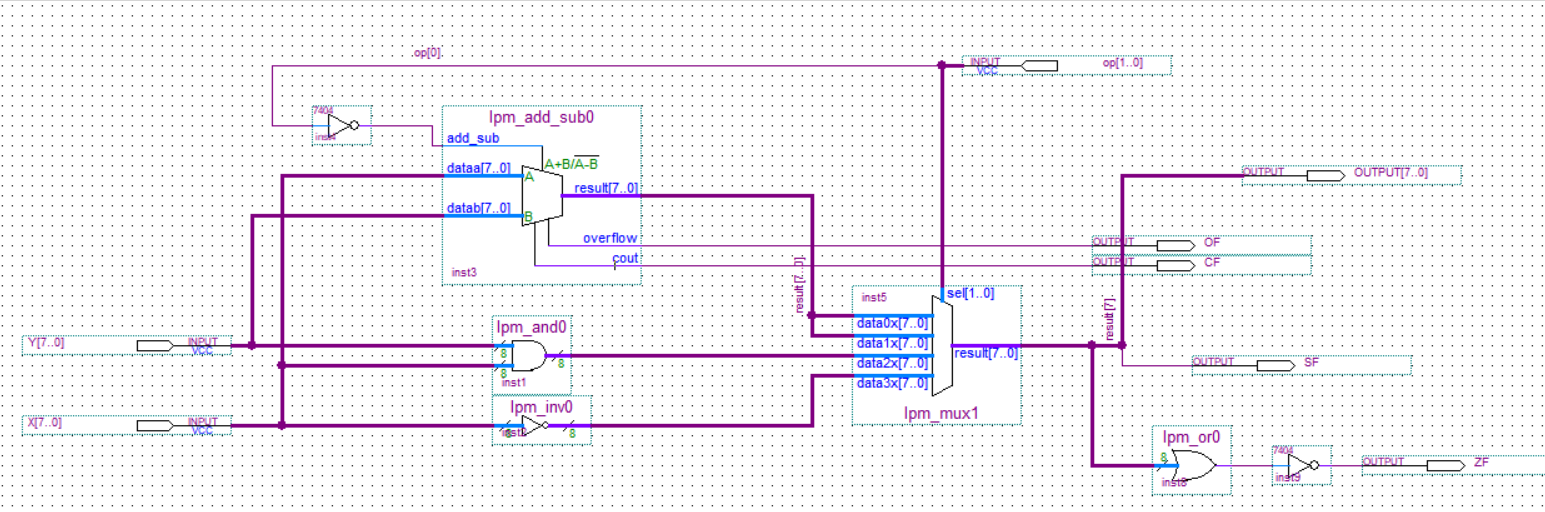
图3 ALU逻辑结构

**2、运算器的设计、实现**

本实验的运算器仅包含支持加法、减法、逻辑与、逻辑非4种功能的8位ALU，能够产生ZF、CF、OF、SF标志位，无需包含暂存结果的锁存器或寄存器组。

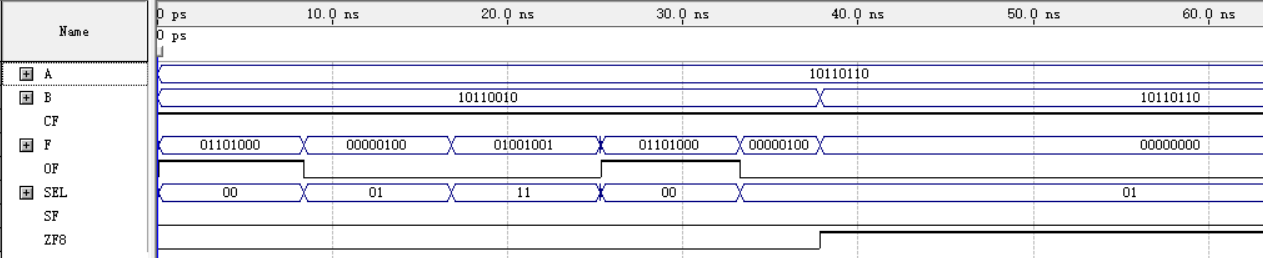
本实验中，运算器的实现要求通过Quartus II的原理图方法进行，实现过程包括：建立/打开工程文件、编辑原理图文件、编译原理图文件。注意，原理图的功能实现有多种方法，原理图文件编译前需先设置顶层文件。

本实验中，运算器的验证可以通过Quartus II仿真实现，或通过FPGA电路实现。基于Quartus II，仿真过程包括：编辑仿真波形文件、生成功能仿真网表、验证仿真波形正确性。通过FPGA电路实现仿真时，验证过程包括：分配电路的I/O管脚、下载电路到FPGA中、连接FPGA引脚到开关/指示灯、根据输入信号观察输出结果。注意，仿真/验证时，输入信号波形应能够反映电路功能的所有特征，如运算溢出、结果符号等特征。



#### 四、实验结果

功能仿真波形图如下：



#### **五、实验小结**

通过本实验，熟悉强化了利用数据选择器进行信号选择，以实现一个模块支持多种功能的方法，了解加深了运算器的基本功能以及基本的设计方法，理解了运算器各种运算具体实现方式以及各种运算的状态变量（比如SF、ZF、OF）的具体作用、含义和计算方法，值得一提的是，在实验过程中由于我们对OF的判断方法理解不够深入，老师提问的时候回答不上来，通过这个实验我们填补了知识上的漏洞，为之后的实验特别是cpu的设计打下了基础。

**实验三 存储器组成实验**

**一、实验目的**

（1）熟悉半导体存储器的存取方法。

（2）掌握存储器的扩展方法。

（3）掌握存储器与总线的连接方法。

**二、实验内容**

（1）验证Quartus II所提供半导体存储器的功能及使用方法。

（2）设计一个读/写端口分离的256×8bit的存储器，地址空间中前一半为只读、后一半为可读可写，并进行存取操作仿真/验证。

（3）将上述存储器连接到地址/数据复用的总线上，并进行存取操作仿真/验证。

**三、实验原理及设计方案**

**1、半导体存储器的功能及操作**

Quartus II提供了多种类型的半导体存储器模块，如lpm\_dq、lpm\_dp、lpm\_rom等，这些模块均均为同步存储器，可以自定义存储字长、存储字数。本实验中要求存储字长为8位即可。

以8位字长、256个存储单元、读/写端口分离的存储器lpm\_dq为例，其I/O引脚如图4所示。其中，data[7..0]、q[7..0]分别为数据输入、数据输出引脚，wren为写操作（写使能）信号、高电平有效，clock为同步操作的时钟脉冲信号。



图4 lpm\_dq引脚示例

对lpm\_dq的操作有读、写两种。写操作（wren=1）时，数据在clock上升沿锁存并写入到存储单元中，地址也由clock上升沿锁存。读操作（wren=0）时，地址由clock上升沿锁存，数据在clock上升沿后一定时间输出到引脚（功能仿真时延迟可忽略、实时仿真时延迟为可见）。

**本实验的存储器要求：**

**1、**设计读/写端口分离的256×8bit的存储器，地址空间中前一半为只读、后一半为可读可写。

**2、**存储器lpm\_dq连接的总线要求是地址/数据复用总线，即存储器的地址引脚、数据引脚连接到同一组总线上。

**3、**总线操作：同时只允许一个设备向总线发送信息，允许多个设备从总线接收信息。

**设计方案：**

**1、**采用ROM与RAM一起进行字扩展，两者地址空间相同，均为128个单元，地址引脚为address[6..0]，数据引脚为q[7..0]，前128个单元为ROM，后128个单元为RAM。

设扩展后存储器的地址引脚为A[7..0]、数据输出引脚为Q[7..0]，则A[7]用于选中ROM或RAM，即控制ROM或RAM的clock时钟信号产生、哪个q[7..0]连接到Q[7..0]。具体实现：将A[7]与时钟脉冲信号进行操作再连接到两个存储器的clock端。两个存储器输出分别接上三态门，将A[7]作为控制信号接入。两者均为A[7]=0时选中ROM，A[7]=1时选中RAM。

**2、**存储器lpm\_dq是读/写端口分离的，即数据输入引脚、数据输出引脚是不同的引脚。因此，本实验中存储器lpm\_dq的address[7..0]、data[7..0]、q[7..0]引脚同时连接到同一组总线bus[7..0]上。而lpm\_rom只有地址和数据输出引脚，故只需将address[7..0]、q[7..0]引脚同时连接到bus[7..0]上。

**3、**存储器lpm\_dq的地址信号、数据输入信号、数据输出信号须分时与总线交互。

因为存储器lpm\_dq与地址/数据复用总线连接，所以地址信号、数据输入信号的分时写入可通过增设地址锁存器实现，lpm\_rom与lpm\_dq的地址线与一同从锁存器获取，lpm\_dq的数据输入信号直接从总线获取。

输入信号、输出信号的分时传送可通过增设三态门实现；同样地，输入部件、存储器的分时传送可通过增设三态门实现。在两个存储器的输出经过三态门选择并合为一条总线后再用三态门选择输出。当选择输入时，输出端信号以及存储器输出总线三态门处置零，输入处置一。选择输出时，反之。

其连接电路原理如图所示。



存储器与地址/数据复用总线的连接

注：存储器由一个rom和一个ram组成

存储器与地址/数据复用总线连接的原理图设计中，输入部件可用输入引脚实现，锁存器可采用lpm\_latch模块、三态门可采用lpm\_bustri模块。

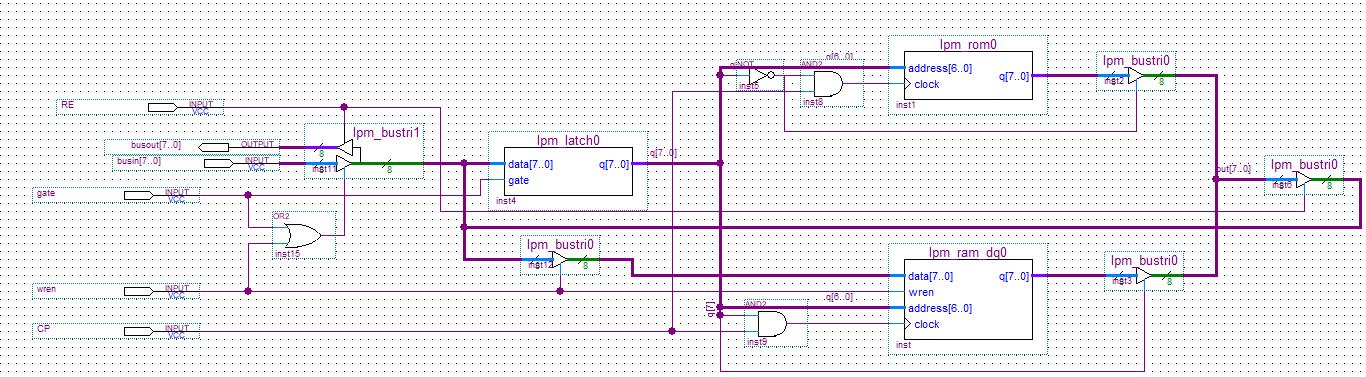
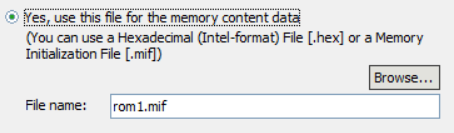
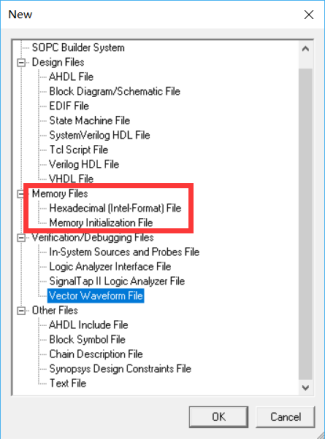
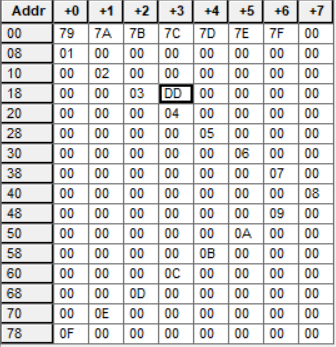
具体实现电路如下：

图1、此外，ROM无法输入数据，故需在开始参数化时导入数据（ram也可以导入）

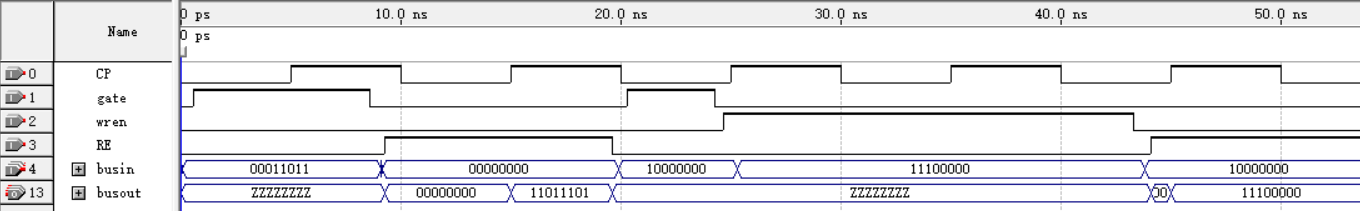
图2、.mif和.hex文件可以在quartus里创建

图3、本实验导入的文件如图

1. 2. 3.

#### 四、实验结果

功能仿真波形图如下：



先输入地址，（再输入数据），再读取数据。

#### **五、实验小结**

通过本实验，我们了解了Quartus的rom和ram的区别以及使用方法，了解了rom和ram的初始化方法，同时我们了解了利用三态门来进行输出控制，防止输出冲突，更加深入地理解了采取总线方案的时候，同时只能有一个输出。学会了利用锁存器来暂时保存数据（这里是保存地址）来防止数据冲突，进一步地强化了总线的连接方式和使用方法，为进一步的实验（CPU通路设计）打下了基础。

**实验四 CPU 数据通路实验**

**一、实验目的**

（1）掌握CPU数据通路的逻辑组成。

（2）了解指令功能的实现过程及其控制方法。

**二、实验内容**

（1）设计一个单总线结构的CPU数据通路，部件包括4种功能的8位ALU、4×8位的寄存器组、256×8位的RAM、8位计数器各一个。

（2）给出相关部件控制信号，分别实现取数、加法、条件转移指令的功能。

**三、实验原理及设计方案**

1、CPU数据通路的设计

本实验要求设计单总线结构数据通路，数据宽度为8位，包括ALU、寄存器组、RAM、计数器4个功能部件，各部件的功能分别是：

ALU：支持加法、减法、逻辑与、逻辑非4种操作（同实验二）

RAM：用作Cache，地址空间为8位、读/写端口分离（同实验三）

计数器：用作程序计数器（记为PC），具有置数、计数功能。

寄存器组：用作通用寄存器组（记为REGs），由4个具有独立读端口、写端口的寄存器组成（同实验一）

本实验的数据通路可以采用如图所示的方案



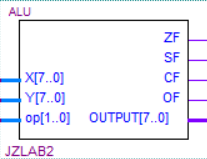
数据通路的逻辑组成

图中，ALU的操作控制信号线(2根)为op[2..0]，REGs的写地址信号线(2根)、读地址信号线(2根)分别为WA[2..0]、RA[2..0]，REGs、RAM的写操作控制信号线分别为r\_wr、m\_wr，计数器的置数控制信号线、时钟信号线分别为ldn、cp。

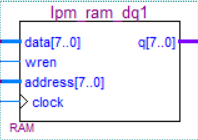
计数器的数据引脚有入端D、出端Q，控制引脚有置数LDN（低电平有效)、时钟CP、同步/异步清零CLRN（低电平有效)等。计数器的真值表如下：CLRN=0时，Q=0（清零）；CLRN=1、LDN=0、CP上升沿时，Q=D（置数）；CLRN=1、LDN=1、CP上升沿时，Q=Q＋1（计数）；其余情况Q保持不变。

根据之前的实验，我们可以对其进行封装来得到上述所需功能部件

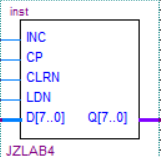
ALU：直接将实验二进行封装



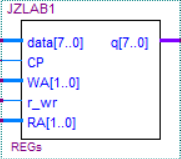
RAM：因本实验的要求和实验三不同，故直接采取使用自带的部件lpm\_ram\_dq



计数器：对lpm\_counter进行简单封装后达到上述要求

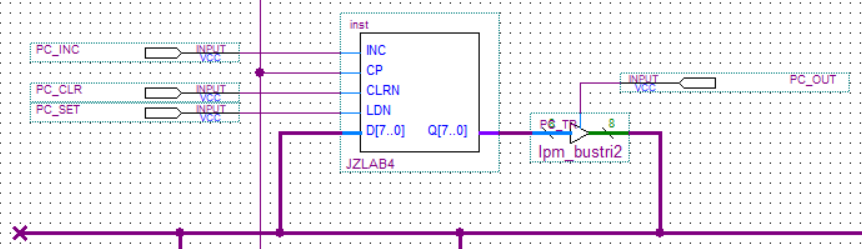


寄存器组：将实验中的八个寄存器改为四个后进行封装

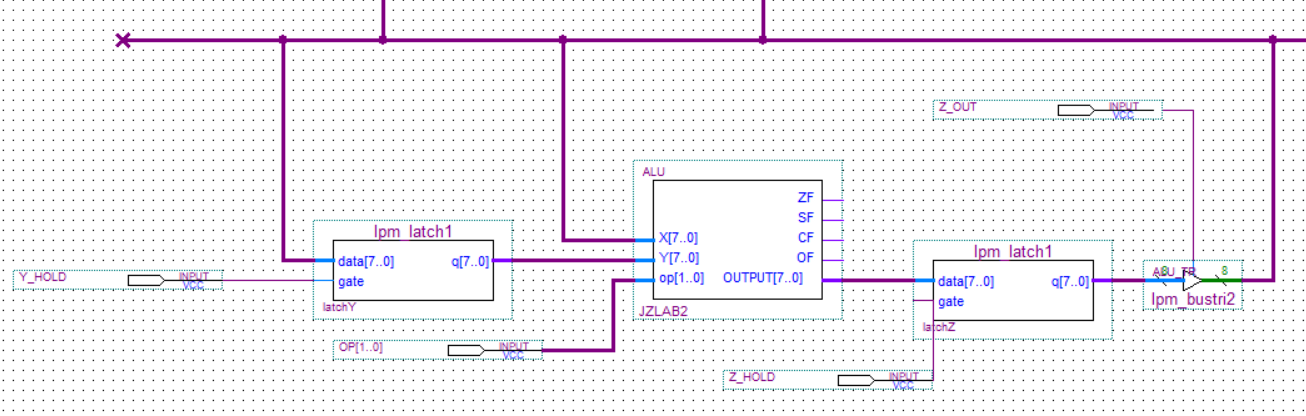


按图中所示结构与总线进行连接：

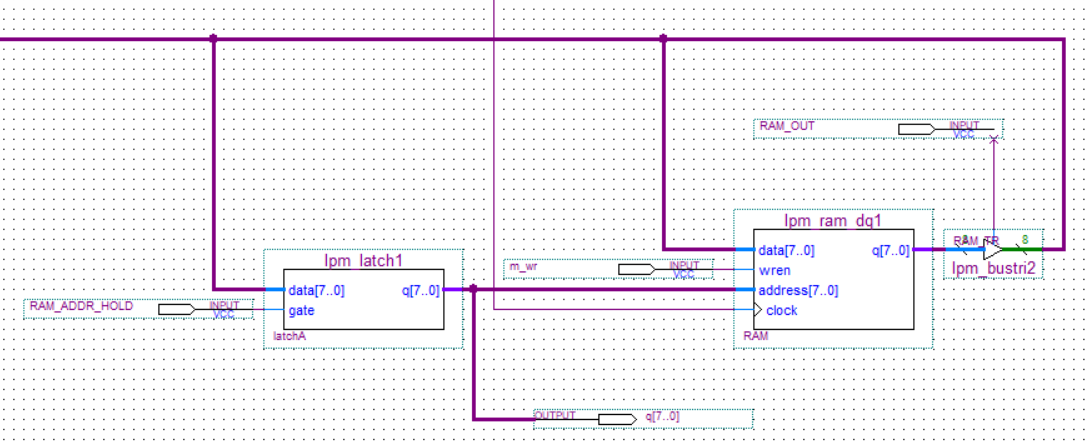
程序计数器PC：



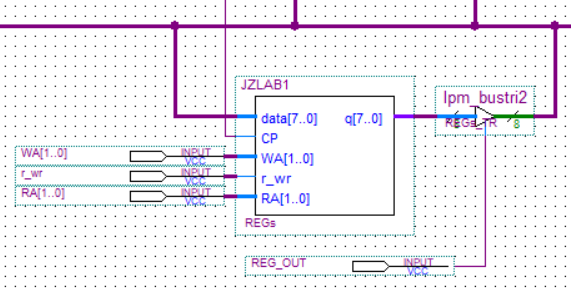
ALU：



DRAM：



寄存器组：



各控制信号如下：



2、指令功能的实现

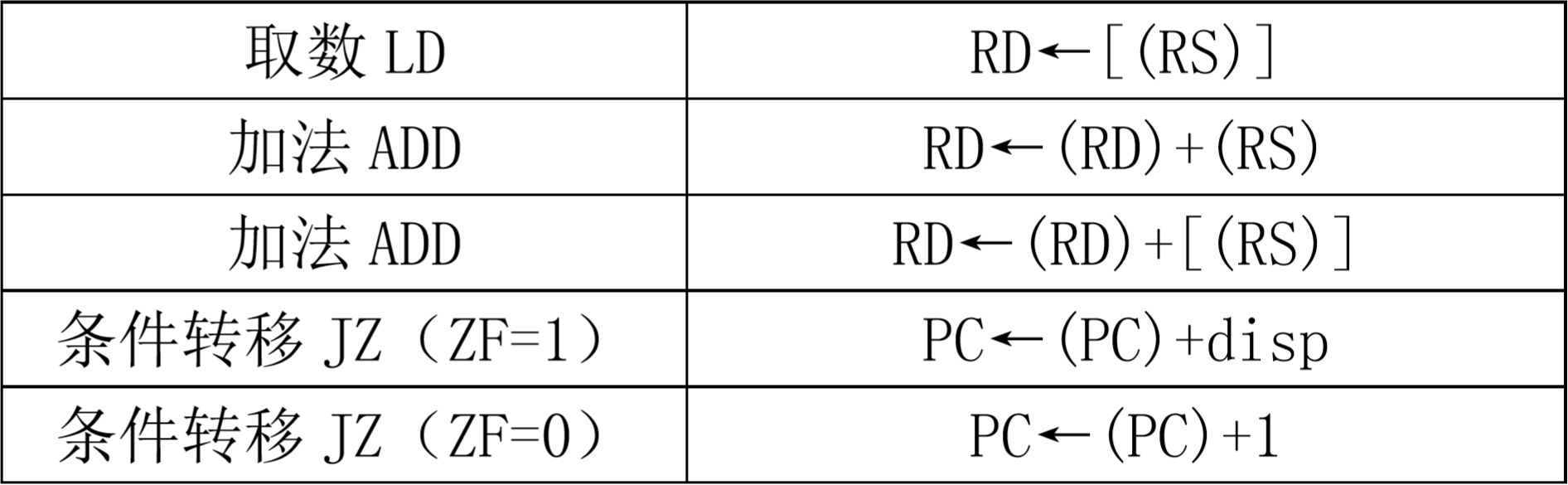
本实验要求的是实现取数、加法、条件转移指令的功能，各指令功能如下：

取数LD -- RD←M[(RS)]

加法ADD -- RD←(RD)＋(RS)及RD←(RD)＋M[(RS)]

条件转移JZ -- ZF=1时，PC←(PC)＋disp；ZF=0时，PC←(PC)＋1

即：



根据每种指令的执行过程，分别设计微操作序列如下：

（注：对于控制信号，不加标记代表置高有效，加下划线代表置低无效。 ）

取数 LD ：RD←[(RS)]

1. RS→RA，REG\_OUT，RAM\_ADDR\_HOLD，RAM\_ADDR\_HOLD，REG\_OUT
2. RD→WA，r\_wr，RAM\_OUT
3. RAM\_OUT，结束

加法 ADD：RD←(RD)+[(RS)]

1. RD→RA，REG\_OUT，Y\_HOLD，REG\_OUT，Y\_HOLD
2. RS→RA，REG\_OUT，RAM\_ADDR\_HOLD，REG\_OUT
3. RAM\_OUT，ADD→OP，Z\_HOLD，RAM\_OUT，RAM\_ADDR\_HOLD，Z\_HOLD
4. RD→WA，r\_wr，Z\_OUT
5. Z\_OUT，结束

加法 ADD：RD←(RD)+ (RS)

1. RD→RA，REG\_OUT，Y\_HOLD，REG\_OUT
2. RS→RA，REG\_OUT，ADD→OP，Z\_HOLD，REG\_OUT，Y\_HOLD，Z\_HOLD
3. RD→WA，r\_wr，Z\_OUT
4. Z\_OUT，结束

条件转移 JZ（ZF=1）： PC←(PC)+disp

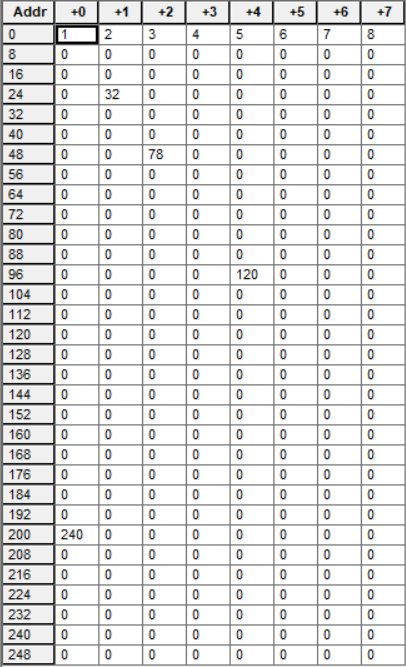
1. PC\_OUT，Y\_HOLD，PC\_OUT
2. BUS\_IN，ADD→OP，Z\_HOLD，BUS\_IN，Y\_HOLD，Z\_HOLD
3. Z\_OUT，PC\_SET，Z\_OUT，结束

条件转移 JZ（ZF=0）： PC←(PC)+1

1. PC\_INS

据此，即可设计波形并仿真

#### 四、实验结果

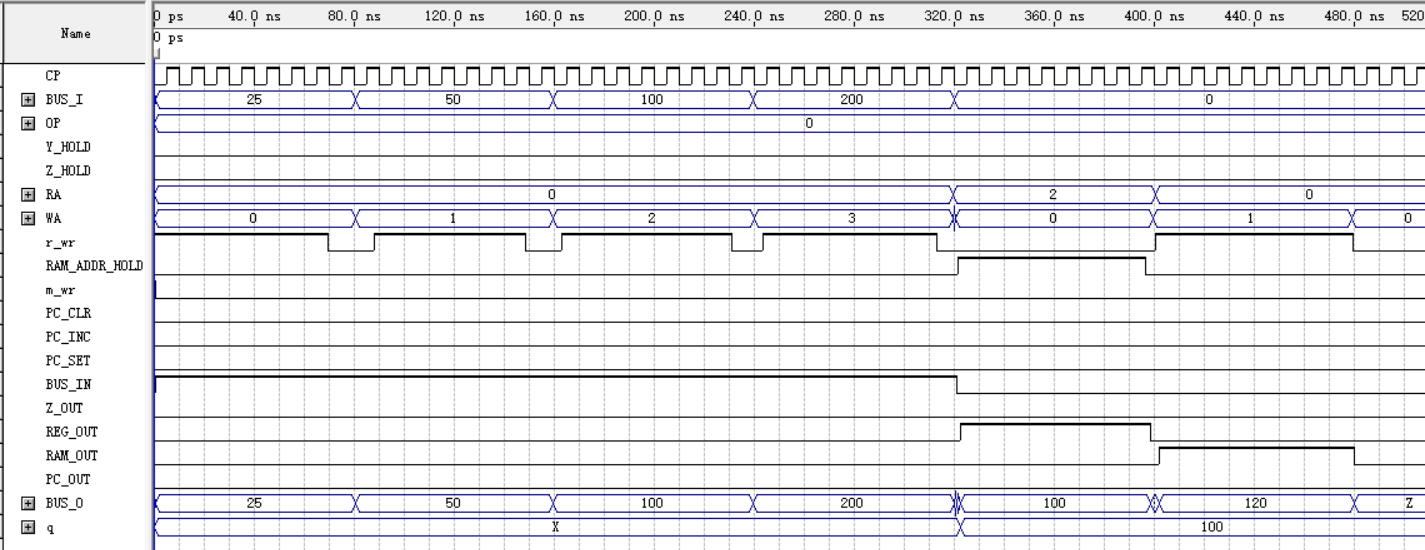


|  |  |
| --- | --- |
| DRAM地址 | DRAM数值 |
| 50 | 78 |
| 100 | 120 |

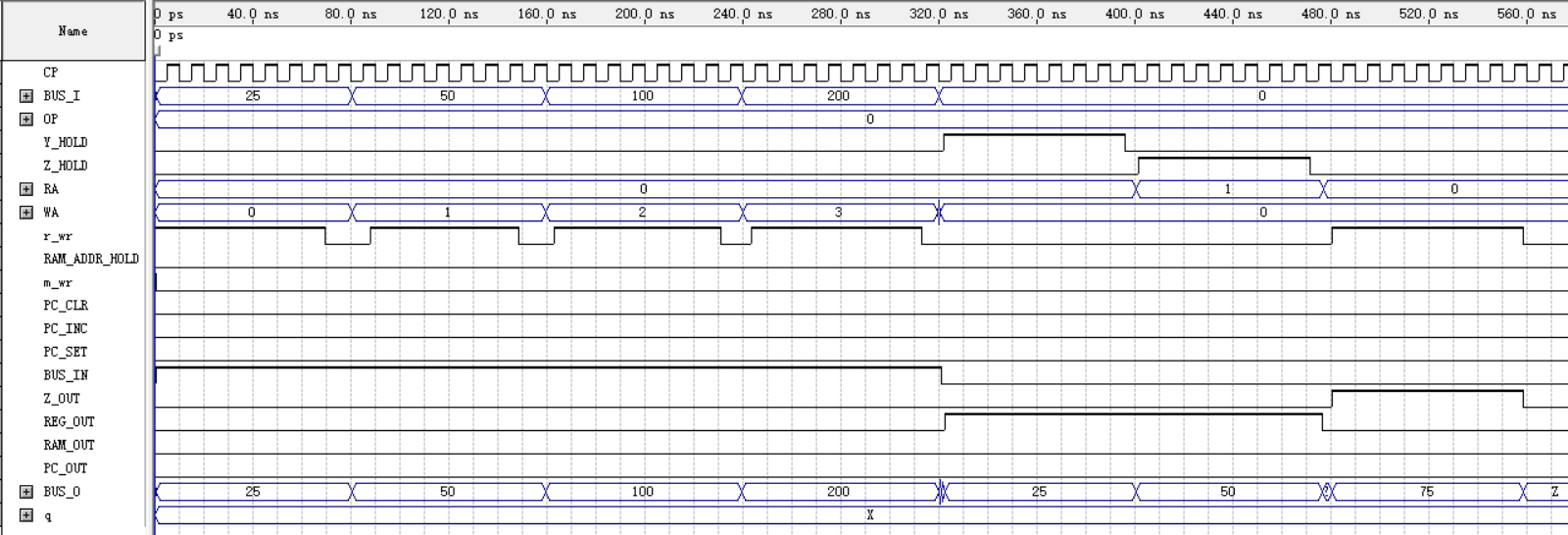
|  |  |
| --- | --- |
| 寄存器地址 | 寄存器数值 |
| 0H | 25 |
| 1H | 50 |
| 2H | 100 |

波形图如下：

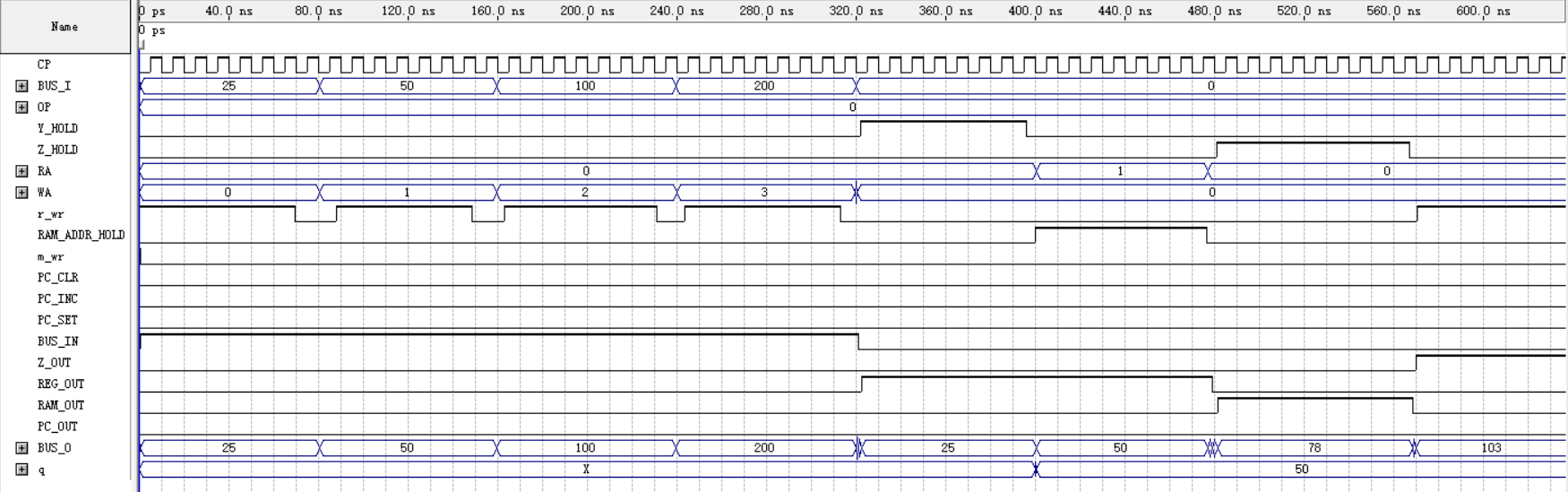
4.1 取数指令 RD←[(RS)]

设 RS =2H，RD=1H，执行结果应该是(RD)=120。 

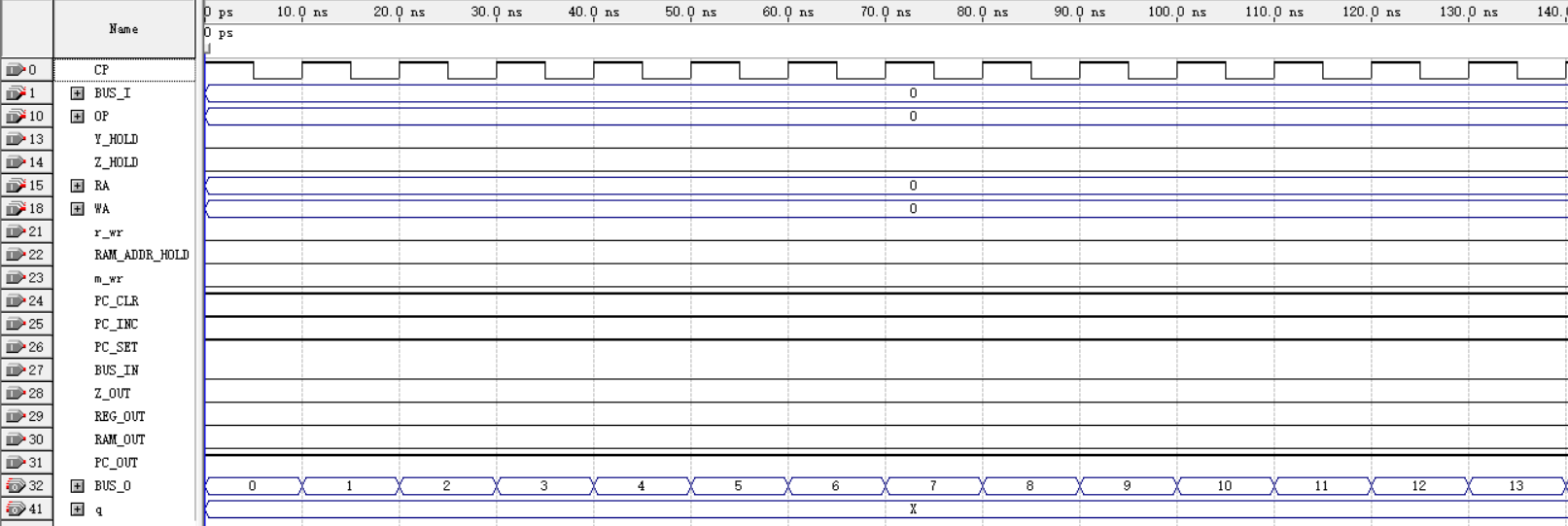
4.2 加法指令RD←(RD)+ (RS)

设 RS =0H，1D=3H，执行结果应该是(RD)=25+50=75。 

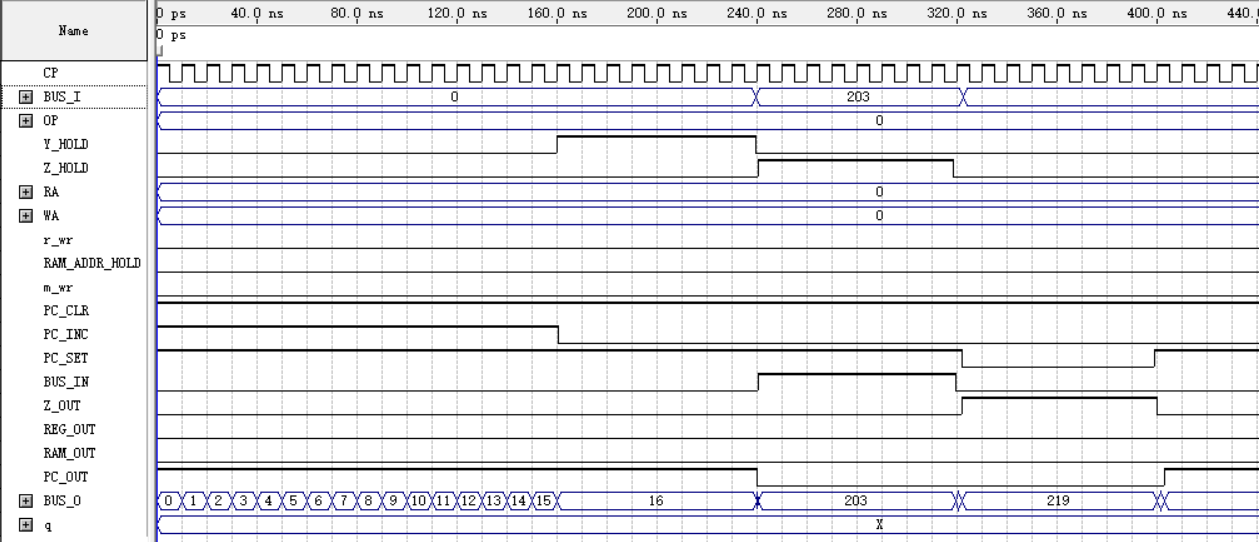
4.3 加法指令RD←(RD)+[(RS)]

设 RS =1H，RD=0H，执行结果应该是(RD)=25+78=103。 

4.4条件转移（ZF=0）： PC←(PC)+1

PC 值随机器运行自动增加。

4.5 条件转移（ZF=1）： PC←(PC)+disp

设 PC =16，disp=203，执行结果应该是 PC=16+203=219。 

#### **五、实验小结**

通过本个综合实验，我们了解了数据通路基本的设计方法，本实验数据通路的设计采用总线结构，更深入地理解了总线的思想，更熟练地利用三态门来控制数据的输出，综合运用前面三次实验的结果完成cpu数据通路的设计，更深入地理解了模块化的设计思想；通过给各个三态门、各个部件（比如运算器、寄存器组、存储器）的添加控制信号，更深入地理解了微命令和微指令的含义。通过仿真模拟，在设计的数据通路上，实现多条机器指令，手动输入微指令，更加深入地理解了计算机的指令系统，深入地理解了给机器指令添加微命令和微指令。