**ALU、译码器、编码器、立即数扩展的设计与综合验证（20分）**

**说明：**

**1．时间安排：第二章课后-第四章课后**

**2．安装、熟悉Logisim平台，对指定电路进行测试**

**3．完成四种运算的32位ALU设计与验证**

**4．立即数扩展电路的设计与验证**

**5．7条MIPS指令的译码电路设计与验证**

**6．相关选择信号的编码器设计与验证**

**7．ALU、译码器、编码器、立即数扩展的综合与验证**

**8．考查理论课中相关知识点的掌握情况**

**9．完成Word报告（注意图表的规范性，含实验小结）**

**一、基于Logisim平台的电路测试（2分）**

**1．在Logisim中打开给定的project.circ**

**2．然后，点击“大作业逻辑图”，查看电路图（此时你可能不理解电路结构），将Shared Label内容修改为与你的学号相关**

**3．然后，点击“应用测试”，看到基于逻辑符号（应有你的学号信息）的电路设计，截图。**

**4．对该电路进行测试，填写下表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **当SEL=00时** | **当SEL=01时** | **当SEL=10时** | **当SEL=11时** |  |
| **输入** | **输出** | **输出** | **输出** | **输出** | **输出** |
| **A B C D** | **OUT3-0** | **OUT3-0** | **OUT3-0** | **OUT3-0** | **FLAG** |
| **0 0 0 0** | **0 0 1 1** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **0** |
| **0 0 0 1** | **0 1 0 0** | **0 0 0 1** | **0 0 0 1** | **0 0 0 0** | **0** |
| **0 0 1 0** | **0 1 0 1** | **0 0 1 0** | **0 0 1 1** | **0 0 0 0** | **0** |
| **0 0 1 1** | **0 1 1 0** | **0 0 1 1** | **0 0 1 0** | **0 0 0 0** | **0** |
| **0 1 0 0** | **0 1 1 1** | **0 1 0 0** | **0 1 1 0** | **0 0 0 0** | **0** |
| **0 1 0 1** | **1 0 0 0** | **1 0 1 1** | **0 1 1 1** | **0 0 0 0** | **0** |
| **0 1 1 0** | **1 0 0 1** | **1 1 0 0** | **0 1 0 1** | **0 0 0 0** | **0** |
| **0 1 1 1** | **1 0 1 0** | **1 1 0 1** | **0 1 0 0** | **0 0 0 0** | **0** |
| **1 0 0 0** | **1 0 1 1** | **1 1 1 0** | **1 1 0 0** | **0 0 0 0** | **0** |
| **1 0 0 1** | **1 1 0 0** | **1 1 1 1** | **1 0 0 0** | **0 0 0 0** | **0** |
| **1 0 1 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **1** |
| **1 0 1 1** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **1** |
| **1 1 0 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **1** |
| **1 1 0 1** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **1** |
| **1 1 1 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **1** |
| **1 1 1 1** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **0 0 0 0** | **1** |

**5．分析电路实现的逻辑功能**

**（1）当IN的值小于等于1001，即输入未超界时：**

**FLAG的值为0；**

**当SEL=00时，out3-0的输出值为IN的余三码；**

**当SEL=01时，out3-0的输出值为IN的2421码；**

**当SEL=10时，out3-0的输出值为IN的格雷BCD码；**

**当SEL=11时，out3-0的输出值均为0000。**

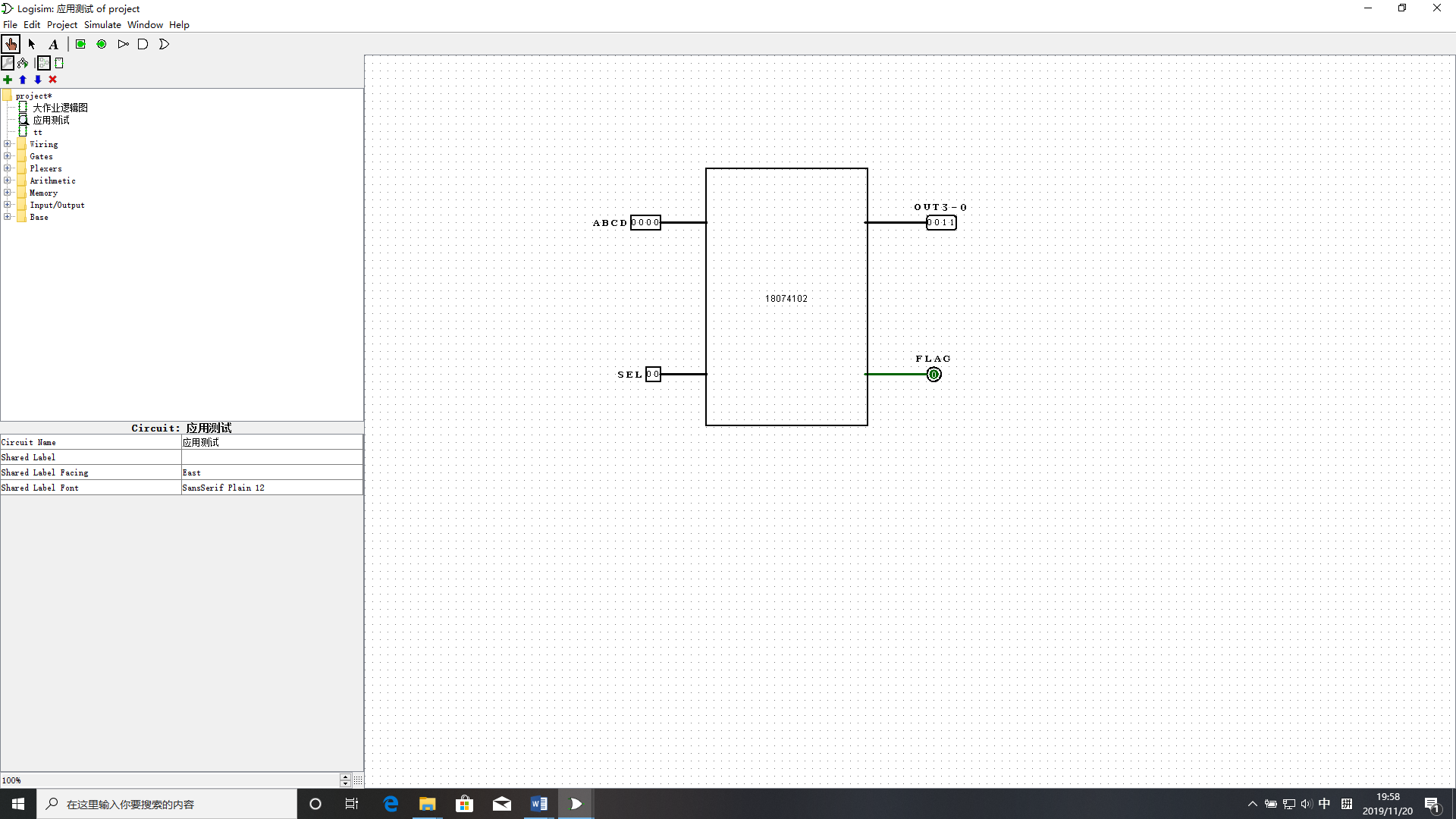
**（2）当IN的值大于1001时：**

**FLAG的值为1，且无论输入何值，out3-0的输出值均为0000。**

**6．说说你对“大作业逻辑图”中电路结构的理解**

**SEL是数据选择器的选择端，控制输入值经历哪一路的计算进行输出。当SEL=00时，IN经加法器加3转换为余三码；当SEL=01时，IN经另一加法器，加法器的另一加数由数据选择器决定，其控制端经一或门连接译码器的5至9端口，当数据为0至4时，控制端为0，加法器加0；当数据为5至9时，控制端为1，加法器加6，转换为2421码；当SEL=10时，out3-0的输出值经过一系列异或运算，根据公式 ,输出IN的格雷BCD码；当SEL=11时，输出值为0000。当输入数超界后，即IN的值大于1001时，译码器的10至15端口经或门输出FLAG为1，同时控制数据选择器不再进行输出。**

**7．截取word报告用图**



二、**设计能完成四种运算的32位ALU（5分）**

**1．利用Logisim中的基本门元件（与门/或门/非门/异或门…），搭建1位全加器电路，并封装电路，编辑逻辑符号（学号+add1）;利用自己封装的1位全加器，搭建32位全加器电路，并封装电路，编辑逻辑符号（学号+add32）；**

**2. 利用上面封装过的32位全加器以及Logisim中的基本门元件搭建32位全减器，并封装电路，编辑逻辑符号（学号+sub32）**

**2.利用上面封装过的32位全法器、全减器以及Logisim中的与门、或门以及选择器，构造能完成四种运算的ALU，功能表如下：**

|  |  |  |
| --- | --- | --- |
| **ALU\_sel** | **ALU\_out** | **说明** |
| **00** | **A+B** | **加法** |
| **01** | **A-B** | **减法** |
| **10** | **A&B** | **按位与** |
| **11** | **A|B** | **按位或** |

**2．添加运算结果为零的标志信号zero**

**3．算术运算（补码）测试：若(A)10 = -5，(B)10 = 5，填表**

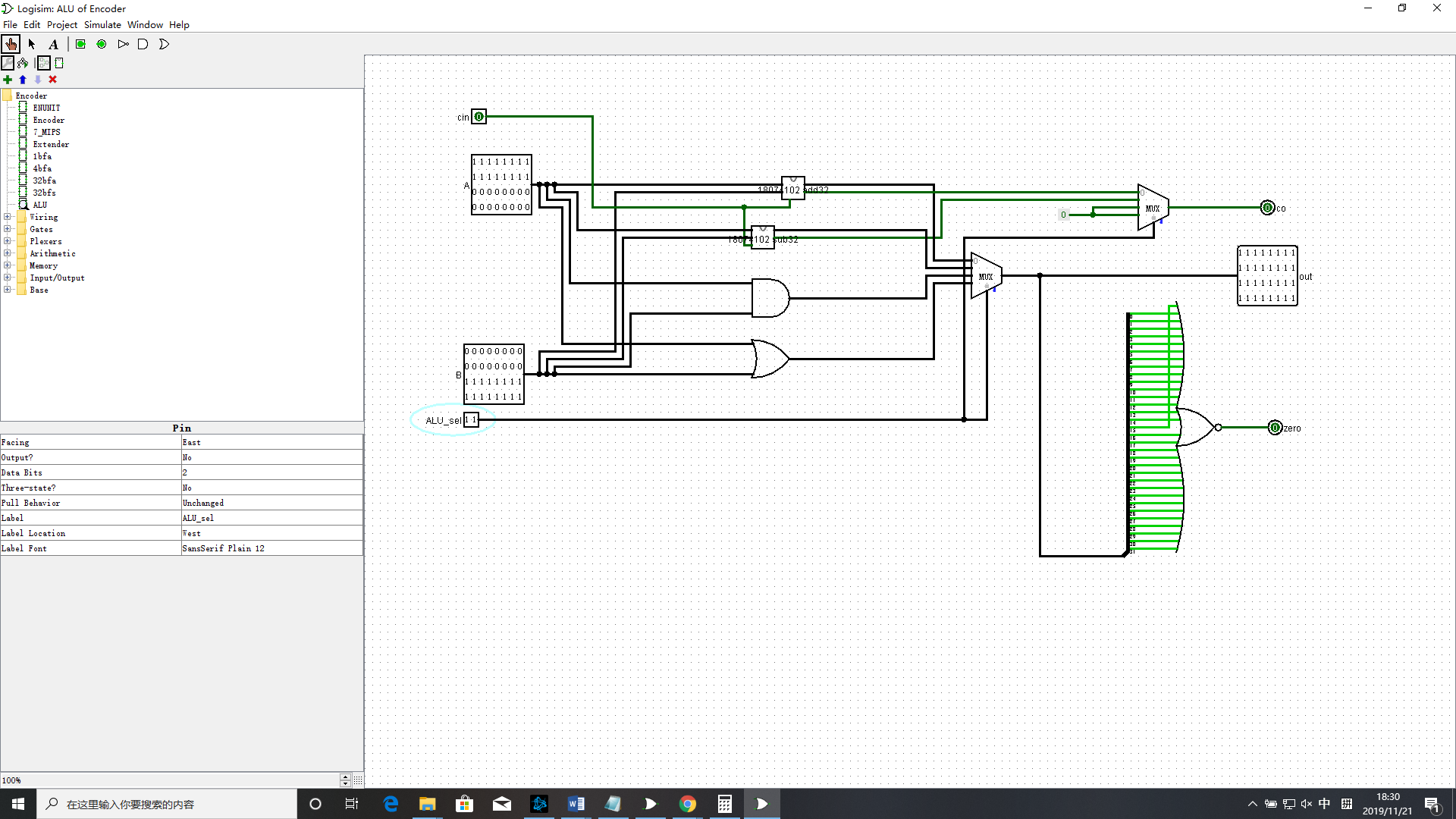
|  |  |  |  |
| --- | --- | --- | --- |
| **ALU\_sel** | **ALU\_out** | **（ALU\_out）10** | **zero** |
| **00** | **0x00000000** | **0** | **1** |
| **01** | **0xFFFFFFFF6** | **-10** | **0** |

**4．逻辑运算测试：若(A)16=FFFF0000，(B)16=0000FFFF，填表**

|  |  |
| --- | --- |
| **ALU\_sel** | **ALU\_out** |
| **10** | **0x00000000** |
| **11** | **0xFFFFFFFFF** |

**5．编辑逻辑符号（ALU-学号），备用**

**6．截取word报告用图**

****

**7．将这个ALU的Verilog HDL描述补充完整（模块名含学号）**

**module ALU\_18074102 (A,B,ALU\_sel,ALU\_out,zero);**

**input [ 31:0 ] A,B;**

**input [ 1:0 ] ALU\_sel;**

**output [ 31:0 ] ALU\_out;**

**output zero;**

**reg [ 31:0 ] ALU\_out ;**

**assign zero= ~|ALU\_out ;**

**always @( A or B or ALU\_sel )**

**case (ALU\_sel)**

**0: ALU\_out=A+B;**

**1: ALU\_out=A-B;**

**2: ALU\_out=A&B;**

**3: ALU\_out=A|B;**

**default: ALU\_out=32’b0;**

**endcase**

**endmodule**

三、**立即数扩展电路的设计与验证（3分）**

**1．按照要求将输入的16位数扩展成32位输出**

**2．功能表**

|  |  |  |  |
| --- | --- | --- | --- |
| **信号名** | **位宽** | **方向** | **说明** |
| **Imm16** | **16** | **输入** | **来自指令寄存器的16位立即数** |
| **Extsel** | **2** | **输入** | **00：无符号扩展，将16位立即数进行0扩展至32位立即数；**  **01：符号扩展，将16位补码立即数扩展成32位补码立即数；**  **10：低位0扩展，将16位立即数移至32位立即数的高16位，低16位补0。** |
| **ExtImm32** | **32** | **输出** | **扩展后的32位立即数** |

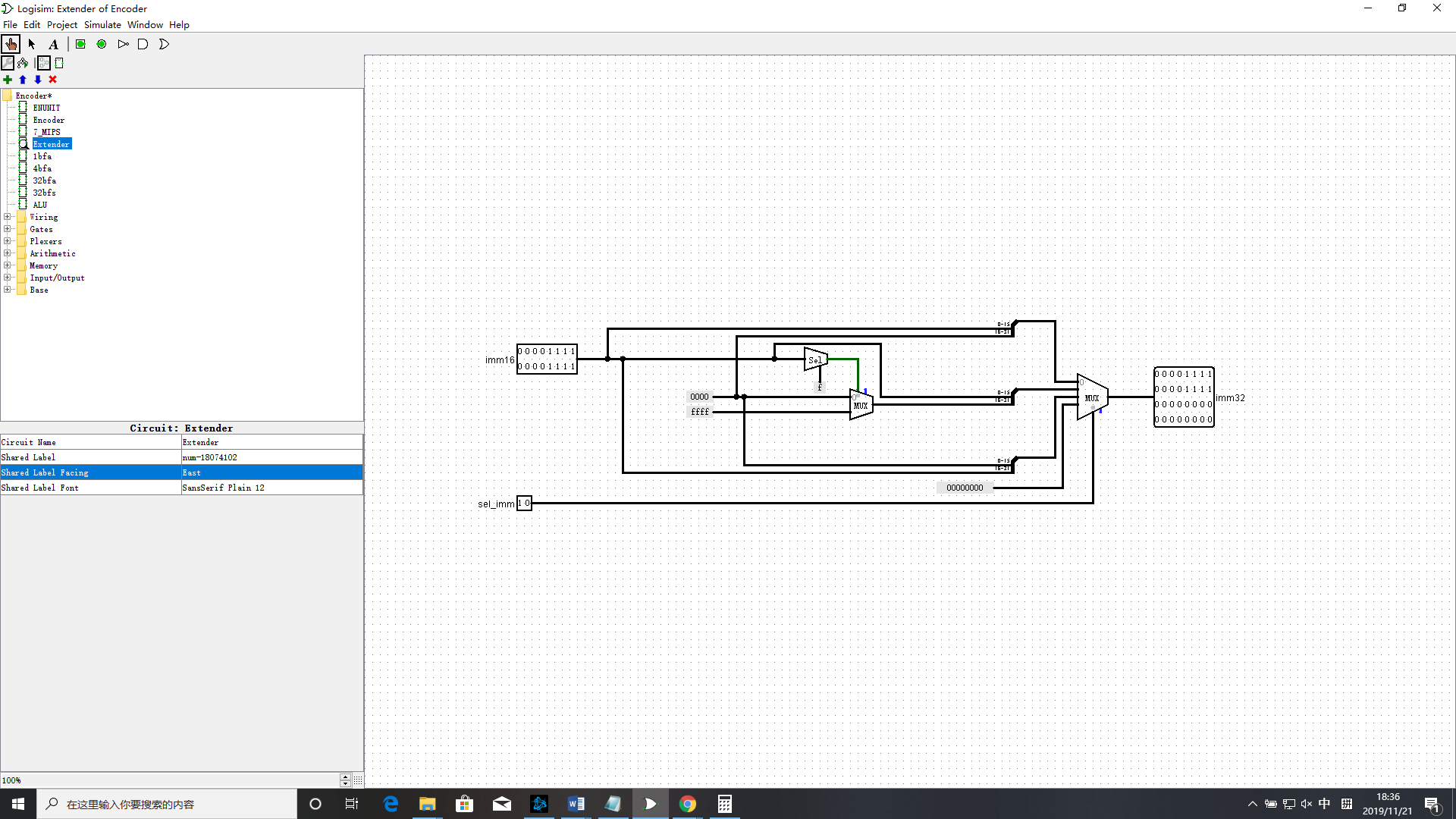
**3．利用Logisim中的“分线器”和“选择器”，构造立即数扩展电路**

**4．模拟验证，填表**

|  |  |  |
| --- | --- | --- |
| **Imm16** | **Extsel** | **ExtImm32** |
| **1111000011110000** | **00** | **00000000000000001111000011110000** |
| **0000000000001111** | **01** | **00000000000000000000000000001111** |
| **1111111111110111** | **01** | **11111111111111111111111111110111** |
| **0000111100001111** | **10** | **00001111000011110000000000000000** |

**5．编辑逻辑符号（num-学号），备用**

**6．截取word报告用图**



**四、7条MIPS指令的译码电路设计与验证（4分）**

**1．7条指令说明见相关ppt，根据7条MIPS指令的特征码“opcode”和“funct”，设计指令译码器**

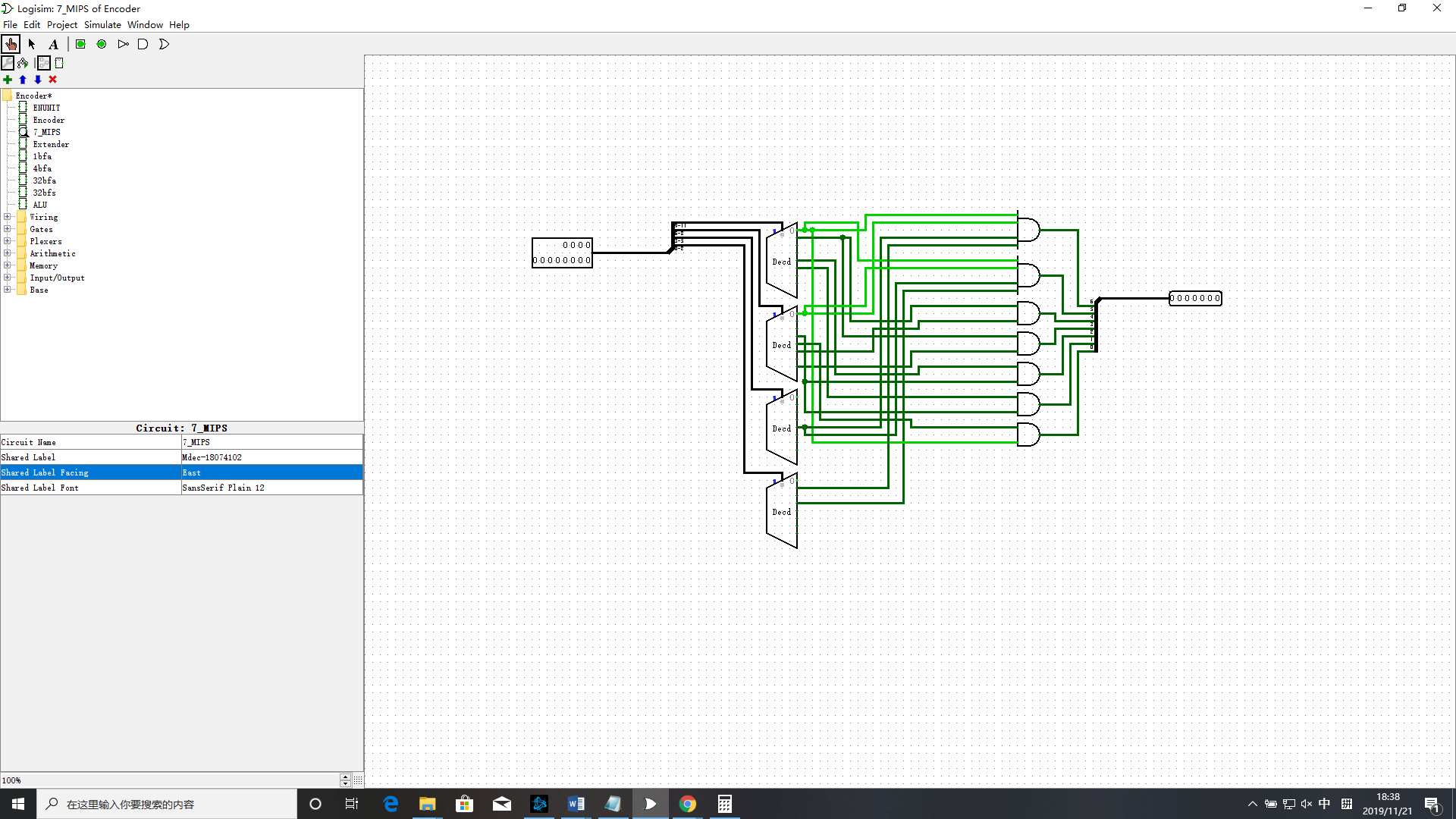
**2．指令译码器功能表**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **输入** | | **输出** | | | | | | |
|  |  | **加法** | **减法** | **或立即数** | **立即数置高位** | **取字** | **存字** | **相等跳转** |
| **opcode** | **funct** | **addu** | **subu** | **ori** | **lui** | **lw** | **sw** | **beq** |
| **000000** | **100001** | **1** | **0** | **0** | **0** | **0** | **0** | **0** |
| **000000** | **100011** | **0** | **1** | **0** | **0** | **0** | **0** | **0** |
| **001101** | **x** | **0** | **0** | **1** | **0** | **0** | **0** | **0** |
| **001111** | **x** | **0** | **0** | **0** | **1** | **0** | **0** | **0** |
| **100011** | **x** | **0** | **0** | **0** | **0** | **1** | **0** | **0** |
| **101011** | **x** | **0** | **0** | **0** | **0** | **0** | **1** | **0** |
| **000100** | **x** | **0** | **0** | **0** | **0** | **0** | **0** | **1** |

**3．在Logisim中设计实现指令译码器，并进行验证**

**4．编辑逻辑符号（Mdec-学号），备用**

**5．截取word报告用图**

****

**五、相关选择信号的编码器设计与验证 （3分）**

**1．当指令被译码后，为完成指令功能，应对数据通路中的选择信号进行编码。例如：addu和subu指令完成A、B两个数的运算；ori指令完成A与立即数的“按位或”逻辑运算；lui指令完成高位“置”有符号立即数；lw和sw指令要通过A+有符号立即数获得数据存放地址；beq指令通过判断A-B的结果是否为零，建立PCsel信号。见图1。**

指令译码

编码器

ALU

立即数扩展

选择器

opcode

funct

Imm16

ExtImm32

A

B

选择控制信号ALU\_sel、B\_sel、Ext\_sel、PC\_sel

zero

zero

ALU\_out

ALU\_sel

B\_sel

Ext\_sel

**图1 简单数据通路示意图**

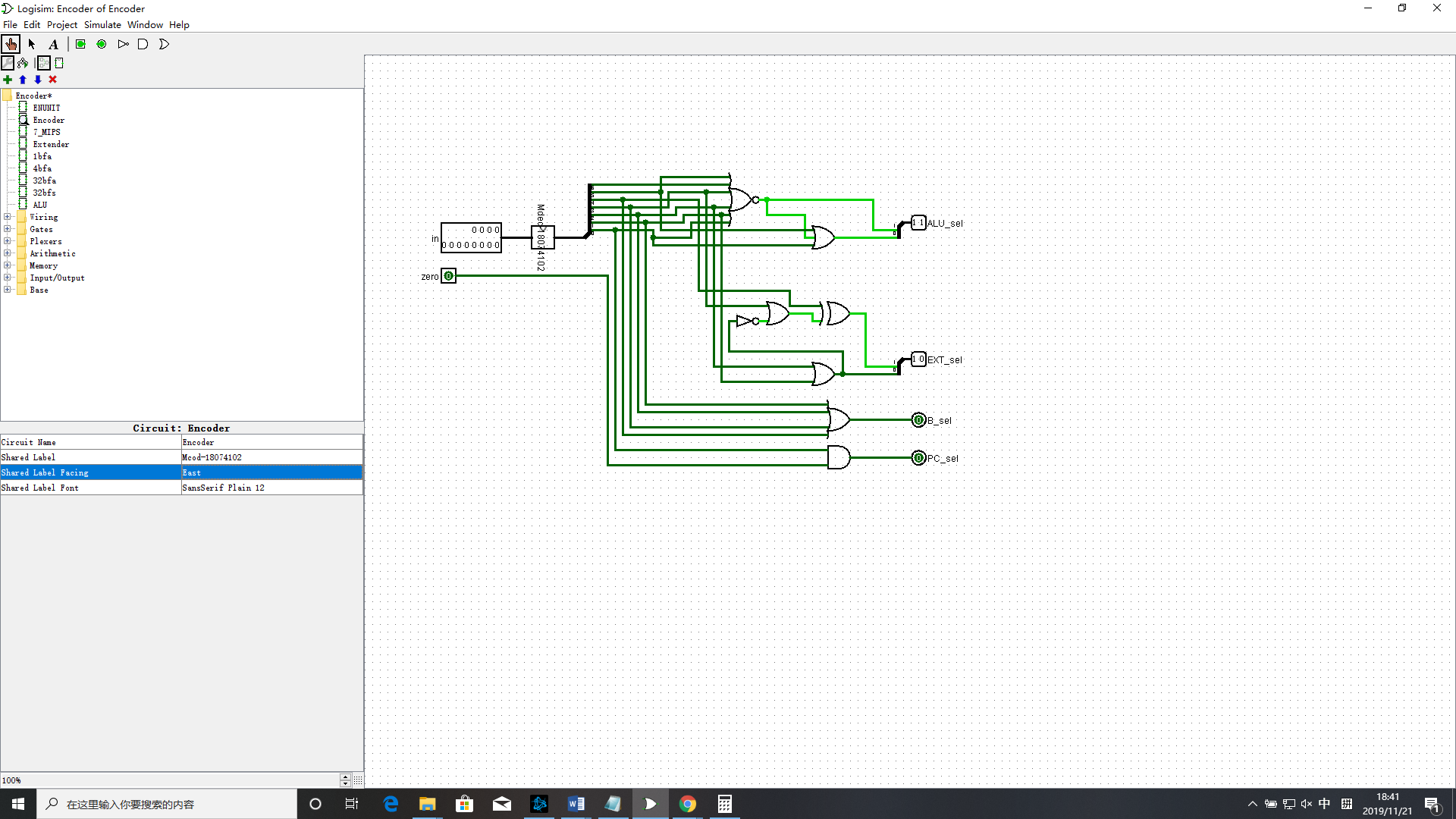
**2．编码器功能表（对照图1，理解编码含义）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **指令** | **ALU\_sel [1:0]** | **EXT\_sel [1:0]** | **B\_sel** | **PC\_sel** |
| **addu** | **00** | xx | 0 | **0** |
| **subu** | **01** | xx | 0 | **0** |
| **ori** | **11** | 00 | 1 | **0** |
| **lui** | **00** | 10 | 1 | **0** |
| **lw** | **00** | 01 | 1 | **0** |
| **sw** | **00** | 01 | 1 | **0** |
| **beq** | **01** | xx | 0 | **1**  **( 当zero=1时 )** |

**3．在Logisim中设计实现编码器，并进行验证**

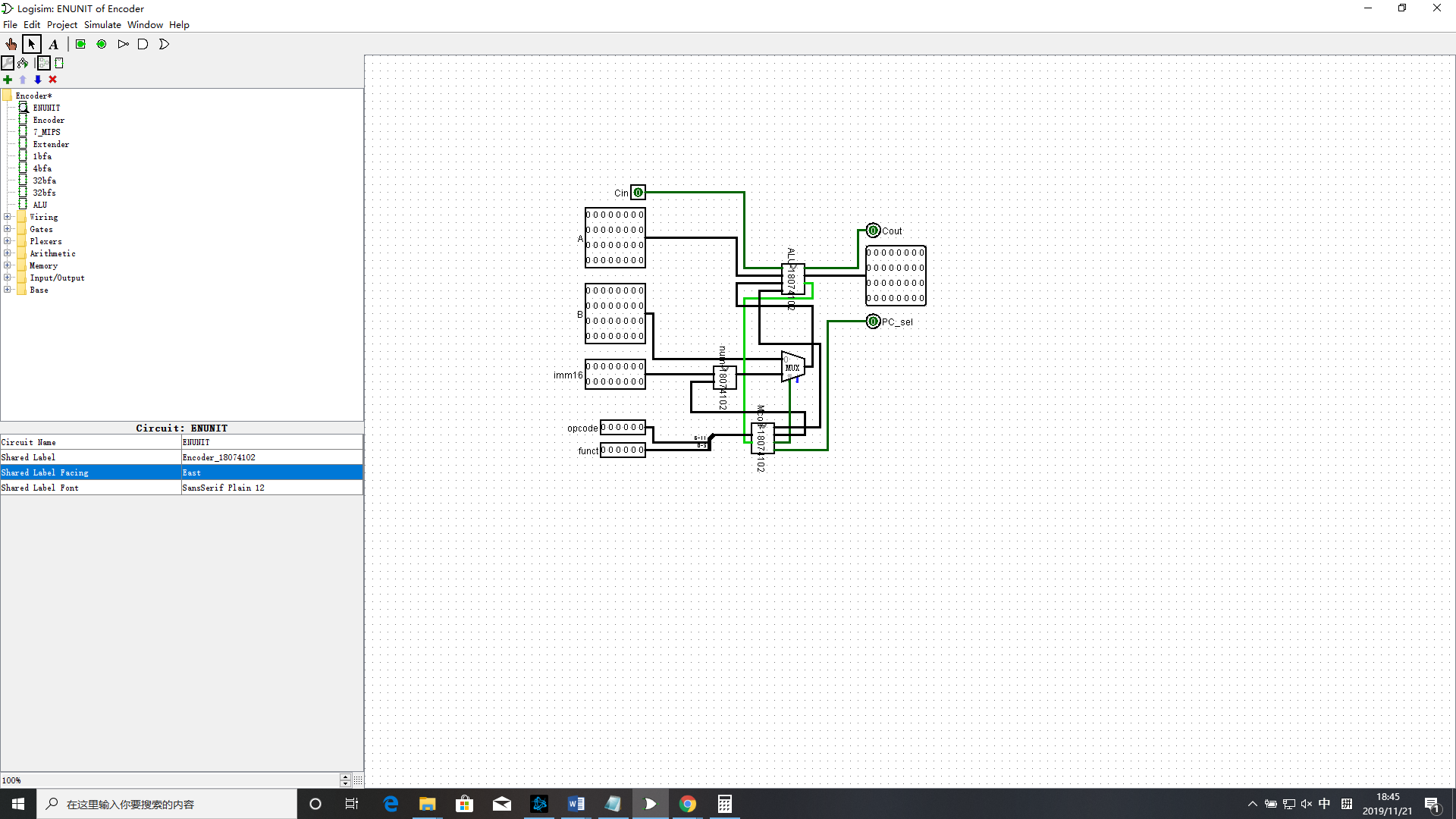
**4．编辑逻辑符号（Mcod-学号），备用**

**5．截取word报告用图**

****

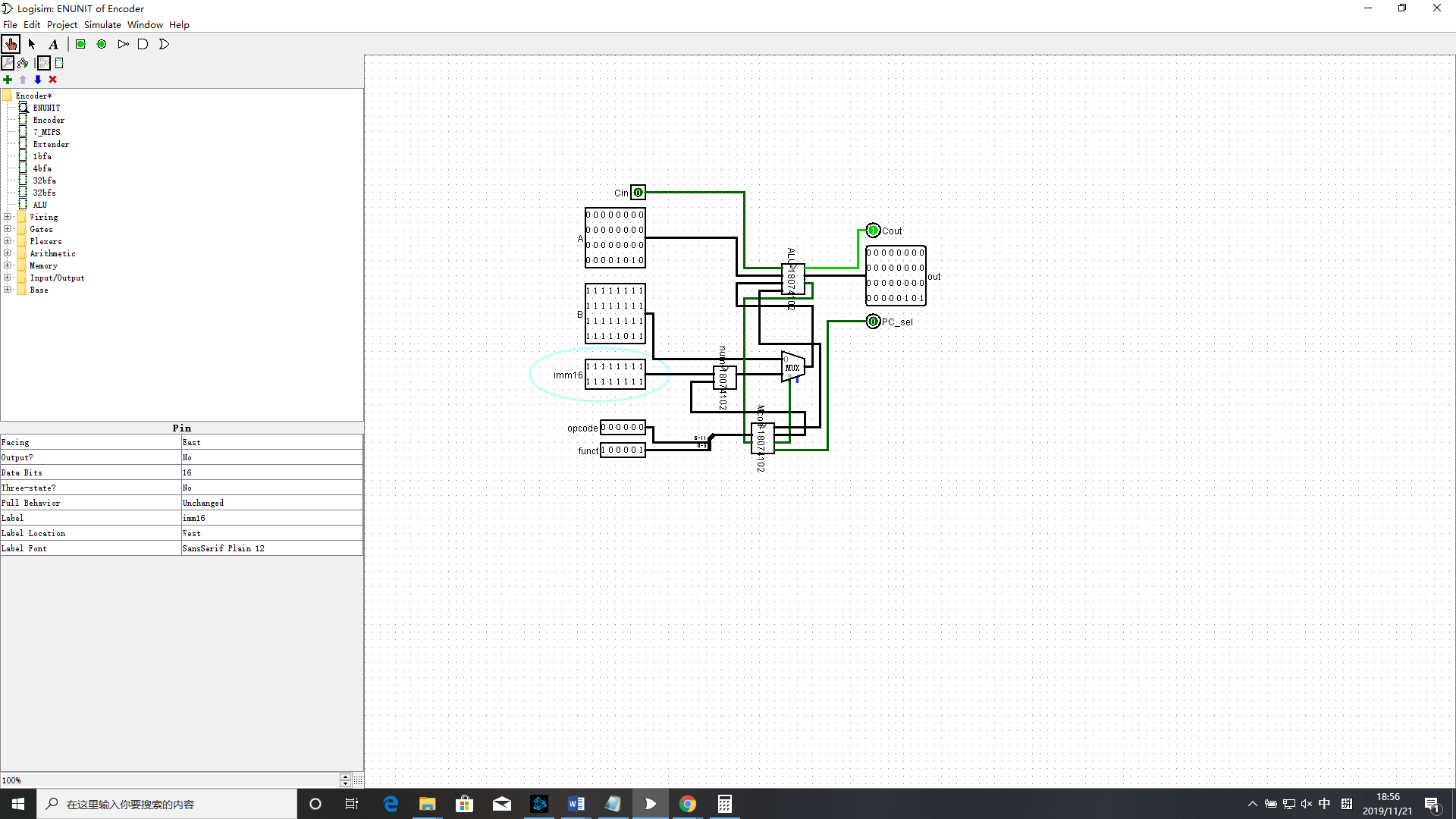
**六、综合验证（2分）**

**1．在Logisim中，按照图1进行电路（逻辑符号）连接**

****

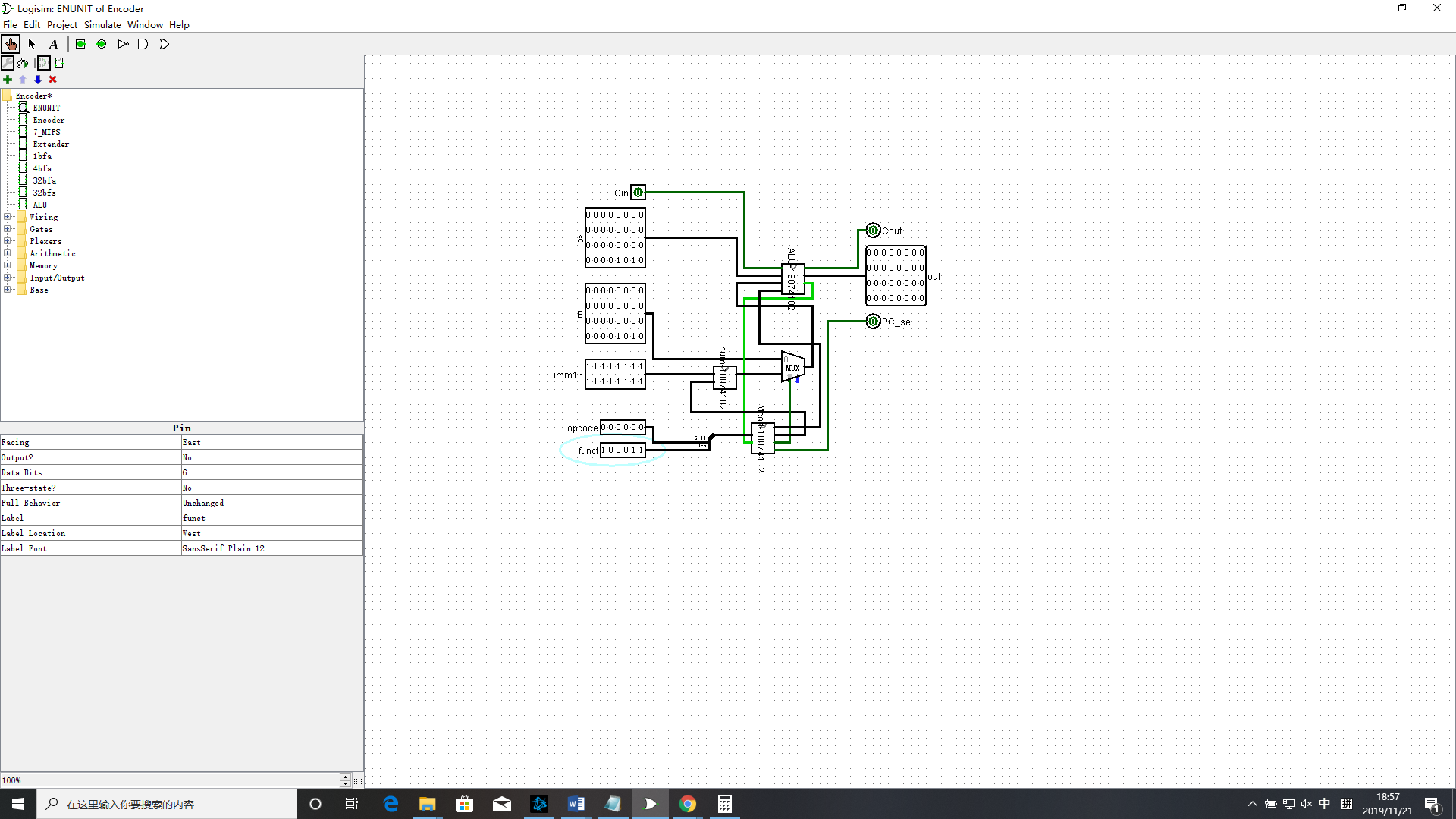
**2．按给定数据（注：十进制）进行测试**

**当A=10、B= -5、Imm16=-1时，完成addu测试，截取测试结果并说明；**

****

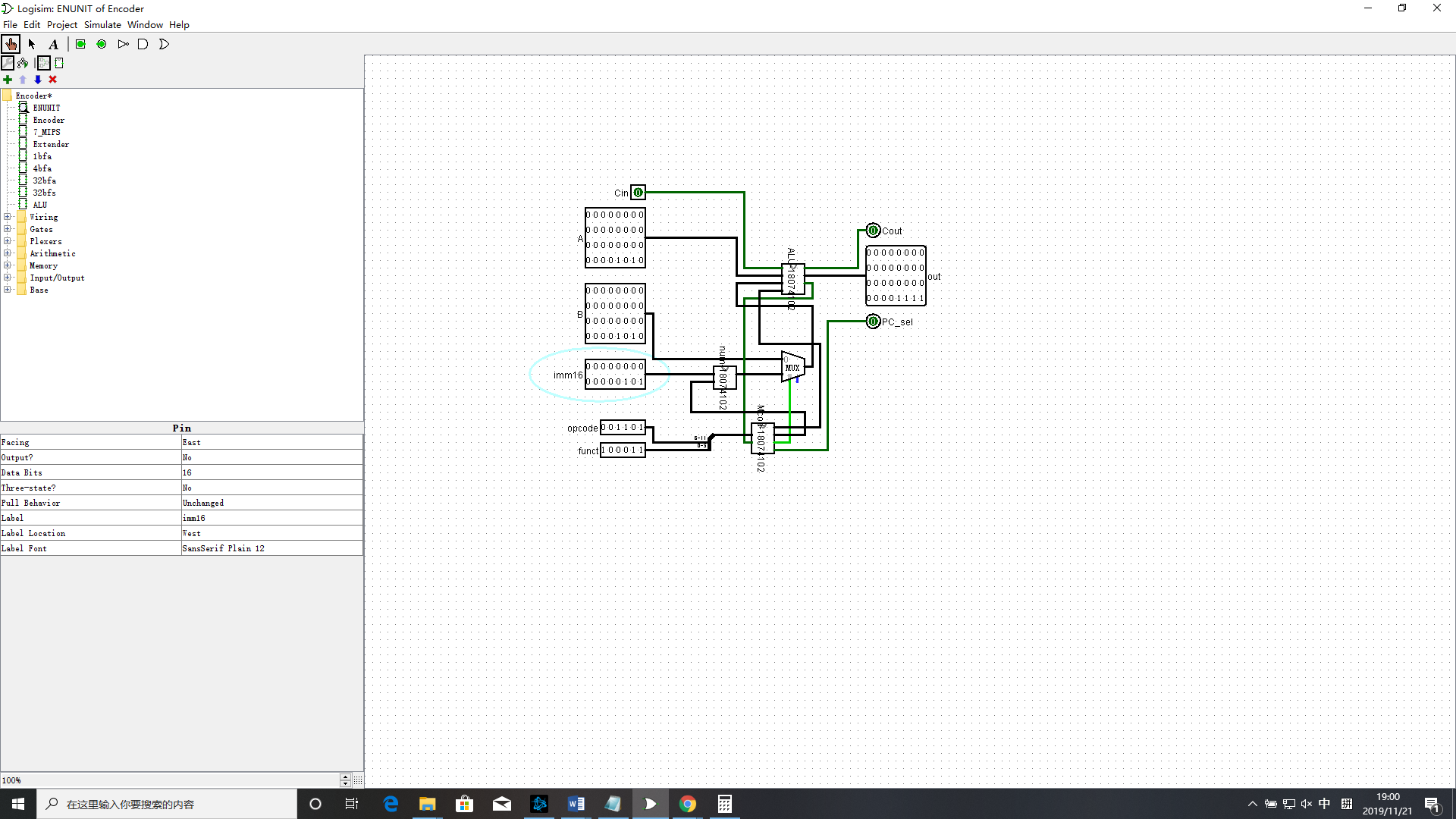
**addu测试，当funct为100001且opcode为000000时，实现addu功能，即两个32位二进制数做全加运算，imm16不影响计算结果，out为0x00000005，PC\_sel为0。**

**当A=10、B=10、Imm16=-1时，完成subu测试，截取测试结果并说明；**

****

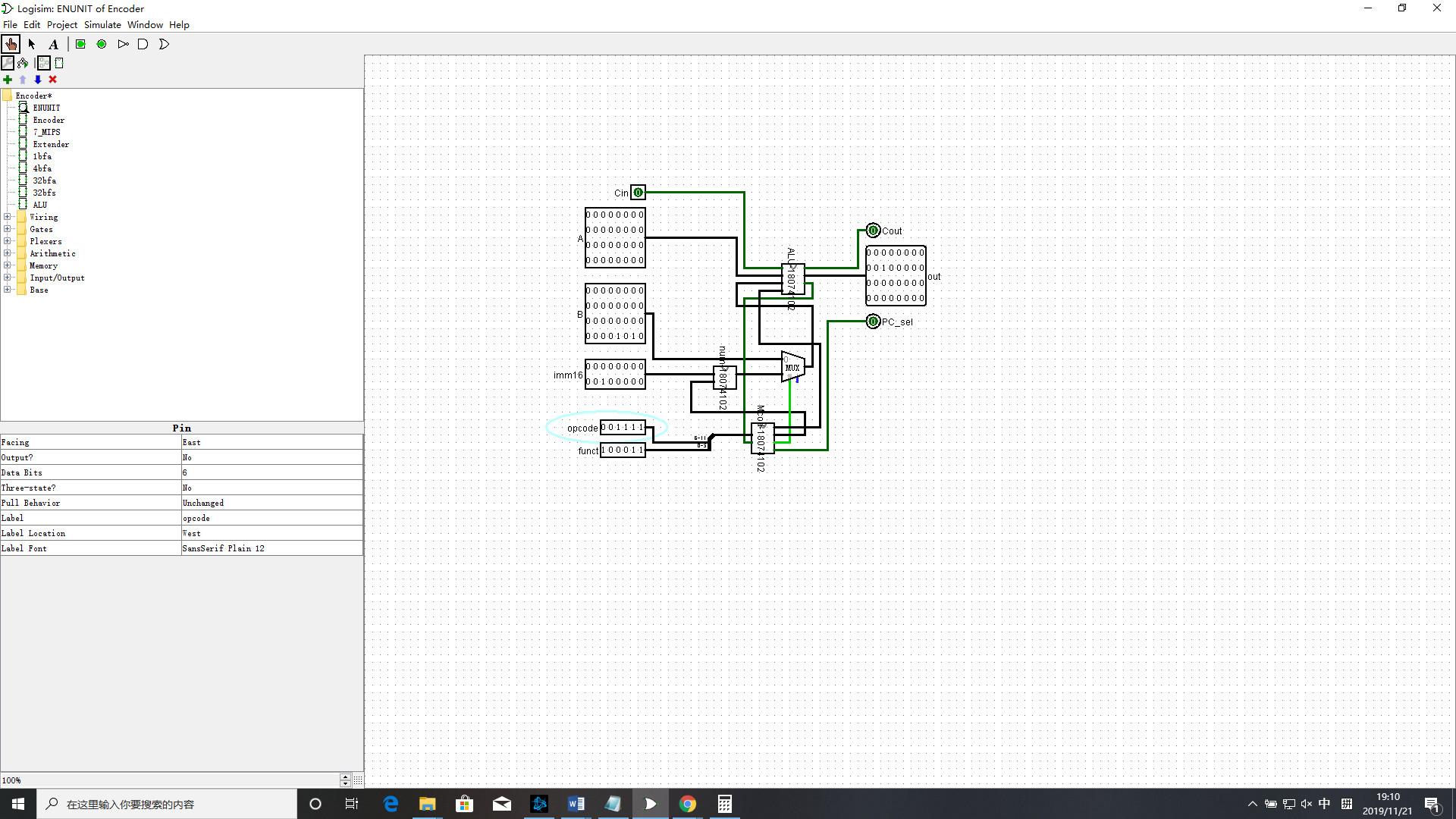
**subu测试，当funct为100011且opcode为000000时，实现subu功能，即两个32位二进制数做全减运算，imm16不影响计算结果，out为0x00000000，PC\_sel为0。**

**当A=10、B=10，Imm16=5时，完成ori测试，截取测试结果并说明；**

****

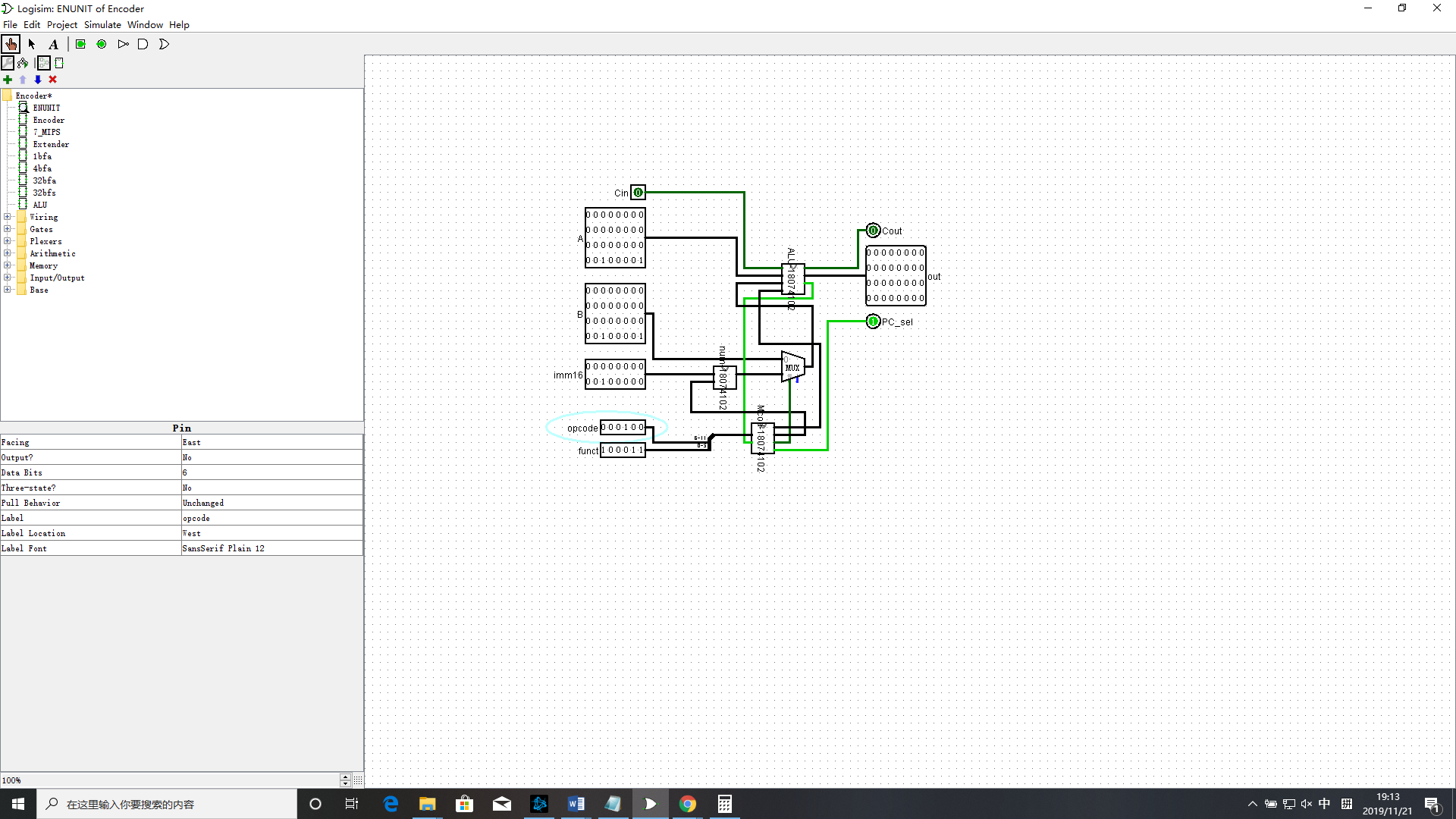
**ori测试，当opcode为001101时，其为按位或功能，立即数向前添16位0，并于A做加法运算，out为0x0000000F。**

**当A=0、B=10，Imm16=32时，完成lui测试，截取测试结果并说明；**

****

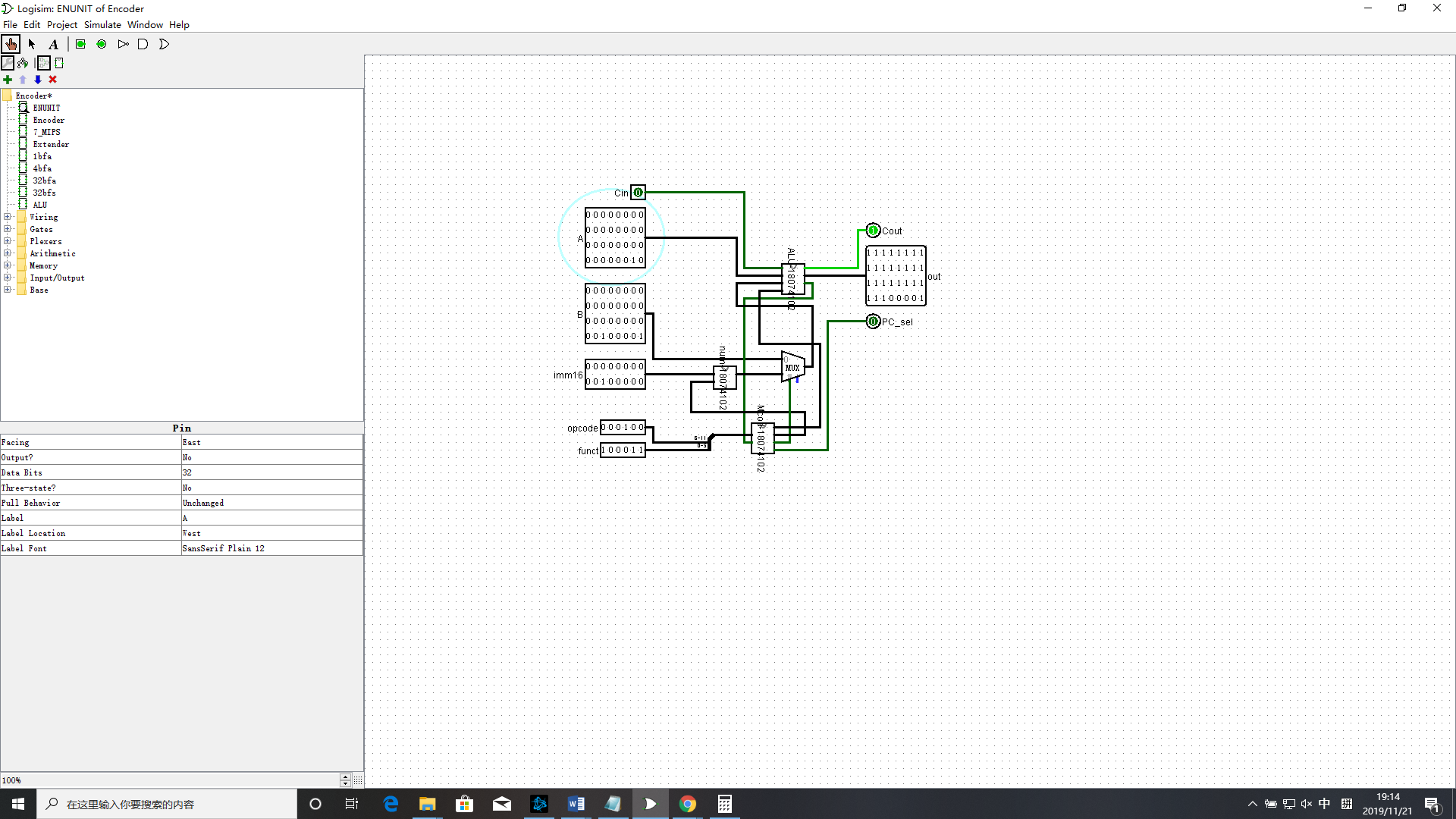
**lui测试，opcode为001111实现高位置立即数扩展功能，imm16经立即数扩展器扩展，结果为0x00200000。**

**当A=33、B=33，Imm16=32时，完成beq测试，截取测试结果并说明；**

****

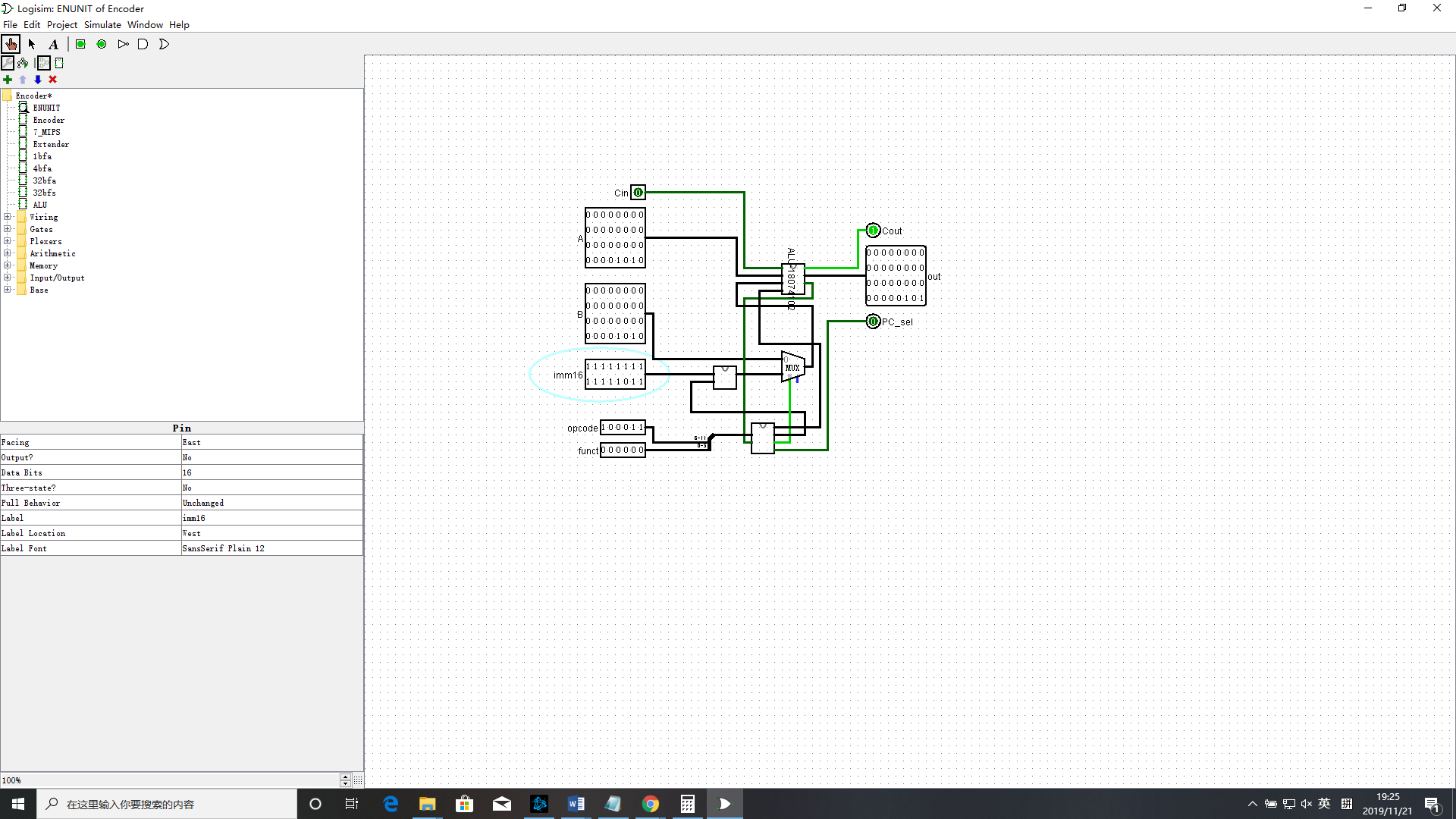
**beq测试，当opcode为000100时实现beq功能，当A和B值相等的时候，PC\_sel输出为1。**

**当A=2、B=33，Imm16=32时，完成beq测试，截取测试结果并说明；**

****

**beq测试，当opcode为000100时实现beq功能，当A和B值不相等的时候，PC\_sel输出为0。**

**当A=10、B=10，Imm16=-5时，完成lw测试，截取测试结果并说明；**

****

**lw测试，当opcode 为100011时，输出值为A加上立即数经符号扩展后的结果。out为0x00000005。**

**七、综合以上内容，写报告和小结（1分）**

**通过32位全加器全减器，ALU，编码器等的编写，通过利用逻辑电路与或非门，位选择器，分线器，译码器等，运用级联的方法，完成了本次数字逻辑大作业，加深了对于级联和电路门的理解。对本门课程的教学内容有很好的帮助，受益良多。**