

《计算机组成原理》教学上机实验报告

原创

司沐夜枫 于 2021-01-10 19:44:54 发布 1817 收藏 12

分类专栏: [实验报告](#) 文章标签: [编程语言](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/WindsZL/article/details/112438900>

版权



[实验报告](#) 专栏收录该内容

7 篇文章 4 订阅

订阅专栏

《计算机组成原理》教学上机实验报告

实验题目:

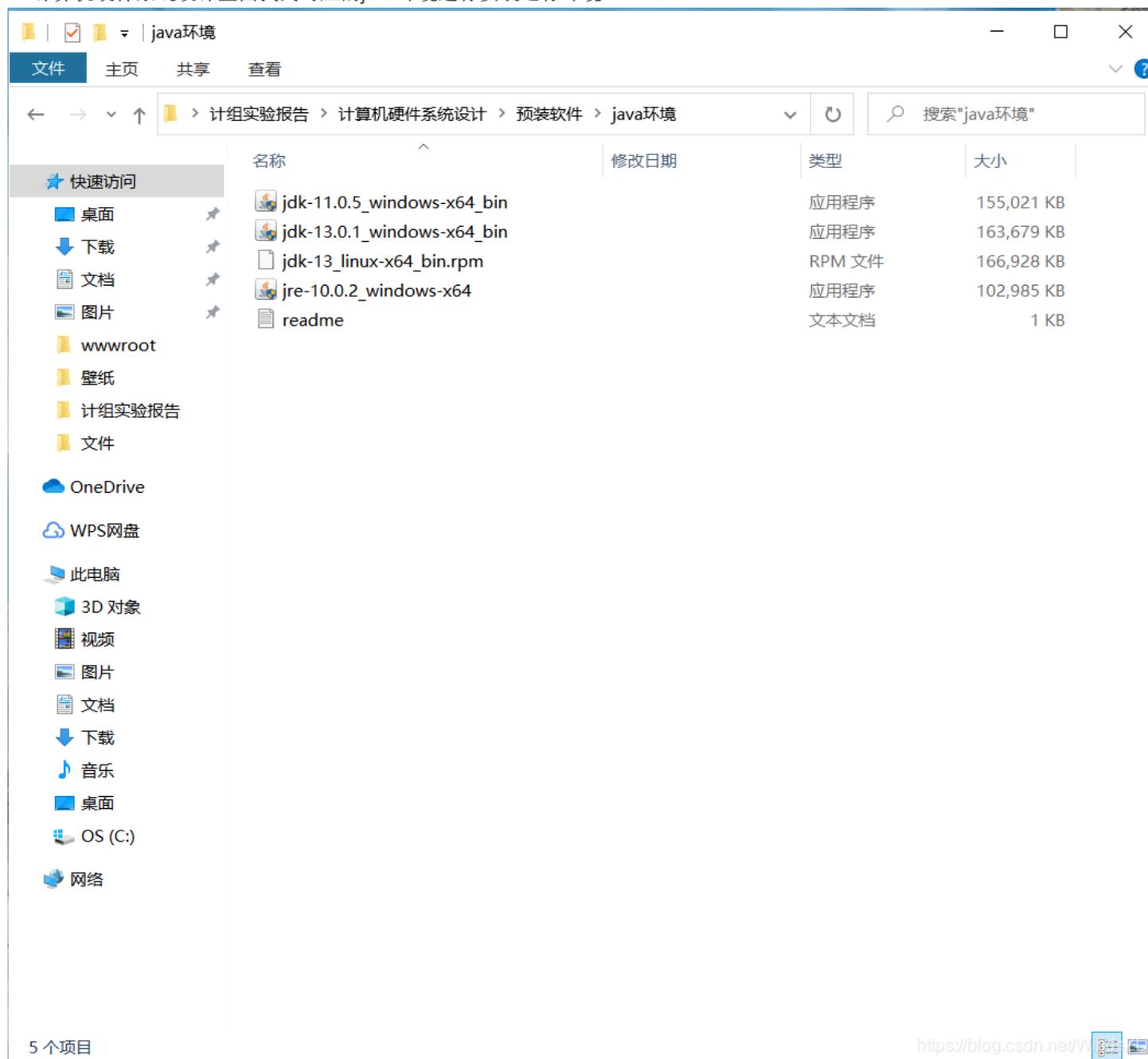
实验一 学习educoder平台, 学习logisim的使用

实验目的和要求:

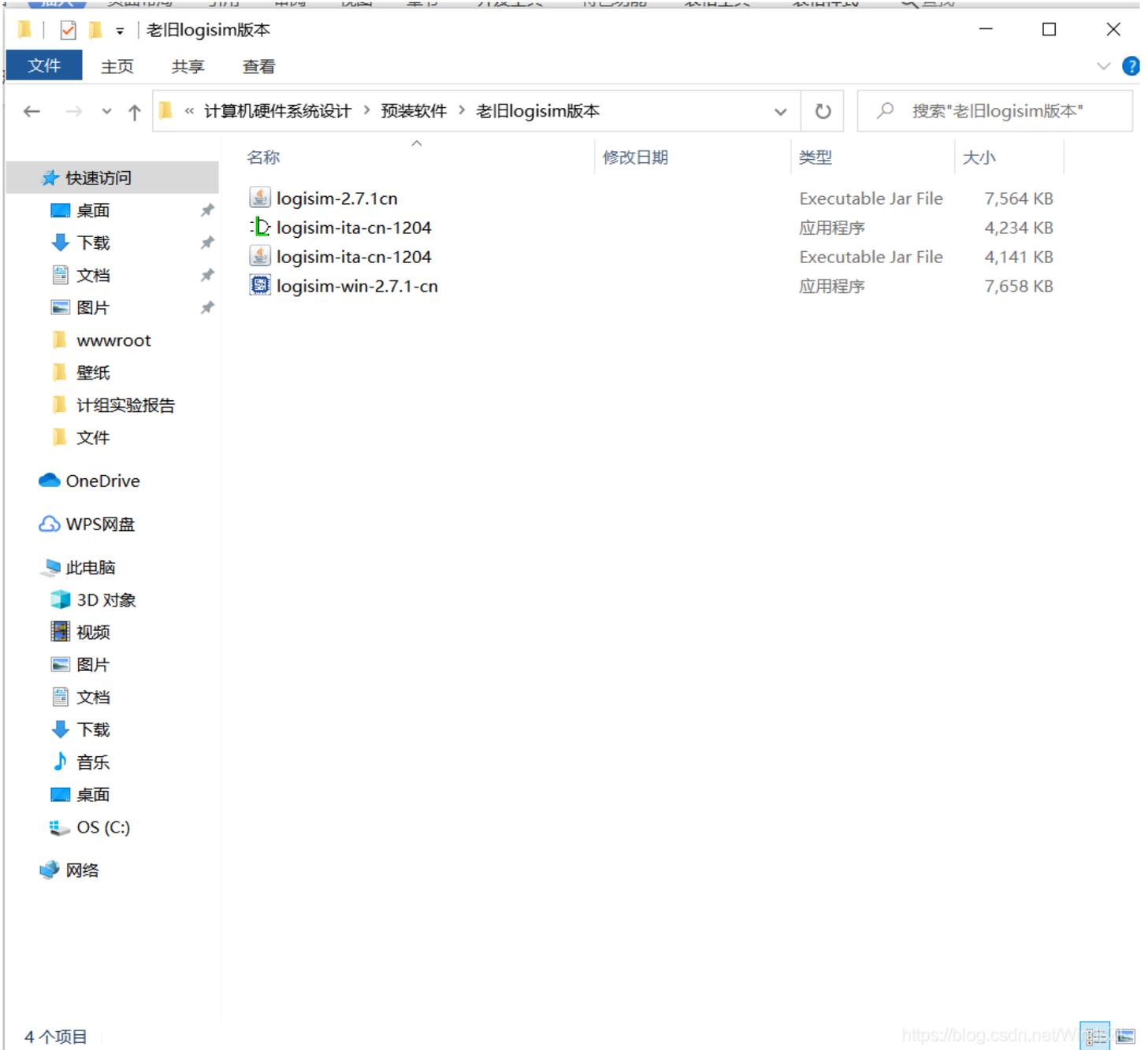
- 1.简单学习logisim使用, 安装好运行环境
- 2.学习educoder平台的使用, 在其提供的配套慕课里学习logisim软件

实验过程:

1. 计算机硬件系统设计里面找到对应的java环境进行安装运行环境



2. 打开并运行logisim软件



3. 加入MOOC课程学习对应的基础操作，练习使用logisim的软件

4. 注册并登录educoder平台，名称要注明班级，学号，姓名，并以学生身份登录，搜索课程号：SGQYO，加入相应的计算机组成原理教学课堂

实验结果：

经过几小时的安装和摸索初步明白了这个软件的基本操作和运用，看到了很多网站上的实验，觉得该课程的实验挺难操作的。

实验分析：

只有不断尝试和摸索，才会取得实验最后的成功，在几个实验的尝试中，发现了一个严重的问题，在educoder平台上代码粘帖不上去，往后的实验去寻找解决吧。

实验题目：

实验二 偶校验编码实验

实验目的和要求：

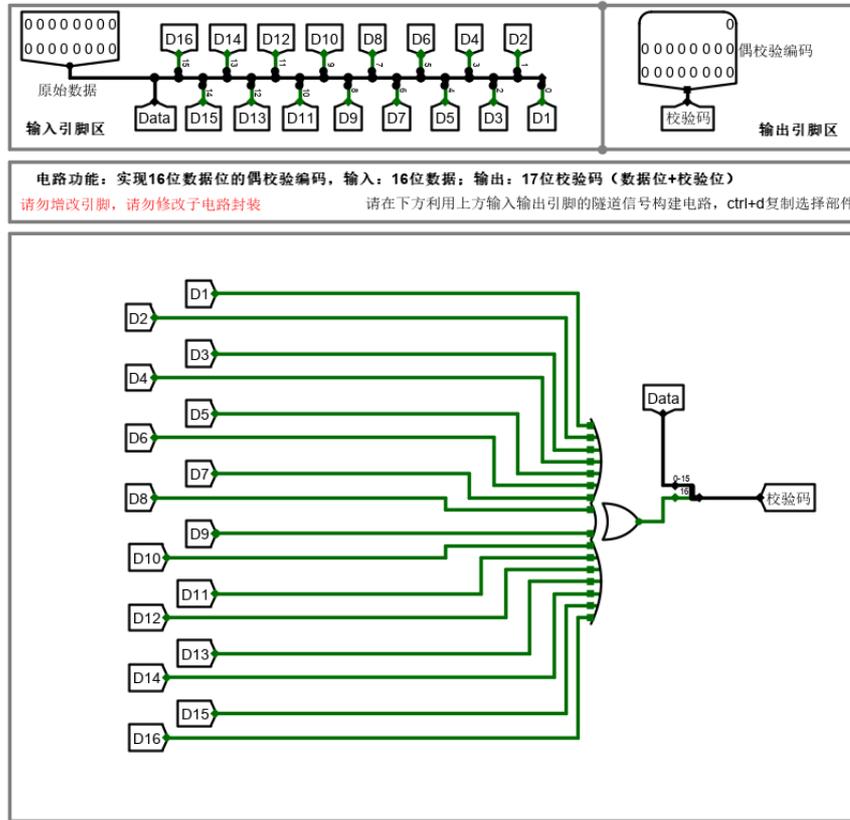
帮助学生掌握奇偶校验基本原理和特性，能在 Logisim 中实现偶校验编码电路，检错电路,理解校验码传输的原理。

实验过程:

在 logisim 中打开实验资料包中的 data.circ 文件，在对应电路中完成偶校验编码电路。实验电路输入输出引脚如图所示。输入：16位原始数据；输出：17位校验码（16位数据位+1位校验位），其中校验位存放在最高位，注意输入16位原始数据的每一位都已经通过分线器利用隧道标签引出，可以直接复制到绘图区使用。

真 窗口 帮助

▷ ◁ ▷



试

1

2

传输

.i1

.i2

.传输

路: ◊ 偶校验编码

◊ 偶校验编码

偶校验编码

上 (北)

SansSerif 标准 12

#000000

<https://blog.csdn.net/WindsZL>

将完成的实验，用记事本打开，粘贴代码，开始运行。

```
00:31:40
代码文件 已保存
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <project source="2.15.0.2.exe" version="1.0">
3   This file is intended to be loaded by Logisim http://logisim.altervista.org
4
5   <lib desc="#Wiring" name="0">
6     <tool name="Splitter">
7       <a name="facing" val="south"/>
8       <a name="fanout" val="4"/>
9       <a name="incoming" val="4"/>
10      <a name="appear" val="left"/>
11      <a name="bit0" val="0"/>
12      <a name="bit1" val="1"/>
13      <a name="bit2" val="2"/>
14      <a name="bit3" val="3"/>
15    </tool>
16    <tool name="Pin">
17      <a name="facing" val="north"/>
```

<https://blog.csdn.net/WindsZL>

实验结果:

测试结果

1/1 全部通过

测试集 1

代码执行时长: 2.7秒 | 消耗内存135.15MB

Cnt	GB2312	ParityD	Cnt	GB2312	ParityD
00	bbaa	0bbaa	00	bbaa	0bbaa
01	d6d0	0d6d0	01	d6d0	0d6d0
02	bfc6	1bfc6	02	bfc6	1bfc6
03	bcbc	0bcbc	03	bcbc	0bcbc
04	b4f3	0b4f3	04	b4f3	0b4f3

https://blog.csdn.net/WindsZL

实验分析:

奇偶校验是一种冗余编码校验,在存储器中是按存储单元为单位进行的,是依靠硬件实现的,因而适时性强,但这种校验方法只能发现奇数个错,如果数据发生偶数位个错,由于不影响码子的奇偶性质,因而不能发现。

对于位数较少,电路较简单的应用,可以采用奇偶校验的方法提高系统的可靠性。

实验题目:

实验三 3、8位可控加减法电路设计

实验目的和要求:

帮助学生掌握一位全加器的实现逻辑,掌握多位可控加减法电路的实现逻辑,熟悉 Logisim 平台基本功能,能在 logisim 中实现多位可控加减法电路。

实验过程:

在 Logisim 模拟器中打开 alu.circ 文件,在对应子电路中利用已经封装好的全加器设计88位串行可控加减法电路,其电路引脚定义如图所示,用户可以直接使用在电路中使用对应的隧道标签,其中 X, Y 为两输入数, Sub 为加减控制信号, S 为运算结果输出, Cout 为进位输出, OF 为有符号运算溢出位。

功能说明: 8位可控加减法器 Sub=0: $S = X + Y$ Sub=1: $S = X - Y$ Cout为最高位进位位, Overflow为有符号溢出标志

8位可控加减法器

运算指示

https://blog.csdn.net/WindsZL

将完成的实验，用记事本打开，粘贴代码，运行成功。

The screenshot shows a Logisim project editor window titled "运算器设计(HUST)" with a timestamp of 01:31:31. The code editor displays XML code for a project named "2.15.0.2.exe". The code includes a library definition for a "Splitter" component with attributes for facing, fanout, incoming, and appear. Below the code editor, the "测试结果" (Test Results) section shows a green checkmark and "1/1 全部通过" (All passed). Under "测试集1" (Test Set 1), it indicates "消耗内存86.23MB" (Consumed memory 86.23MB) and "代码执行时长: 2.47秒" (Code execution time: 2.47s). A comparison table shows "预期输出" (Expected Output) and "实际输出" (Actual Output) for 10 test cases, with all values matching. At the bottom, it shows "本关最大执行时间: 20秒" (Max execution time for this level: 20s) and "本次评测耗时(编译、运行总时间): 2.011 秒" (Time for this evaluation (compilation, total runtime): 2.011s). A URL "https://blog.csdn.net/..." and a "评测" (Evaluate) button are also visible.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project source="2.15.0.2.exe" version="1.0">
  This file is intended to be loaded by Logisim http://logisim.altervista.org
  <lib desc="#Wiring" name="0">
    <tool name="Splitter">
      <a name="facing" val="north"/>
      <a name="fanout" val="4"/>
      <a name="incoming" val="4"/>
      <a name="appear" val="left"/>
    </tool>
  </lib>
</project>
```

—— 预期输出 ——						—— 实际输出 ——							
1	X	Y	Sub	S	Cout	OF	1	X	Y	Sub	S	Cout	OF
2	10	21	0	31	0	0	2	10	21	0	31	0	0
3	7f	02	0	81	0	1	3	7f	02	0	81	0	1
4	ff	fe	0	fd	1	0	4	ff	fe	0	fd	1	0
5	81	82	0	03	1	1	5	81	82	0	03	1	1
6	10	df	1	31	0	0	6	10	df	1	31	0	0
7	7f	fe	1	81	0	1	7	7f	fe	1	81	0	1
8	ff	02	1	fd	1	0	8	ff	02	1	fd	1	0
9	81	7e	1	03	1	1	9	81	7e	1	03	1	1
10	00	00	0	00	0	0	10	00	00	0	00	0	0

实验结果：

据实验要求，按步骤慢慢完成就OK了，实验过程比较难受，尤其电脑端代码粘贴不上，手机端剪贴版不够大，分步粘贴的艰辛。

实验分析：

这个实验的本意是用一位全加器和逻辑门搭建电路。用全加器和异或门也可以完成。这里涉及两个“异或”的知识点：

- 1.位变量a和0异或，结果为a；和1异或，结果为~a（取反）
- 2.溢出标志可以用最高位（符号位）进位和次高位进位异或得到。即这两个进位相同则不溢出，否则溢出。

实验题目：

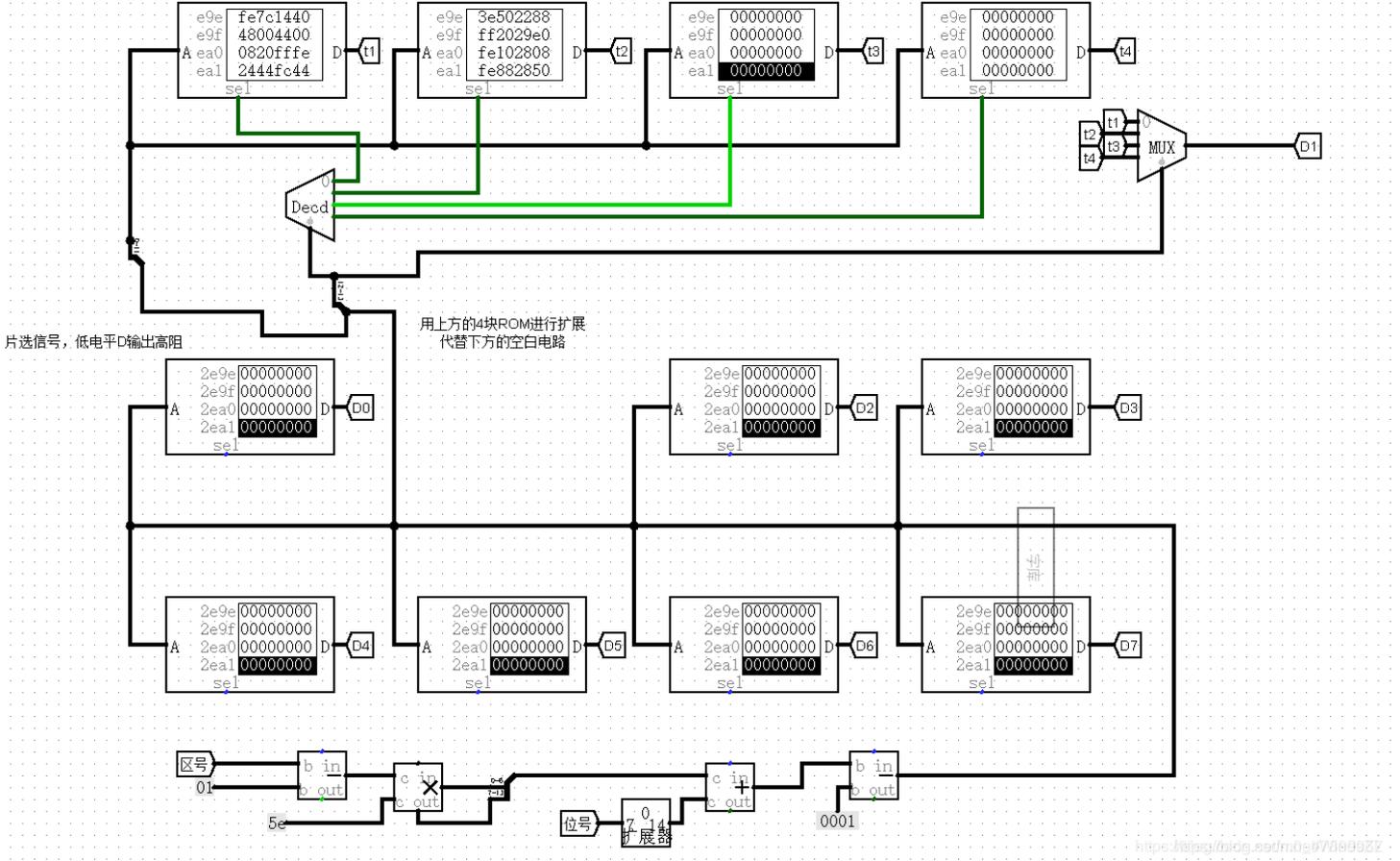
实验四 汉字字库存储芯片扩展实验

实验目的和要求：

理解存储系统进行位扩展、字扩展的基本原理，能利用相关原理解决实验中汉字字库的存储扩展问题，并能够使用正确的字库数据填充。

实验过程:

现有如下 ROM 组件，4片4K32位 ROM，7片16K32位 ROM，请在 Logisim 平台构建 GB2312 汉字编码的16K16点阵汉字字库，电路输入为汉字区号和位号，电路输出为8×32位（16K16=256位点阵信息），待完成的字库电路输入输出引脚见后图，具体参见工程文件中的 storage.circ 文件，图中左侧是输入引脚，分别对应汉字区位码的区号和位号，中间区域为8个32位的输出引脚，可一次性提供一个汉字的256位点阵显示信息，右侧是实际显示区域，用于观测汉字显示是否正常。待完成字库子电路封装已经完成，请勿修改以免影响后续自动测试功能。



将完成的实验，用记事本打开，粘贴代码，运行成功。

存储系统设计(HUST)

00:00:40

退出实训

代码文件 已保存

```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <project source="2.15.0.2.exe" version="1.0">
3   This file is intended to be loaded by Logisim http://logisim.altervista.org
4
5   <lib desc="#Wiring" name="0">
6     <tool name="Splitter">
7       <a name="facing" val="north"/>
8       <a name="fanout" val="4"/>
9       <a name="incoming" val="4"/>
10      <a name="appear" val="left"/>
11      <a name="bit0" val="0"/>
12      <a name="bit1" val="1"/>
13      <a name="bit2" val="2"/>
14      <a name="bit3" val="3"/>
15    </tool>
16    <tool name="Pin">
17      <a name="facing" val="north"/>
18      <a name="output" val="false"/>
```

测试结果

1/1 全部通过

测试集1 消耗内存90.06MB 代码执行时长: 2.37秒

本关最大执行时间: 20秒 本次评测耗时(编译、运行总时间): 1.998 秒

<https://blog.csdn.net/windsZL> 评测

实验结果:

1/1 全部通过

测试集1 代码执行时长: 2.47秒 | 消耗内存99.76MB

- 预期输出 -

1	CNT	Qu	Wei	Chip1Data
2	00	1b	0a	08600858
3	01	36	30	7ffe0104
4	02	1f	26	0850f890
5	03	1c	1c	13fc1048
6	04	14	53	01000100
7	05	31	07	00201110
8	06	2a	27	0fe00820
9	07	19	5a	5ff44024
10	08	1c	32	40027ffe
11	09	1d	2c	09447f40
12	0a	33	5d	0400fffe
13	0b	12	1f	7f440244
14	0c	36	11	01007ffc
15	0d	2a	54	20003ff8
16	0e	15	24	7f7e2244
17	0f	28	0b	04400280

- 实际输出 -

```
1 六月 10, 2020 8:21:02 上午 java.util.pref
2 INFO: Created user preferences directory.
3 CNT Qu Wei Chip1Data
4 00 1b 0a 08600858
5 01 36 30 7ffe0104
6 02 1f 26 0850f890
7 03 1c 1c 13fc1048
8 04 14 53 01000100
9 05 31 07 00201110
10 06 2a 27 0fe00820
11 07 19 5a 5ff44024
12 08 1c 32 40027ffe
13 09 1d 2c 09447f40
14 0a 33 5d 0400fffe
15 0b 12 1f 7f440244
16 0c 36 11 01007ffc
17 0d 2a 54 20003ff8
```

实验分析：

这次实验让我学习到了很多。一开始，我从上个周开始就在做这个汉字字库存储芯片扩展实验，但是画的电路交上去测评一直都是错的。好在今天，经同学点拨，要将ROM里的内容复制到自己的设计画出的存储器中。这样一修改测评就对了。总之，完成这次实验挺不容易的，也让我学到了许多。