**组成原理实验课程第 3 次实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验名称 | 寄存器堆实现 | | | 班级 | 李涛老师 |
| 学生姓名 | 曹瑜 | 学号 | 2212794 | 指导老师 | 董前琨 |
| 实验地点 | 实验楼A区306 | | 实验时间 | 2024.04.25 | |

**1、实验目的**

1.熟悉并掌握MIPS计算机中寄存器堆的原理和设计方法。

2.初步了解MIPS指令结构和源操作数/目的操作数的概念。

3.熟悉并运用verilog语言进行电路设计。

4.为后续设计cpu的实验打下基础。

**2、实验内容说明**

1.学习MIPS计算机中寄存器堆的设计及原理；

2.自行设计本次实验的方案，画出结构框图，详细标出输入输出端口，本次实验建议设计为异步读同步写的寄存器堆；

3.本次实验建议寄存器堆设计为1个写端口和2个读端口；

4.根据设计的实验方案，使用verilog编写相应代码；

5.将以上设计作为一个单独的模块，设计一个外围模块去调用该模块，外围模块中需调用封装好的LCD触摸屏模块，显示寄存器堆的读写端口地址和数据，最好能扫描出所有寄存器的值显示在LCD触摸屏上，并且需要利用触摸功能输入寄存器堆的读写地址和写数据；

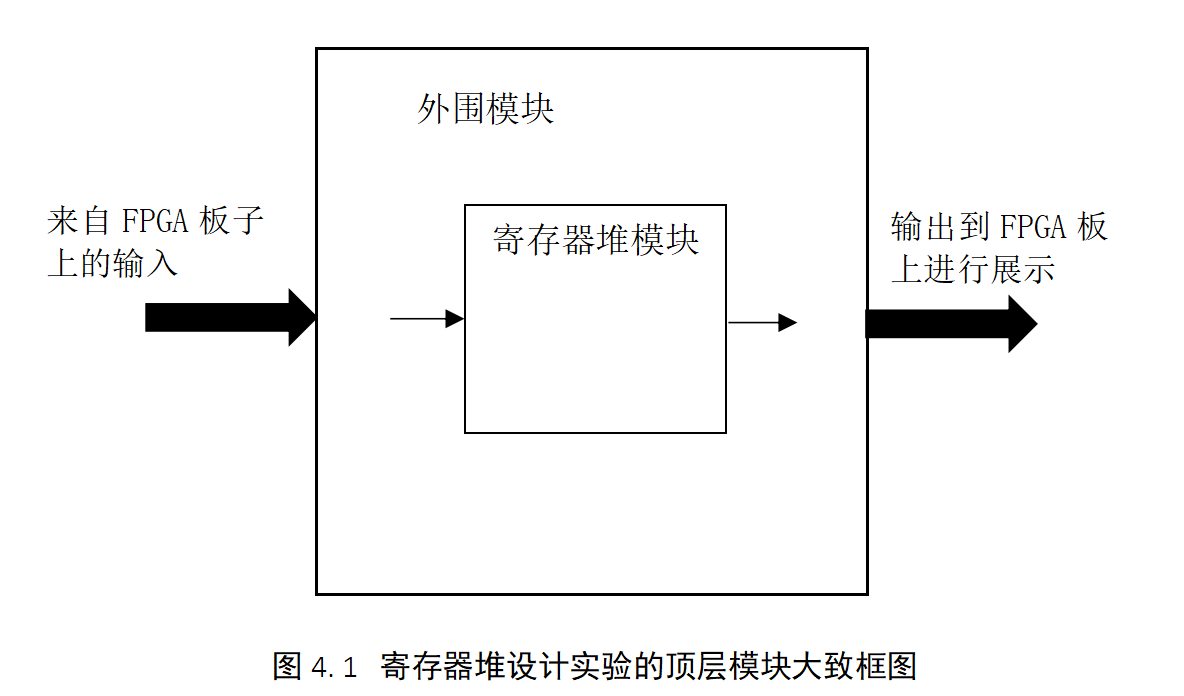
6.将提供的存储32位数的32个寄存器改为16个64位寄存器；

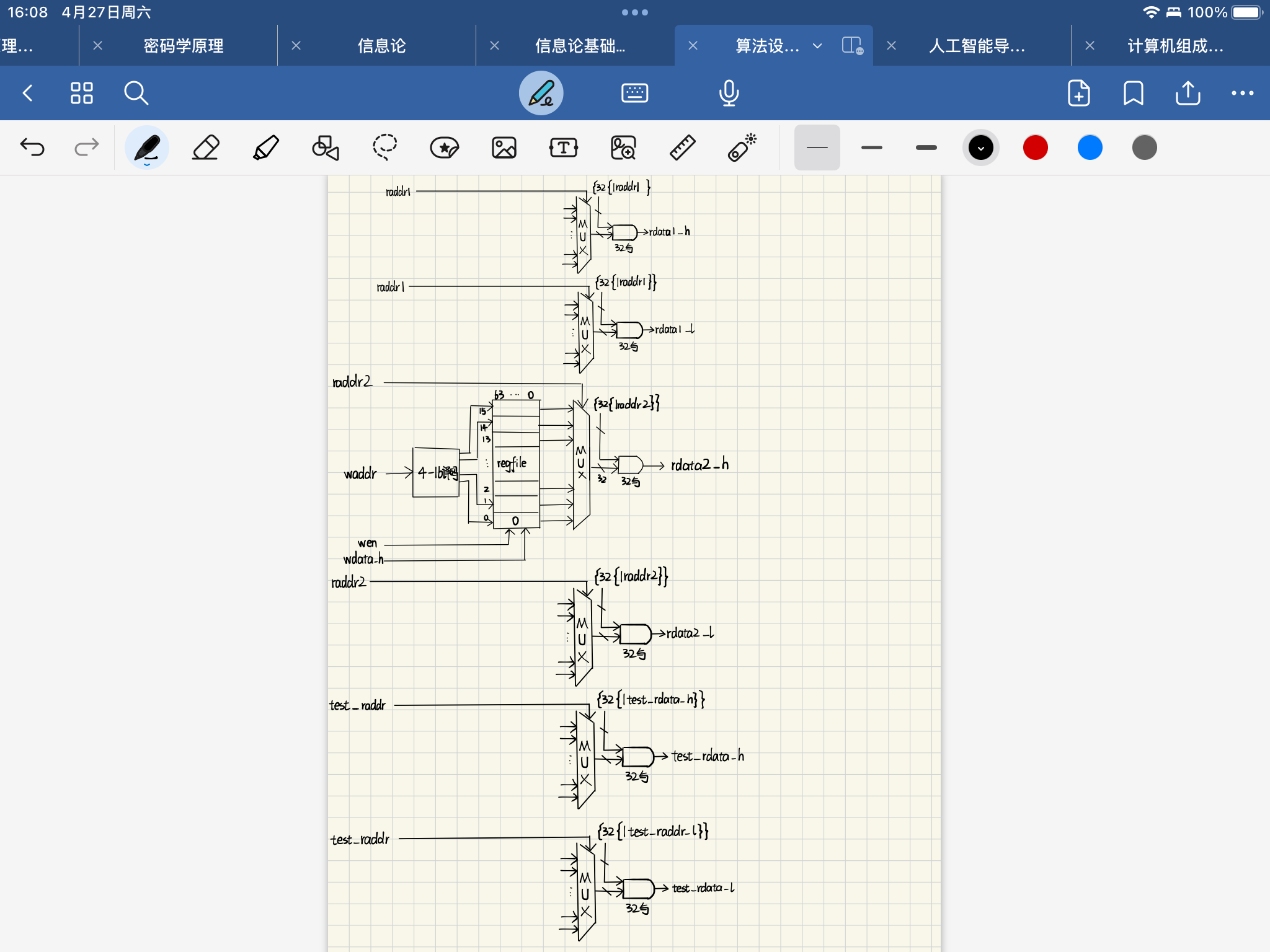
（**设计思想：将64位数拆分成高低各32位放到两个32位寄存器中存储**）

1. **实验原理图**

test\_addr为寄存器堆的调试读端口，而test\_data为通过该端口读出的数据；“display\_name”域有特殊处理，其头3个字符显示“REG”，第4个字符表示“H”或者“G”，表明该寄存器存的是高/低位，但对于第5位字符(0~F)的赋值，在设计LCD屏显示字符时，给第5位字符赋值{4'b0011,test\_addr[3:0]}；

共实现16个64位寄存器，其中0号寄存器读出的值恒为0，寄存器堆为异步读同步写，共有2个写端口和6个读端口，读端口高位和低位两两一组，其中2个读端口为调试读端口，用于扫描读出16个寄存器；





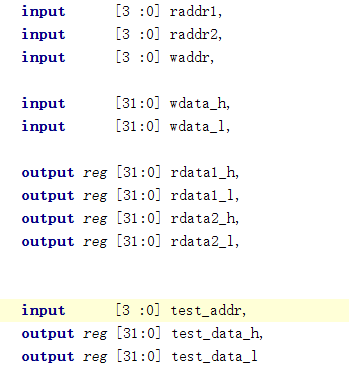
1. **实验步骤**

**[regfile]模块：**

1. 在regfile模块里将读写的三个地址改为4位（表示16个）；

将rdata1/2分为高位和低位，定义32位宽的rdata\_h和rdata\_l；

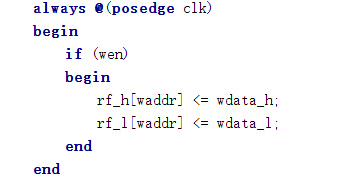
对应修改test\_addr和test\_data\_h 、test\_data\_l



2.定义两组16个存储32位数的寄存器堆，存储高低位；

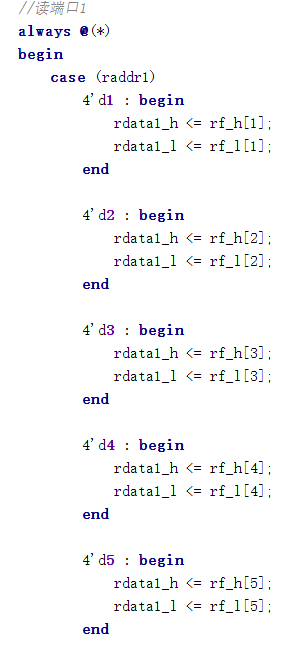
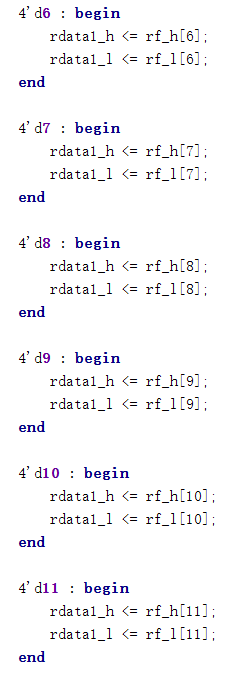
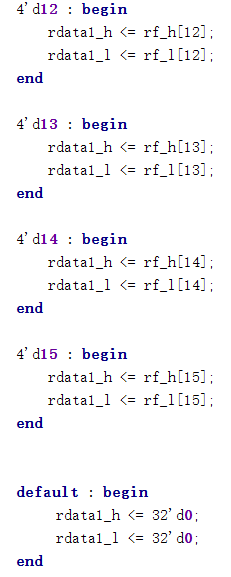


3.接收到时钟信号时，将写数据端的两个值分别输出到两个目标寄存器中

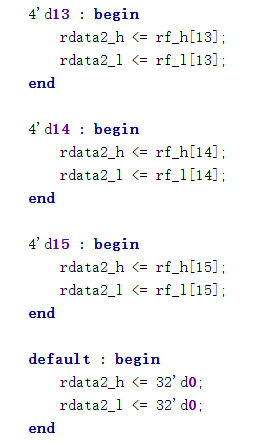
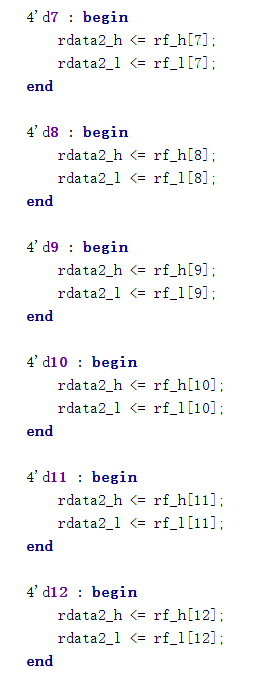
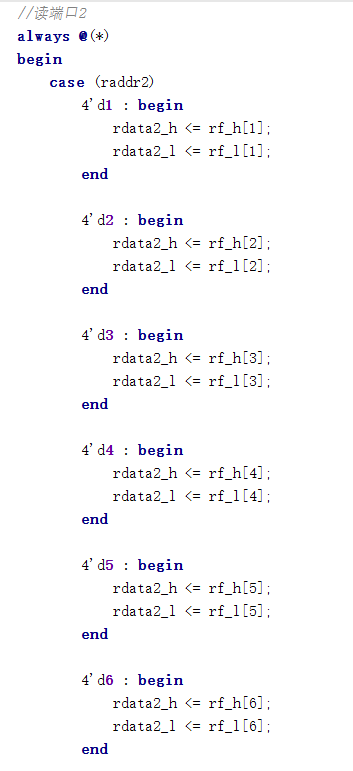


1. 依次修改raddr1为不同值时对应的读数据操作，

raddr1为0时默认读到的为64位0；

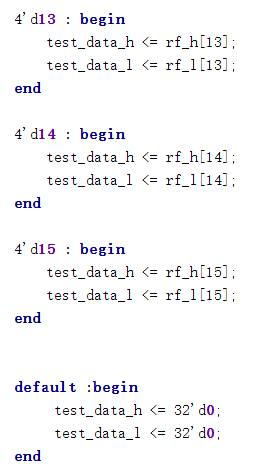
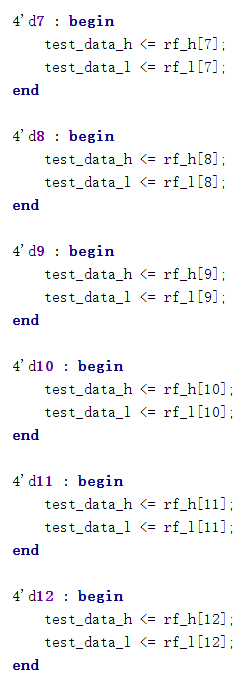
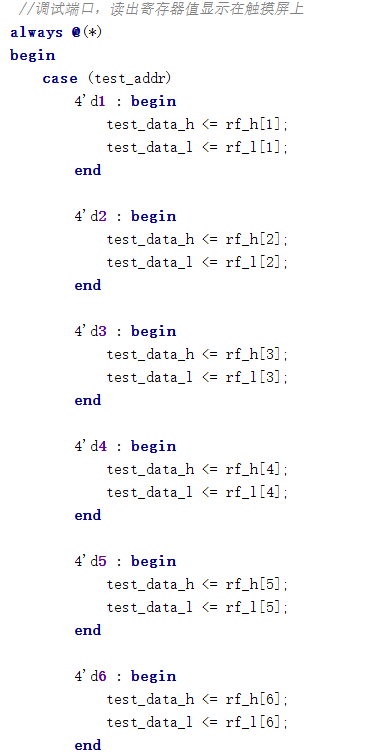
  

1. 对读端口2也做同样的修改：



1. 对调试端口进行输出对应修改：

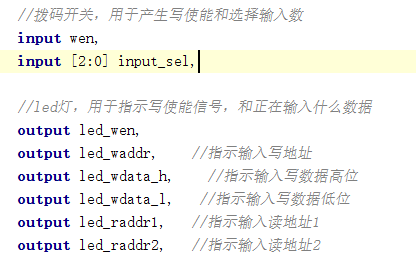
不同的test\_addr值对应输出两个高低位的值；



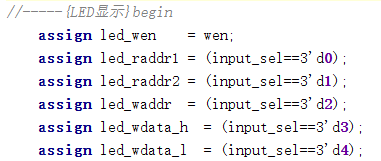
**[regfile\_display]模块：**

1. 将拨码开关新增为3个，选择对应的5个功能

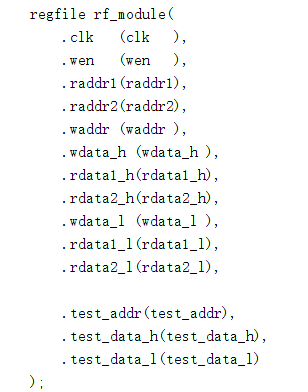
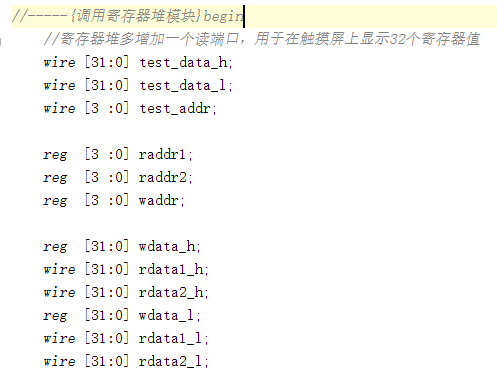
Led灯更新为5个，写地址/写数据高位/写数据低位/输入读地址1/输入低地址2;



1. 修改led显示：input\_sel对应的数值为0/1/2/3/4时分别对应不同的操作；



1. 在寄存器堆模块处将几个地址修改为4位宽，将读写数据相关变量修改为两个 32位宽的形式，修改对应参数传递调用对应值；



1. 修改input\_sel对应的不同操作：

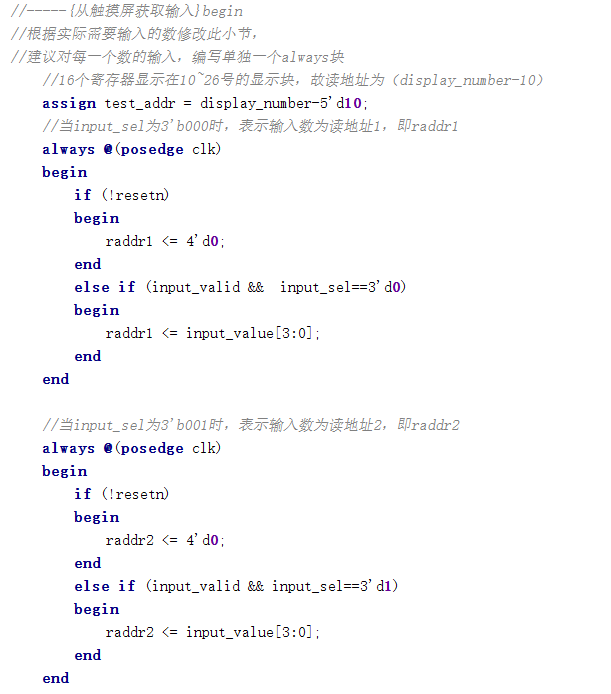
input\_sel值为000（0）时，表示输入raddr1；

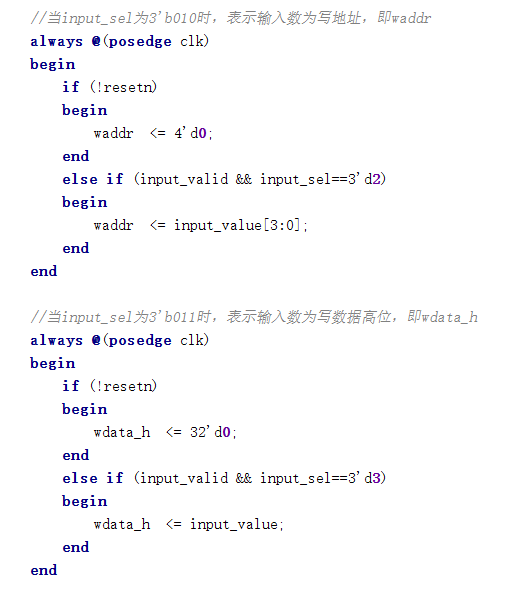
input\_sel值为001（1）时，表示输入raddr2；

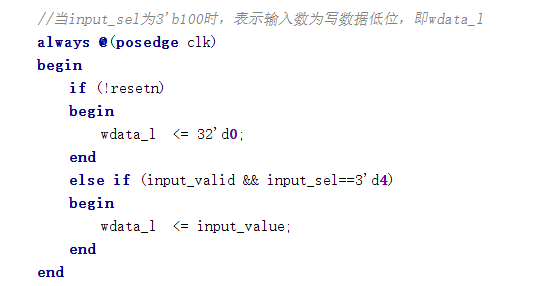
input\_sel值为010（2）时，表示输入waddr；

input\_sel值为011（3）时，表示输入wdata\_h；

input\_sel值为100（4）时，表示输入wdata\_l；



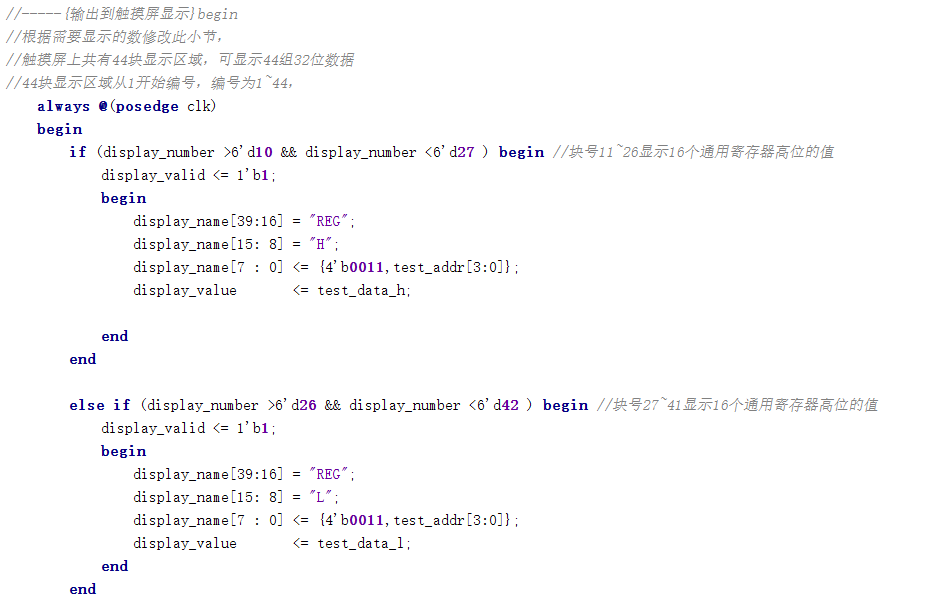




1. 修改触摸屏显示：

块号11~26显示16个通用寄存器高位的值；

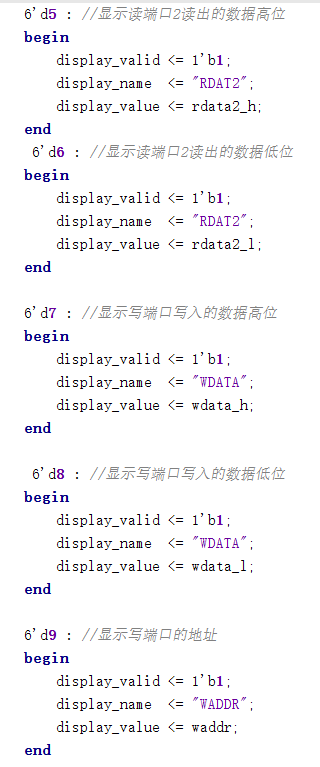
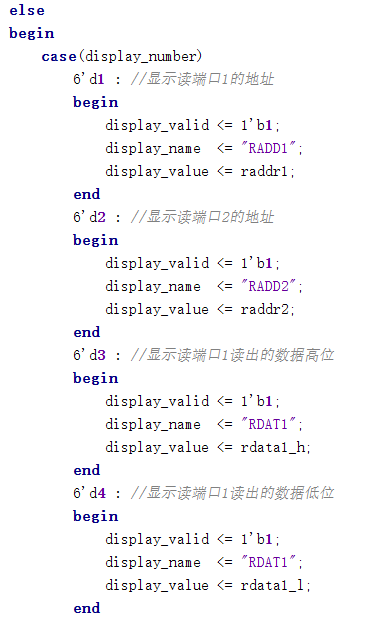
块号27~41显示16个通用寄存器低位的值；



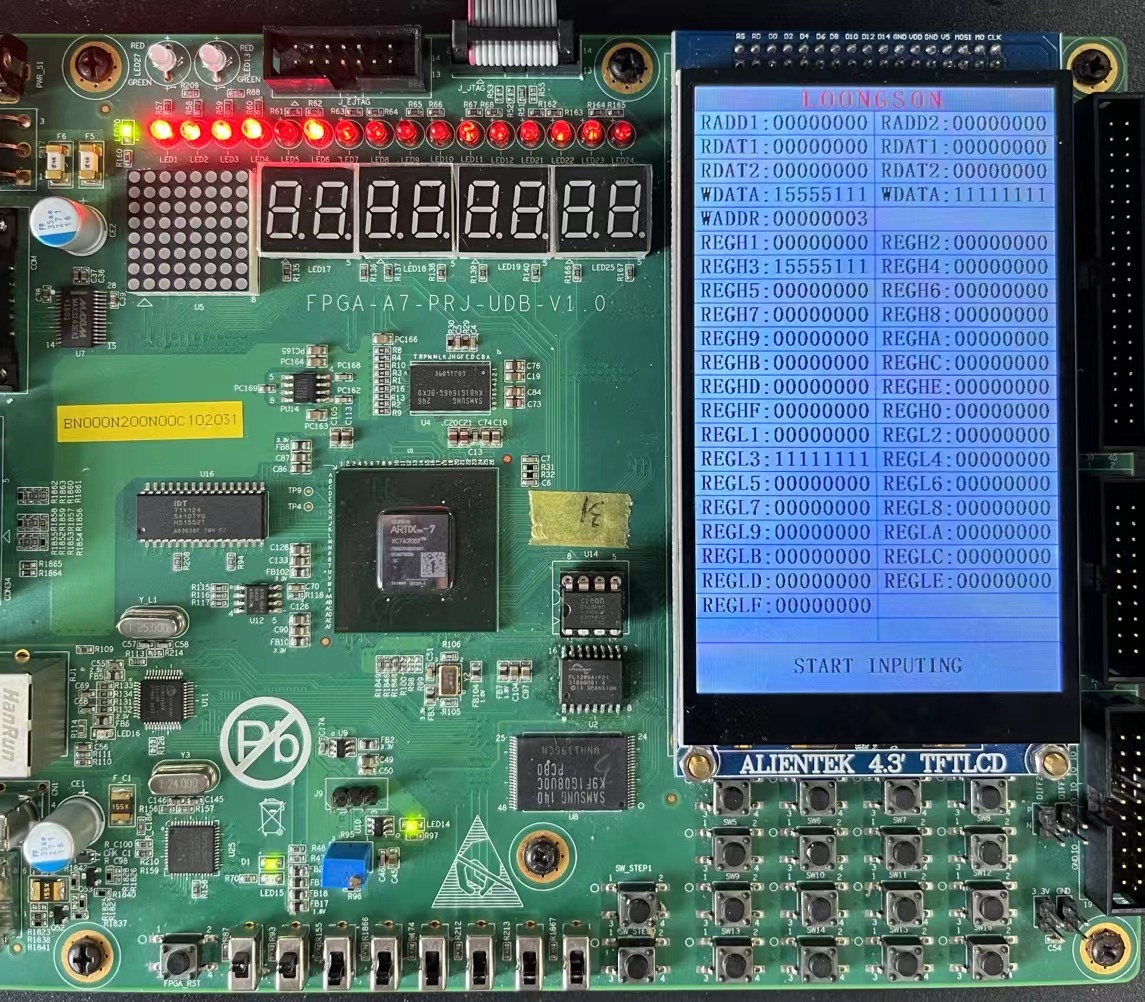
1. 修改led屏前10块的显示：

依次显示:

|  |  |
| --- | --- |
| raddr1 | raddr2 |
| rdata1\_h | rdata1\_l |
| rdata2\_h | rdata2\_l |
| wdata\_h | wdata\_l |
| waddr |  |



**5、实验结果分析**

****

此时的面板显示**：**

第一行为radd1，radd2，

第二行左边为rdata1的高32位显示，右边为rdata1的低32位显示；

第三行左边为rdata2的高32位显示，右边为rdata2的低32位显示；

第四行左边为wdata\_h,右边为wdata\_l；

然后REGH依次为16个寄存器的高32位显示；

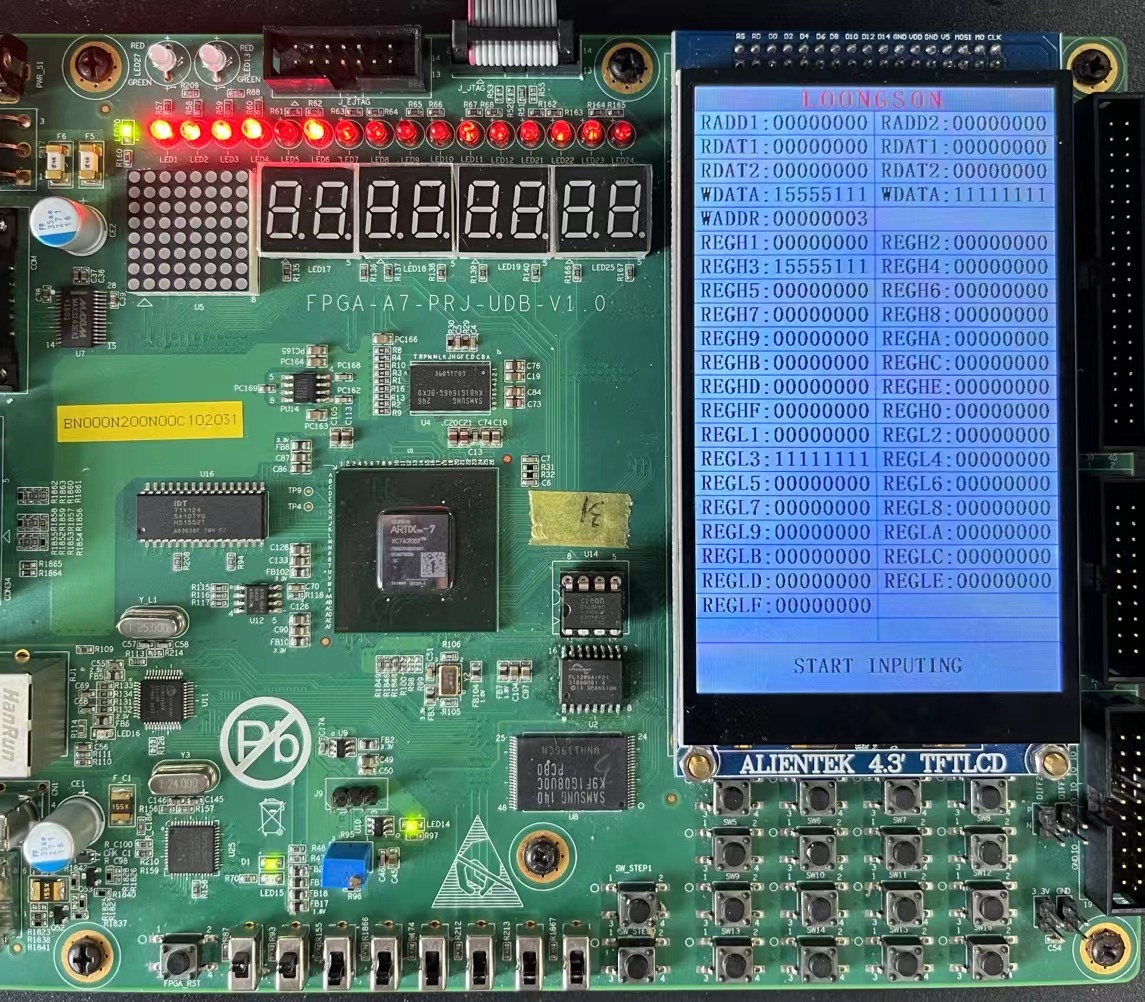
REGL依次为16个寄存器的低32位显示；

**操作1：写数**

往wdata的高低位分别写入15555111和11111111；

Waddr输入3，表示写入3号寄存器；

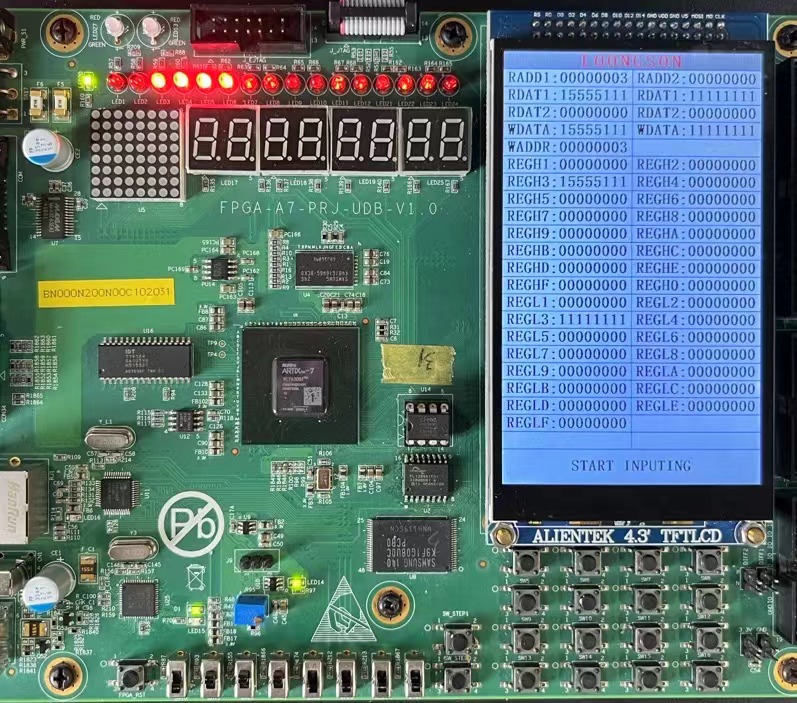
给出wen信号后可见REGH3和REGL3内都写入了对应的数；

****

**操作2：读数data1**

在radd1处输入3，表示读取3号寄存器内的数；

给出wen信号后，可见rdat1\_h和rdat2\_l处显示出了对应的3号寄存器存储数值15555111和11111111 即1555511111111111

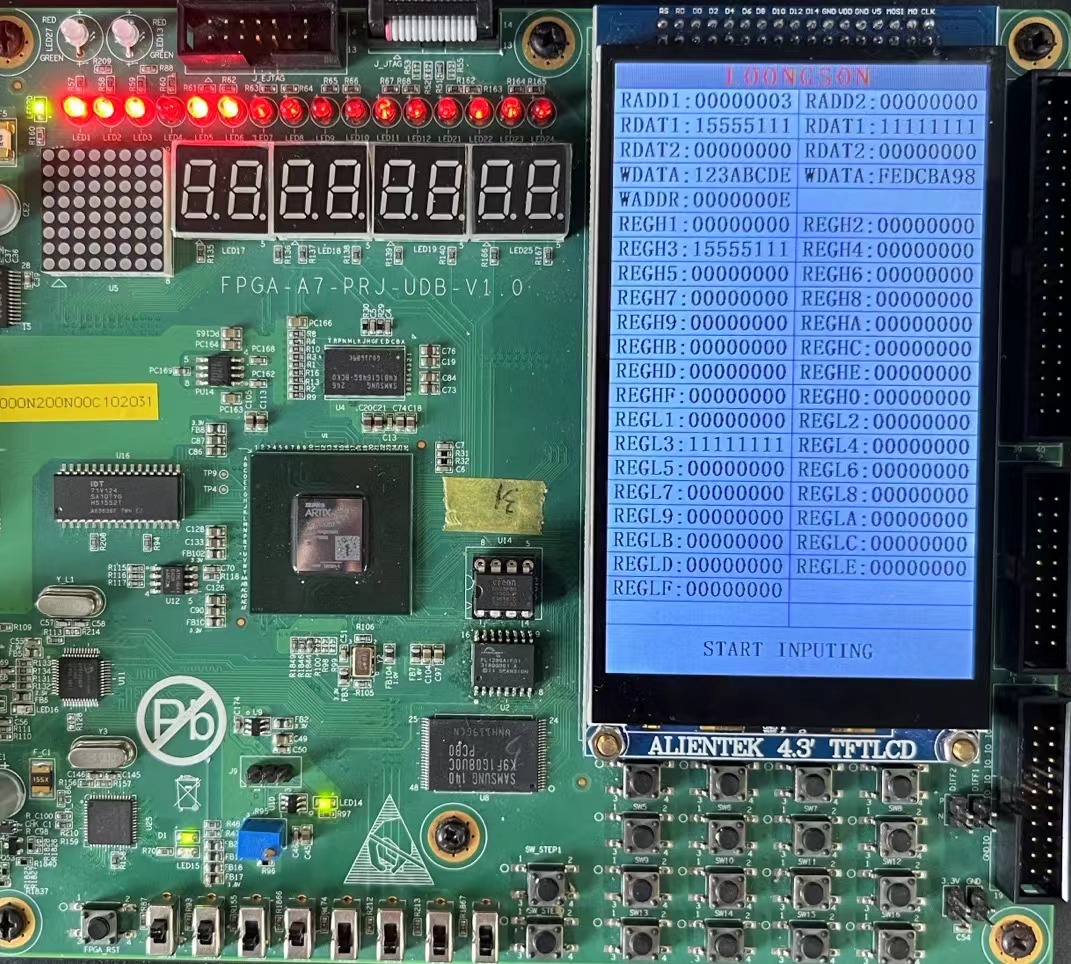
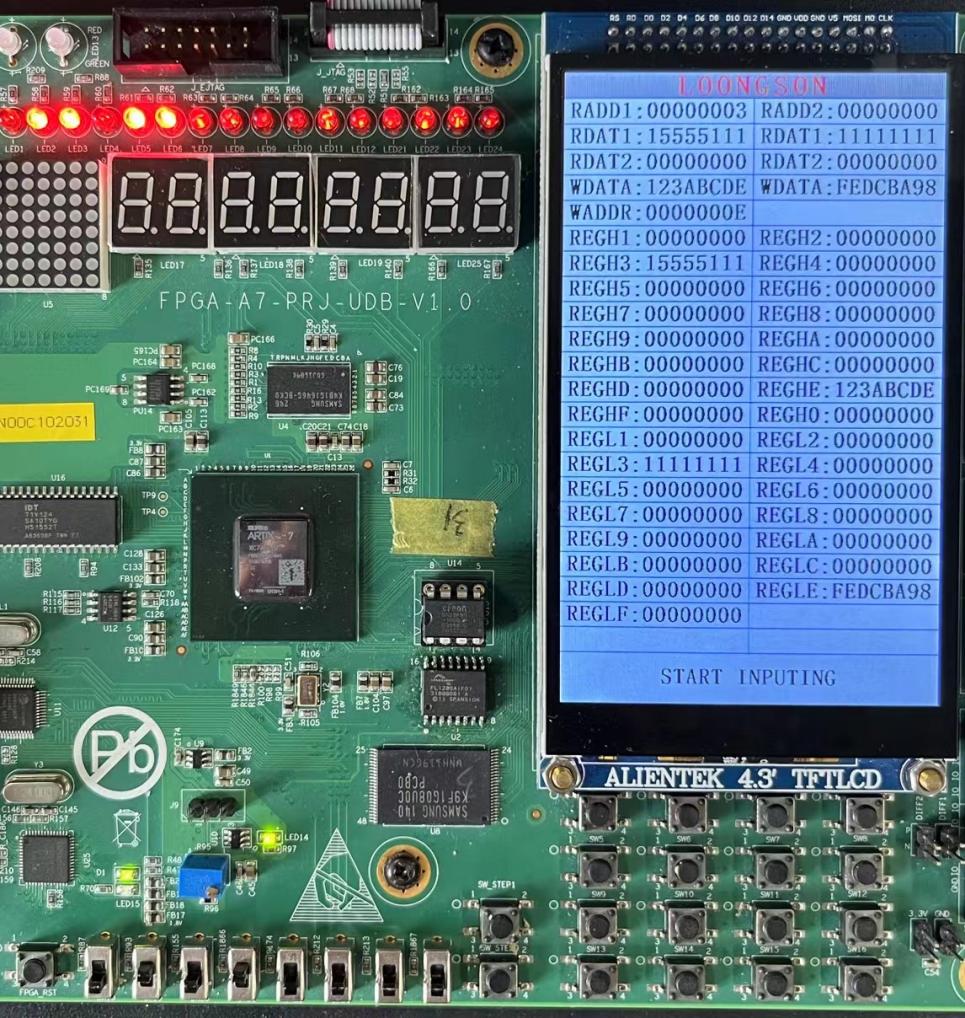
****

**操作3：写数**

往wdata的高低位分别写入123ABCDE和FEDCBA98；

waddr输入E，表示写入E号寄存器；

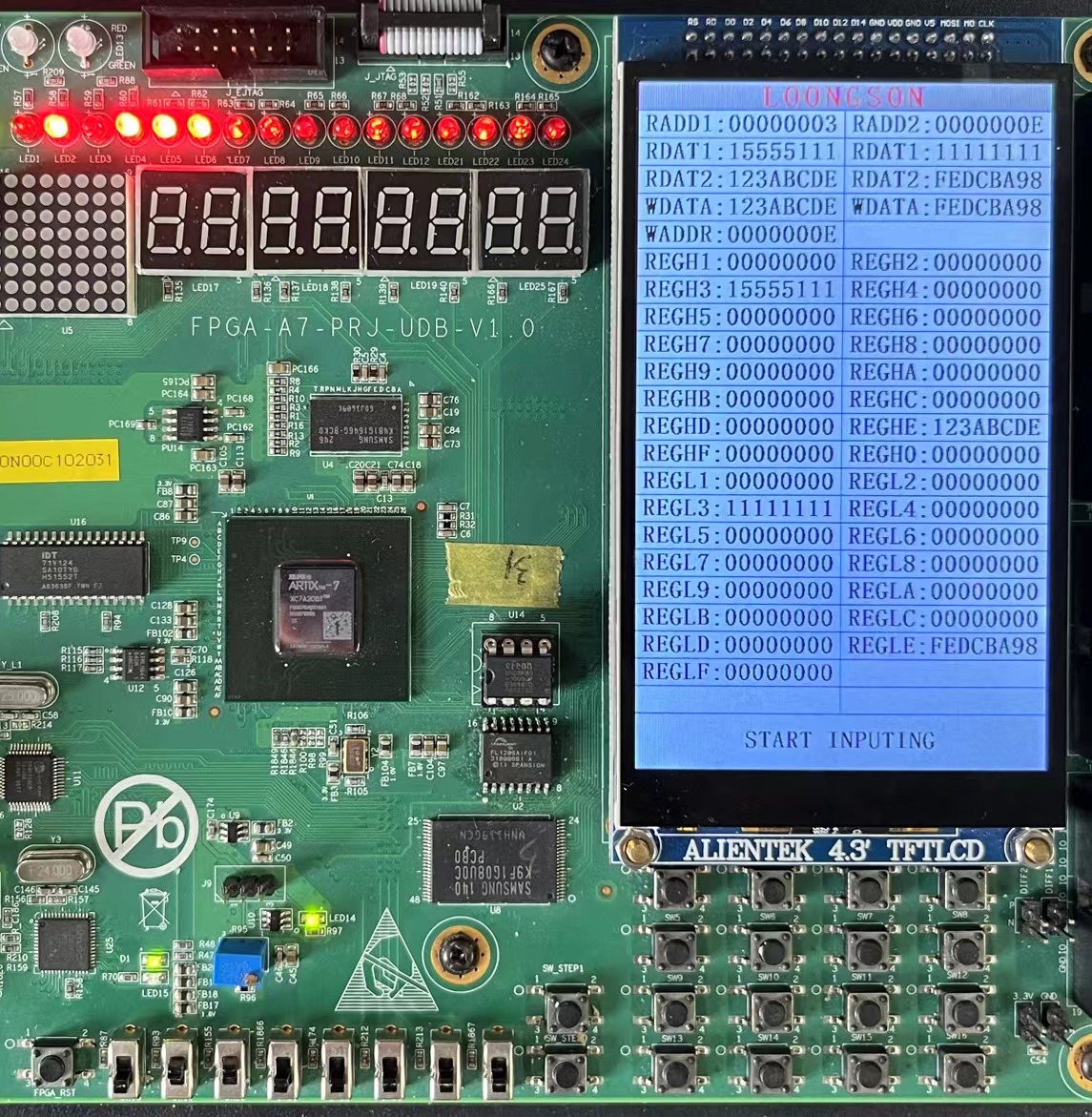
给出wen信号后可见REGHE和REGLE内都写入了对应的数；

** 给出wen信号： **

**操作4：读数data2**

在radd1处输入E，表示读取E号寄存器内的数；

给出wen信号后，可见rdat1\_h和rdat2\_l处显示出了对应的E号寄存器存储数值： 123ABCDE和FEDCBA98 即123ABCDEFEDCBA98；

****

**6、总结感想**

本次实验中学习了MIPS计算机中寄存器堆的设计原理，在理解了32个32位寄存器堆的设计原理后，自行完成了16个64位寄存器堆的设计并在实验箱上完成了功能验证，对寄存器堆的结构和数据的读写原理有了更深一步认知。