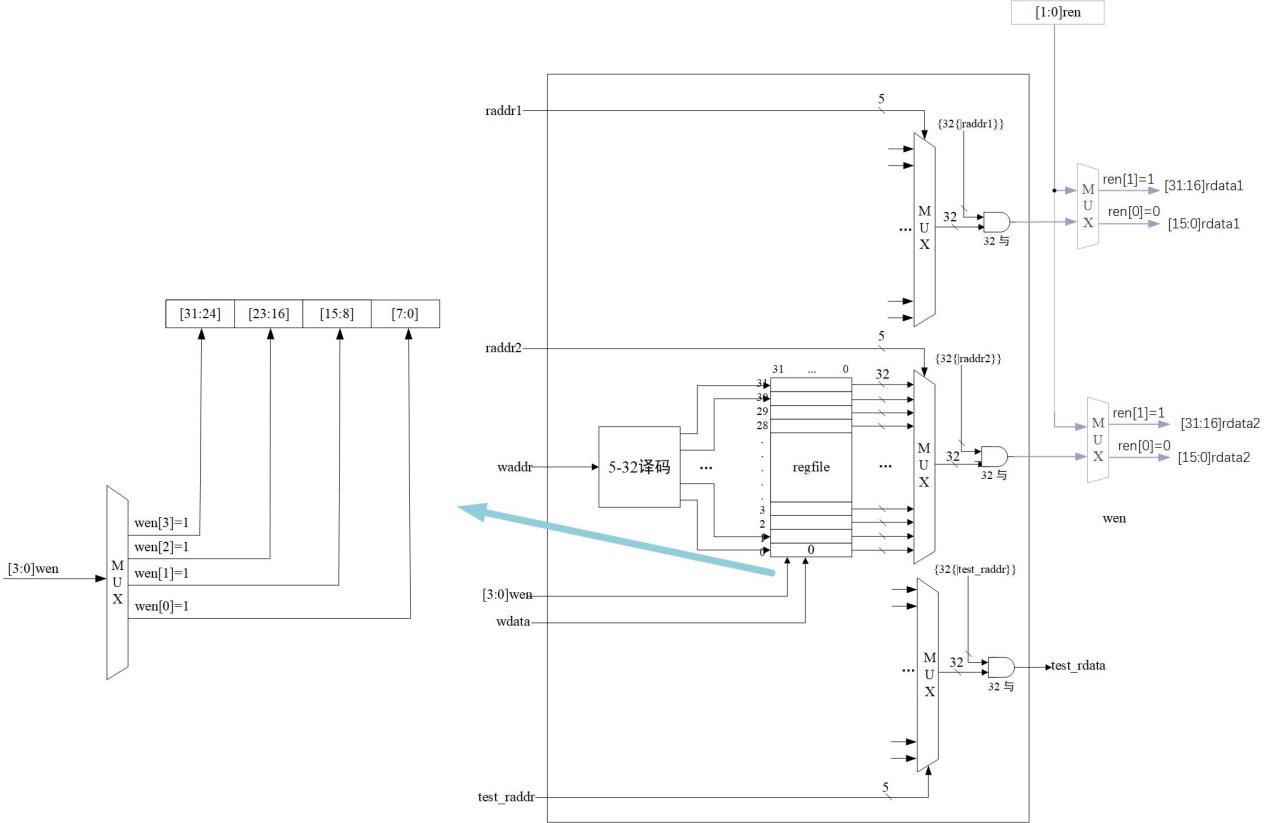
**组成原理实验课程第 三 次实报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验名称 | 寄存器堆的实现 | | | 班级 | 李涛 |
| 学生姓名 | 李佳豪 | 学号 | 2111252 | 指导老师 | 董前琨 |
| 实验地点 | A306 | | 实验时间 | 2023.4.18 | |

1. **实验目的**
2. 熟悉并掌握MIPS计算机中寄存器堆的原理和设计方法。
3. 初步了解MIPS指令结构和源操作数/目的操作数的概念。
4. 熟悉并运用verilog语言进行电路设计。
5. 为后续设计cpu的实验打下基础
6. **实验内容说明**

学习寄存器堆的概念，实验verilog语言在ViVado上实现寄存器堆并上实验箱实验；此外在原有寄存器堆代码的基础上，实现将输入数据按BYTE输入（原数据32位），并使用四个开关控制；读出数据指定寄存器高低16位的值，使用两个开关分别控制读取read1和read2的高低位。

1. **实验原理图**

写入数据时，根据wen的四位输入决定将wdata（8位）写入指定寄存器，右上角是读出数据，根据ren的取值，来决定输出指定寄存器的高低位。

1. **实验步骤**
2. regfile.v 文件中将寄存器堆改变wen位4位，增加两位ren输入， 并调整相关操作逻辑
3. 输入部分

wen修改为4位，控制输入的位置；增加ren接口，控制输出的高低位；将写入wdata改为8位

module regfile(

input clk,

input [3:0] wen,

input [1:0] ren,

input [4 :0] raddr1,

input [4 :0] raddr2,

input [4 :0] waddr,

input [7:0] wdata,

1. 逻辑部分

wen[0]输入最低8位、wen[1]输入次低8位

wen[2]输入次高8位、wen[3]输入最高8位

always @(posedge clk)

begin

if(wen[0] ==1)

begin

rf[ waddr ][7:0] <= wdata ;

end

if (wen[1] == 1)

begin

rf[waddr][15:8] <= wdata ;

end

if (wen[2] == 1)

begin

rf[waddr][23:16] <= wdata ;

end

if (wen[3] == 1)

begin

rf[waddr][31:24] <= wdata ;

end

end

ren = 11全部输出、 ren = 10输出高16位

ren = 01输入低16位、 ren = 00输出32’d0

//读取rdata2高低位置

always @(\*)

begin

if (ren==2'b10)

begin

rdata2 <= rf[raddr2][31:16];

end

if (ren==2'b01)

begin

rdata2 <= rf[raddr2][15:0];

end

if (ren==2'b11)

begin

rdata2 <= rf[raddr2];

end

else

begin

rdata2 <= 32'd0;

end

end

//读取rdata1高低位置

always @(\*)

begin

if (ren==2'b10)

begin

rdata1 <= rf[raddr1][31:16];

end

if (ren==2'b01)

begin

rdata1 <= rf[raddr1][15:0];

end

if (ren==2'b11)

begin

rdata1 <= rf[raddr1];

end

else

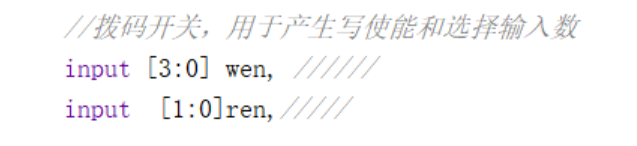
begin

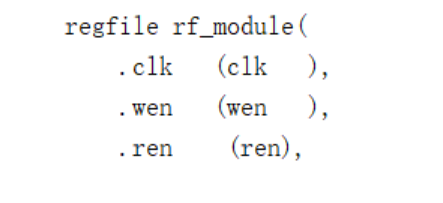
rdata1 <= 32'd0;

end

end

1. regfile.display 文件中修改相应输入、led接口、实例化等





1. refile.xdc 文件中修改对应开关接口以及删除多余led灯

拨码开关从左向右依次控制写入位置的wen、控制当前操作方式的input\_select、控制读高低位的ren。

#led灯连接，用于输出

#set\_property PACKAGE\_PIN K23 [get\_ports led\_wen1]

#set\_property PACKAGE\_PIN F7 [get\_ports led\_wen2]

#set\_property PACKAGE\_PIN G8 [get\_ports led\_wen3]

#set\_property PACKAGE\_PIN J21 [get\_ports led\_wen4]

#set\_property PACKAGE\_PIN H7 [get\_ports led\_wen]

set\_property PACKAGE\_PIN D5 [get\_ports led\_waddr]

set\_property PACKAGE\_PIN A3 [get\_ports led\_wdata]

set\_property PACKAGE\_PIN A5 [get\_ports led\_raddr1]

set\_property PACKAGE\_PIN A4 [get\_ports led\_raddr2]

#拨码开关连接，用于输入，依次为sw0,sw1,sw7

set\_property PACKAGE\_PIN AC21 [get\_ports wen[0]]

set\_property PACKAGE\_PIN AD24 [get\_ports wen[1]]

set\_property PACKAGE\_PIN AC22 [get\_ports wen[2]]

set\_property PACKAGE\_PIN AC23 [get\_ports wen[3]]

set\_property PACKAGE\_PIN AA7 [get\_ports input\_sel[1]]

set\_property PACKAGE\_PIN Y6 [get\_ports input\_sel[0]]

set\_property PACKAGE\_PIN AB6 [get\_ports ren[1]]

set\_property PACKAGE\_PIN W6 [get\_ports ren[0]]

set\_property IOSTANDARD LVCMOS33 [get\_ports clk]

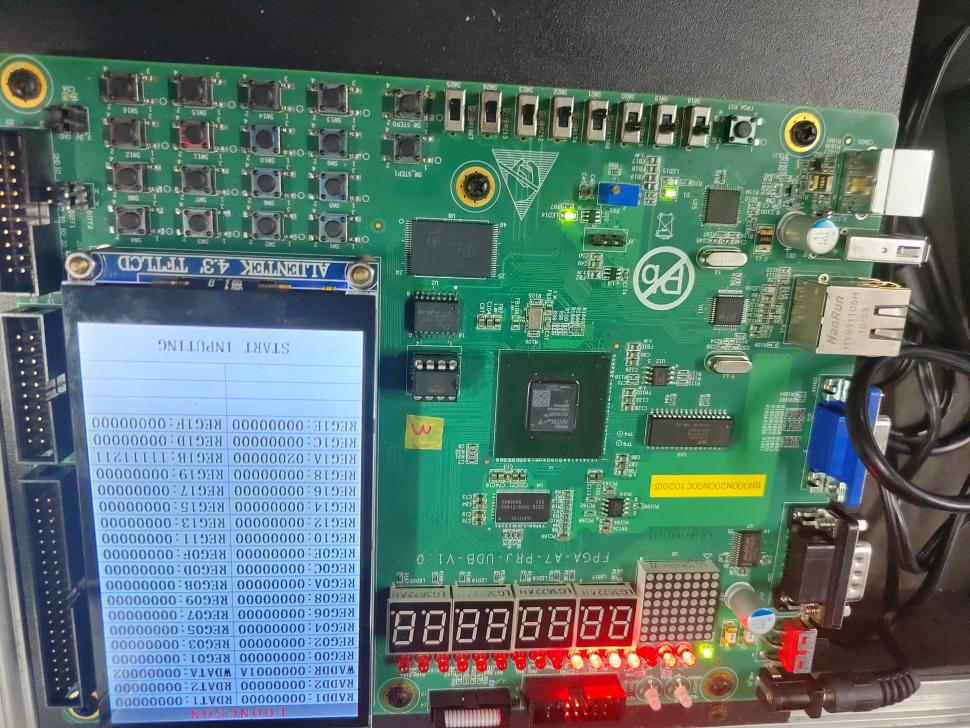
set\_property IOSTANDARD LVCMOS33 [get\_ports resetn]

# set\_property IOSTANDARD LVCMOS33 [get\_ports led\_wen]

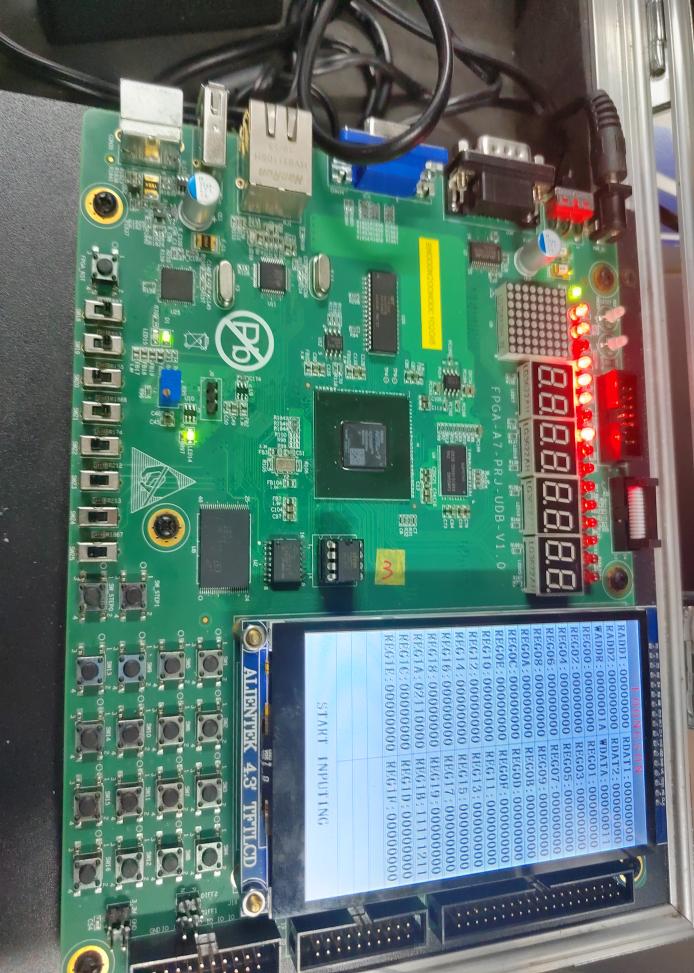
1. **实验结果分析**
2. **输入部分，向1A写入0251125DD**

输入最高2位：将wen的四个开关置为0（即前四个开关向上），input\_sel置10（即开关56置下上），向WADDR写入1A（即向1A写入数据），再将56置为11，紧接着四次写入代写数据。

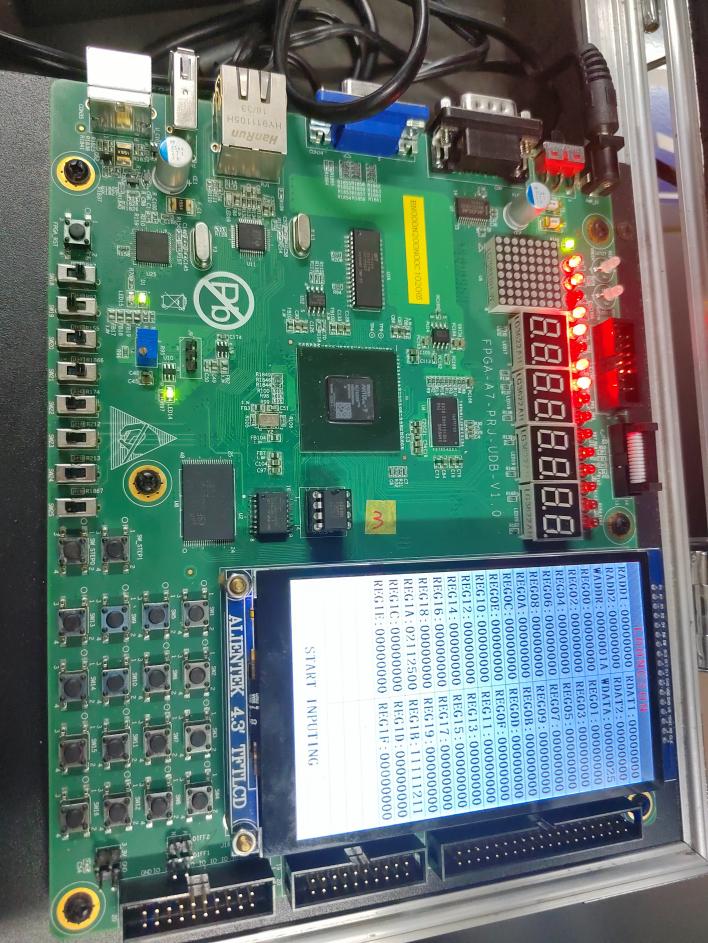
1. 写入02，拨动开关4向下，将02写入，如图：



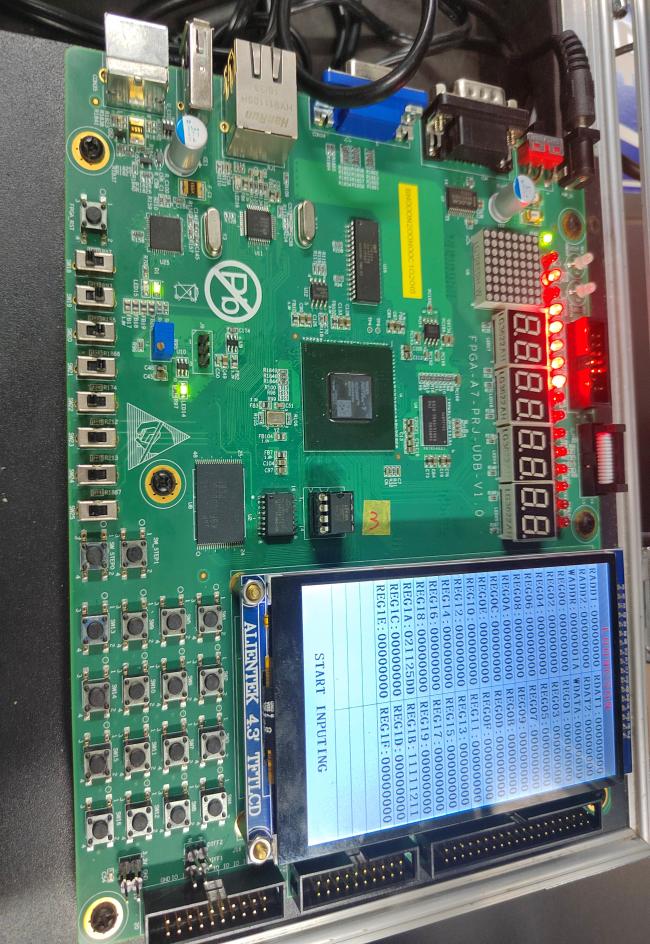
1. 写入11，拨动开关3向下，将11写入，如图：



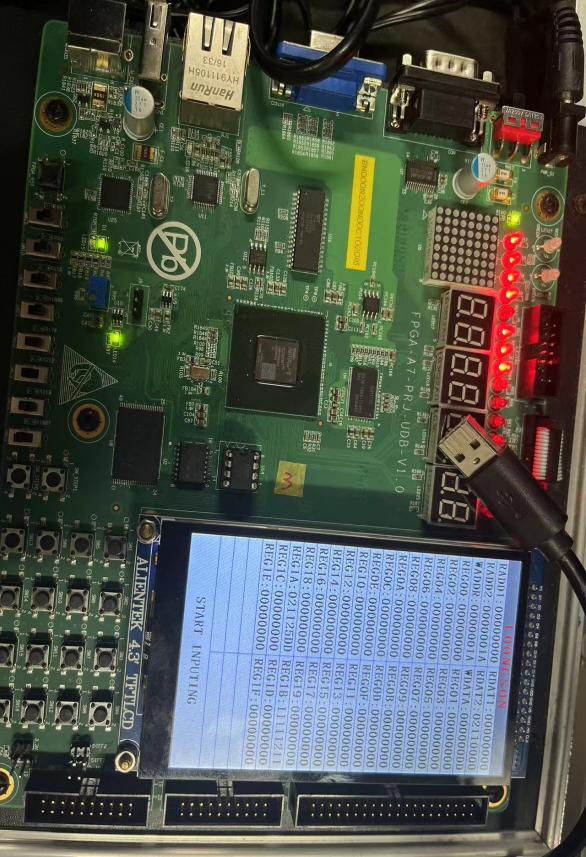
1. 写入25，拨动开关2向下，将25写入，如图：



1. 写入DD，拨动开关1向下，将DD写入，如图：



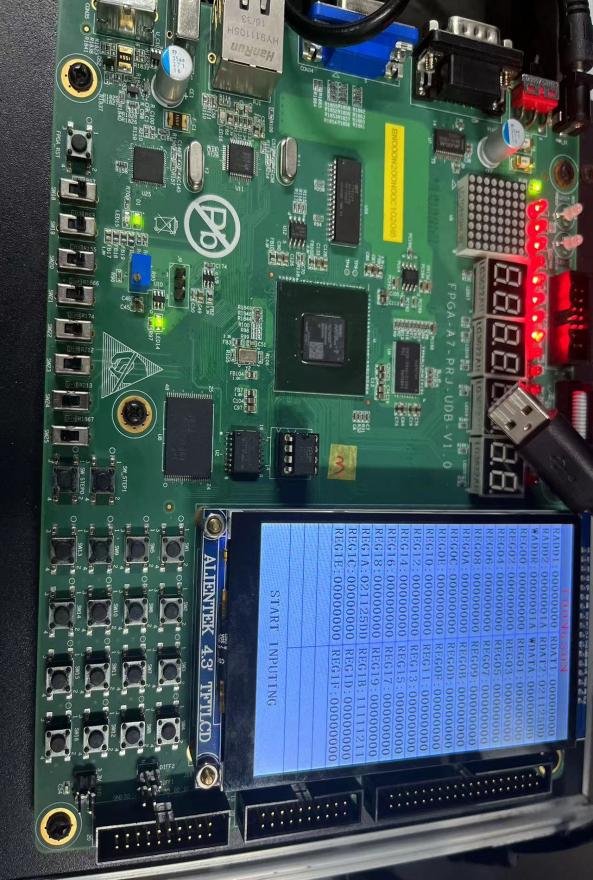
1. **读数据**
2. 读取1A的高16位，如图



1. 读取1A的低16位，如图



1. 读取1A的全32位，如图



1. **总结感想**
2. 对于理论课学习的寄存器堆的概念更加清晰，记忆更加深刻，对verilog编写寄存器堆代码有了认知，此外在修改原代码使之可以按4个高低位置输入时，并在对verilog语言不熟悉的基础上，经历了诸多难题，对这门语言的运用更熟练
3. 对我动手操作实验提升许多，在开始得到代码上箱操作时，连基本操作都很凌乱，更何况按要求还要增加4个开关。所以先认真审计代码，读懂inp\_sel的作用以及使用方法（00、01、10、11分别读、写目的寄存器、写数据），读懂wen为1时改变寄存器的值...最后再加上修改后的开关和逻辑，共8开关，慢慢最后得到实验结果。