计算机组成原理 实验二报告

姓名：曾郅琛　学号：PB20071431　实验日期：2022-3-31

一、实验题目：

Lab02 寄存器堆与存储器及其应用

二、实验目的：

了解、掌握寄存器堆（Register File） 和存储器的功能、 时序及其应用；掌握数据通路和控制器的设计和描述方法。

三、实验平台：

Vivado

四、实验过程：

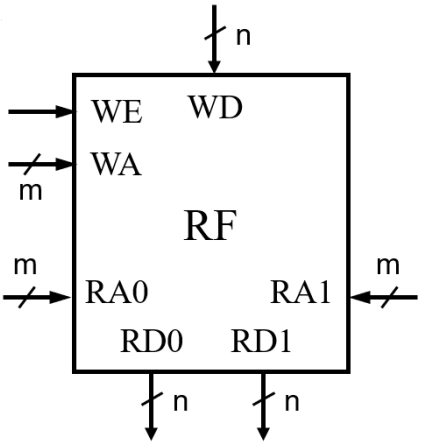
A. 寄存器堆

1. 逻辑设计

三端口的 2^m ╳ n 位寄存器堆

1 个写端口 WA：写地址 WD：写入数据 WE：写使能

2 个读端口 RA0、RA1：读地址 RD0、RD1：读出数据



2. 代码展示

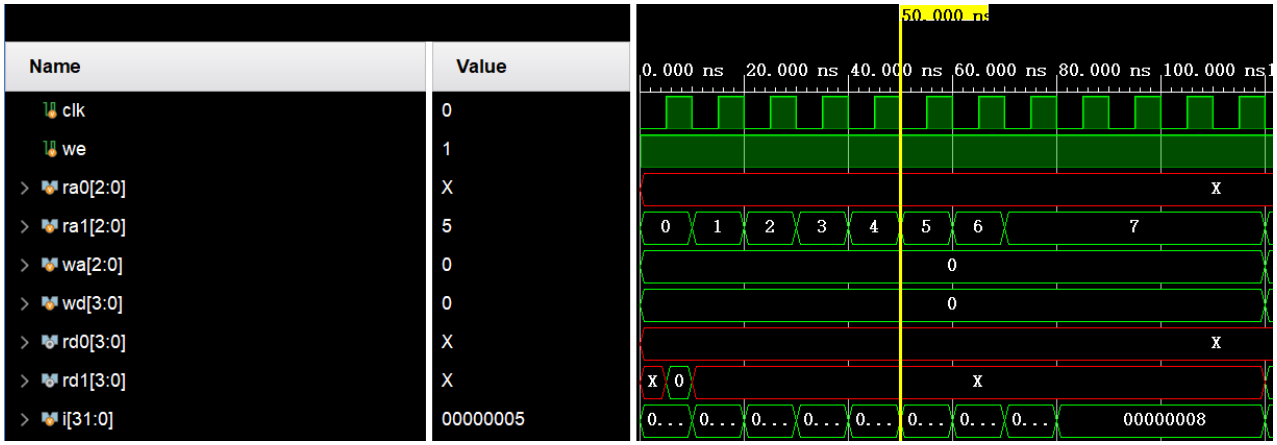
```
1. `timescale 1ns/1ns
2. module register_file
3.  #(parameter
4.     WIDTH = 32
5.     )
6.     (
7.     input clk,
8.     input [2:0] ra0, ra1,
9.     output [WIDTH - 1:0] rd0,
10.    output [WIDTH - 1:0] rd1,
```

```
    input [2:0] wa,
    input we,
    input [WIDTH - 1:0] wd
    );
    reg [WIDTH - 1:0] regfile[0: 7];
    assign rd0 = regfile[ra0];
    assign rd1 = regfile[ra1];

    always @(posedge clk) begin
        if (we)
            regfile[wa] = wd;
    end
endmodule
```

3. 仿真模拟

对寄存器堆进行波形模拟：



B. 存储器 IP 核

1. 分布式

```
    dist_mem_gen_0   distributed (
    .a(a),          // input wire [15 : 0] a
    .d(d),          // input wire [11 : 0] d
    .dpra(dpra),    // input wire [15 : 0] dpra
    .clk(clk),      // input wire clk
    .we(we),        // input wire we
    .dpo(dpo)      // output wire [11 : 0] dpo
);
```

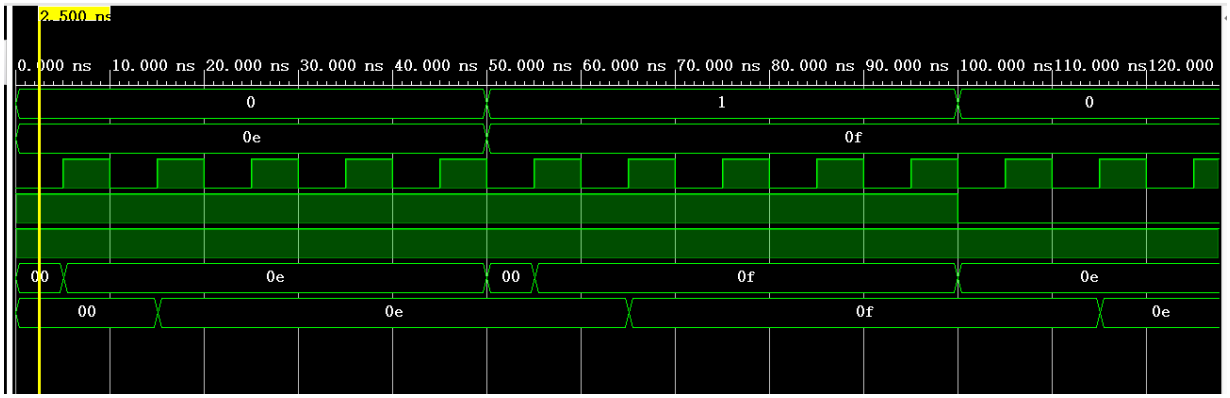
2. 块式

```
blk_mem_gen_0 block (
.clka(clka), // input wire clka
.ena(ena), // input wire ena
.wea(wea), // input wire [0 : 0] wea
.addra(addra), // input wire [3 : 0] addra
```

```
6.  .dina(dina), // input wire [7 : 0] dina
7.  .douta(douta) // output wire [7 : 0] douta
8.  );
```

3.仿真波形

将两者波形对比仿真：



可以看出分布式和块式读写过程大致相同，高电平时，输入的地址被当作写地址，输出的信号为写入的数据；低电平时读写相反。但在读写时间周期上有一定区别。

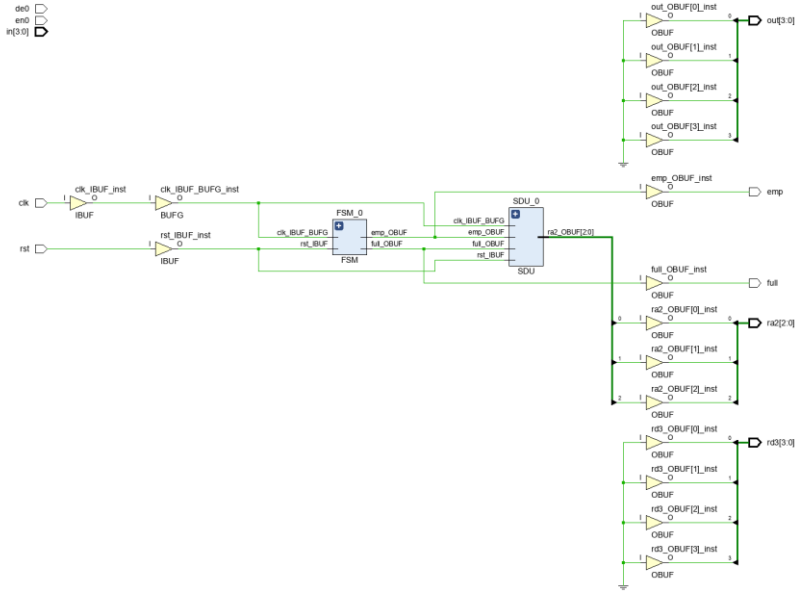
具体如下：

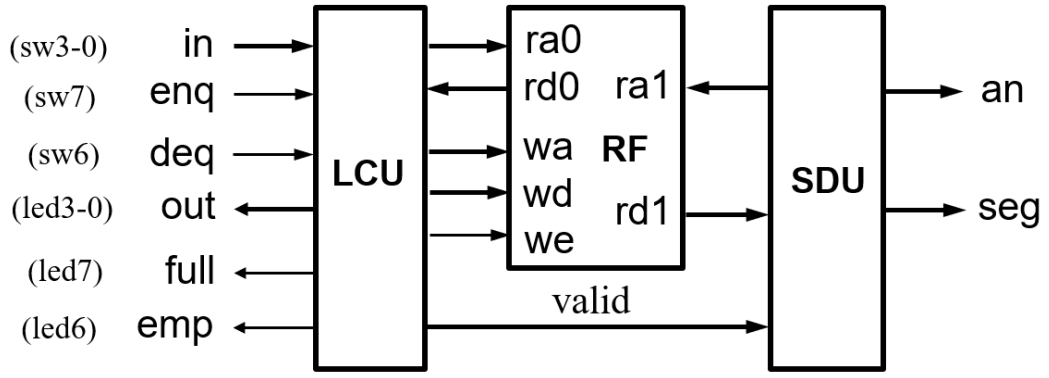
分布式为异步读， 块式为同步读。

分布式 RAM 和块式都是同步写， 但分布式要快一个时钟周期。

C.FIFO

1.数据通路





LCU:

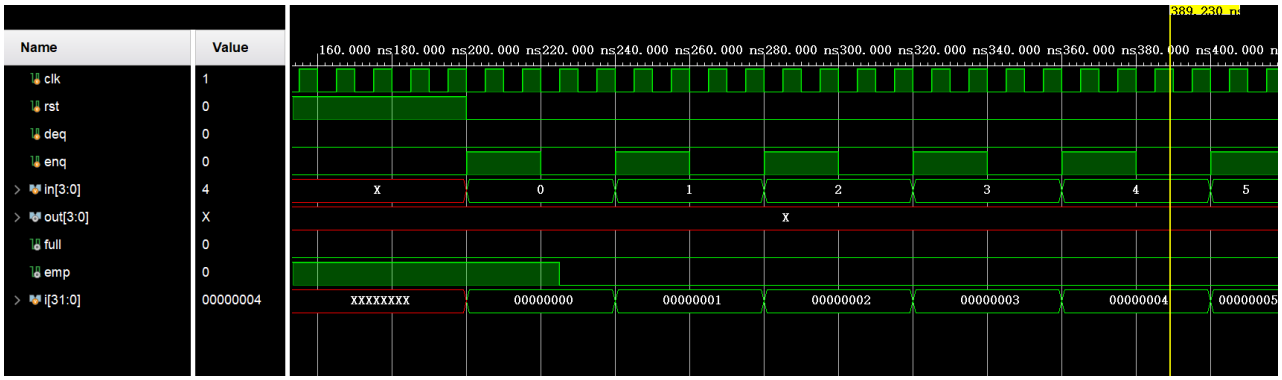
```
always@(posedge clk or posedge rst) begin
    if(rst) cs<=0;
    else cs<=ns;
end

always@(*)begin
case(cs)
    s0: begin if(en_edge) ns<=s1;
              else ns<=s0;end
    s1: begin if(en_edge) begin
        if(((tail+1)==head)||(tail==7&&head==0)) ns<=s2;
                  else ns<=s1;end
              else if(de_edge) begin
                   if(((head+1)==tail)||(head==7&&tail==0))
                       ns<=s0;
                   else ns<=s1;end
              else ns<=s1;
        end
    s2: begin if(de_edge) ns<=s1;
              else ns<=s2;end
    default: ns<=s0;
endcase
end
always@(posedge clk)begin
case(cs)
    s0: begin if(en_edge)
begin tail<=1;head<=0;empty<=0;valid[0]<=1;end
              else begin tail<=0;head<=0;empty<=1;full<=0;val
id<=0;end
        end
    s1: begin if(en_edge) begin tail<=tail+1;head<=head;valid
[tail]<=1;end
              else if(de_edge) begin head<=head+1;tail<=tail;
valid[head]<=0;end
        end
    s2: begin if(de_edge) begin head<=head+1;tail<=tail;full<
=0;valid[head]<=0;end
              else begin head<=head;tail<=tail;full<=1;valid<
=valid;end
        end
    default: begin head<=head;tail<=tail;valid<=valid;end
endcase
end
```

SDU：

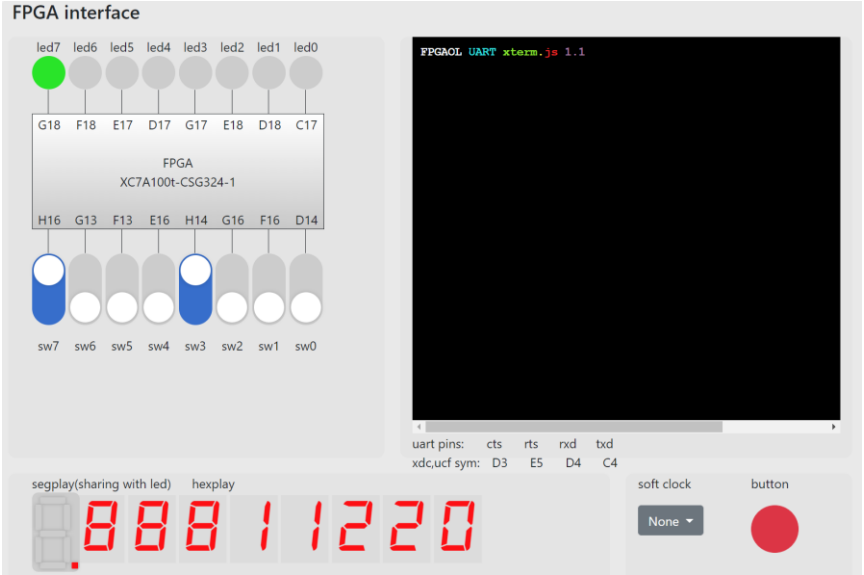
```
1. initial begin
2. cnt<=0;clk_cnt<=0;display<=0;
3. end
4.
5. always@(posedge clk or posedge rst)begin
6.     if(rst) begin cnt<=0;clk_cnt<=0;end
7.     else begin
8.          if(cnt==100000) cnt<=0;
9.          else cnt<=cnt+1;
10.         if(cnt>100000/2) clk_cnt<=1;
11.         else clk_cnt<=0;
12.         end
13.end
14.
15.always@(posedge clk_cnt or posedge rst)begin
16.    if(rst) display<=0;
17.    else display<=display+1;
18.end
19.
20.assign ra1=display;
21.
22.always@(posedge clk)begin
23.    if(valid==0) begin
24.        seg<=0;an<=0;
25.        end
26.    else if(valid[display])begin
27.            seg<=rd1;an<=display;
28.            end
29.End
```

3.波形仿真

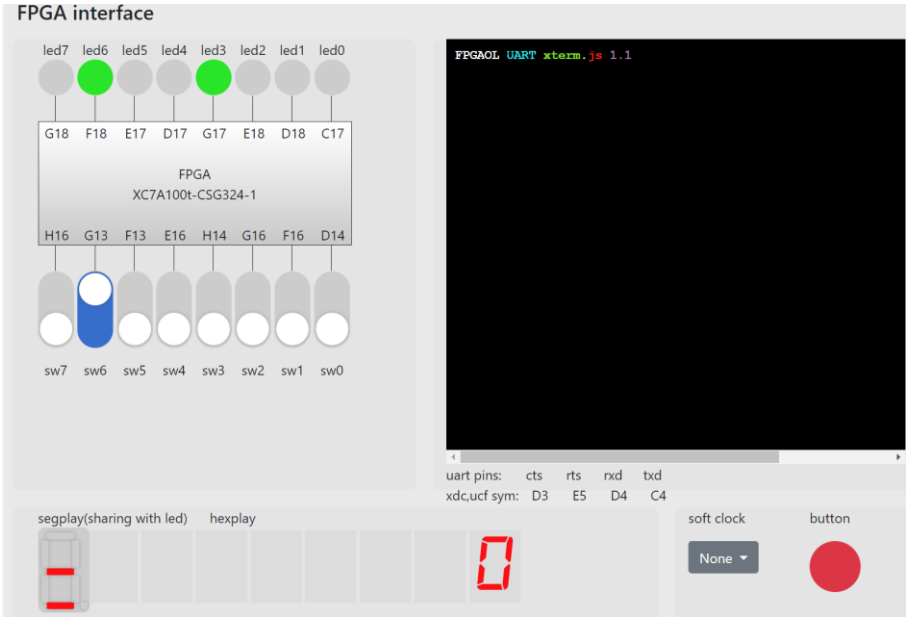


4. FPGA 烧板

进队到满：



出队到空：



六、心得体会：

本次实验完成寄存器堆比较容易解决，IP 核那块做了很久，因为不太理解这种存储，还有 vivado 第一次用这个，FIFO 代码段也很难写，后来问同学还有参考其他组助教给同学的讲解最后才算完成。