《ＥＤＡ技术》试验

实验八
复杂系统设计与实现 2

专业：　　计算机科学与技术

姓名：　　高星杰

学号：　　2021307220712

班级：　　计科 2102

报告上交时间：2023 年　5 月 23 日

一、实验目的

1、复杂系统设计与实现

二、实验任务及要求

任务一：采用字扩展方式将 4 片 256*8 位的 RAM 接成一个 1024*8 位的 RAM。

要求：波形仿真或用 signaltap 调试结果。

提示：可以先用宏功能模块实现 256*8 的 RAM（也可以用 verilog 语言自己写一个 256*8 的 RAM）,然后再用字扩展方式实现 1024*8 的 RAM.

任务二：附加题（具体题目请参加页面最后的附加题）

由于实验七已经写了两个附加题所以本次从附加题第三题中开始写

附加题 3：路口交通灯控制器设计与实现。A 方向红灯 55s、黄灯 5s、绿灯 40s，B 方向红灯 45s，黄灯 5s，绿灯 50s；

三、实验原理与步骤

任务一：采用字扩展方式将 4 片 256*8 位的 RAM 接成一个 1024*8 位的 RAM。

实验原理：首先写了 256*8 的 RAM，包括 A(地址)，CS（片选信号），RW（写使能信号），IO（总线），dataIN（数据输入），实现了读取和写入的功能。然后再进行字拓展，将 A（地址）的前两位判断产生片选信号，选中一片 ram 再将数据和总线传入，最后就实现了功能。

Verilog 代码：

1024*8 位的 RAM：

```
module task1(A,IO,RW,dataIN);
input [9:0]A;
input [7:0]dataIN;
output reg [7:0]IO;
input RW;
reg [3:0]Y;
wire  [7:0]IO1,IO2,IO3,IO4;
always@(A)
begin
   case({A[9],A[8]})
   2'b00:begin Y<=4'b1110; IO<=IO1;end
   2'b01:begin Y<=4'b1101;IO<=IO2;end
   2'b10:begin Y<=4'b1011;IO<=IO3;end
   2'b11:begin Y<=4'b0111;IO<=IO4;end
   endcase
end
ram ram1(A[7:0],RW,Y[0],IO1,dataIN);
ram ram2(A[7:0],RW,Y[1],IO2,dataIN);
ram ram3(A[7:0],RW,Y[2],IO3,dataIN);
ram ram4(A[7:0],RW,Y[3],IO4,dataIN);
endmodule
```

256*8 Ram：

```
module ram(A,RW,CS,IO,dataIN);
input[7:0]A;
input RW,CS;
output reg [7:0]IO;
input [7:0]dataIN;
reg [7:0]data[255:0];
always@(CS)
begin
   if(CS==0)
   begin
      if(RW==1)
      begin
         data[A]<=dataIN;
      end
      IO<=data[A];
   end
end
endmodule
```

附加题 3：路口交通灯控制器设计与实现。A 方向红灯 55s、黄灯 5s、绿灯 40s，B 方向红灯 45s，黄灯 5s，绿灯 50s；

实现原理：使用分频器产生基本的时间 1s 累加，100s 一循环，然后

再根据不同方向的信号灯的特点显示，由于板子上没有那么多颜色，所以我对颜色进行编码以代替颜色。红色是 1，黄色是 2，绿色是 3，然后一个方向上 2 数码管上显示倒计时，1 个数码管显示颜色的编码。总共使用了 6 个数码管

Verilog 代码：

总程序：

```
module task3(clk,hex0,hex1,hex2,hex3,hex4,hex5);
(*chip_pin="Y2"*)input clk;
(*chip_pin="H22,J22,L25,L26,E17,F22,G18"*)output[6:0]hex0;
(*chip_pin="U24,U23,W25,W22,W21,Y22,M24"*)output[6:0]hex1;
(*chip_pin="W28,W27,Y26,W26,Y25,AA26,AA25"*)output[6:0]hex2;
(*chip_pin="Y19,AF23,AD24,AA21,AB20,U21,V21"*)output[6:0]hex3;
(*chip_pin="AE18,AF19,AE19,AH21,AG21,AA19,AB19"*)output[6:0]hex4;
(*chip_pin="AH18,AF18,AG19,AH19,AB18,AC18,AD18"*)output[6:0]hex5;
reg [35:0]num;
reg [6:0]sec;
always@(posedge clk)
begin
   if(sec<0)sec<=100;
   num=(num+1)%50000000;
   if(num==1)begin
      sec<=(sec-1)%100;
   end
end
showSEC1(sec,hex0,hex1,hex2);
showSEC2(sec,hex3,hex4,hex5);
endmodule
```

A 方向显示模块：

```
module showSEC1(sec,hex0,hex1,hex2);
task show;//根据in输出信号
   input [6:0]in;
   output [6:0]hex;
   begin
      case(in)
      6'b000000:hex<=7'b1000000;
      6'b000001:hex<=7'b1111001;
      6'b000010:hex<=7'b0100100;
      6'b000011:hex<=7'b0110000;
      6'b000100:hex<=7'b0011001;
      6'b000101:hex<=7'b0010010;
      6'b000110:hex<=7'b0000010;
      6'b000111:hex<=7'b1111000;
      6'b001000:hex<=7'b0000000;
      6'b001001:hex<=7'b0010000;
      endcase;
   end
endtask
input [6:0]sec;
output [6:0]hex0;
output [6:0]hex1;
output [6:0]hex2;
reg [3:0] shi,ge;
reg [3:0] color;
parameter red=6'b000001,yellow=6'b000010,green=6'b000011;
always@(sec)
```

```
parameter red=6'b000001,yellow=6'b000010,green=6'b000011;
always@(sec)
begin
    if(sec<=55)
    begin
    shi<=sec/10;
    ge<=sec%10;
    color<=red;
    end
    else if(sec>55&&sec<=60)
      begin
      shi<=0;
      ge<=sec-55;
      color<=yellow;
      end
    else begin
      shi<=(sec-60)/10;
      ge <=(sec-60)%10;
      color<=green;
      end
    show(shi,hex1);
    show(ge,hex0);
    show(color,hex2);
end
endmodule
```

B 方向显示模块：

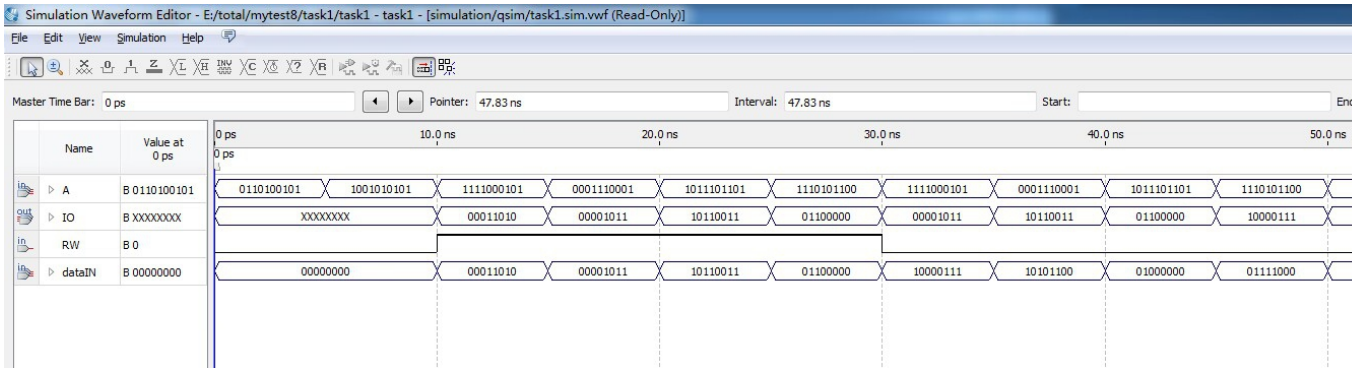
```
module showSEC2(sec,hex3,hex4,hex5);
task show;//根据in输出信号
    input [6:0]in;
    output [6:0]hex;
    begin
        case(in)
        6'b000000:hex<=7'b1000000;
        6'b000001:hex<=7'b1111001;
        6'b000010:hex<=7'b0100100;
        6'b000011:hex<=7'b0110000;
        6'b000100:hex<=7'b0011001;
        6'b000101:hex<=7'b0010010;
        6'b000110:hex<=7'b0000010;
        6'b000111:hex<=7'b1111000;
        6'b001000:hex<=7'b0000000;
        6'b001001:hex<=7'b0010000;
        endcase;
    end
endtask
input [6:0]sec;
output [6:0]hex3;
output [6:0]hex4;
output [6:0]hex5;
reg [3:0] shi,ge;
reg [3:0] color;
parameter red=6'b000001,yellow=6'b000010,green=6'b000011;
always@(sec)
```

```
parameter red=6'b000001,yellow=6'b000010,green=6'b000011;
always@(sec)
begin
    if(sec<=45)
    begin
    shi<=sec/10;
    ge<=sec%10;
    color<=red;
    end
    else if(sec>45&&sec<=50)
      begin
      shi<=0;
      ge<=sec-45;
      color<=yellow;
      end
    else begin
      shi<=(sec-50)/10;
      ge <=(sec-50)%10;
      color<=green;
      end
    show(shi,hex4);
    show(ge,hex3);
    show(color,hex5);
end
endmodule
```

四、实验结果与分析

任务一：采用字扩展方式将 4 片 256*8 位的 RAM 接成一个 1024*8 位的 RAM。

波形仿真：



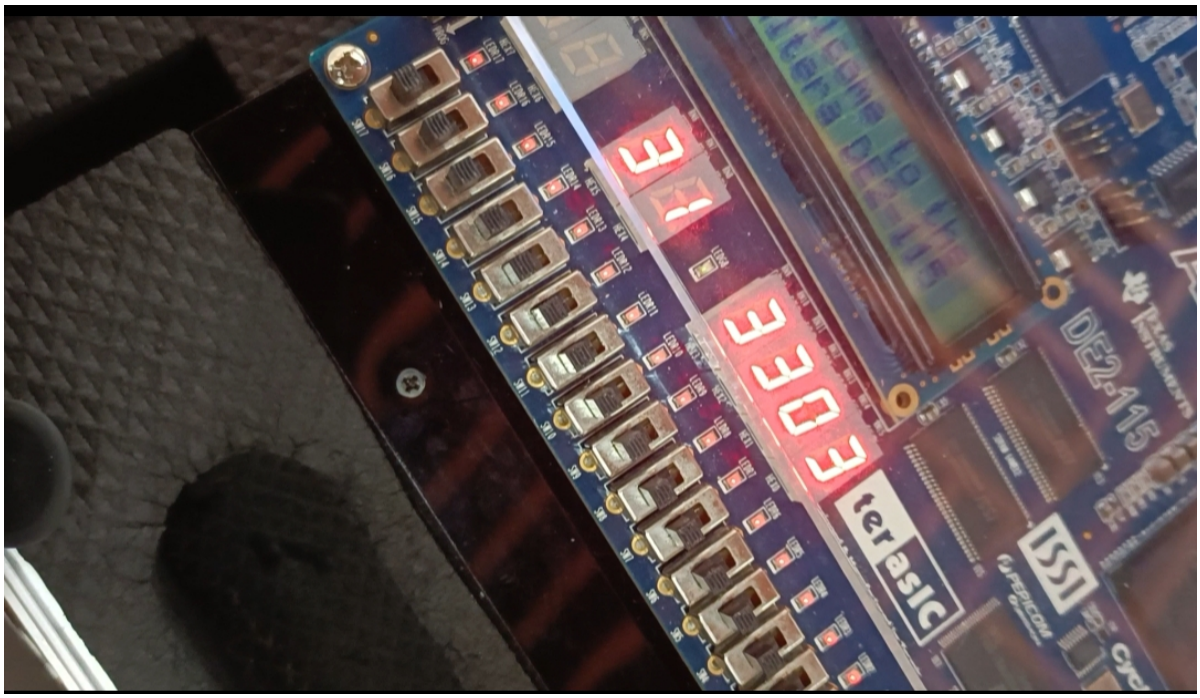
可以看到在写使能信号为1有效将数据00011010写入了1111000101地址的内存，将信号改为0再写入后读取，发现数据没有变化仍为00001011，其他地址的内存也如此可以从图中看出，所以任务设计成功。

附加题 3：路口交通灯控制器设计与实现。A 方向红灯 55s、黄灯 5s、绿灯 40s，B 方向红灯 45s，黄灯 5s，绿灯 50s；

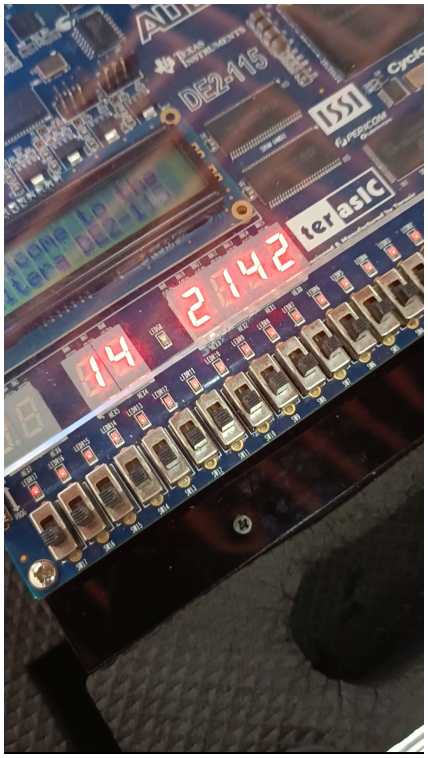
开发板结果：

绿灯时相差 10s

13s 和 3s



到红灯时候小于 45s 后倒计时相同：



根据开发板上显示结果可以得出结论设计成功。

五、实验体会与讨论

本次又学会用新的工具，task 相当于原本的函数，可以省去一大部分的重复代码，使程序更加简洁，也更符合结构化设计的原则，使程序的结构分明，更加易读。

不知不觉实验课程已经结束了，这八次实验已经让我的语法逐渐趋于完善和熟练，直接的写程序已经基本没报错了，行为描述，数据流描述，结构描述，都使用过都基本掌握。可以说本学期的 eda 实验还是挺有趣的，不仅学习了知识，也锻炼了编程的思维，同时也为计算机组成原理的实验打下了坚实的基础，尤其使最终的 cpu 设计中，发挥了举足轻重的作用。