中山大学数据科学与计算机学院本科生实验报告

（2019 学年秋季学期）

课程名称：**计算机组成原理实验**　　任课教师：郭雪梅　　助教：汪庭葳、刘洋旗

年级&班级	2018 级计科一班	专业(方向)	计算机类
学号	18340013	姓名	陈琮昊
电话	15734867907	Email	1062080021@qq.com
开始日期	2019.11.13	完成日期	2019.11.14

一、实验题目：

实验 6-存储器实验

二、实验目的：

1. 掌握存储器的设计原理，并且自己设计一个大小合适的 ROM。
2. 掌握存储器的存储和取数的过程，并将存储器的内容通过数码管显示出来。
3. 学会封装 IP 核以及调用方法。

三、实验内容

1. 实验步骤：

（I）新建工程文件建立存储器系统并仿真

i.添加调用 rom IP 核

ii.添加 design source(此过程需要将 ROM 的实例化代码放入该文件)

iii.功能仿真

（II）加入数码管显示，修改文件

i.添加数码管设计源文件文件 display

ii.在 rom_top 文件中添加调用数码管显示所读出的 rom 内容

iii.添加约束文件

iv.Generate Bitstream

v.烧板

2. 实验原理：本次实验主要是模拟数据在存储器中的存取过程，先初始化 ROM 存储器中的内容，再通过开关选择相应的地址，将对应的存储器中内容读出来，并通过显示管显示。实验的原理图如下图 1 所示：

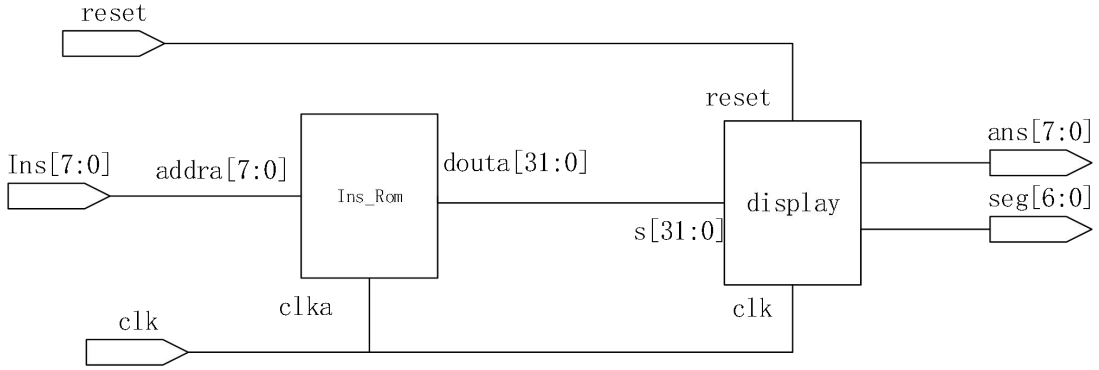


图 1 存储器原理图

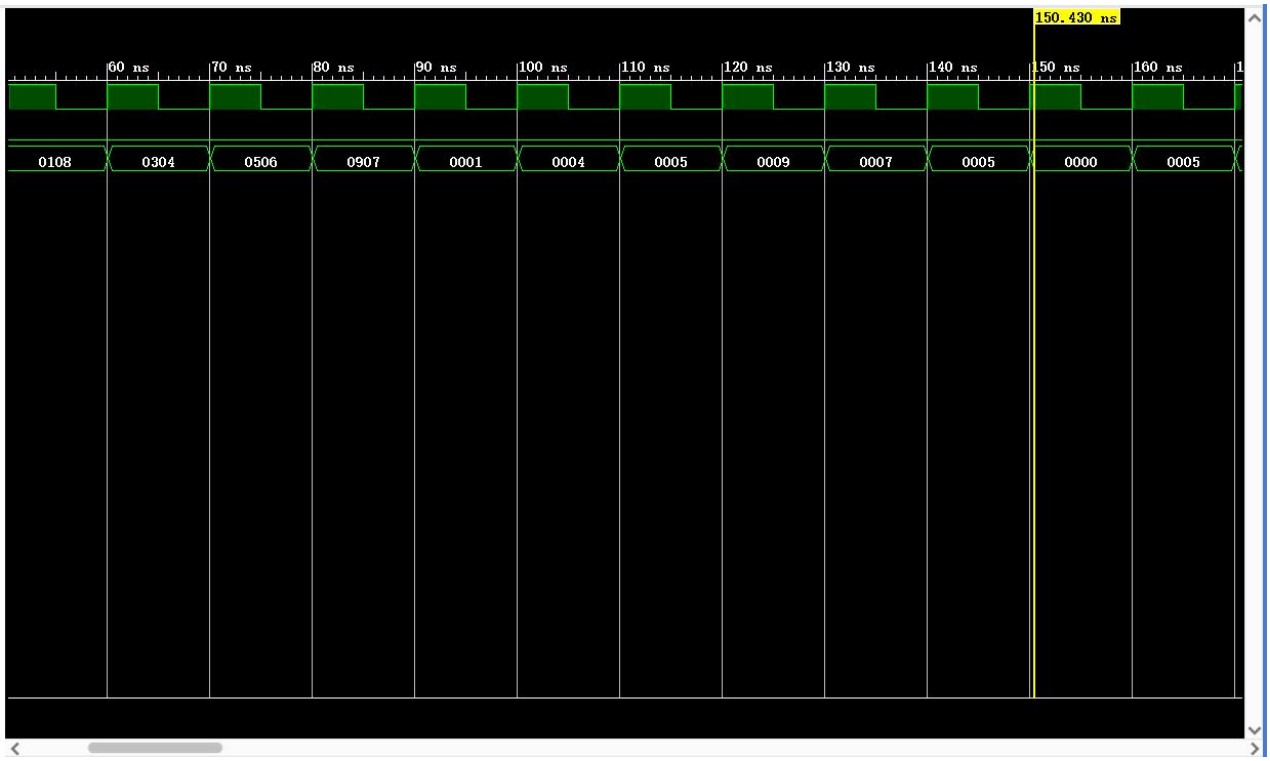
我们使用的 IP 核，ins_ROM 核是系统自带的，通过该 IP 核我们实现数据的存取，通过 8 位地址开关的选择，将 ROM 中对应的 16 位数据取出来并送往 display。通过 display 可以将 ROM 中取出的 16 位数据在数码管上显示出来。

存储容量是指存储器可以容纳的二进制信息量，用存储器中存储地址数与存储字位数的乘积表示。

四、实验结果：

（1）仿真结果如下：

该图为一部分的截图。可以看到是按照 coe 文件的内容显示的，故仿真成功。



（2）烧板结果：见文件夹附件视频。

五、实验感想：本次实验主要是设计存储器，在有了上一次的经验后这次实验明显顺利许多，在创建 IP 核时有好多注意的点需要细心，一旦疏忽到后面就会转化为大的错误，那时候再 Debug 就很难了；还有就是烧板的时候要控制频率，因此需要添加一段代码来确定数码管显示时的频率，根据老师的提议选择四分频最好，添加代码后就能够显示在板子上并顺利观察了。

附录：

由于仿真所需文件与烧板所需文件不同，故给出两部分的代码。

（1）仿真所需文件：

（i）prgmip16.coe 文件：

```
memory_initialization_radix=16;
memory_initialization_vector=1834,0013,
0108,0304,
0506,0907,
0001,0004,
0005,0009,
0007,0005,
0000,0005,
8888,8888,
9999,9999,
Aaaa,aaaa,
Bbbb,bbbb,
1234,5678,
12ab,cdef;
```

（ii）设计源文件：

```
module rom_top(Clk,Rst,data);
  input Clk;//系统时钟
  input Rst;//高电平复位
  output [15:0] data;//输出数据
  reg [7:0]addr;//rom 地址
  always@(posedge Clk or negedge Rst)
        if(Rst)
                addr<=8'd0;
          else if(addr == 8'd255)
                  addr<=8'd0;
          else
                addr<=addr+1'b1;
//下面这部分代码来自于 Ins_Rom.veo 文件（即实例化文件）
Ins_Rom rom (
                    .clka(Clk),       // input wire clka
                    .ena(1'b1),          // input wire ena 数据输出允许
                    .addra(addr),    // input wire [7 : 0] addra
                    .douta(data)    // output wire [15 : 0] douta
                  );
```

```
endmodule
```
（iii）仿真文件：
```
`define clk_period 10

module rom_tb;


// Inputs
        reg Clk;
        reg    Rst;
        // Outputs
        wire [15:0] dout;


        rom_top rom_top(
        .Clk(Clk),
        .Rst(Rst),
        .data(dout));
        initial
        begin
        Clk = 1;
        Rst=0;
        end
        always #(`clk_period/2)Clk = ~Clk;
              initial begin
                        Rst=1'b1;
                        #(`clk_period*3);
                          Rst=1'b0;
                          #(`clk_period*1000);


                          $stop;
                    end
endmodule
```
(2)烧板所需文件：

（i）rom_top.v:

```
module rom_top(Clk,Rst,seg,sm_wei);
   input Clk;//系统时钟
   input Rst;//高电平复位
  // output [15:0] data;//输出数据
   output [6:0] seg;//段码
   output [3:0] sm_wei;//哪个数码管
  wire [15:0] data;
  reg [7:0]addr;//rom 地址
  integer clk_cnt;
reg clkin;//1.33 秒一个时钟周期，以此为 cpu 的时钟周期10^8/0.75*10^8=4/3=1.33
always @(posedge Clk)
if(clk_cnt==32'd75_000_000)
begin
clk_cnt <= 1'b0;
clkin <= ~clkin;
end
else
clk_cnt <= clk_cnt + 1'b1;
//数码管显示频率
integer clk_cnt1;
reg clkin1;//1.33 秒一个时钟周期，以此为 cpu 的时钟周期    10^8/0.75*10^8=4/3=1.33
always @(posedge Clk)
if(clk_cnt1==32'd75_000_000)
```

```
begin
clk_cnt1 <= 1'b0;
clkin1 <= ~clkin1;
end
else
clk_cnt1 <= clk_cnt1 + 3'b100;                    //四分频
   always@(posedge clkin or negedge Rst)
           if(Rst)
                      addr<=8'd0;
            else if(addr == 8'd255)
                        addr<=8'd0;
            else
                       addr<=addr+1'b1;
Ins_Rom rom (
                          .clka(clkin),        // input wire clka
                          .ena(1'b1),           // input wire ena
                          .addra(addr),    // input wire [7 : 0] addra
                          .douta(data)    // output wire [15 : 0] douta
                        );
 display dis
                            (.clk(clkin1),
                              .data (data),
                              .sm_duan(seg),
                              .sm_wei(sm_wei));
```

```
endmodule
```

（ii）display.v:

```
module display(clk,data,sm_wei,sm_duan);
input clk;
input [15:0] data;
output [3:0] sm_wei;
output [6:0] sm_duan;
//-------------------------------------------------------
//分频
//位控制
reg [3:0]wei_ctrl=4'b1110; always @(posedge clk)
wei_ctrl <= {wei_ctrl[2:0],wei_ctrl[3]}; //段控制
reg [3:0]duan_ctrl;
always @(wei_ctrl)
case(wei_ctrl)
4'b1110:duan_ctrl=data[3:0];
4'b1101:duan_ctrl=data[7:4];
4'b1011:duan_ctrl=data[11:8];
4'b0111:duan_ctrl=data[15:12];
default:duan_ctrl=4'hf;
  endcase
//-------------------------------------------------------
//解码模块
reg [6:0]duan;
```

```
always @(duan_ctrl)
case(duan_ctrl)
4'h0:duan=7'b100_0000;//0
4'h1:duan=7'b111_1001;//1
4'h2:duan=7'b010_0100;//2
4'h3:duan=7'b011_0000;//3
4'h4:duan=7'b001_1001;//4
4'h5:duan=7'b001_0010;//5
4'h6:duan=7'b000_0010;//6
4'h7:duan=7'b111_1000;//7
4'h8:duan=7'b000_0000;//8
4'h9:duan=7'b001_0000;//9
4'ha:duan=7'b000_1000;//a
4'hb:duan=7'b000_0011;//b
4'hc:duan=7'b100_0110;//c
4'hd:duan=7'b010_0001;//d
4'he:duan=7'b000_0111;//e
4'hf:duan=7'b000_1110;//f
// 4'hf:duan=7'b111_1111;//不显示
default : duan = 7'b100_0000;//0
endcase
//-------------------------------------------------------
assign sm_wei = wei_ctrl;
assign sm_duan = duan;
```

endmodule

（iii）prgmip16.coe:

memory_initialization_radix=16;

memory_initialization_vector=1834,0013,

0108,0304,
0506,0907,
0001,0004,
0005,0009,
0007,0005,
0000,0005,
8888,8888,
9999,9999,
Aaaa,aaaa,
Bbbb,bbbb,
1234,5678,
12ab,cdef;

（iv）约束文件：

```
set_property PACKAGE_PIN U7 [get_ports {seg[6]}]
set_property PACKAGE_PIN V5 [get_ports {seg[5]}]
set_property PACKAGE_PIN U5 [get_ports {seg[4]}]
set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
set_property PACKAGE_PIN U8 [get_ports {seg[2]}]
set_property PACKAGE_PIN W6 [get_ports {seg[1]}]
set_property PACKAGE_PIN W7 [get_ports {seg[0]}]
set_property PACKAGE_PIN U2 [get_ports {sm_wei[0]}]
set_property PACKAGE_PIN U4 [get_ports {sm_wei[1]}]
set_property PACKAGE_PIN V4 [get_ports {sm_wei[2]}]
set_property PACKAGE_PIN W4 [get_ports {sm_wei[3]}]
set_property PACKAGE_PIN W5 [get_ports Clk]
set_property PACKAGE_PIN U18 [get_ports Rst]


set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sm_wei[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sm_wei[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sm_wei[1]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {sm_wei[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports Clk]
set_property IOSTANDARD LVCMOS33 [get_ports Rst]
```

（v）测试文件：

```
`define clk_period 10
module rom_tb;

// Inputs
	reg Clk;
	reg    Rst;
	// Outputs
	wire [15:0] dout;

	rom_top rom_top(
	.Clk(Clk),
	.Rst(Rst),
	.data(dout));
	initial
	begin
	Clk = 1;
	Rst=0;
	end
	always #(`clk_period/2)Clk = ~Clk;
		initial begin
			Rst=1'b1;
			#(`clk_period*3);
			  Rst=1'b0;
			  #(`clk_period*1000);

			  $stop;
		end
endmodule
```