实验报告

实验名称：存储器实验

院系：计算机学院 19 级计算机科学与技术

班级：　计科 2 班

姓名：　施天予

指导老师：陈刚

一、 实验内容

1. 使用 Block Memory Generator 生成单端口 ROM，并将指令 coe 文件加载；
2. 将 ROM 中对应的 32 位指令取出并送往 7-seg 数码管显示。
3. 将地址线接到拨码开关上，验证 ROM 读取的值。

二、实验原理

本次实验使用 Vivado 的 Block Memory Generator 模拟数据在存储器中的存取过程。实验使用单端口 ROM。初始化 ROM 存储器中的内容，通过开关选择相应的地址，将对应的存储器中内容读出来，并通过七段数码管显示。实验原理如图 1 所示：

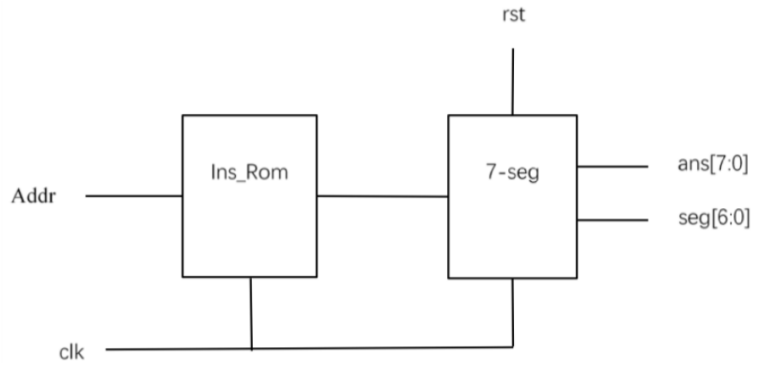


图 1

三、实验设计

设置 sw0-sw7 为 addr 的低到高位

约束文件如下：

```
set_property PACKAGE_PIN W5 [get_ports clk]
    set_property IOSTANDARD LVCMOS33 [get_ports clk]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

set_property PACKAGE_PIN V17 [get_ports {addr[0]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[0]}]
set_property PACKAGE_PIN V16 [get_ports {addr[1]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[1]}]
set_property PACKAGE_PIN W16 [get_ports {addr[2]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[2]}]
set_property PACKAGE_PIN W17 [get_ports {addr[3]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[3]}]
set_property PACKAGE_PIN W15 [get_ports {addr[4]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[4]}]
```

```
set_property PACKAGE_PIN V15 [get_ports {addr[5]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[5]}]
set_property PACKAGE_PIN W14 [get_ports {addr[6]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[6]}]
set_property PACKAGE_PIN W13 [get_ports {addr[7]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {addr[7]}]

#7 segment display
set_property PACKAGE_PIN W7 [get_ports {seg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]
set_property PACKAGE_PIN W6 [get_ports {seg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
set_property PACKAGE_PIN U8 [get_ports {seg[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {seg[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
set_property PACKAGE_PIN V5 [get_ports {seg[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
set_property PACKAGE_PIN U7 [get_ports {seg[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]

set_property PACKAGE_PIN U2 [get_ports {ans[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {ans[0]}]
set_property PACKAGE_PIN U4 [get_ports {ans[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {ans[1]}]
set_property PACKAGE_PIN V4 [get_ports {ans[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {ans[2]}]
set_property PACKAGE_PIN W4 [get_ports {ans[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {ans[3]}]
##Buttons
set_property PACKAGE_PIN U18 [get_ports rst]
        set_property IOSTANDARD LVCMOS33 [get_ports rst]

```

top. v

```
`timescale 1ns / 1ps

module top(
    input clk,
    input rst,
    input [7:0] addr,
    output [3:0] ans, //select for seg
    output [6:0] seg  //segment digital
    );
    wire [31:0] data;
    wire [15:0] data1;
    assign data1=data[15:0];
    Ins_Rom rom(
        .clka(clk),
```

```
        .addra(addr),
        .douta(data)
        );
    display U2(.clk(clk),.reset(rst),.s(data1),.ans(ans),.seg(seg));
endmodule
```

Ins_Rom 例化

```
ENTITY Ins_Rom IS
  PORT (
    clka : IN STD_LOGIC;
    addra : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
    douta : OUT STD_LOGIC_VECTOR(31 DOWNTO 0)
  );
END Ins_Rom;
```

display.v

```
`timescale 1ns / 1ps

module display(
    input wire clk,reset,
    input wire [31:0]s,
    output wire [6:0]seg,
    output reg [3:0]ans
    );
    reg [20:0]count;
    reg [4:0]digit;
    always@(posedge clk,posedge reset)
    if(reset)
        count = 0;
    else
        count = count + 1;

    always @(posedge clk)
    case(count[20:19])
        0:begin
            ans = 4'b1110;
            digit = s[3:0];
        end

        1:begin
            ans = 4'b1101;
            digit = s[7:4];
        end

```

```
        2:begin
            ans = 4'b1011;
            digit =s[11:8];
        end

        3:begin
            ans = 4'b0111;
            digit = s[15:12];
        end
    endcase

    seg7 U4(.din(digit),.dout(seg));
endmodule
```

seg7.v

```
`timescale 1ns / 1ps

module seg7(
    input wire [4:0]din,
    output reg [6:0]dout
    );

    always@(*)
    case(din)
        5'h0:dout = 7'b000_0001;
        5'h1:dout = 7'b100_1111;
        5'h2:dout = 7'b001_0010;
        5'h3:dout = 7'b000_0110;
        5'h4:dout = 7'b100_1100;

        5'h5:dout = 7'b010_0100;
        5'h6:dout = 7'b010_0000;
        5'h7:dout = 7'b000_1111;
        5'h8:dout = 7'b000_0000;
        5'h9:dout = 7'b000_0100;
        5'ha:dout = 7'b000_1000;
        5'hb:dout = 7'b110_0000;
        5'hc:dout = 7'b011_0001;
        5'hd:dout = 7'b100_0010;
        5'he:dout = 7'b011_0000;
        5'hf:dout = 7'b011_1000;
        default:dout = 7'b111_1111;
    endcase
endmodule
```

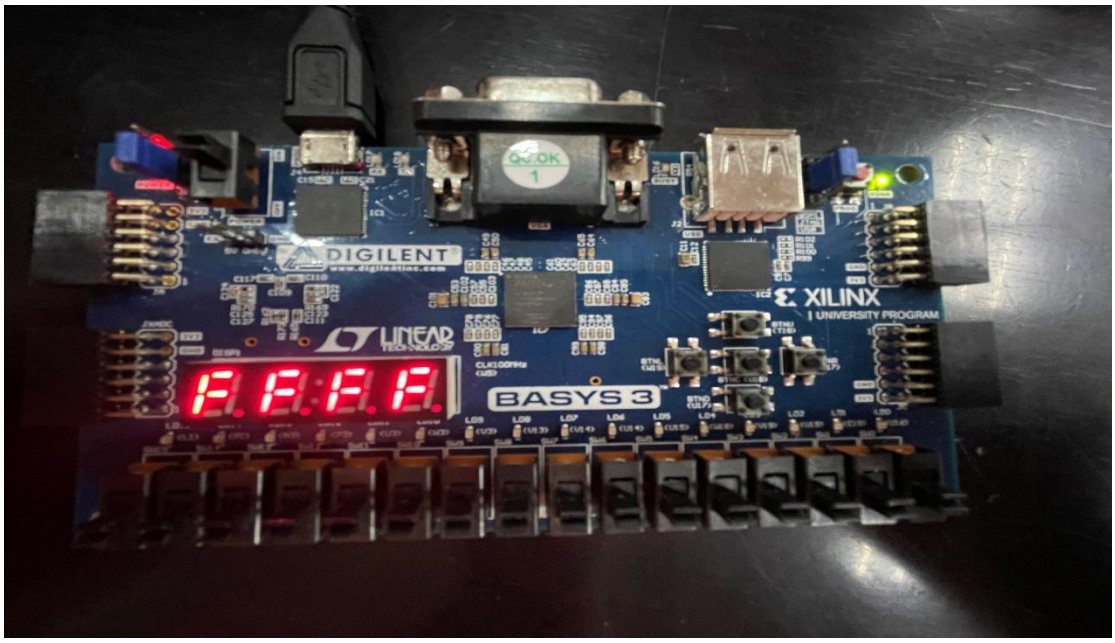
四、实验结果与分析

设置 Rom 初值的文件 prgmip32.coe

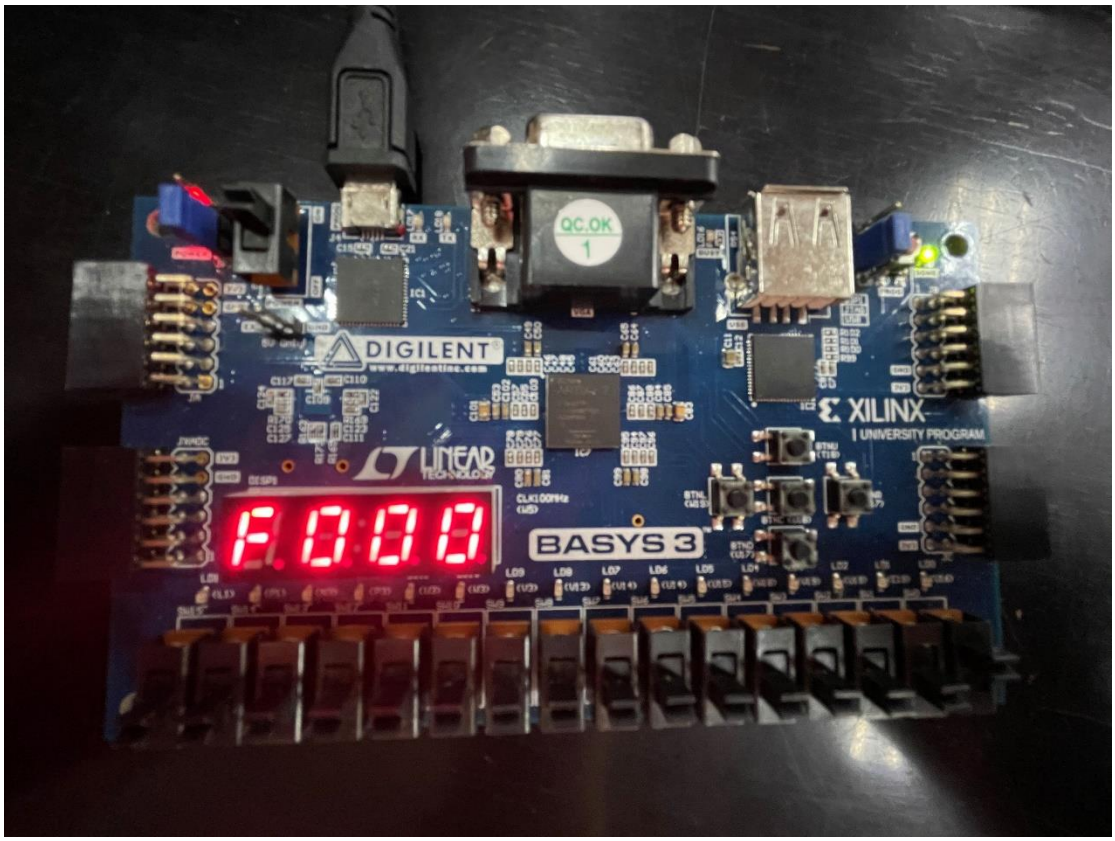
```
memory_initialization_radix = 16;
memory_initialization_vector =
3c01ffff,
343cf000,
8c190004,
8c180008,
8c17000c,
8c160010,
8c150014,
8c140018,
8c13001c,
8c120020,
8c110024,
8c1d0000,
23bdffff,
17a0fffe,
03e00008,
af980c60,
0c00000b,
af970c60,
```

对随机五个地址进行测试

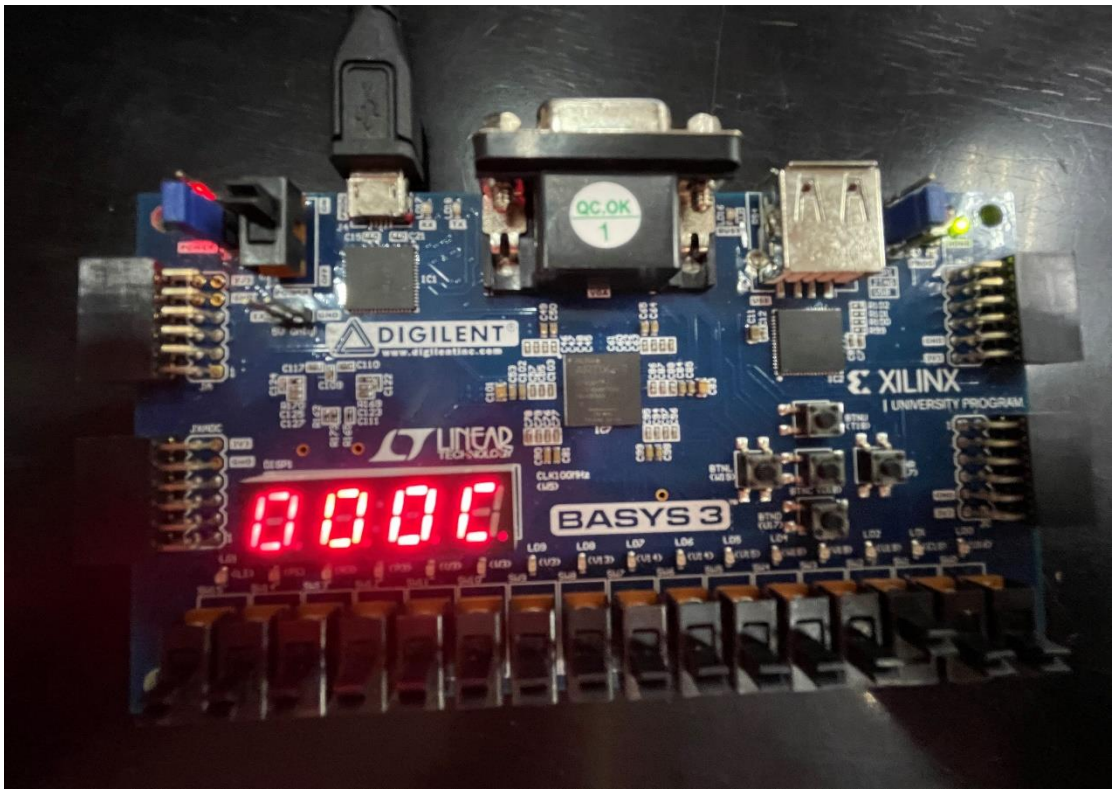
addr = 8’b00000000 七段数码管显示低 16 位 FFFF 正确



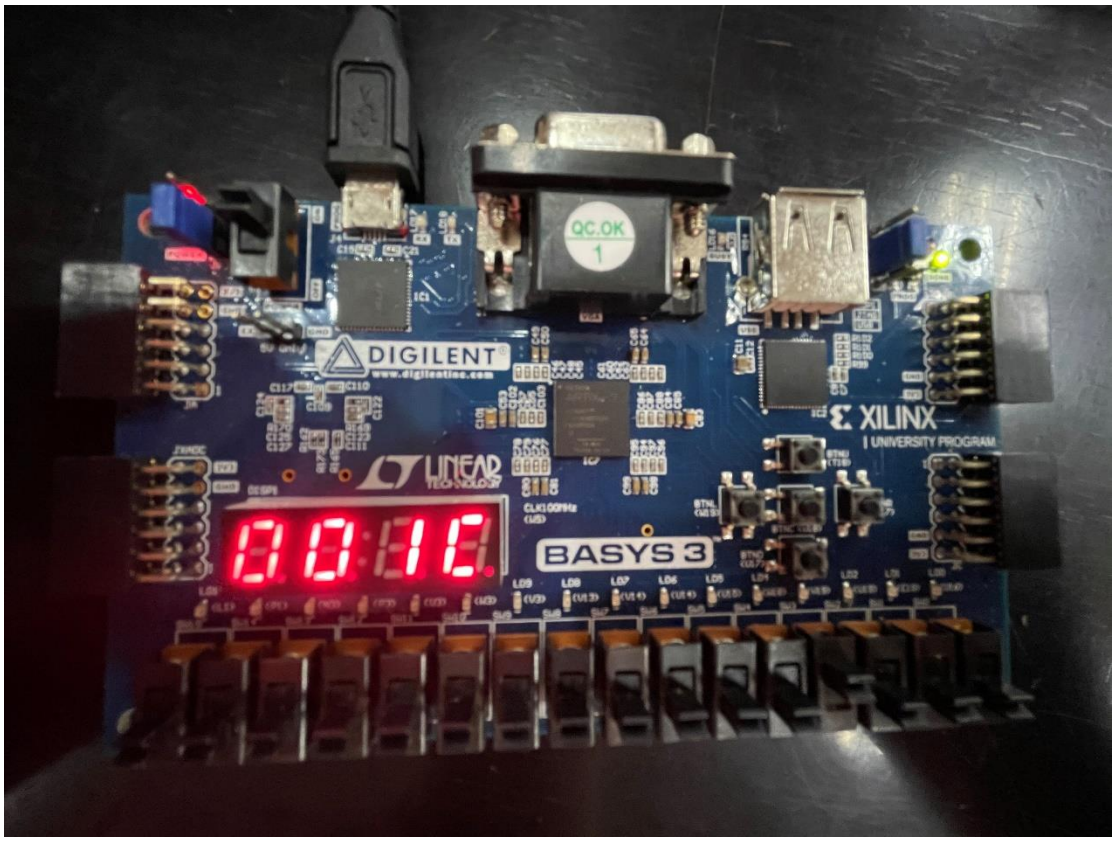
addr = 8’b00000001 七段数码管显示低 16 位 F000 正确



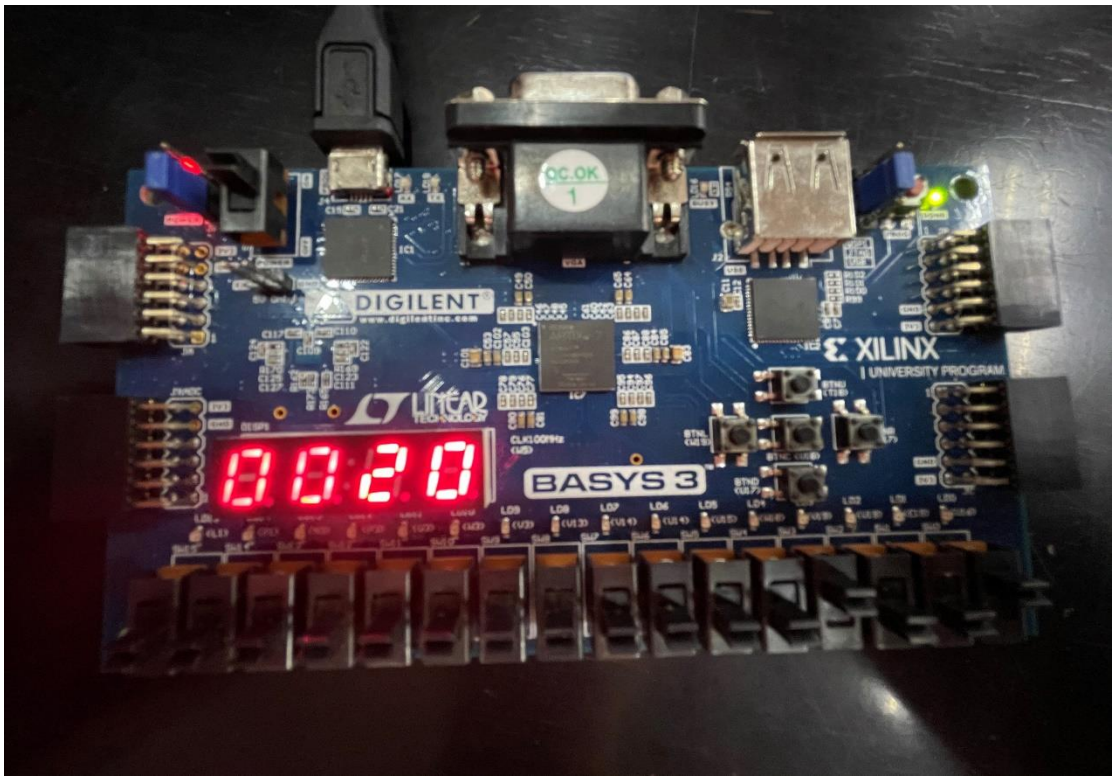
addr = 8’b00000100 七段数码管显示低 16 位 000C 正确



addr = 8’b00001000 七段数码管显示低 16 位 001C 正确



addr = 8’b00001001 七段数码管显示低 16 位 0020 正确



五、实验总结

这次存储器的实验比较简单，但我也学到了很多。管脚分配我更熟练地掌握了，也学会了如何在 Vivado 中添加 ip 核，并使用它的存储功能，还明白了如何在顶层文件中调用 ip 核，收获颇丰。