

## 存储器实现 & 存储器读写

### 实验目的

目标：存储器读写。自选合适的方式选择地址写入32位数据，并能按任意地址读出存储器数据。系统按16位zjie编址。

输入：

开 关：八个开关表示8位按16位zjie编址的存储器地址。

按 钮：[可选]当按字读出时，最右按/不按选择显示大头/小头不同设计。

输出：

发光管：自定。

四数码：显示存储器读写地址(按zjie寻址)。

八数码：显示32位存储器读写数据。

### 实验过程

首先编写 COE 文件如下：

```
memory_initialization_radix = 16;
memory_initialization_vector =
    00010203,
    04050607,
    08090A0B,
    0C0D0E0F,
    00102030,
    40506070,
    8090A0B0,
    C0D0E0F0
;
```

接着利用 IP cores 生成一个 word size 为 32 bits，并且 words 的数量为 8 的 Block Memory

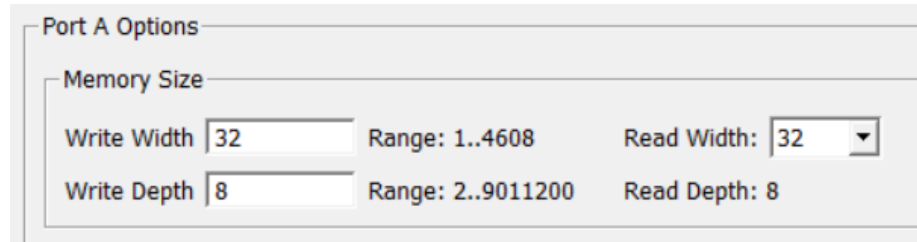


图 1: Memory Settings

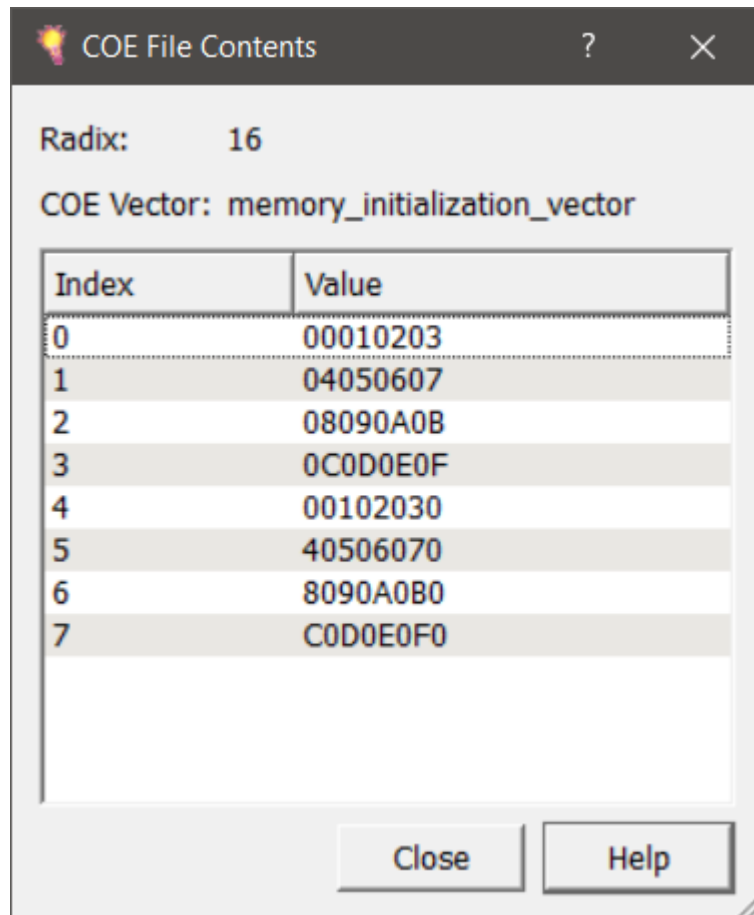
然后导入 COE

确认无误以后，编写控制代码如下：

```
Ram32b myram(.clka(clk), .wea(write_en), .dina(m_write_data), .addra(addr[3:1]), .douta
```

```
always @(*) begin
    if (is_little_endian) begin
        if (addr[0] == 0) begin
            read_data_1 = m_read_data[15:0];
        end else begin
            read_data_1 = m_read_data[31:16];
        end
    end else begin
        if (addr[0] == 0) begin
            read_data_1 = {m_read_data[7:0], m_read_data[15:8]};
        end else begin
            read_data_1 = {m_read_data[23:16], m_read_data[31:24]};
        end
    end
end

if (addr[0] == 0) begin
    m_write_data = {m_read_data[31:16], write_data};
end else begin
    m_write_data = {write_data, m_read_data[15:0]};
end
```



The image shows a software dialog box titled "COE File Contents". It has a standard Windows-style title bar with a question mark icon and a close button (X). The main area of the dialog contains the text "Radix: 16" and "COE Vector: memory\_initialization\_vector". Below this is a table with two columns: "Index" and "Value". The table contains eight rows of data, with alternating light and dark gray background colors for each row. At the bottom of the dialog, there are two buttons: "Close" and "Help".

Index	Value
0	00010203
1	04050607
2	08090A0B
3	0C0D0E0F
4	00102030
5	40506070
6	8090A0B0
7	C0D0E0F0

图 2: COE Config

```
end  
end
```

测试代码，波形图呈现在实例分析小节。

编写 Top 模块，实现相应功能：

输入：

开 关：八个开关表示8位按16位zjie编址的存储器地址。

按 钮：[可选]当按字读出时，最右按/不按选择显示大头/小头不同设计。

输出：

发光管：自定。

四数码：显示存储器读写地址(按zjie寻址)。

八数码：显示32位存储器读写数据。

形成最终项目：

## 实例分析

波形图如下：

可以看出，在 big-endian 的模式下，原本的储存的数值 0x0203 输出方式成功改变成了 0x0302。接着可以观察到，当我改变 0x1 地址下的值为 0x4141 以后，当前地址的值的的确被改变了。将读取地址变为 0x0 之后，可以看到储存在 0x0 的值并没有随之发生变化，因此可以证明对 half-word 的操作不会影响到储存器的其他值。

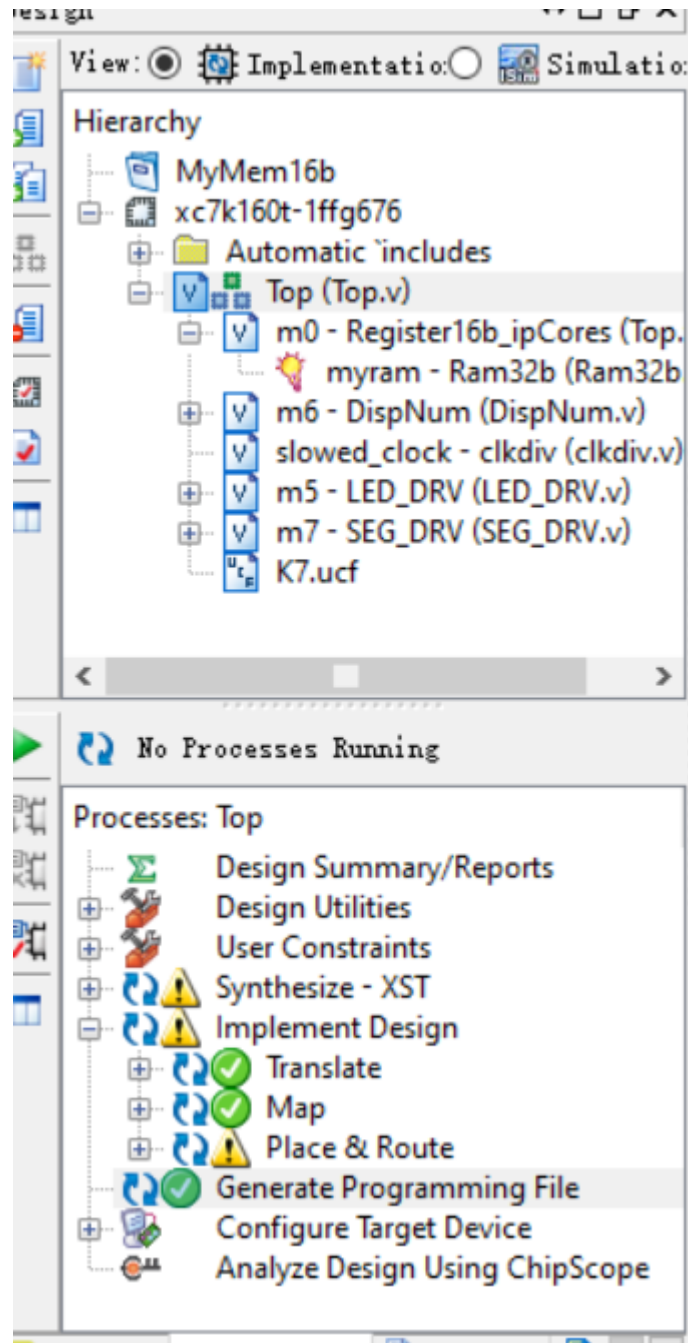


图 3: Project

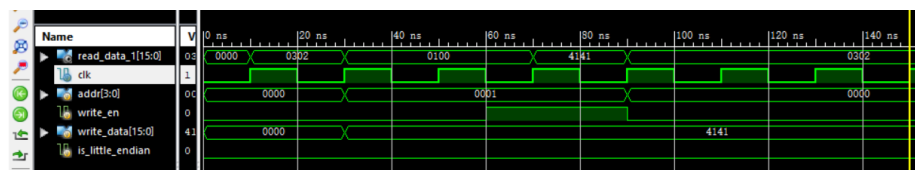


图 4: 波形图