**组成原理实验课程第 五 次实报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验名称 | 存储器实现 | | | 班级 | 李涛 |
| 学生姓名 | 艾明旭 | 学号 | 2111033 | 指导老师 | 董前琨 |
| 实验地点 | A306 | | 实验时间 | 2023.5.23 | |

1. **实验目的**

（1）了解只读存储器 ROM 和随机存取存储器 RAM 的原理。

（2）理解 ROM 读取数据及 RAM 读取、写入数据的过程。理解计算机中存储器地址编址和数据索引方法。

（3）理解同步 RAM 和异步 RAM 的区别。

（4）掌握调用 xilinx 库 IP 实例化 RAM 的设计方法。

（5）熟悉并运用 verilog 语言进行电路设计。

（6）为后续设计 cpu 的实验打下基础。

（7）理解 ROM 读取数据及 RAM 读取、写入数据的过程。**实验内容说明**

（概述本次实验要做什么，参见实验要求）

1. **实验内容说明**

• 学习同步和异步下 rom 和 ram 存储器，并观察同步异步下两种存储器的代码差异

• 实验箱对同步、异步下的 rom 和 ram 存储器进行实验

1. **实验原理图**

**同步RAM的顶层展示模块**

**同步ROM的顶层模块**



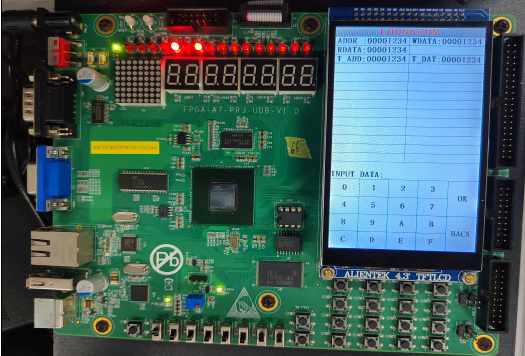
1. **实验步骤**

复现代码上箱测试即可，无代码修改部分。

1. **实验结果分析**

5.1 同步 ram

将前两个开关向上，之后四个开关置为 1，即 input\_select 控制信号设置为 00，四位 wen 控制信号置均为 1，写入读写地址 1234，将开关置为 01，写入数据 1234，发现 rdata 显示我们写入的数据，将开关置为 10，在 test\_addr 输入 1234, 发现 t\_data 显示出我们要的数据。



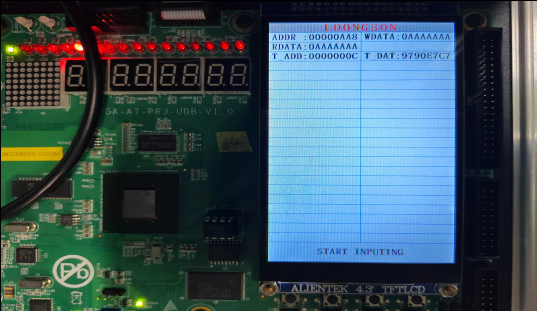
**5.2 同步 rom**

将开关置于向上，此时输入读地址 18，即输入第 floor(（1\*16+8）/4)=6，即第 7 个数据（按照字节编码，4 个字节一个数据），是 00A23027



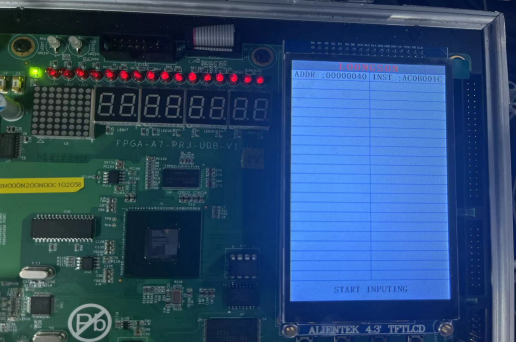
5.3 异步 ram

首先还是将最左侧两个开关置为 00，将 wen 信号均为 1，再输入读写地址 c，然后写入9790E7C7；然后重新写入再输入读写地址 AA8，然后写入 0AAAAAAA，此时 rdata 显示该数据；开关置于 10. 写测试地址 C，发现测试 data 显示我们之前写入的 9790E7C7。验证成功



**5.4 异步 rom**

和同步 rom 一样，俩开关置于上，写入地址 40，40 出应该写的是 4\*16/4=16，也就是第 17 条指令，观察表格为 ACB001C，而试验箱结果也为它，验证成功！



1. **总结感想**

6.1 ram or rom 的区别，如下：

• 直观来看，ram：Random Access Memory，rom：Read-Only Memory 是可读可写的，但 rom 只可以读，且读的数据是写死在硬件中的。

• 从深层次看，rom 往往存储的是计算机内部常用固定的数据，如指令及存储器便应该使用 rom，因为数据是固定的；而 ram 具有更强的灵活性，即它可以任意读写任意地址，比如可以用来存储计算机内临时产生的信息。

6.2 同步 or 异步的区别，如下：

• 同步存储器是在特定时钟上升或者下降沿进行数据读写，而异步则不管时钟，只要读写信号出现，便执行，二者更像是时序电路和简单逻辑电路之间的区别。以 rom 代码为例，看看代码上的区别，如下：

可以看到：异步 rom 的读代码 always 根本没用时钟信号，同步 rom 的读代码 always 使用时钟下降沿

|  |
| --- |
| //建议对每一个数的输入，编写单独一个always块 |
| //当input\_sel为2'b00时，表示输入数为读写地址，即addr |
| always @(posedge clk) |
| begin |
| if (!resetn) |
| begin |
| addr <= 32'd0; |
| end |
| else if (input\_valid && input\_sel==2'd0) |
| begin |
| addr[31:2] <= input\_value[31:2]; |
| end |
| end |

同步 rom 的读代码 always 使用时钟下降沿

|  |
| --- |
| //读数据,取4字节 |
| always @(\*) |
| begin |
| case (addr) |
| 5'd0 : rdata <= DM[0 ]; |
| 5'd1 : rdata <= DM[1 ]; |
| 5'd2 : rdata <= DM[2 ]; |
| 5'd3 : rdata <= DM[3 ]; |
| 5'd4 : rdata <= DM[4 ]; |
| 5'd5 : rdata <= DM[5 ]; |
| 5'd6 : rdata <= DM[6 ]; |
| 5'd7 : rdata <= DM[7 ]; |
| 5'd8 : rdata <= DM[8 ]; |

二者使用场景的思考：因为二者的有无时序控制，导致同步存储器的稳定性高、可以保

证数据的一致性和时序性，所以性能相较于异步存储器高，更可以胜任大规模、高性能、

更复杂的场景。

而异步没用时钟信号去读写数据，所以其速度更快、电路更简单，在一些小数据量、系

统较为简单的地方应用更适合！