|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **学 期** | 2022-2023学年第1学期 | | | | **实验日期** | | 2022/10/28 | | |
| **学 院** | 信息学部 | | | | **专 业** | | 计算机科学与技术（实验班） | | |
| **班 级** | 210710 | | **学 号** | | 21071003 | | **姓 名** | | 高立扬 |
| **组 号** | 43 | | **学 号** | | 21071004 | | **姓 名** | | 石昊阳 |
| **评 阅 内 容** | | | | | | | | | |
| **任务一** | | **任务二** | | **总结** | | **格式** | | **成 绩** | |
|  | |  | |  | |  | |  | |
|  | | | | | | | | | |
| **题 目** | 实验2：存储器电路设计 | | | | | | | | |

1. **实验目的**

通过本实验，掌握基本存储电路的分析方法和设计方法；学会使用Verilog HDL设计存储器电路。

1.通过触发器、锁存器、寄存器、存储器等时序电路的设计与测试，掌握基本存储电路的分析方法和设计方法；

2.学会使用Verilog HDL设计存储器电路

1. **任务一设计与实现**
2. 要求

(1) 用行为描述方式实现D锁存器，并对该器件进行封装。

(2) 用行为描述方式实现D触发器，并对该器件进行封装。

(3) 用图形方式调用D锁存器和D触发器，两个器件共用同一个输入端D，并用拨动开关控制，输出端Q分别接至LED灯，EN接开关，CLK接按键，下载后进行功能测试并记录实验现象。

(4) 请编写Testbench测试程序完成仿真。

1. 设计思路

先用行为描述的方式实现D锁存器与D触发器，分别对其进行封装。在Block Diagram/Schematic File中使两个模块共用一个输入端D，并分配引脚，即可完成。

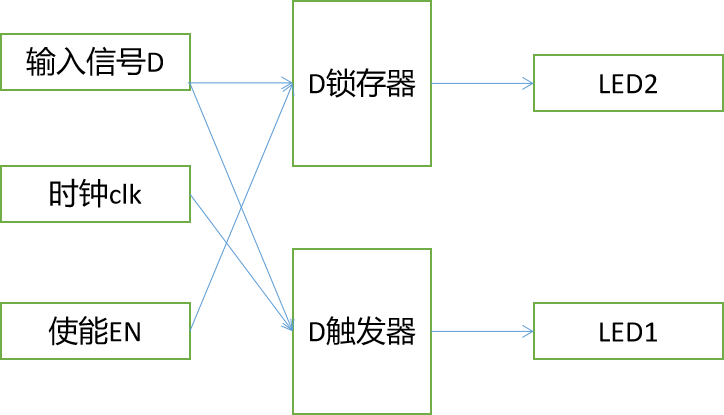


图1.D锁存器和D触发器思路

1. 详细设计

表1.D锁存器功能表

|  |  |  |  |
| --- | --- | --- | --- |
| 使能 | 输入 | 输出 | 显示 |
| EN | D | Q | LED灯 |
| 1 | 0 | 0 | 亮 |
| 1 | 1 | 1 | 灭 |
| 0 | d | 保持上个输出 | 保持上个状态 |

表2.D编译器功能表

|  |  |  |  |
| --- | --- | --- | --- |
| 时钟信号 | 输入 | 输出 | 显示 |
| CLK | D | Q | LED灯 |
| ↑ | 0 | 0 | 亮 |
| ↑ | 1 | 1 | 灭 |
| 0 | d | 保持上个输出 | 保持上个状态 |

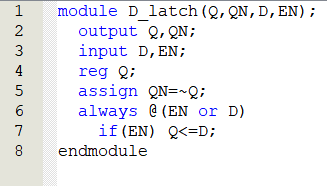
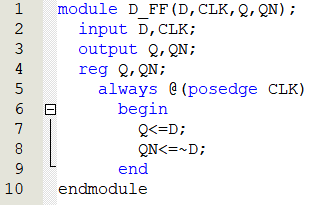
 

图2. D锁存器 图3.D触发器

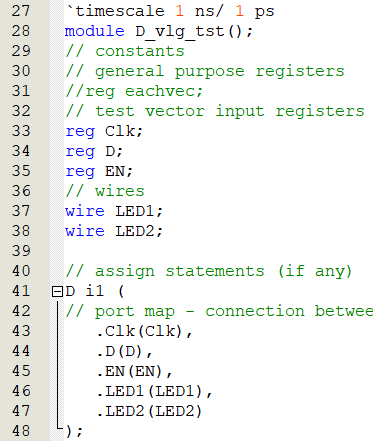
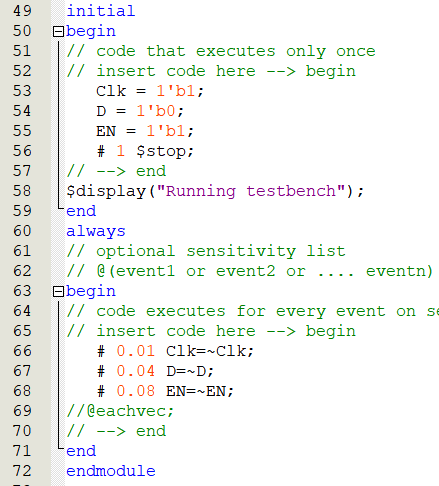
 

图4，5.Testbench

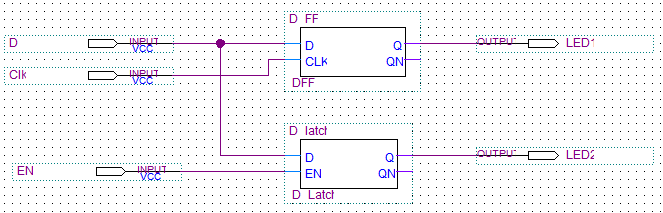


图6.D锁存器+D触发器原理图

1. 仿真验证

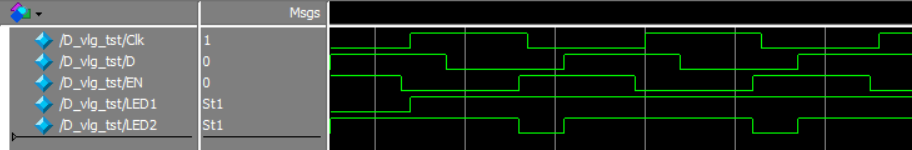


图7.testbench仿真图线

由图。LED1接的是D触发器，因此在Clk为上升沿的时候，根据D的初态1，LED1的数值也就变成了1。LED2接的是D锁存器，因此在EN为1的时候，LED2的数值跟随D的改变而改变，否则LED2的数值锁定。

1. 引脚分配

引脚分配见下表

表3.引脚分配表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 端口名称 | 输入端 | | | 输出端 | |
| 使能端EN | 时钟信号Clk | 待显示数据D | LED1 | LED2 |
| 引脚编号 | AB17 | AB15 | N18 | U12 | V12 |
| 平台端口 | SW9 | F1 | SW1 | LED1 | LED2 |

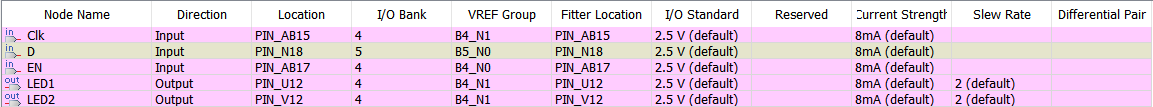


图8.引脚分配

1. 实验现象

LED1：先设置D的数值，按Clk按钮的时候观察LED1的亮灭情况。当松开按钮前D为1时，小灯泡会灭，证实了程序正确。

LED2：先设置D的数值，然后调整EN的数值，当EN为1，小灯泡随着D数值的改变而改变亮灭，一旦EN设置为0，则无论D的改变，小灯泡的亮灭情况不会改变。

1. **任务二设计与实现**
2. 要求

(1) 设计一个由8个4位寄存器构成的寄存器堆，该寄存器堆支持寄存器地址选择，能够写入任意4位值，并且可从任意一个寄存器中读取数据。

(2) 读操作不需要时钟控制，写操作需要在上升沿写入。

(3) 用地址译码、寄存器组、数据选择电路等基本电路实现。

1. 设计思路

通过分析寄存器堆的组成，发现他需要由四个模块组成。分别是地址，译码器，数据选择器，寄存器组，和三态门。分别将其用verilog语言写出并封装，再将其合理连接至一起，组成寄存器堆。

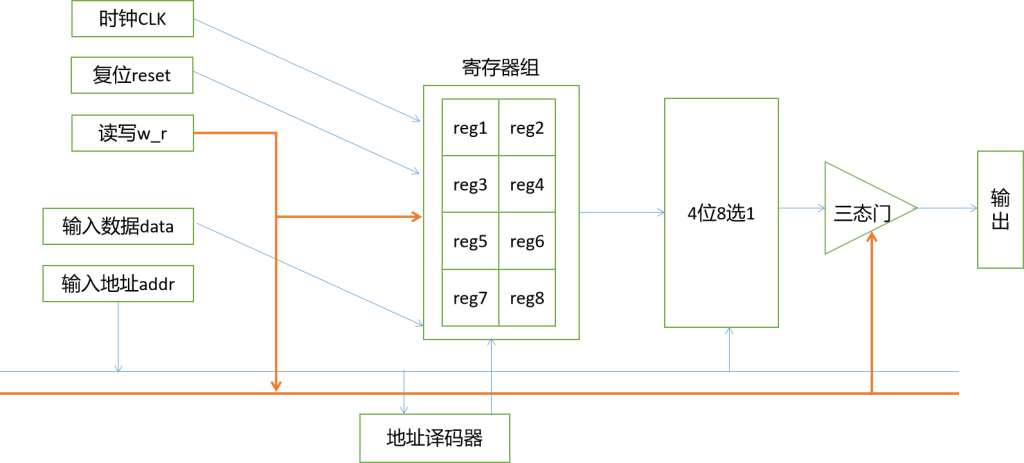


图9.寄存器堆的设计思路

1. 详细设计

表4.地址选择器功能表

|  |  |
| --- | --- |
| 输入 | 输出 |
| 地址编码addr | 片选out |
| 000 | 01111111 |
| 001 | 10111111 |
| 010 | 11011111 |
| 011 | 11101111 |
| 100 | 11110111 |
| 101 | 11111011 |
| 110 | 11111101 |
| 111 | 11111110 |

表5.数据选择器功能表

|  |  |  |
| --- | --- | --- |
| 输入 | | 输出 |
| 地址编码addr | 输入数据(4位)in | 数据选择out |
| 000 | in1~in8 | in1 |
| 001 | in2 |
| 010 | in3 |
| 011 | in4 |
| 100 | in5 |
| 101 | in6 |
| 110 | in7 |
| 111 | in8 |

表6.四位寄存器功能表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入 | | | | | 输出 |
| 时钟信号Clk | 置零信号reset | 4位输入data | 片选信号cs | 读写权限w\_r | 4位输出(寄存)q |
| x | 1 | xxxx | x | x | 置零 |
| x | 0 | xxxx | 0 | x | 保持 |
| ↑ | 0 | data | 1 | 1 | data |
| 其他 | 0 | xxxx | 1 | 1 | 保持 |

表7.三态门功能表

|  |  |  |
| --- | --- | --- |
| 输入 | | 输出 |
| 读写权限w\_r | 4位输入data | 4位输出out |
| 0 | data | data |
| 1 | data | 0000 |

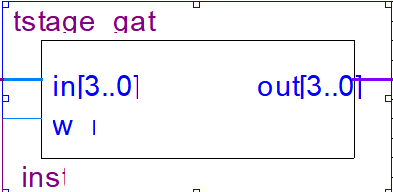
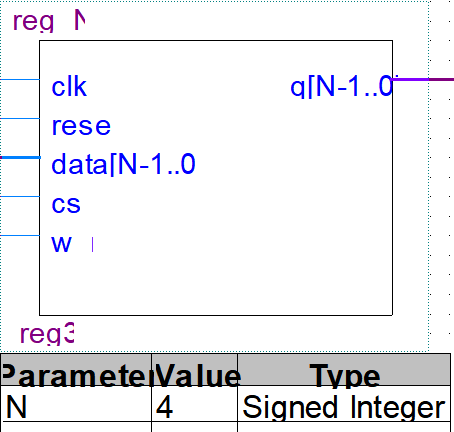
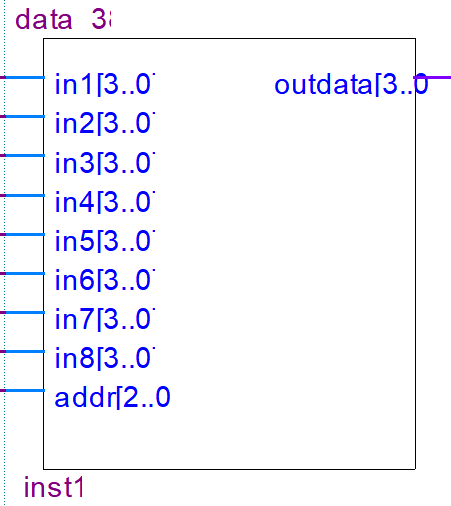
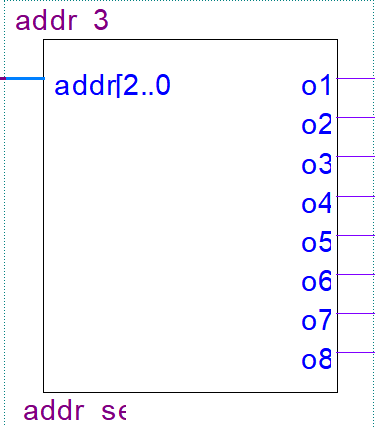


图10,11,12,13 依次为：地址译码器，数据选择器，4位寄存器，三态门

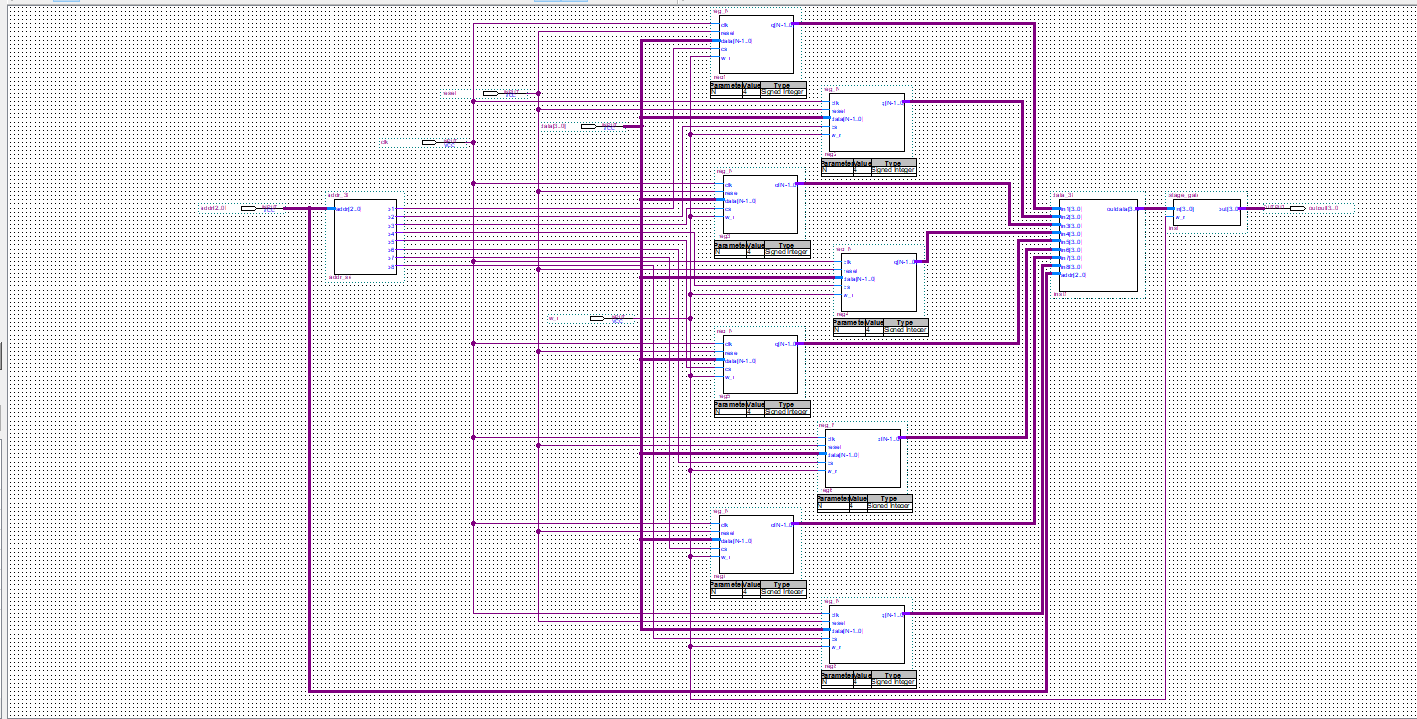
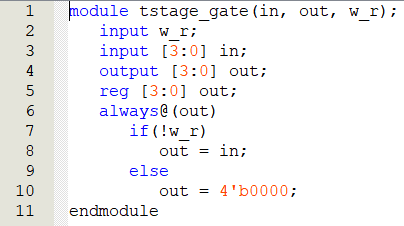
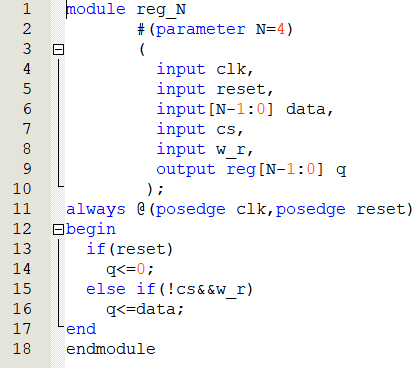
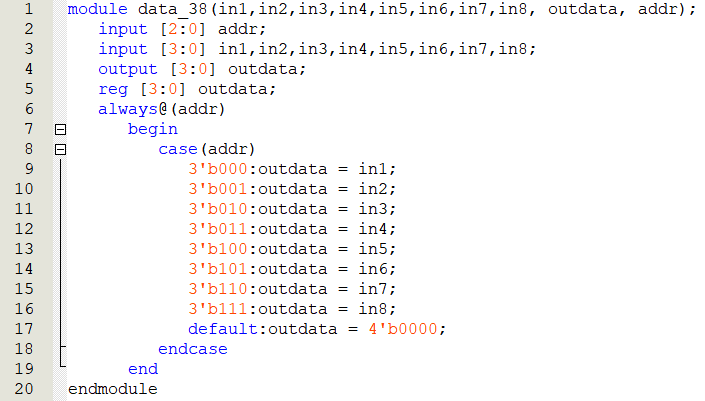
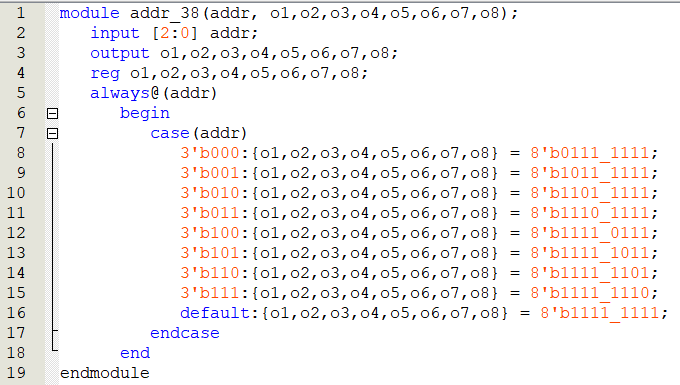


图14,15,16,17,18 依次为：地址译码器，数据选择器，4位寄存器，三态门，寄存器堆总设计原理图

1. 仿真验证

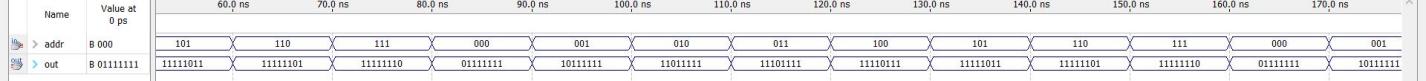


图19.地址译码器仿真图

由上图，可发现地址编码从000~111，对应第1~8个寄存器的片选信号被激活，符合预期功能。

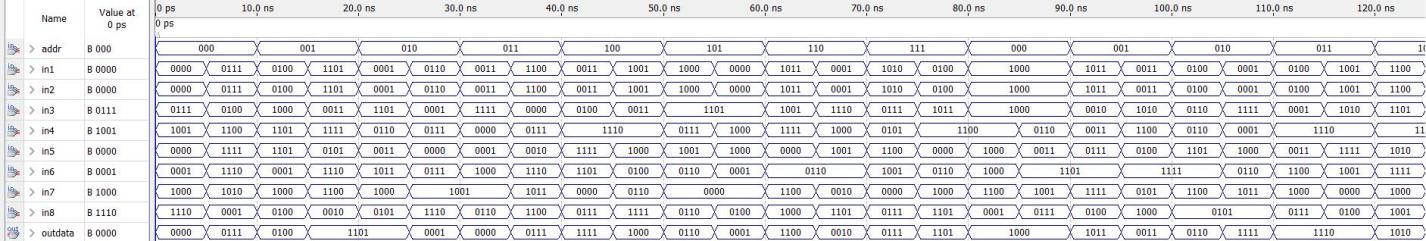


图20.数据选择器仿真图

由上图，可以发现地址编码从000~111对应outdata=in1~in8。波形图证明了本模块设计无误。

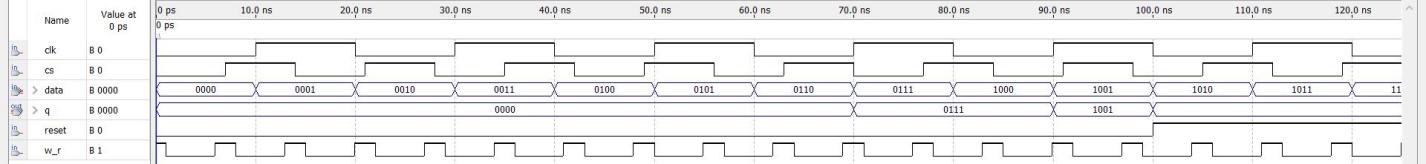


图21.4位寄存器仿真图

由上图，70ns之前Clk上升沿的瞬间要么w\_r是0，要么cs是1，所以寄存器无法写入数据，为0000。70ns的时候，w\_r为1，cs为0，Clk上升沿，因此数据被成功写入。后续波形仍能符合上述写入逻辑，验证了寄存器设计的正确性。

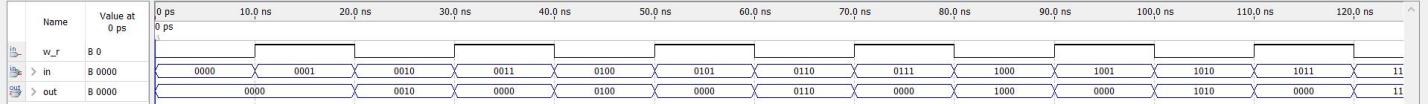


图22.三态门仿真图

由上图，每当w\_r为1，也就是读写权限为“写”时，“读”权限被禁止，因此out恒为0000。反之，三态门输出读取到的数据。整个波形验证了三态门设计的正确性。

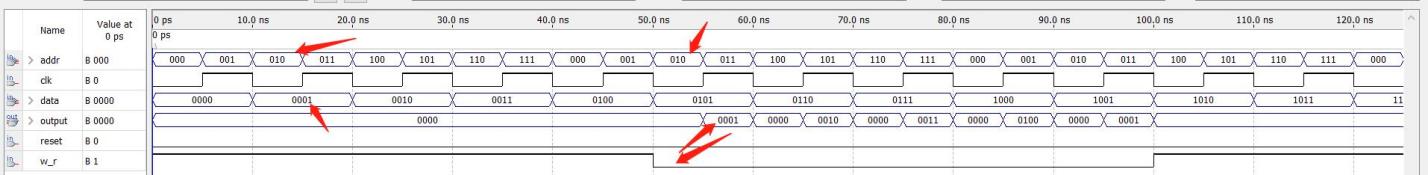


图23.寄存器堆仿真图

由上图红色箭头标注的地方，在15ns的时候有写入权限，片选信号选择第3片寄存器，此时为Clk上升沿，因此第3片寄存器被写入了数据0001，由于此时为写入模式，所以读取不到任何信息，读取数据为0000。55ns时，片选信号又选择了第3片寄存器，但是此时虽然Clk为上升沿，但是由于读写权限转换为了只读，所以无法写入数据，却能读取出第3片寄存器的数据，可见，读取数据正是之前写入的0001。完全符合预期。

1. 引脚分配

表8.引脚分配

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 端口名称 | 输入端 | | | | | | | 输出端 | | |
| 读写权限w\_r | 置零信号reset | 时钟信号clk | 待显示数据data | | | | 地址选择addr | | |
| data[3] | data[2] | data[1] | data[0] | addr[2] | addr[1] | addr[0] |
| 引脚编号 | AB17 | F8 | AB15 | N18 | M20 | AA15 | V13 | AB17 | AB18 | C3 |
| 平台端口 | SW9 | SW8 | F1 | SW1 | SW2 | SW3 | SW4 | SW9 | SW10 | SW11 |

表8.续表

|  |  |  |  |
| --- | --- | --- | --- |
| 输出端 | | | |
| LED1 | LED2 | LED3 | LED4 |
|
| U12 | V12 | V15 | W13 |
| LED1 | LED2 | LED3 | LED4 |

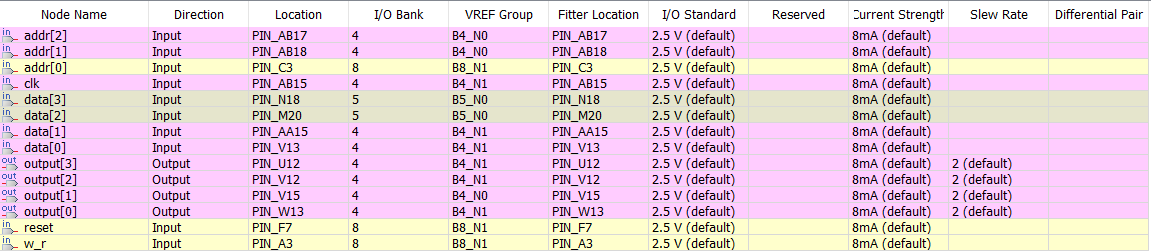


图24.引脚锁定

1. 实验现象

通过读写权限，片选信号和Clk的配合，8片寄存器每一片存入了我们想要的数字，然后调整读写权限后读取数据，LED的亮灭均符合每一片寄存器寄存的数据。最后调置零信号为1，所有寄存器的数据全部被清空，无论选择哪一片寄存器，读取出来的数据都是0000，小灯泡全部亮起。

1. **拓展任务设计与实现**

1.设计思路

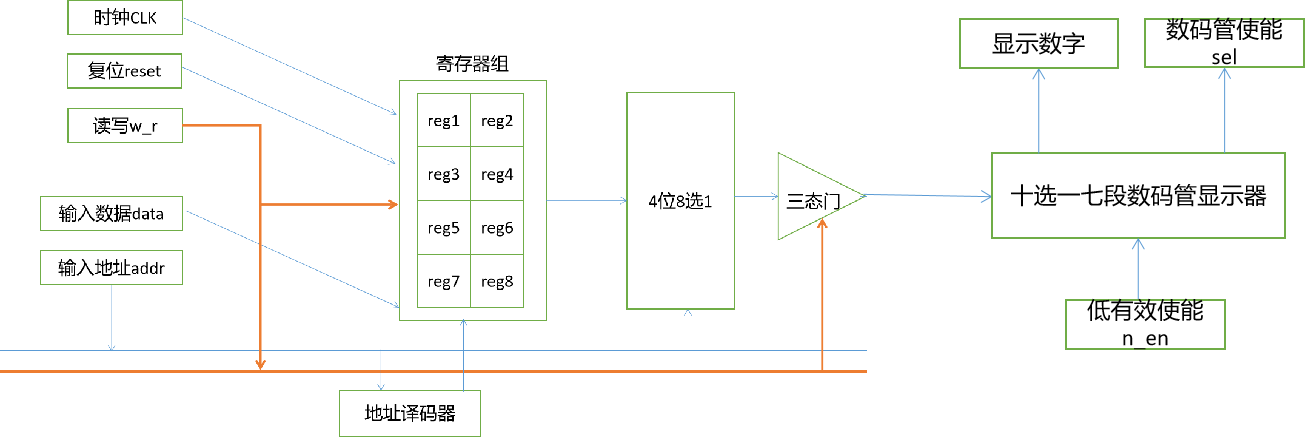


图25.拓展任务的设计思路

寄存器存储的数据是4位，片数是8，因此我们联想到了上次实验中显示学号的LED模块。将寄存器堆和LED显示模块连接起来，实现学号的储存和显示，而不是临时、逐个地调整想要的学号。

1. 详细设计

仅在任务2的基础上，新添加了七段数码管模块，因此下面只展示本模块的功能表

表9.七段数码管模块功能表

|  |  |  |  |
| --- | --- | --- | --- |
| 使能 | 输入 | 输出 | 显示 |
| n\_en | in | out |
| 0 | 0000 | 111\_1110 | 0 |
| 0 | 0001 | 011\_0000 | 1 |
| 0 | 0010 | 110\_1101 | 2 |
| 0 | 0011 | 111\_1001 | 3 |
| 0 | 0100 | 011\_0011 | 4 |
| 0 | 0101 | 101\_1011 | 5 |
| 0 | 0110 | 101\_1111 | 6 |
| 0 | 0111 | 111\_0000 | 7 |
| 0 | 1000 | 111\_1111 | 8 |
| 0 | 1001 | 111\_1011 | 9 |
| 1 | dddd | 000\_0000 | 全灭 |
| 0 | 1010~1111 | 000\_0000 | 全灭 |

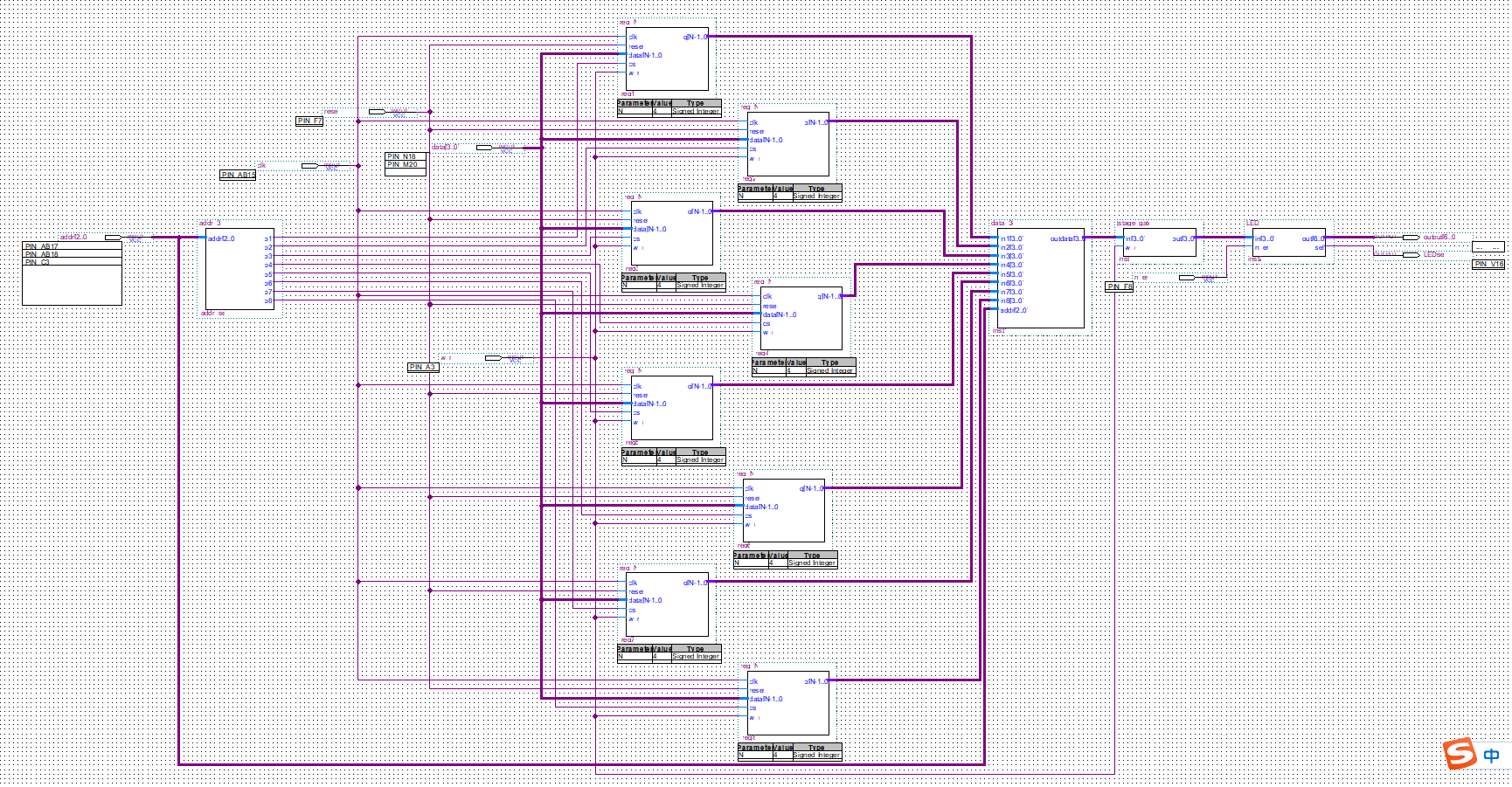


图26.拓展任务原理图实现

1. 仿真验证

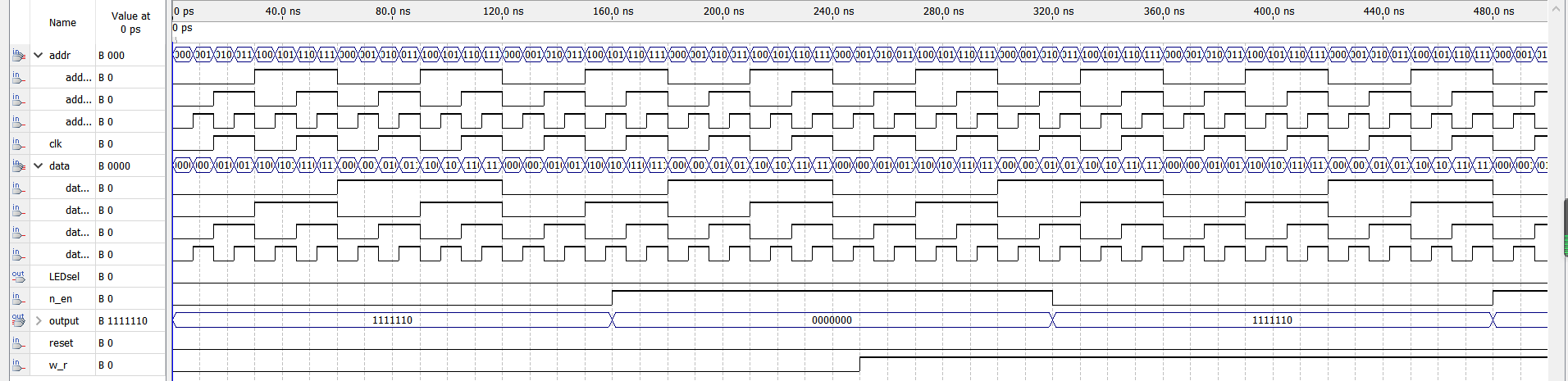


图27.波形图

结合实验一中的七段数码管模块，配合上文已验证过的寄存器堆波形图，本波形图符合预期。

1. 引脚分配

表10.引脚分配

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 端口名称 | 输入端 | | | | | | |
| 读写权限w\_r | 置零信号reset | 时钟信号clk | 待显示数据data | | | |
| data[3] | data[2] | data[1] | data[0] |
| 引脚编号 | AB17 | F8 | AB15 | N18 | M20 | AA15 | V13 |
| 平台端口 | SW9 | SW8 | F1 | SW1 | SW2 | SW3 | SW4 |

表10.续表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 输出端 | | | | | | | | | | |
| 地址选择addr | | | LED | | | | | | | 选位 |
| addr[2] | addr[1] | addr[0] | output[6] | output[5] | output[4] | output[3] | output[2] | output[1] | output[0] | sel |
| AB17 | AB18 | C3 | AA20 | W20 | R21 | P21 | N21 | N20 | M21 | V16 |
| SW9 | SW10 | SW11 | LA | LB | LC | LD | LE | LF | LG | DS8 |

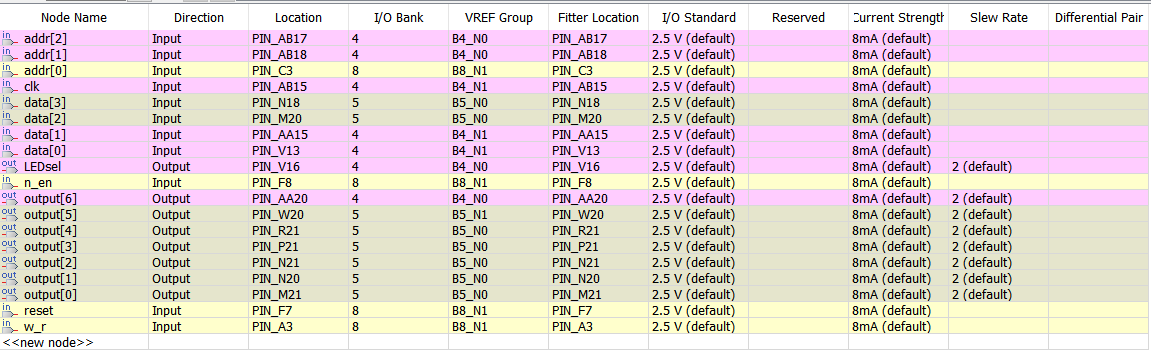


图27.引脚锁定

5.实验现象

预先设置好8片寄存器的数据为21071003，然后调整读写为只读，通过依次调整地址编码，可见LED逐步显示了学号。符合实验预期。

1. **总结**

收获：学会了testbench的编写和仿真方法，巩固了图形编程的技能。

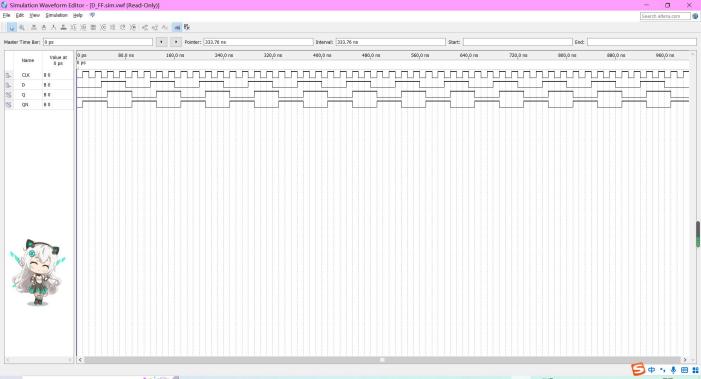
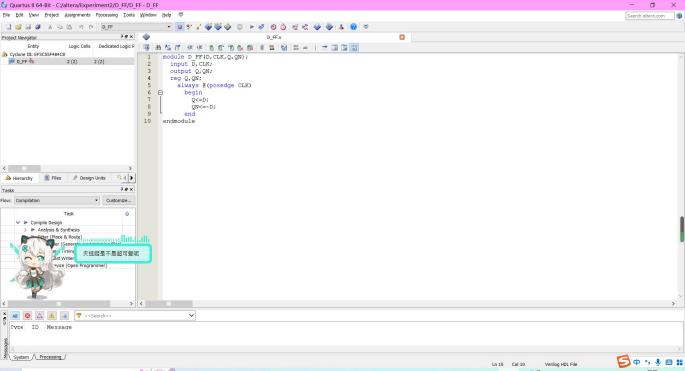
遇到的问题或者发现：

①Clk按钮抬起的时候，才是上升沿，任务1中的小灯泡在抬起的一瞬间改变亮或灭的状态，寄存器在抬起的一瞬间被写入数据。

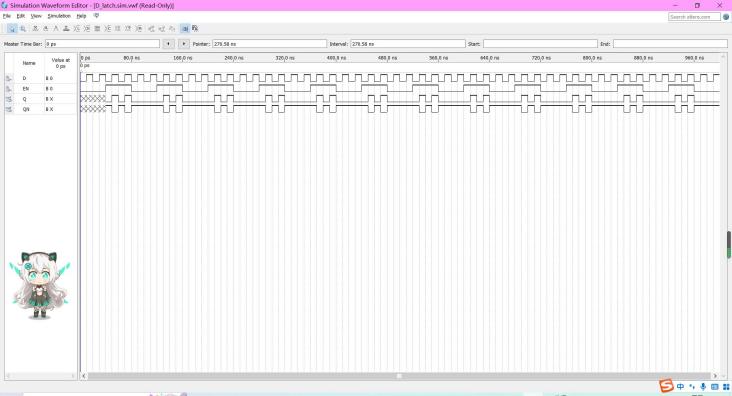
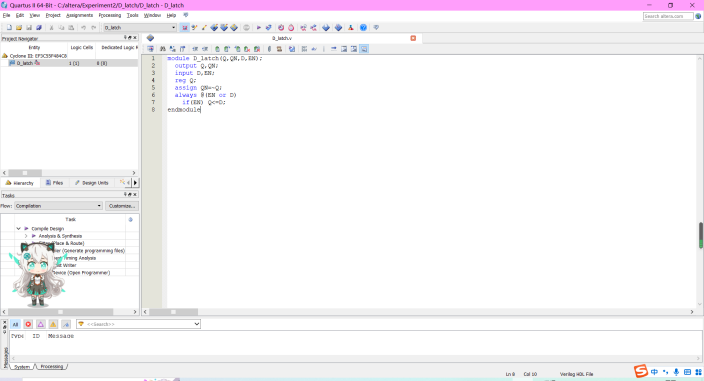
②Testbench生成的文件所在文件夹交modelsim文件夹而不是qsim文件夹

③自动生成testbench之前，需要设置好setting，包括modelsim所在文件夹，vt文件所在文件夹等。

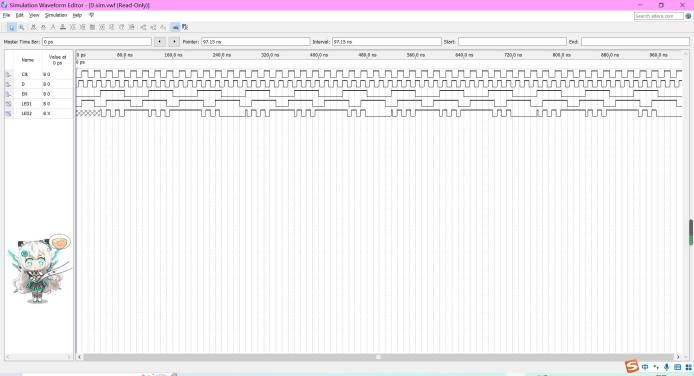
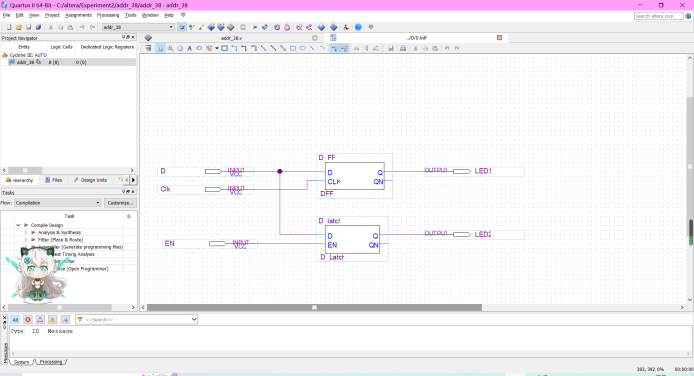
④testbench编写还是不熟练，实际仿真的时间间隔和编程写的不太相同，课下我们会多加练习，查资料。



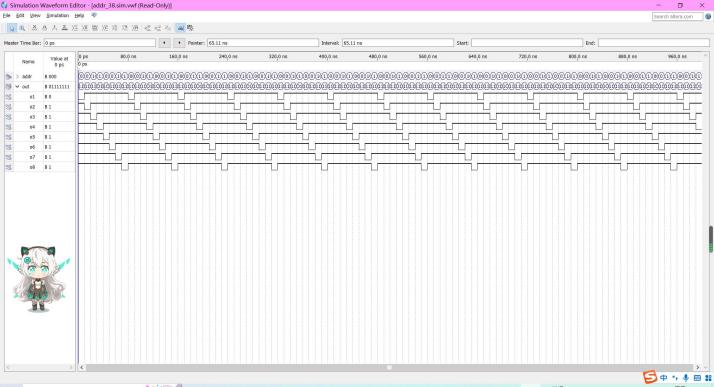
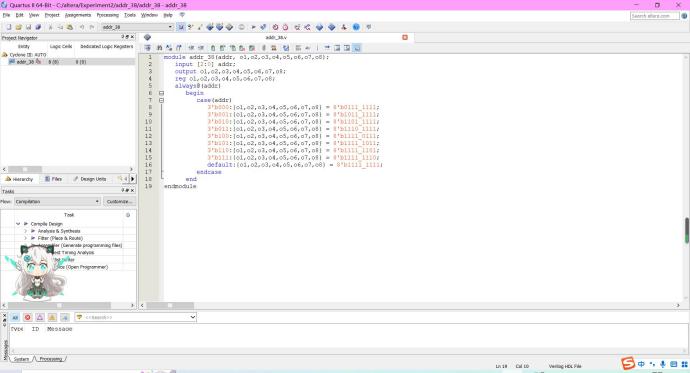
附图1：D触发器代码和波形



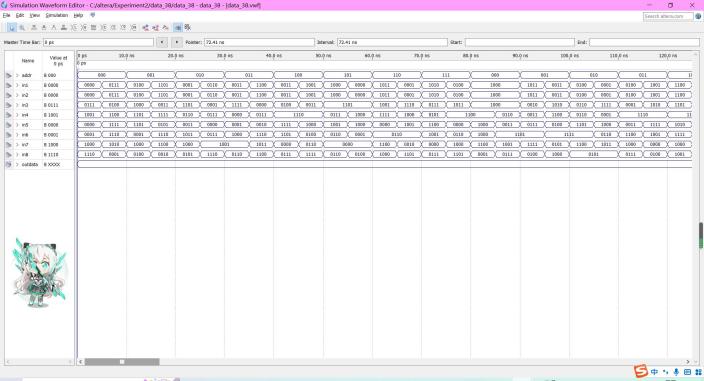
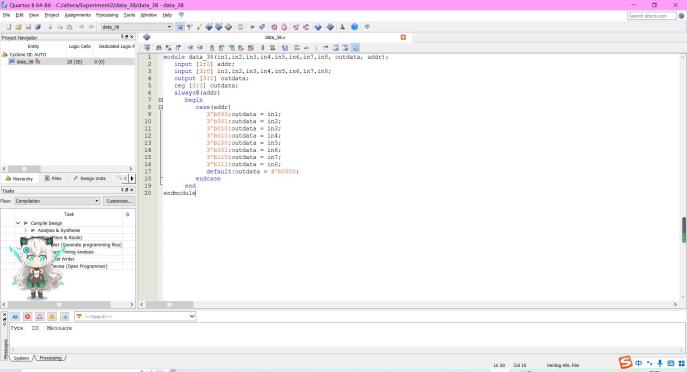
附图2：D锁存器代码和波形



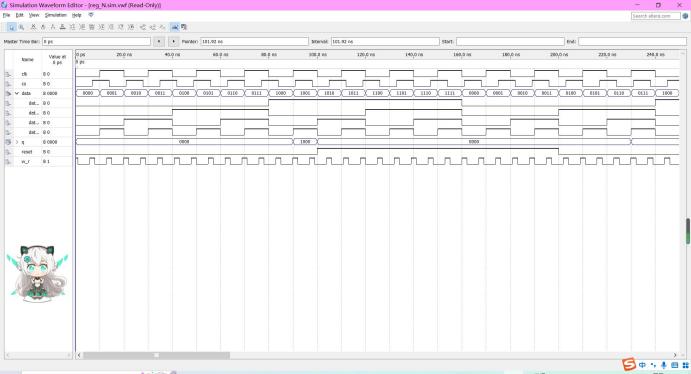
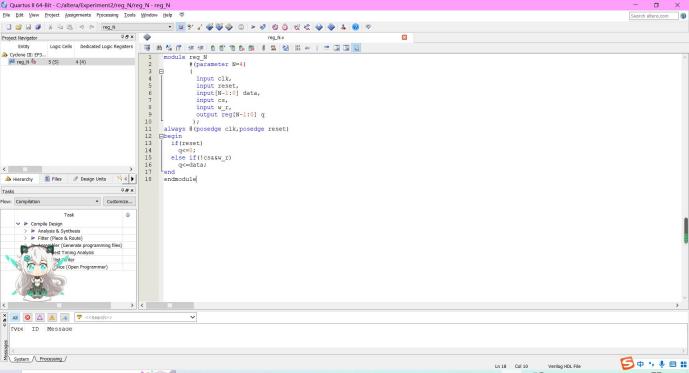
附图3：D触发器+D锁存器原理图和波形



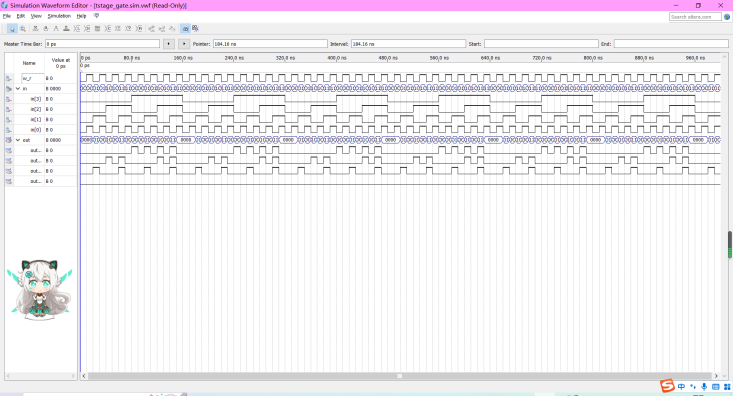
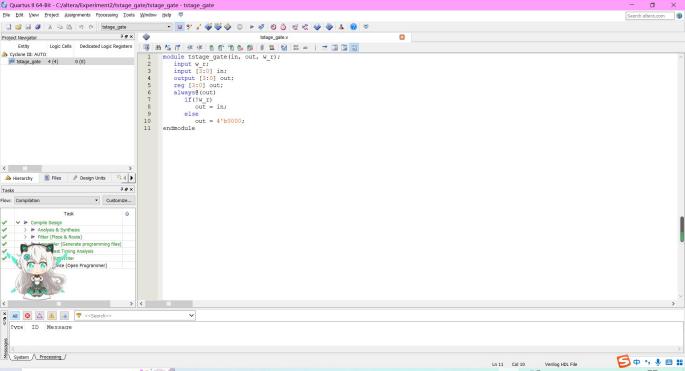
附图4：地址译码器代码和波形



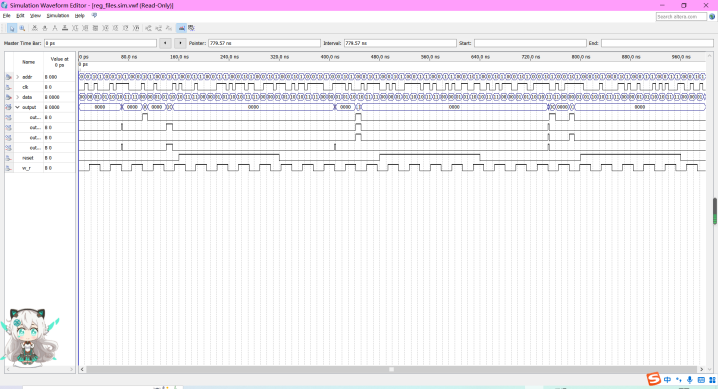
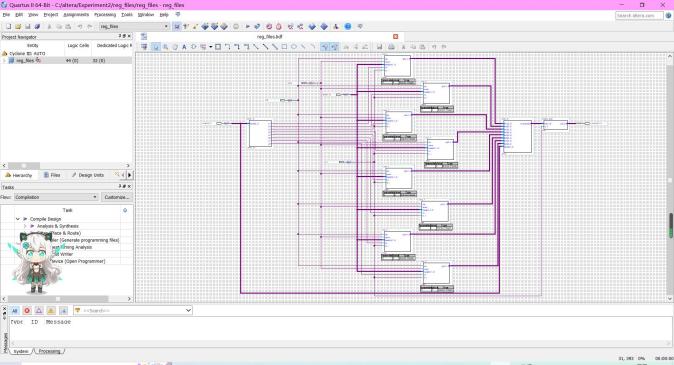
附图5：数据选择器代码和波形



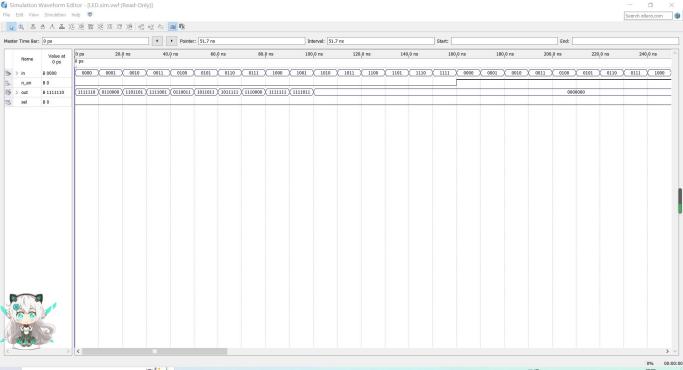
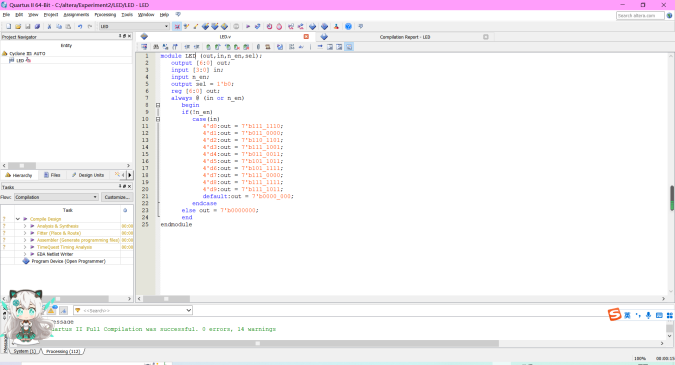
附图6：寄存器代码和波形



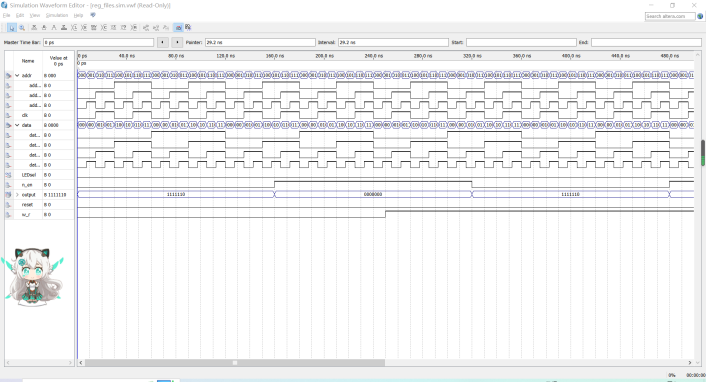
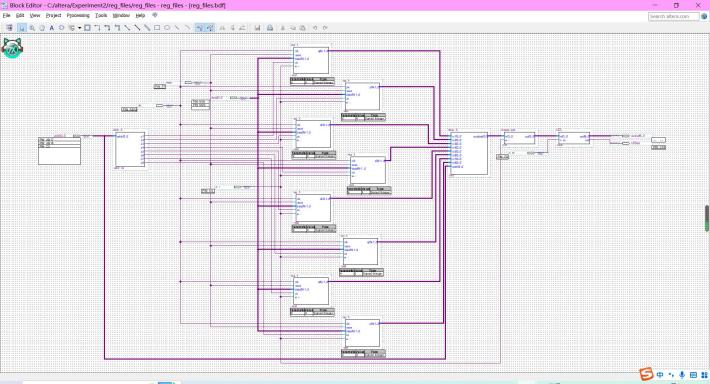
附图7：三态门代码和波形



附图8：寄存器堆原理图和波形



附图9：七段数码管代码和波形



附图10：扩展任务原理图和波形