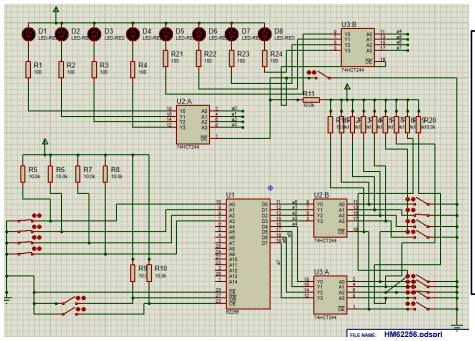
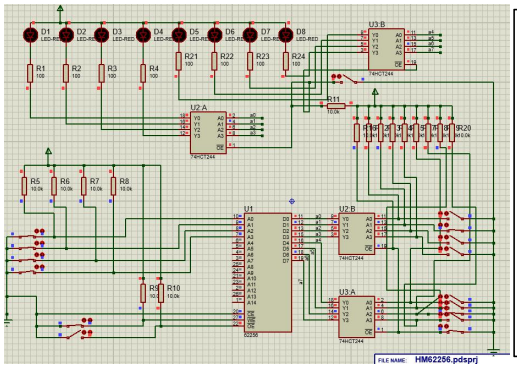
c 级任务 1

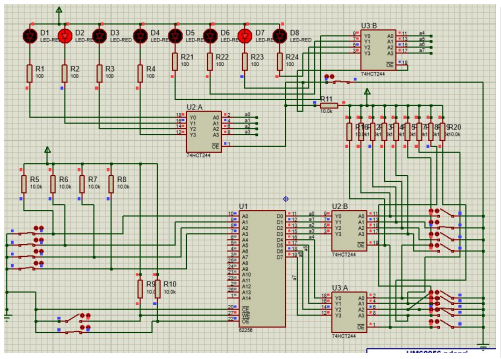
HM62256 电路增强数据线电路由 4 位至 8 位和数据输出显示部分



4 位地址线，8 位数据输出开始仿真



写入信号置为 0，右下角数据输入信号使能置为有效，地址 0000 处写入数据 10111101

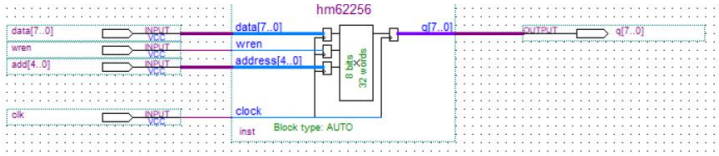


断开 we 非，打开 oe 非读信号，右下角数据输入信号使能置为无效，右上角读出数据信号使能置为有效，最终在二极管上显示符合实验结果读出 0000 地址数据 10111101

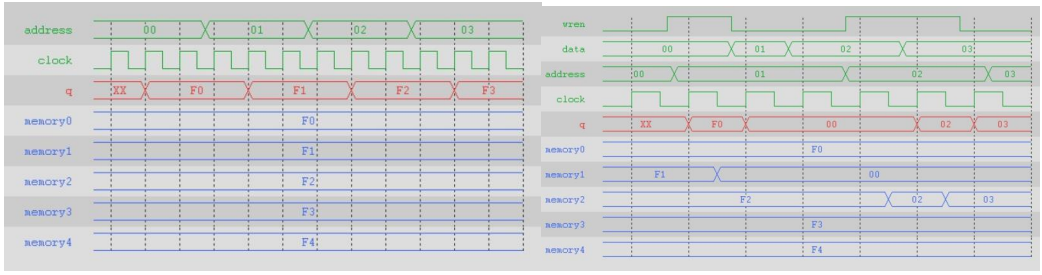
c 级任务 2

定制开发一个 1-port RAM 的 IP 核

顶层电路

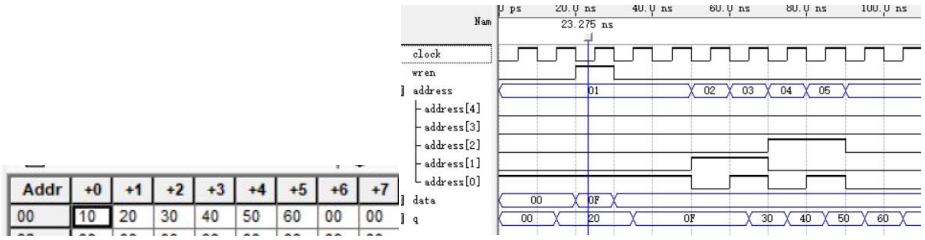


生成目录下的波形报告



第一个波形是在时钟上升沿对应的读出相应地址的数据（可能存在延时的问题，总在地址囊括的三个周期中的第二个周期读出对应地址的数据）

第二个波形是加入了写信号后的波形，在第一个写信号有效，并且在第二个时钟上升沿，00 数据被写入地址 01 处于是内存 memory1 的数据由 F1 变为 00，同时由于存在延时，读的时候总在地址囊括的三个时钟周期的第二个时钟上升沿才会读出对应地址数据，所以第三个时钟周期读出数据由 F0 变为写入后的数据 00；第二个写信号改变地址 2 的数据改变了两次相继变为 02，03，读数据如上分析（以上数据都为 16 进制）

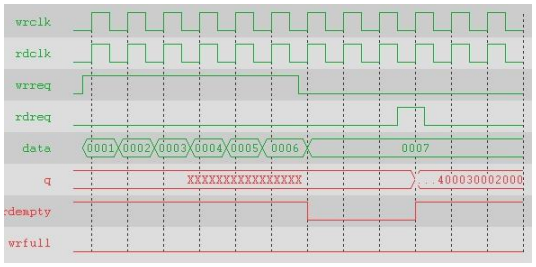


用的是生成的 vhd 文件

wren 有效将 01 地址数据 20 写为 0F 并最后在时钟上升沿都相继输出 20 和 0F 验证结果正确

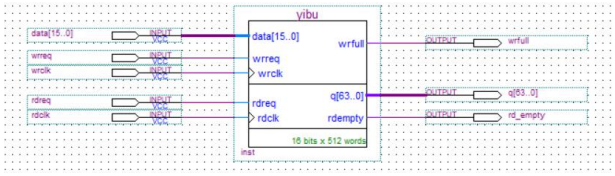
B 级任务

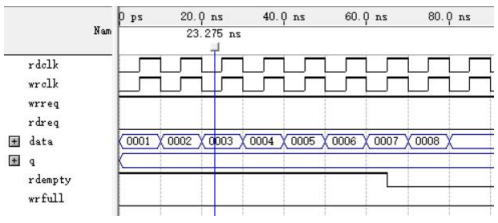
生成波形



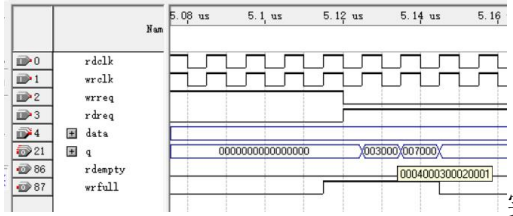
第一个波形 wrreg 信号有效写入 6 个时钟周期 6 个数据，然后 rd 信号有效读出 6 个数据 6->1 第二个波形写满之后 wrfull 为 1，同时 rd 信号有效读出数据，wrfull 信号持续一共三个时钟周期。

Quartus 顶层电路





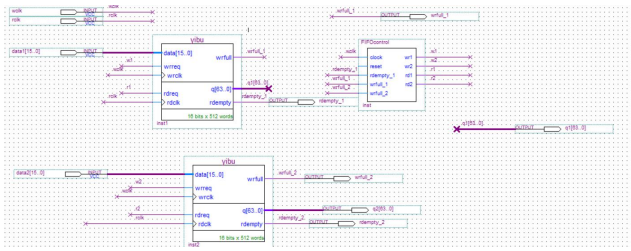
开始写信好有效输入数据 16 位从输入八个 1 到 8 由于容量为 512，时钟周期是 10ns 所以当到达 5.12us 就写满，于是设置读信号有效，一次读出的是 64 位



实验结果符合预期（用的是生成的 vhd 文件）

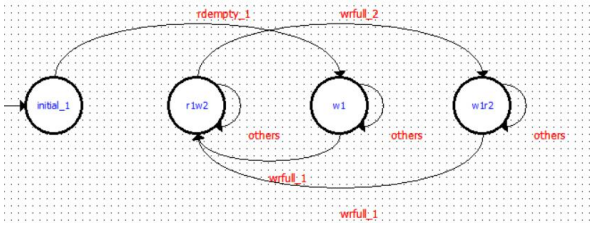
A 级任务

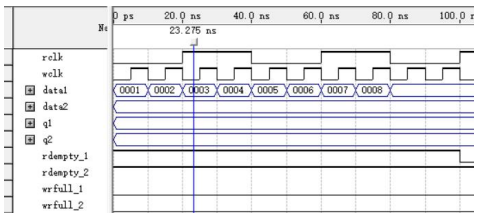
乒乓操作顶层电路



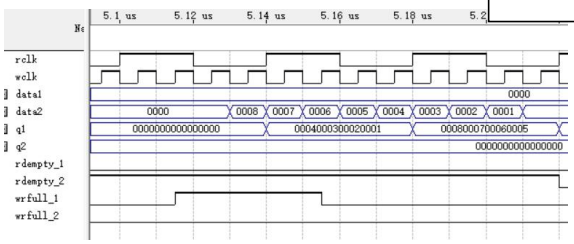
左边两个代表异步的 FIFO 右上角是控制电路，控制两个 FIFO 什么时候读和写（结合 FIFO 的生成波形绘制四个状态图）

FIFO 控制状态图

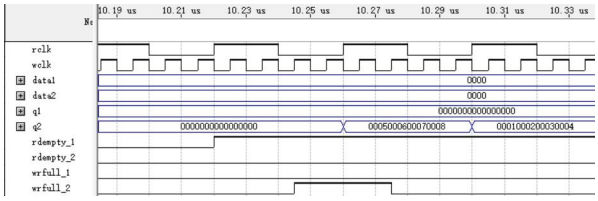




开始写入数据 1->8 之后写入数据 0 一直写到 5.12us 也就是 512 个容量全部写完



此时第一个 FIFO 写满了开始读数据所以读出 q1 同时开始写第二个 FIFO，这时数据设置为 8->1，同样写 512 个容量这个时候时 10.24us 会有现象



这个时候第二个 FIFO 写满了 wrfull_2 为 1 同时读出数据，此外 rdempty_1 为 0 因为当时已经读完了现在要重新开始写

总结：

（思考了一下为什么读的频率是写的频率的 4 倍，因为存在一个数据输出是数据输入的 4 倍，这样做可以方便观察现象，之前实验了一下当读的频率和写的频率一致发生的情况，发现读会保留最后一个数据，也就是你输入的其中一个数据（这和 FIFO 存储有关，也和输出形式有关（我忘了当时保留的是哪个数据了但是我知道是和 FIFO 存储有关，好像保留的是数据 8（如果输入的是 1->8））））然后他就会在第 512/4+512 这个时间点开始读出的 q1 会一直都是 8（前面 15 位 0）本次实验增强了对缓存的理解，了解到 FIFO 的功能和 HM62256 芯片的作用更是掌握了 quartus 自带的定制 RAM 的操作，也加强了对 proteus 软件的使用，但最终因为篇幅限制和能力有限不能挑战 s 级任务，本次实验还是收获满满，任务的渐进程度也能得以接受，在不断探索中学习和摸索，并不会觉得任务之间跨度太大而没有做实验的思路。