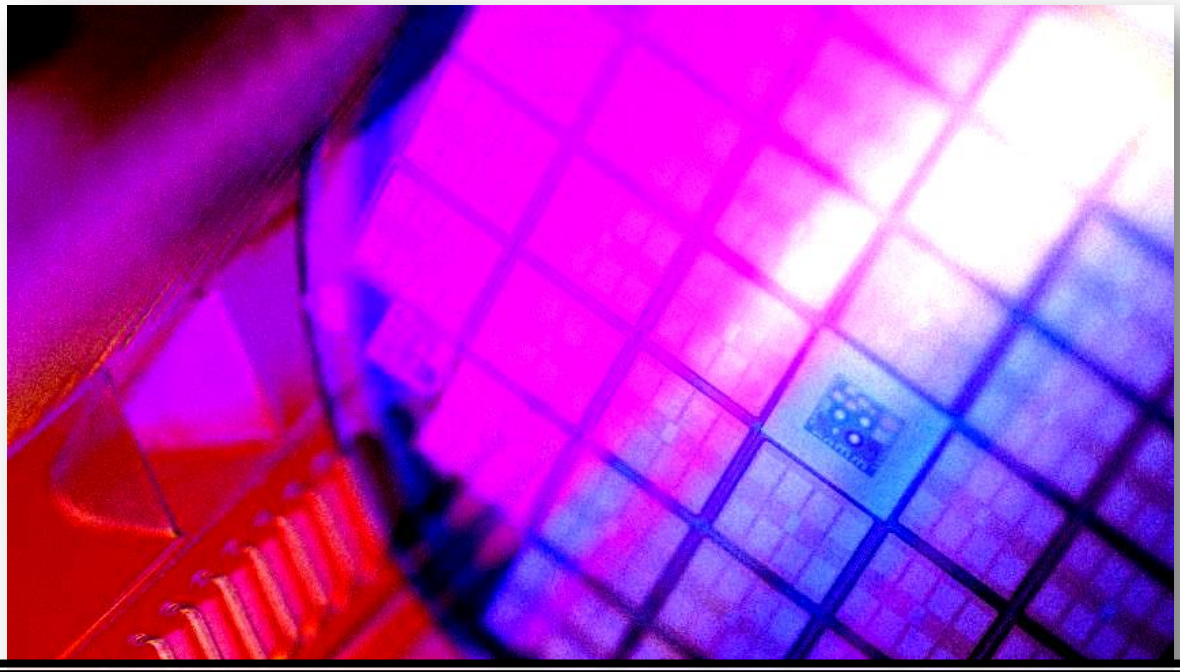
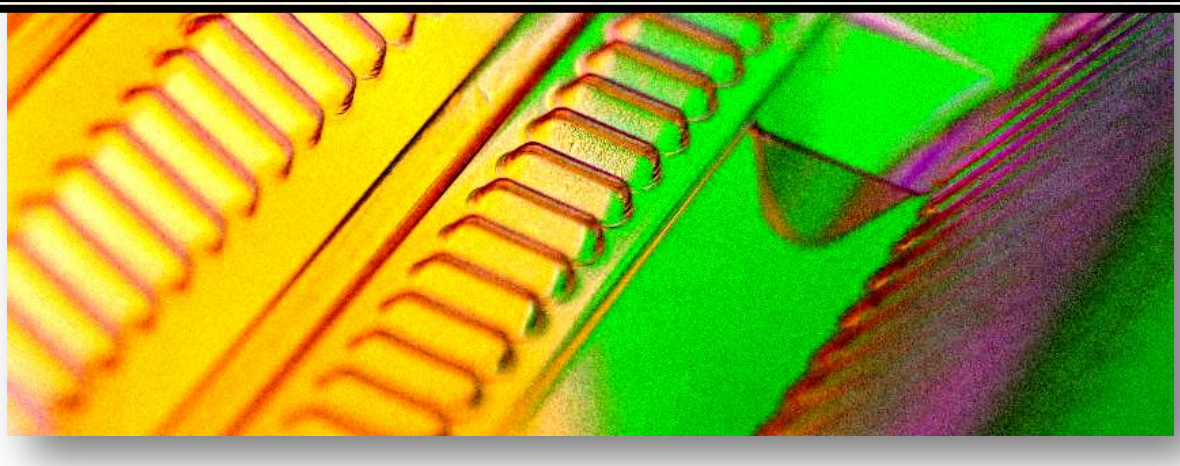
FEC1001T07-V1.0



zrtech

FPGA/CPLD 开发套件实验教程

--仿真，调试，设计篇



WWW.ZR-TECH.COM

实验七、我有存储器—MAXII 的 UFM FLASH

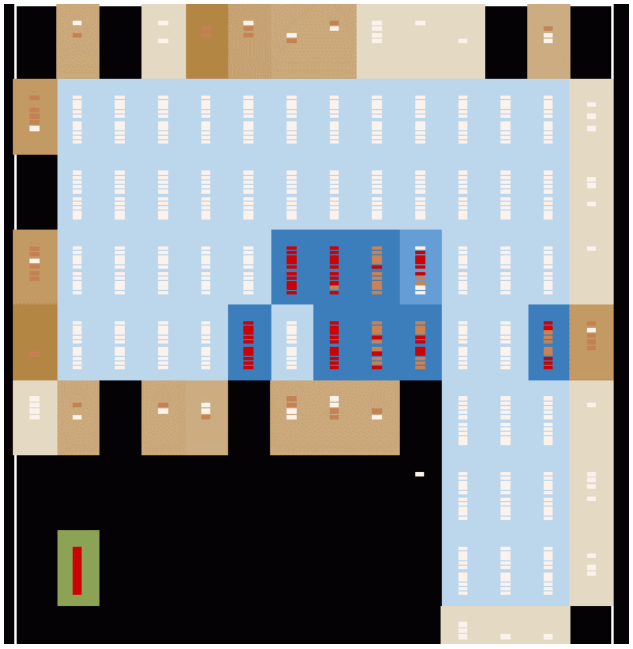
实验目的：

通过这个基础实验，使用户了解 UFM FLASH 的使用方法。

实验原理：

1、 UFM 简介

传闻说 CPLD 有个缺陷，就是内部没有存储模块，所以不能对 RAM,ROM 等操作，但其实，MAXII 的 CPLD 内部有 8192bits 的 UFM—User Flash Memory，很容易就可以实现 RAM、ROM 的功能，能满足一时的需要。



如上图所示，MAX II 芯片 Chip Planner： 左下角这块黑色区域是用户不可用资源区，而在这片不可用区域里有一块绿色的方块是可用的。这块不可用的黑色区域叫做 CFM block(配置 Flash 存储区)，而那个绿色方块叫做 UFM（用户可用的 Flash 存储区）。

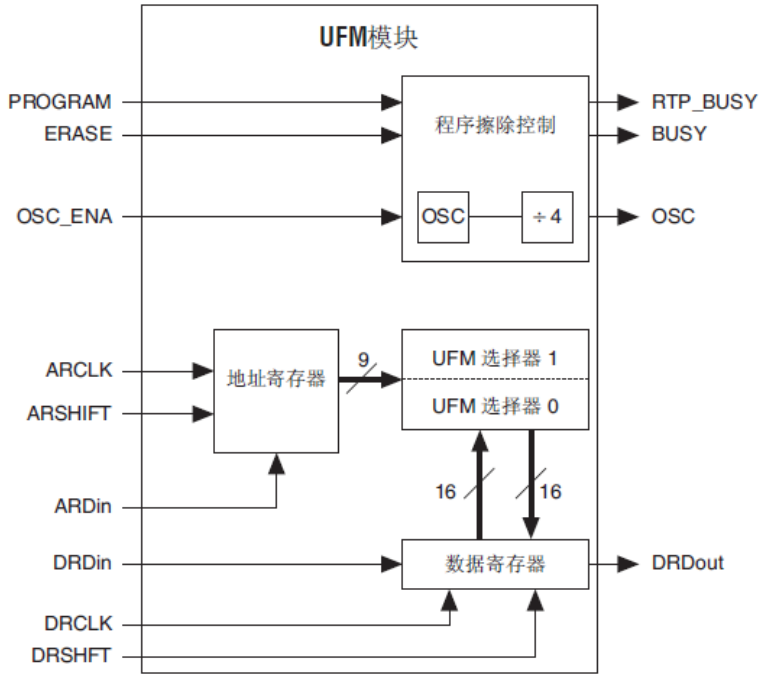
以下是 UFM 的实际应用和特性：

■可以利用 UFM 来存储关键的非易失信息。

■其特性包括并口和串口，例如串行通信接口(SCI)、串行外设接口(SPI)、内置集成电路(I2C)、Microwire 等其他专用协议。MAX II 器件在接口上比商用 EEPROM 器件更加灵活。

■ UFM 的内部振荡器能够满足所有设计的时钟需求；它不需要外部时钟电路，因此节省了空间和成本。在设计电路板时，您可以采用 MAX II CPLD 来合并逻辑和存储器，从而缩短了芯片至芯片延时，减小了电路板面积，降低了系统总成本。UFM

划分成两个区，每个 4 Kbits。地址寄存器指示数据写入和读出的 UFM 存储器地址。数据寄存器保存向 UFM 写入或者读出的数据。程序擦除控制模块用于对 UFM 的编程或者擦除，还可以使能内部振荡器。下图所示为 UFM 的结构图。



UFM 被分成两个区：0 区和 1 区。每一区都是 4096 比特，地址范围分别为 000h 到 0FFh 和 100h 到 1FFh。地址长度为 9 比特，每一地址位置可存储 8 位数据。通过读/ 数据流读、编程和擦除操作来装入并修改 UFM 的数据。读/ 数据流读操作用于读取地址寄存器所指向地址的内容。递增地址寄存器(数据流读) 可实现对存储器的连续读操作。编程操作用于把数据装入 UFM，而修改 UFM 内容时则采用擦除操作。但是，不能只擦除单个地址位置。擦除操作要么擦除整个 UFM (A2A1A0 = 111)，要么擦除地址 MSB 所指向的 UFM 区。UFM 还含有一个可以使能的内部振荡器。通过逻辑阵列模块来传送其输出信号，还可以把该信号反馈回 ARCLK 和 DRCLK。在 Quartus II 对于这块存储区读写接口 altera 提供了四种通用的接口供用户选择。

■ I2C

■ SPI

■ Parallel

■ None (Altera Serial Interface)

最后一种就是不需要占用器件额外逻辑资源的接口，上面三种是需要消耗器件逻辑资源的接口，在 EPM240 中实现不太经济，在此我们选择 Parallel 并行传输模式，一是简单，而是高速，没必要去写 SPI、I2C 这些在 verilog 中书写复杂的协议。 此时并行模式需要的连线如下表：

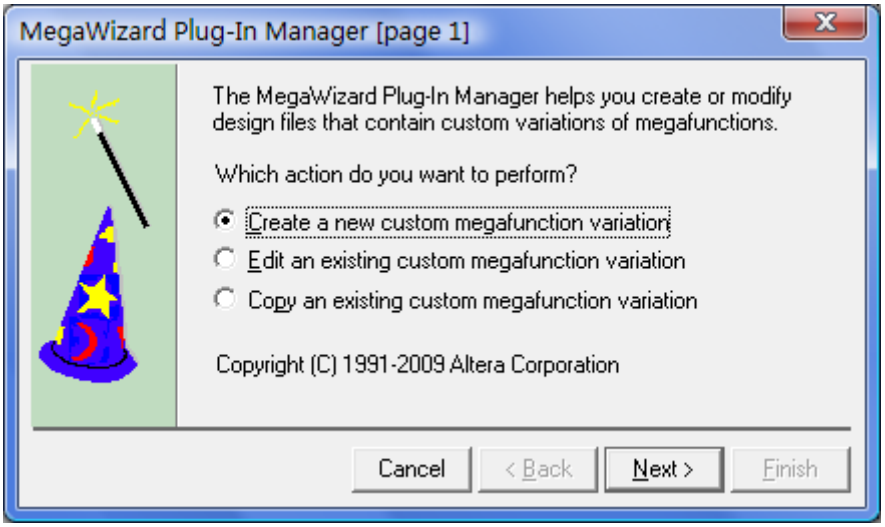
Pin	Description	Function
DI[15:0]	16-bit data Input	Receive 16-bit data in parallel. You can select an optional width of 3 to 16 bits using the altufm megafunction.
DO[15:0]	16-bit data Output	Transmit 16-bit data in parallel. You can select an optional width of 3 to 16 bits using the altufm megafunction.
ADDR[8:0]	Address Register	Operation sequence refers to the data that is pointed to by the address register. You can determine the address bus width using the altufm megafunction.
nREAD	READ Instruction Signal	Initiates a read sequence.
nWRITE	WRITE Instruction Signal	Initiates a write sequence.
nERASE	ERASE Instruction Signal	Initiates a SECTOR-ERASE sequence indicated by the MSB of the ADDR [] port.
nBUSY	BUSY Signal	Driven low to notify that it is not available to respond to any further request.
DATA_VALID	Data Valid	Driven high to indicate that the data at the DO port is the valid data from the last read address for read request.

实验结果：

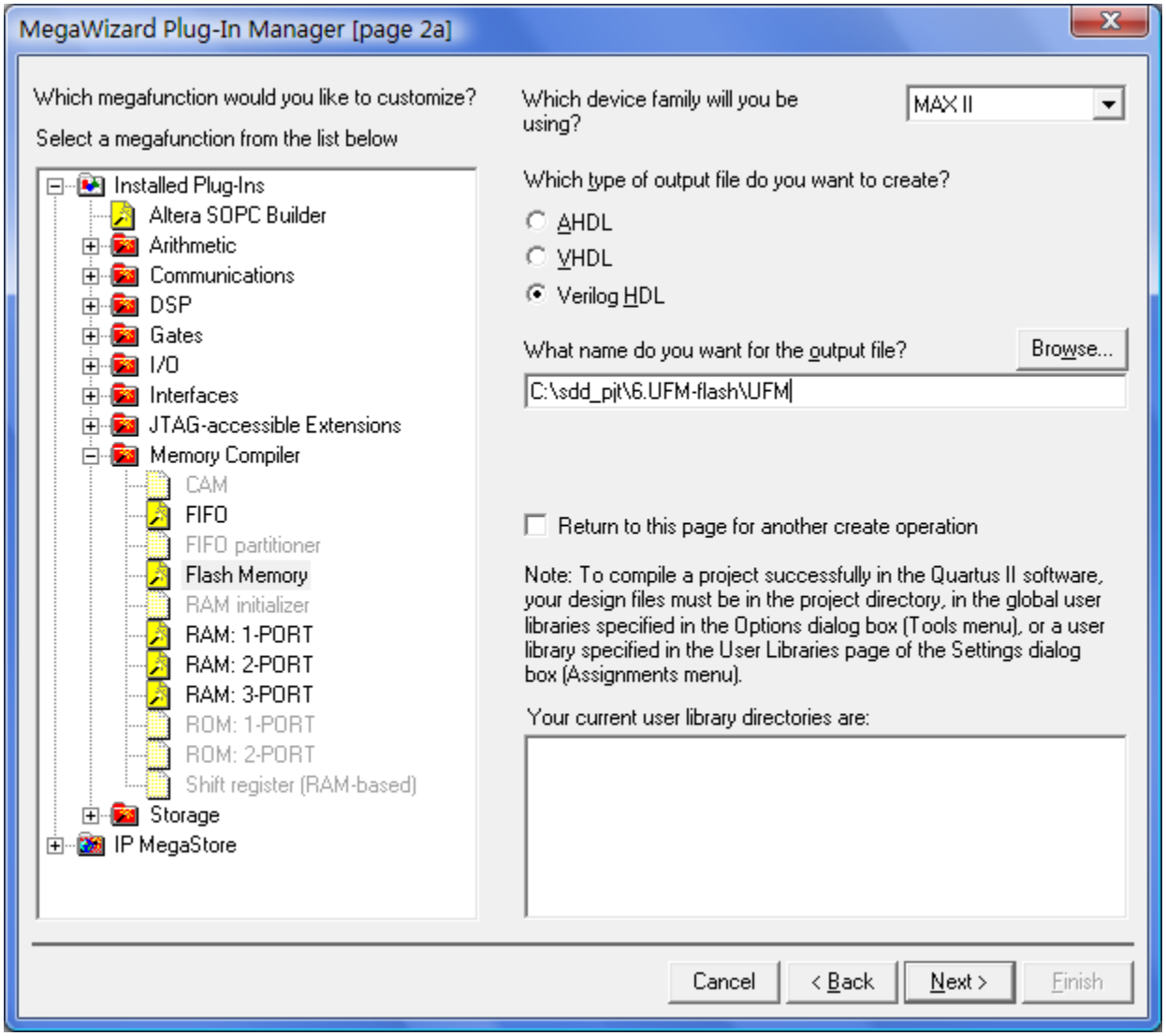
使用 modelsim 仿真测试 Parallel 并行接口 UFM FLASH。

具体步骤：

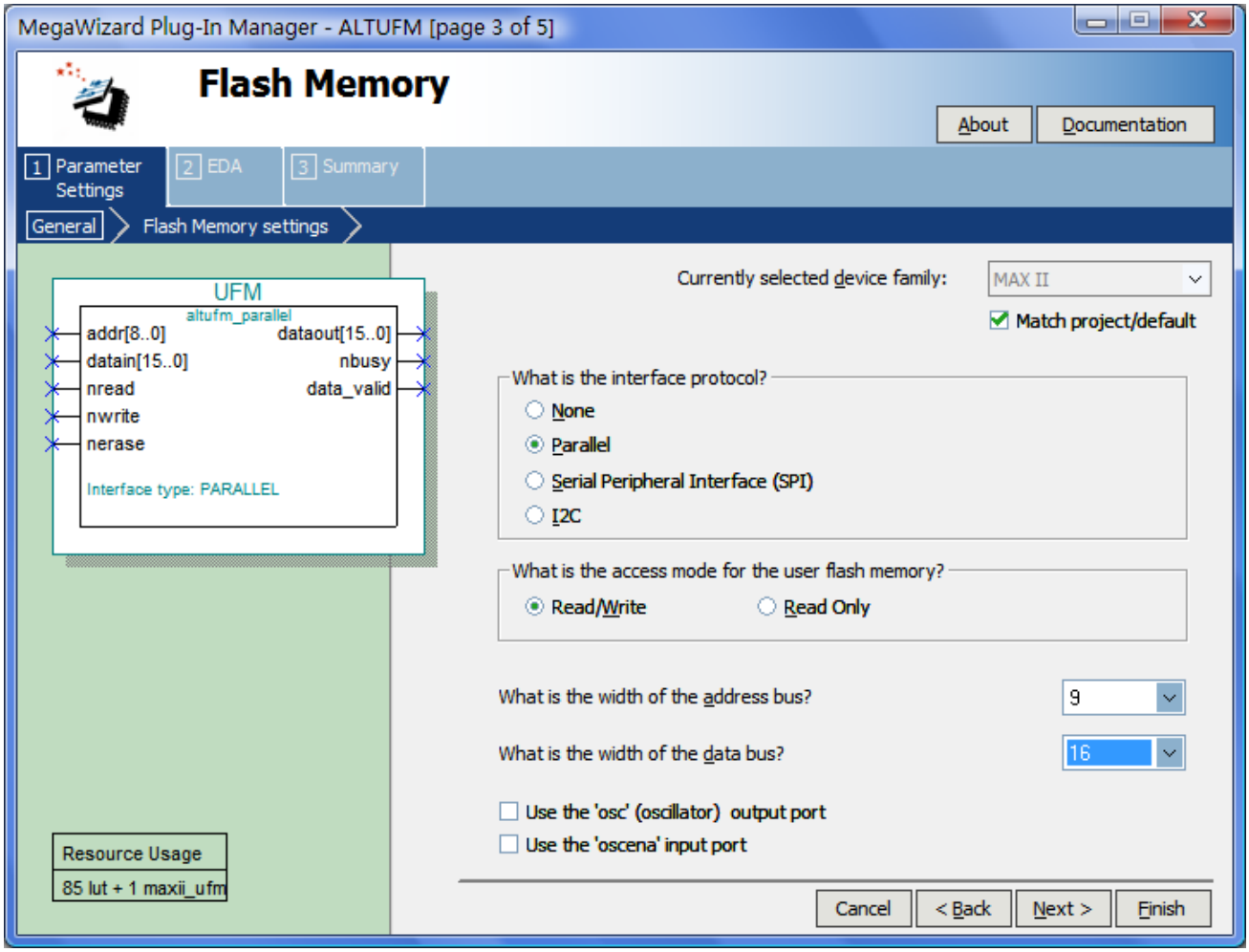
新建工程，加入 bdf 作为顶层文件，点击 Tool—Megawizard Plug-In Manager



接着选择 Memory Compiler 下的 Flash Memory，然后在 What name do you want for the output file?下路径的最后添加输出文件名为 UFM,点击 next.



选择并行 Paralell 只读模式 ,并制定位宽。



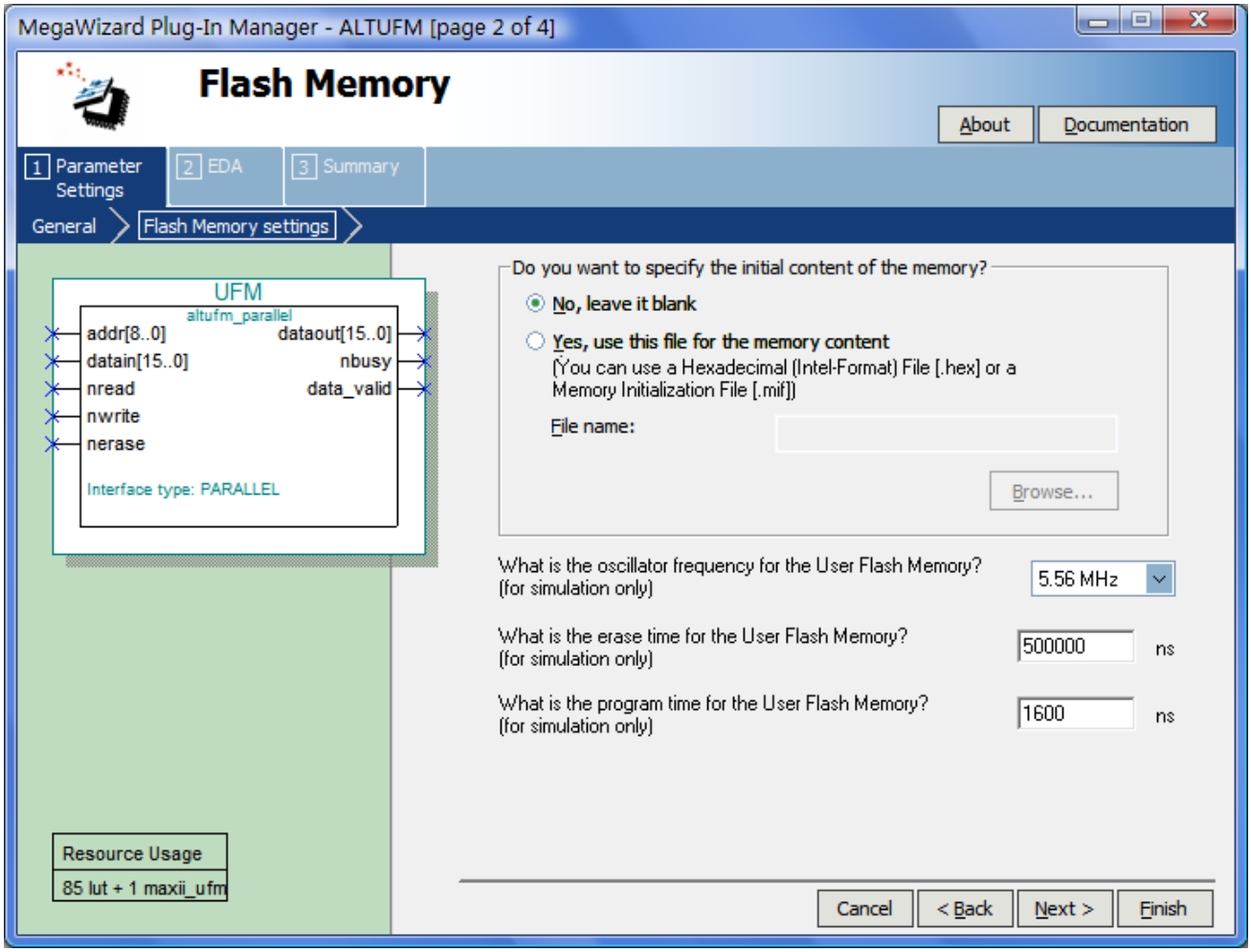
选择：

<1>晶振 Oscillator：5.56MHZ 或者 3.33MHZ(再次测试我用的是 5.56MHZ)

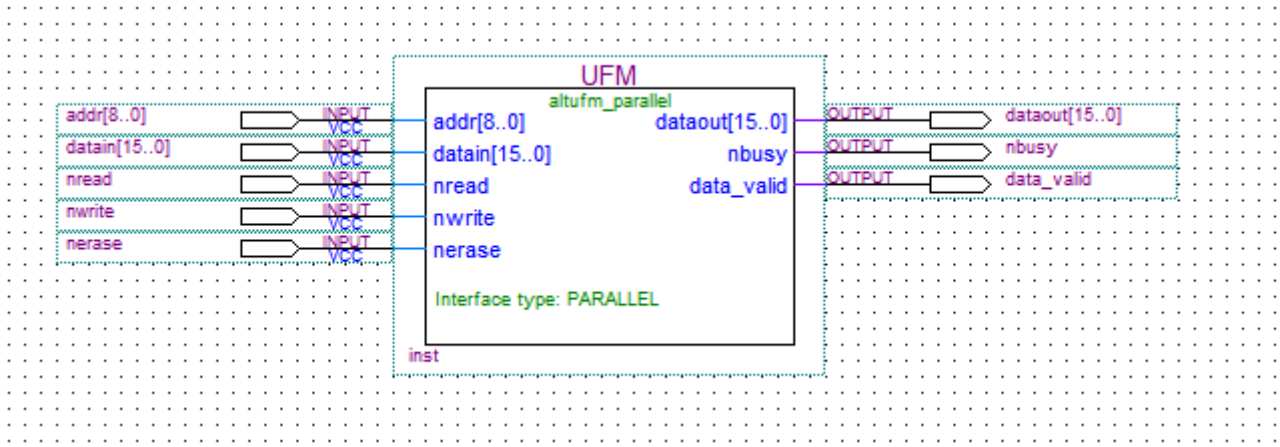
<2>擦除 Flash 的时间：默认 500000ns

<3>Flash 编程时间：默认 1600ns

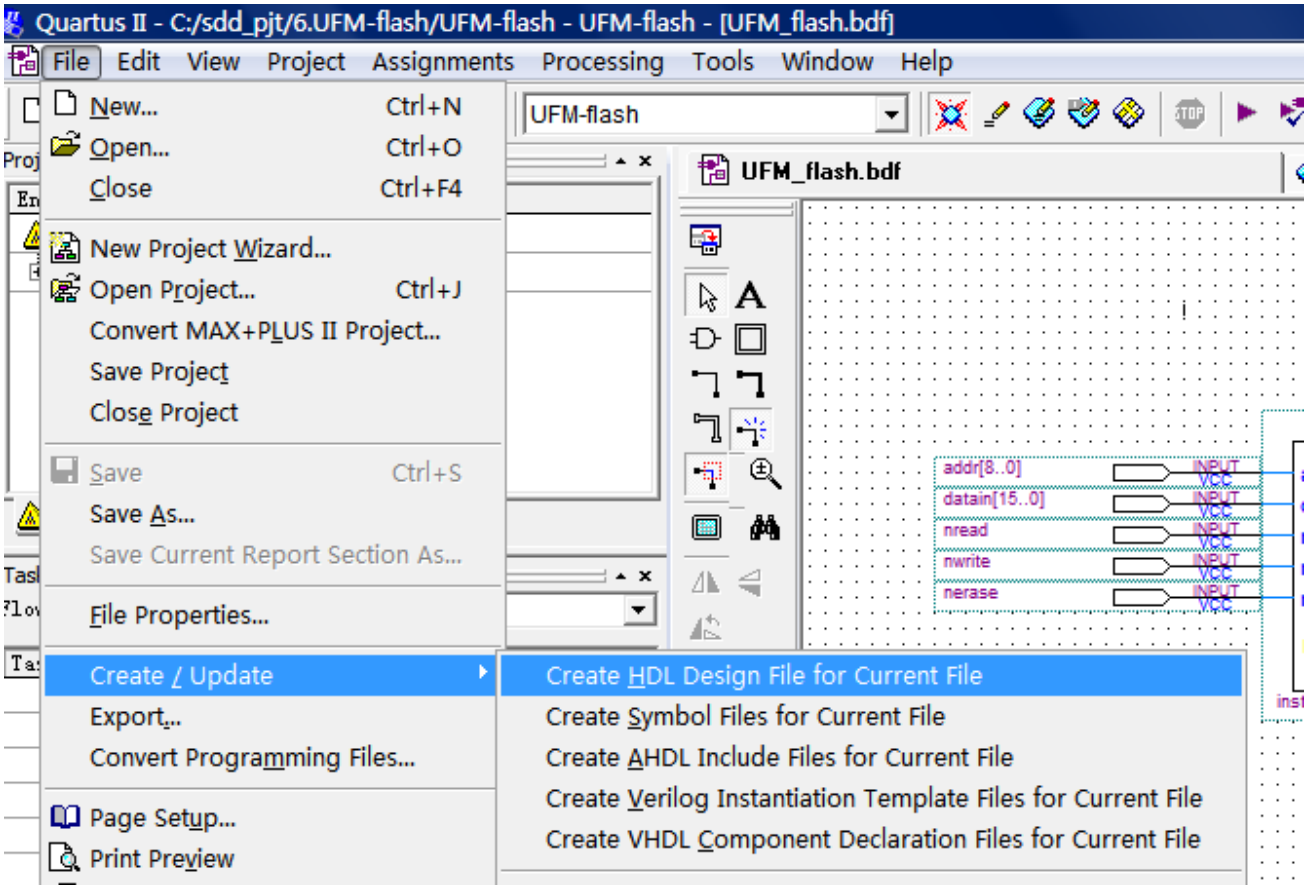
<4> File Name 中留空，不初始化。



点击下一步，结束，然后新建一个 bdf，加入 UFM 模块，并添加管脚



生成对应的.v 文件，这个文件是需要在 modelsim 里仿真使用的。



下面编写 UFM_flash_test.v 测试程序，程序的内容很简单，即在 0 地址写入数据，然后读出 0 地址的数据。

```
`timescale 1ns/1ns
module UFM_flash_test();

//input
wire[15:0] dataout;          //Flash 数据总线
wire data_valid;         //Flash 数据输出有效信号
wire nbusy;                  //Flash 忙信号

//output
reg[15:0] datain;          //Flash 数据总线
reg[8:0] addr;               //Flash 地址总线
reg nerase;                  //擦除 Flash 某一扇区信号
reg nread;                 //读 Flash 信号
reg nwrite;                  //写 Flash 信号
```

```
 UFM_flash       UFM_flash_t(
                 .datain(datain),
                 .dataout(dataout),
                 .addr(addr),
                 .nerase(nerase),
                 .nread(nread),
                 .nwrite(nwrite),
                 .data_valid(data_valid),
                 .nbusy(nbusy)
               );

parameter   DELAY_600US     = 600_000,       //600us 延时
            DELAY_5US    = 5_000;          //5us 延时

initial begin
        nerase = 1;
        nread = 1;
        nwrite = 1;
        addr = 0;
    #DELAY_600US; //0 地址写入数据 99
        datain = 99;
        addr = 9'd0;
        nwrite = 0;
        @ (posedge nbusy);
        #DELAY_5US;
        nwrite = 1;
    #DELAY_5US;     //0 地址读出数据，
        addr = 9'd0;
        nread = 0;
        @ (posedge data_valid);
        #DELAY_5US;
        nread = 1;
    #DELAY_600US;
         $stop;

end
```

UFM 接口模块实际上是在 nread 信号的上升沿锁存地址数据，在 nread 的下降沿开始读过程 ，写操作亦类似，具体参考如下波形图。

Figure 9–40. Parallel Interface Timing Waveform

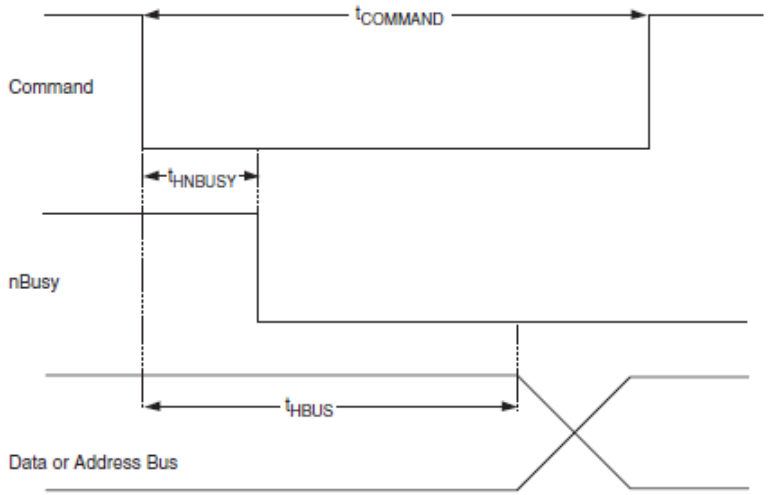
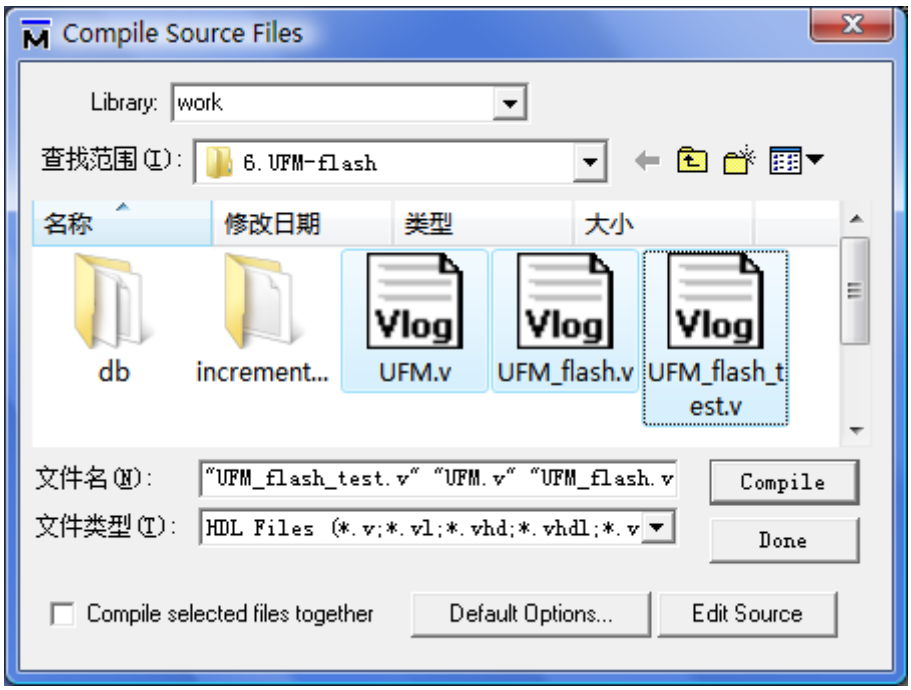


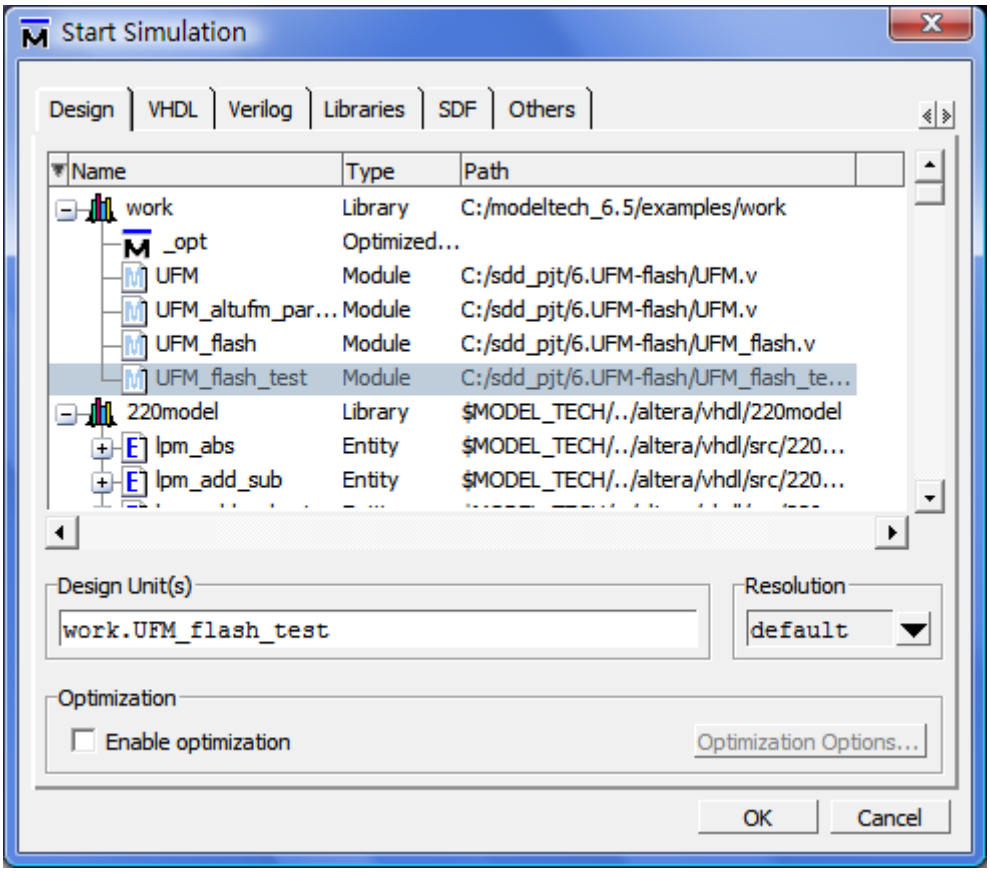
Table 9–16. Parallel Interface Timing Parameters

Symbol	Description	Minimum (ns)	Maximum (ns)
$t_{COMMAND}$	The time required for the command signal (nREAD/nWRITE/nERASE) to be asserted and held low to initiate a read/write/erase sequence	600	3,000
t_{HNBUSY}	Maximum delay between command signal's falling edge to the nBUSY signal's falling edge	—	300
t_{HBUS}	The time that data and/or address bus must be present at the data input and/or address register port after the command signal has been asserted low	600	—

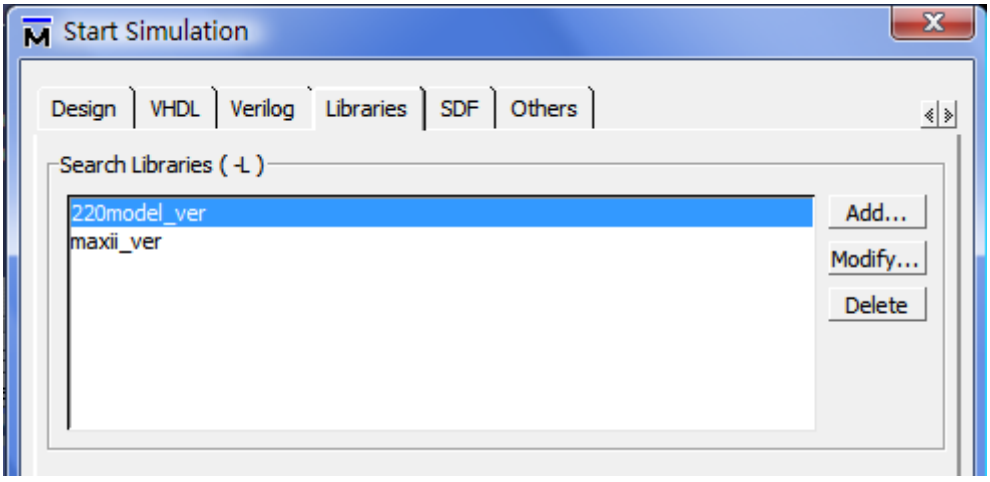
打开 modelsim，加入顶层例化程序，UFM 的源程序以及测试程序进行编译



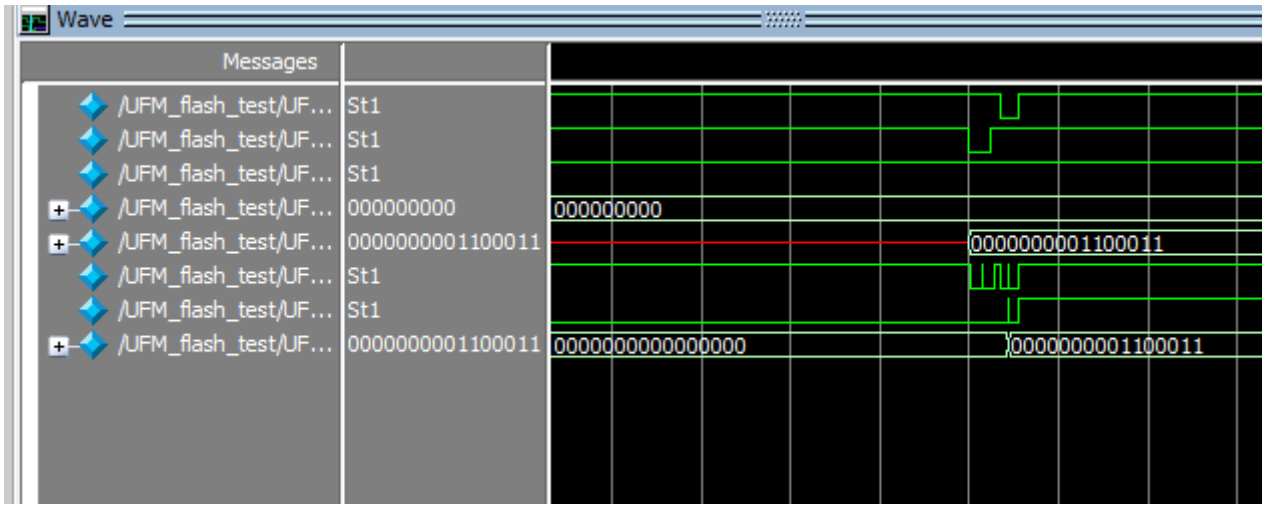
点击工具栏上的 Simulation->Start simulation，选择测试激励文件



加入与程序相关的编译库，点击 OK，开始仿真。



加入信号，查看最后的仿真结果：



可见数据顺利的被写入，并且被正确读出了。

实验总结：

MAXII的UFM是一个非常有趣的模块，其不但可以作为FLASH使用，也可以作为RAM进行读写操作。这弥补了其内部没有RAM模块的缺点。

课后作业：

试着用MAXII UFM存一些程序中需要的非易失数据，用实际逻辑去读出这些数据，此外可以尝试一下另外几种访问UFM的方法，如SPI,I2C等。

文档内部编号：FEC1001T07

编号说明：
首一字母：F-FPGA系列
首二字母：L-理论类 E-实验类 T-专题类
首三字母：C-普及类 Q-逻辑类 S-软核类
数字前两位：代表年度
数字后两位：同类文档顺序编号
尾字母/数字：C目录，T正文，数字表示章节号

修订记录

版本号	日期	描述	修改人
1.0	11.1.2	初稿完成	左超