11 IP 调用之 ROM

工程源码	
相关视频课程	

章节导读

ROM:（只读内存(Read-Only Memory)简称）英文简称 ROM。ROM 所存数据，一般是先写好的，工作过程中只能读出，而不像随机存储器那样能快速、方便地加以改写。ROM 所存数据稳定，断电后所存数据也不会改变。

11.1 实验任务

本章我们将对 PDS 软件生成的 ROM IP 核进行读出测试，并向大家介绍 ROM IP 核的使用方法。在 ROM 存储器中存储一组固定的数据（正弦波形表），然后通过仿真以及板级调试的方法，查看 ROM 存储器的数据读取内容和时序，从而掌握 ROM 存储器 IP 的使用方法。

新建 PDS 工程，首先在菜单栏里选择“Tools”然后单击 “IP Compiler”选项，如图 11-1 所示。

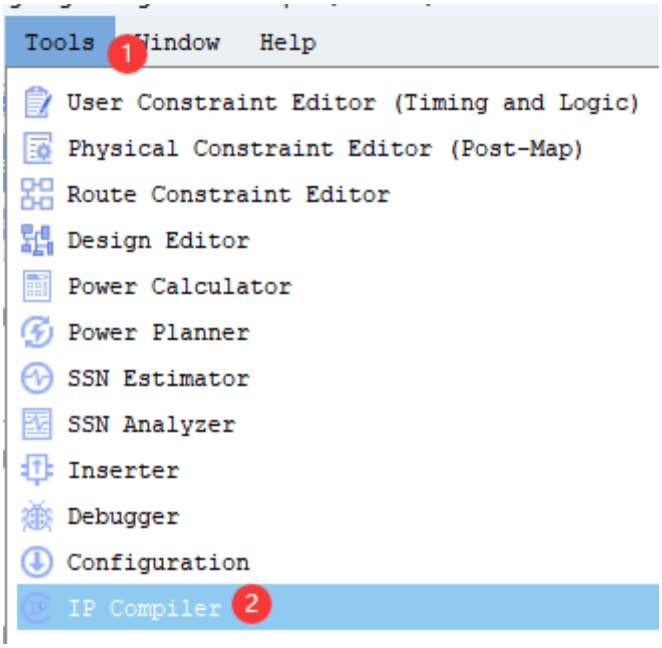


图 11-1“IP Compiler”页面

点击图 11-1 中的“IP Compiler”后会跳转图 11-2 页面。在图 11-2 中我们

可以看到有两个与 ROM 有关的 IP，一个是 “Distributed ROM”，另一个是“DRM Based ROM”。先简单说说两个的差别，两者最主要的差别是生成的 Core 所占用的 FPGA 资源不一样。从“Distributed ROM”生成的 ROM Core 占用的资源是 LUT（查找表，查找表本质就是一个小的 RAM）；而“DRM Based ROM”生成的 ROM Core 占用的资源是 Block Memory（嵌入式的硬件 RAM）。

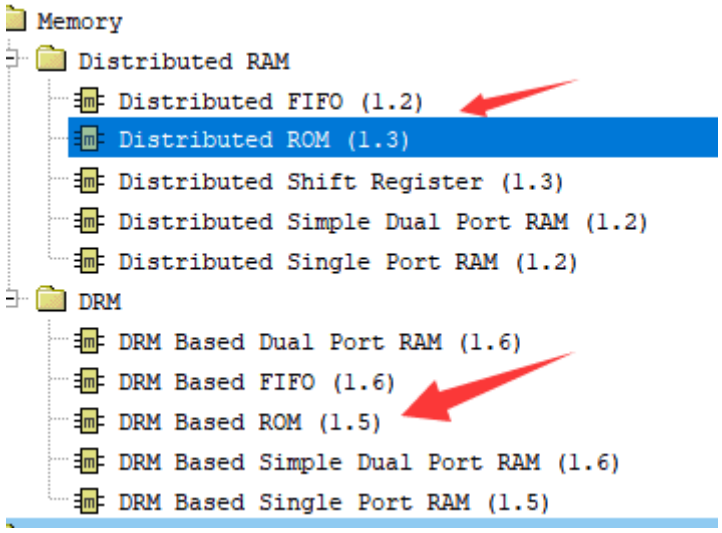


图 11-2 ROM IP 核类型

本章实验本节主要讲解 ROM 的使用，由于加载的 dat 文件占用使用 Core 资源较少，所以选择“Distributed ROM”去生成 ROM IP。 ROM IP 生成时，通过 PDS 加载一个内存初始化文件（dat 文件）来初始化该 ROM IP，这样 ROM 存储的数据就固定了。我们选择“Distributed ROM”，首先我们点击图 11-3 页面中标签 1 中的“Distributed ROM”按钮，选择创建 ROM 核。然后将标签 2 中“Instance Name”选项命名为“rom”。最后点击标签 3 中的“Customize”按钮进入 ROM IP 参数配置页面。

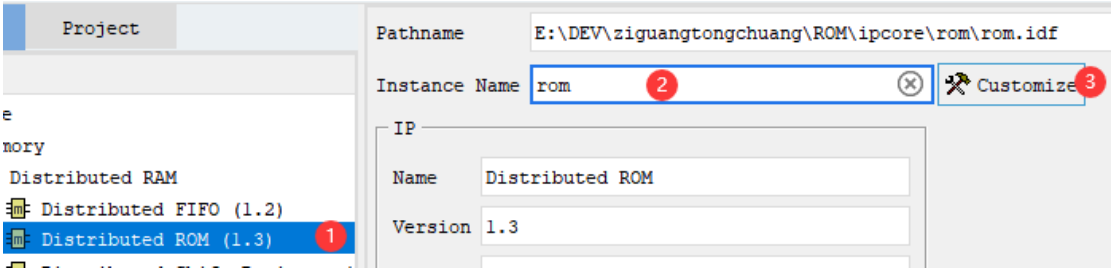


图 11-3 IP 选择页面

点击图 11-3 中的“Customize”按钮进入图 11-4 中。

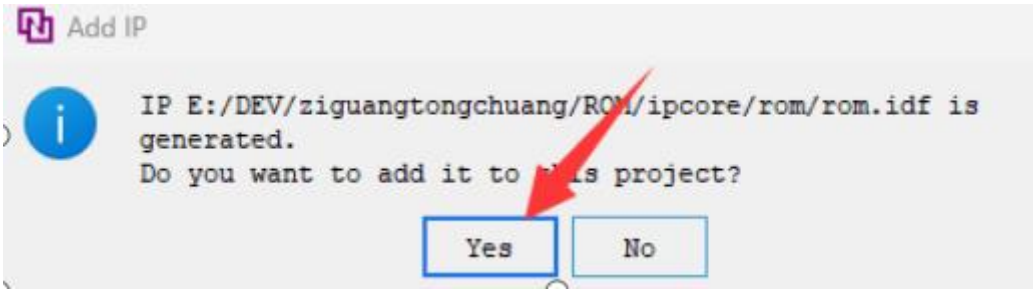


图 11-4“Add IP 弹窗

图 11-4 中表示的是 rom.idf 文件已经生成，询问我们是否将该文件加载到我们的工程中。因为这个 IP 我们后续是要在工程中用到的，因此这里直接点击图 11-4 中的“Yes”按钮即可。然后软件会自动进入 ROM 存储器的参数配置界面，如图 11-5 所示。

1. “Address Width”该项指定了 ROM 存储器的地址宽度，地址宽度越大，存储器的容量也就越大。本案例，我们设置位宽为 10，则 rom 存储的存储容量为 1024 个存储单元；
2. 数据位宽，代表了 ROM 存储器每个地址（存储单元）中存储的数据的位宽，可根据实际应用需求，在 1 位到 256 位之间任意设定；
3. 标签 3 的位置是加载初始化文件存放的位置（初始化文件是我们公司提供的.dat 文件，用于产生正弦波）；
4. 选择初始化文件里数据的格式，这里我们选择十六进制；
5. 点击图 11-5 中的“Generate”按钮；
6. 方框中的“Enable Output Register”表示的使能控制输出寄存器，勾选该选项后会对输出数据增加一级寄存器，这里我们不做勾选。

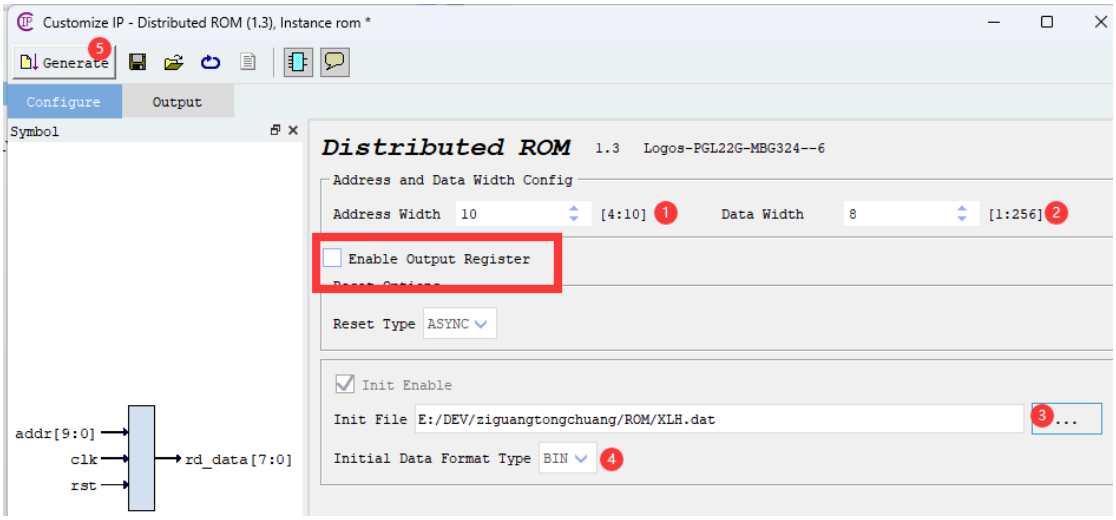


图 11-5 IP 设置页面

之后会弹出图 11-6，图 11-6 中表示的是：对 IP 核设置的数据将会保存，是否继续操作。我们继续点击图 11-6 中箭头指向的“OK”按钮。

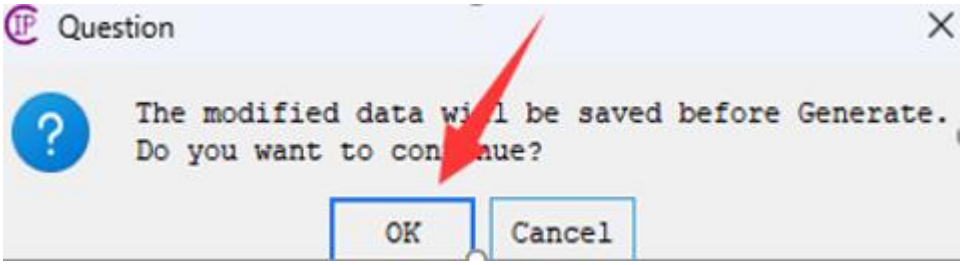


图 11-6“Question”页面

最终会弹出图 11-7，至此 ROM IP 核配置成功。

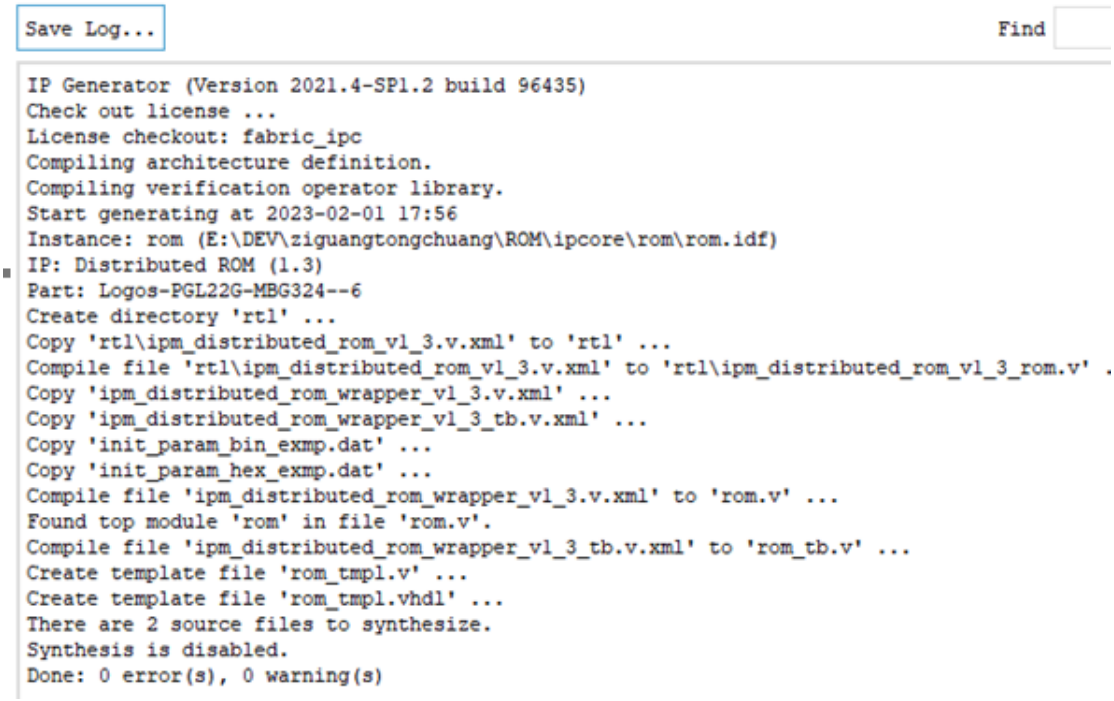


图 11-7 ROM IP 配置成功

IP 核配置成功后会自动弹出图 11-8 即这个 IP 的例化模板文件，接下来我们需要例化这些自动生成的代码。

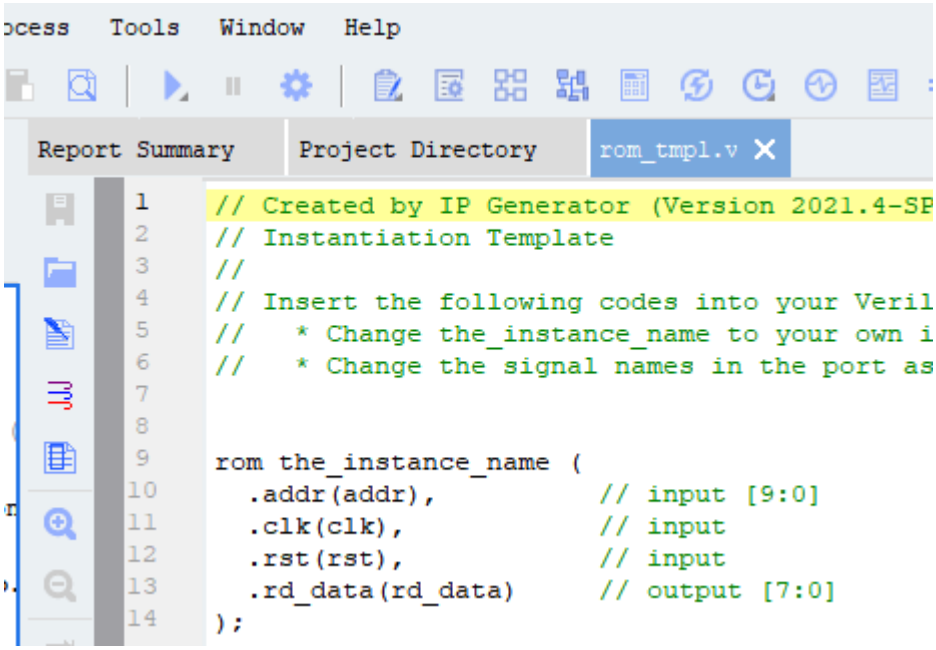


图 11-8 例化模板文件

将 IP 核配置过程中弹出的界面关闭，返回 Source 面板，如图 11-9 所示。

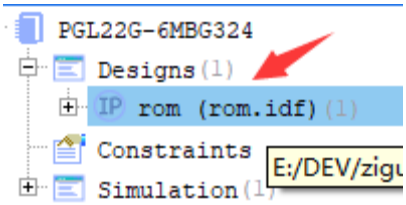


图 11-9 Source 面板

点击图 11-9 页面中 ROM IP 左侧加号位置，可以看到 ROM IP 核下有两个文件，点击图 11-10 中箭头所指的位置，打开 rom.v 文件。

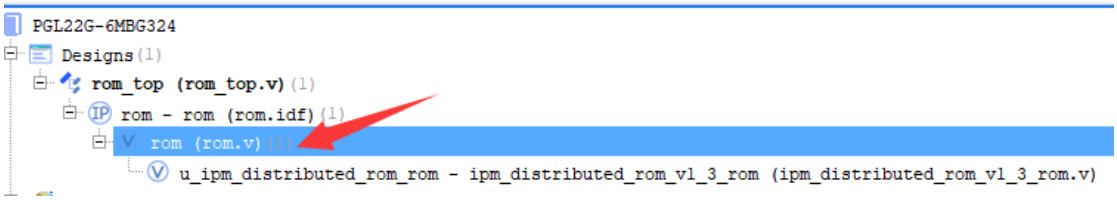
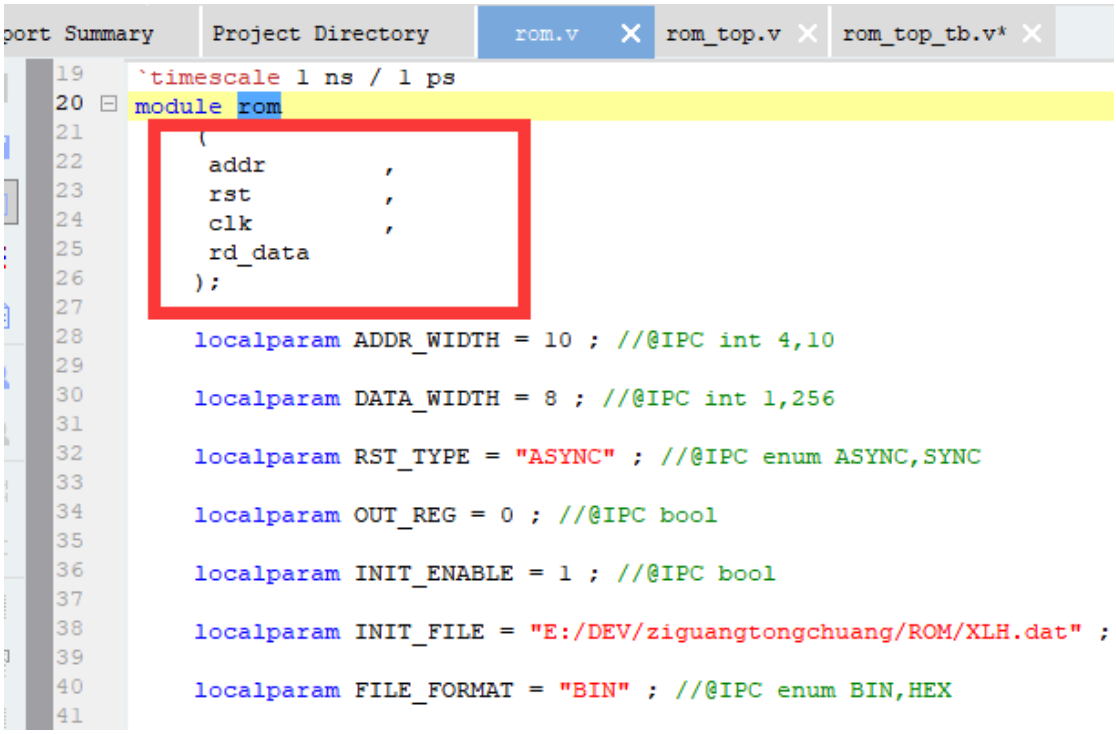


图 11-10 IP 核包含的文件

部分 rom.v 文件内容如图 11-11 所示，若要调用 ROM IP，我们可以选择将



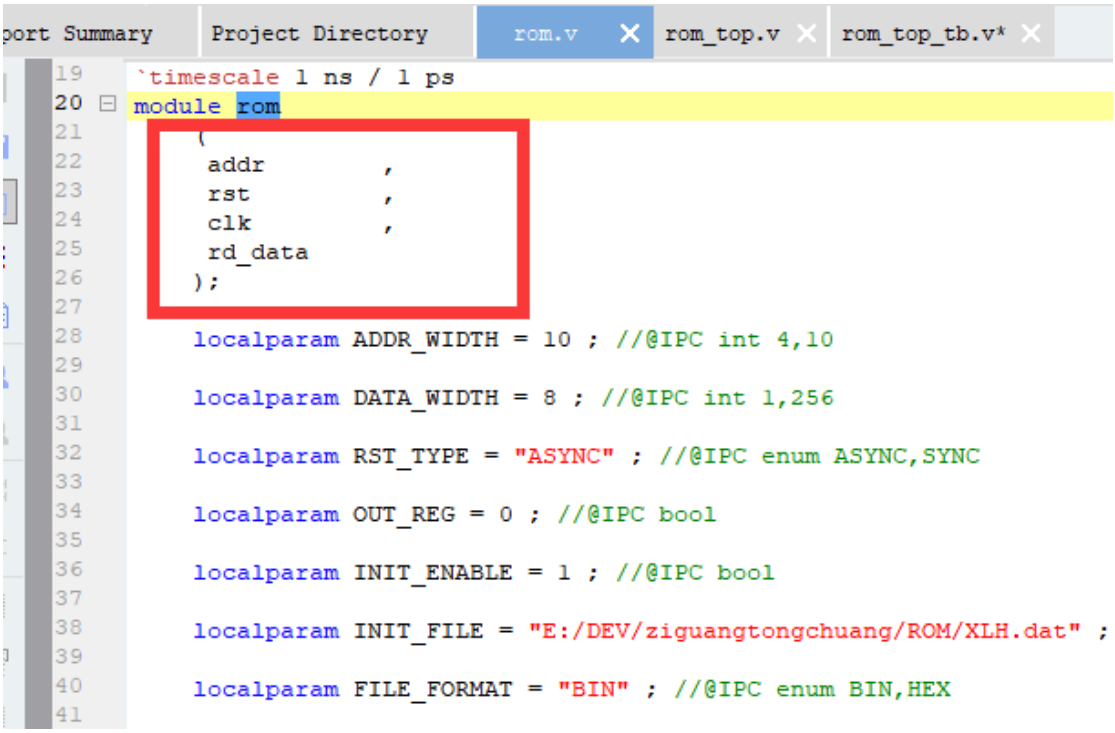


图 11-11 中的端口例化到需要调用 ROM IP 的模块中，或者是直接例化图 11-8 中内容。

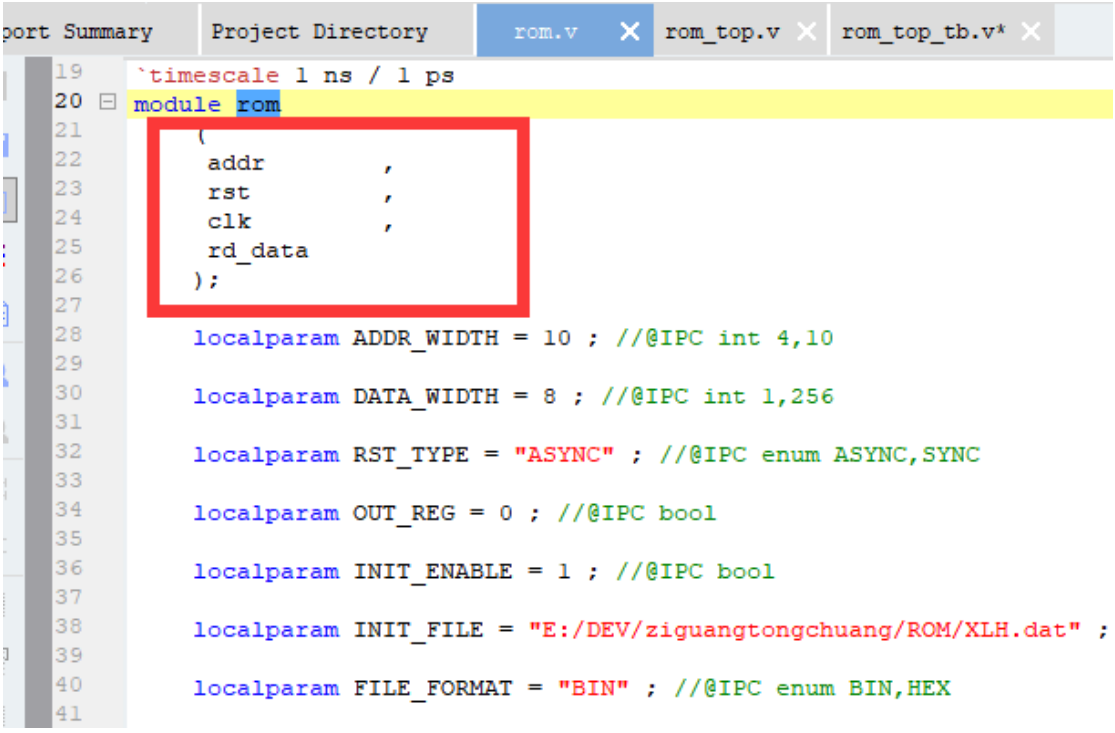


图 11-11rom.v 文件内容

接下来我们创建一个 verilog 源文件，其名称为 rom_top，代码如下：

```
module rom_top(
    clk,
    reset_n,
    dout
);

  input clk;
  input reset_n;
  reg [9:0]addr;
  output [7:0]dout;

always@(posedge clk or negedge reset_n)
  if(!reset_n)
    addr<=0;
  else if(addr==10'd1023)
    addr<=0;
  else
    addr<=addr+1'b1;

rom rom  (
  .addr(addr),          // input [9:0]
  .clk(clk),            // input
  .rst(~reset_n),            // input
  .rd_data(dout)     // output [7:0]
);
endmodule
```

程序中例化了 ROM IP 并且还定义了一个 addr 变量，addr 变量的计数范围在 0~1023，所以将 addr 变量设置成从 0 自加到 1023。

11.2激励创建及仿真测试

当设计文件创建好之后，我们可以通过仿真方式来对顶层模块进行测试，进而分析和验证该 IP 在逻辑设计时的读取方法以及读取时序。这里编写一个简单的 testbench 来进行仿真测试，testbench 代码如下所示。

```
`timescale 1ns/1ns
`define CLK_PERIOD 20

module rom_top_tb();
```

```
 reg clk;
 reg  reset_n;
 wire [7:0]dout;

initial clk=1;
always #(`CLK_PERIOD/2) clk = ~clk;

initial begin
   reset_n=0;
   #21;
   reset_n=1;
   #2000000;
  end

rom_top rom_top(
  .clk(clk),
    .reset_n(reset_n),
  .dout(dout)
);
endmodule
```

由于 ROM 的最大数据个数是 1024，所以将 addr 设置成自加到 1023， 实现了地址数从 0 自加到 1023。

在上面的代码中，我们设置完初始值以后，延时时间为#21 而不是时钟周期的整数倍，目的是为了让地址变化避开时钟沿的变化，从而能够在行为级层面稳定的模拟仿真地址变化。

如何对模块仿真这里我们不再赘述，仿真波形如图 11-12 所示：

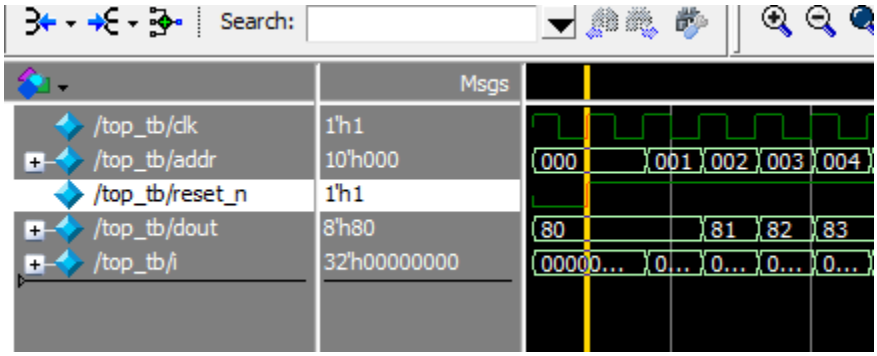


图 11-12 仿真输出数据波形

从图 11-12 中可以看出，当复位信号“reset_n”拉高时 “addr”信号开始发生变化。“addr”信号为 0 时对应“dout”信号的第一个数据；“addr”信号为 1 时对应“dout”信号的第二个数据，依次类推。这是因为在图 11-5 中的“Enable Output Register”选项我们没有勾选，文件中的数据与图 11-12 中仿真输出的数据一致。通过上述分析，可以确认在给出 addr 值后，对于分布式 rom，如果不加上输出寄存器，dout 的值会立即更新为该 addr 指定的存储器中的值。对于功能仿真，不考虑门级延迟的情况，仿真波形中的与 addr 对齐的位置的 dout 端口上的值就是该 addr 的存储器中存储的数据。而对于实际 fpga 中，由于存在门级延迟，所以 dout 的值会比 addr 的值变化稍晚一点。

ROM IP 加载的文件内容如图 11-13 所示。文件中的数据与图 11-12 中仿真输出的数据一致。

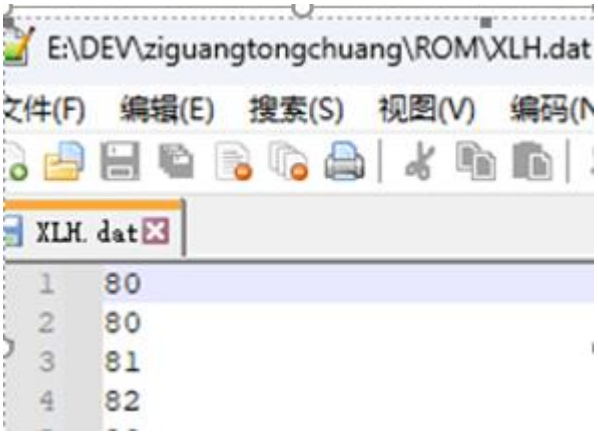


图 11-13 文件内容

对于时序性能要求较高的场合，可以通过对输出加上寄存器的方法，让存储器的数据经过一级 D 触发器后再输出，这样就能缩短数据从存储器到目的寄存器的传输路径，优化整体设计的时序性能。下图为勾选“Enable Output Register”选项即开启输出寄存器后的仿真波形图，如图 11-14。我们可以看到当复位信号“reset_n”拉高时 “addr”信号开始发生变化。“addr”信号为 1 时对应“dout”信号的第一个数据；“addr”信号为 2 时对应“dout”信号的第二个数据，依次类推。使用输出寄存器，虽然能够提升系统时序性能，但是也对数据的输出引入了一拍的延迟（也称潜伏期），在使用 rom 时，如果用到了输出寄存器，需要特别注意这个时序差别。

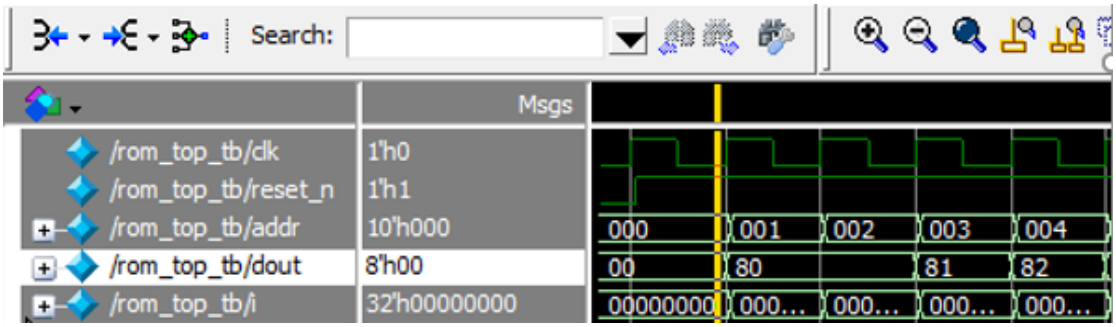


图 11-14 仿真输出数据波形

通过上述仿真波形分析，我们已经了解并验证了分布式 ROM 存储器的接口时序。由于我们在 ROM 中存储的数据为一个正弦波的数据内容，我们还可以借助 Modelsim 的模拟显示功能，将 rom 的输出数据以模拟波形的形式进行展示，以让数据效果更加直观。

右击图 11-15 中信号“dout”，选择“Format”选项，然后选择“Analog(automatic)”选项，将输出波形转化为模拟量形式。

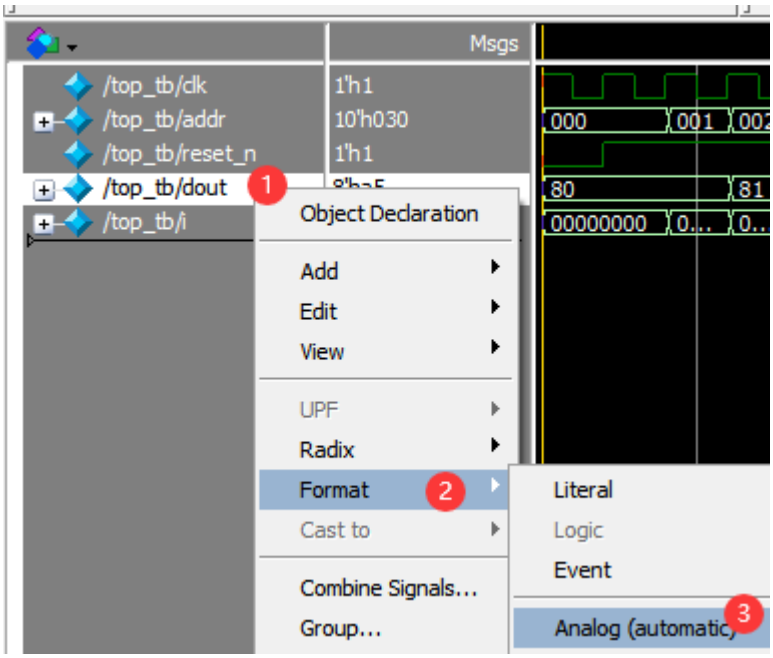


图 11-15 输出波形转换为模拟量形式

然后再次右击图 11-16 中信号“dout”，选择“Radix”选项设置显示数值的进制，然后选择“Unsigned”选择无符号的十进制数。

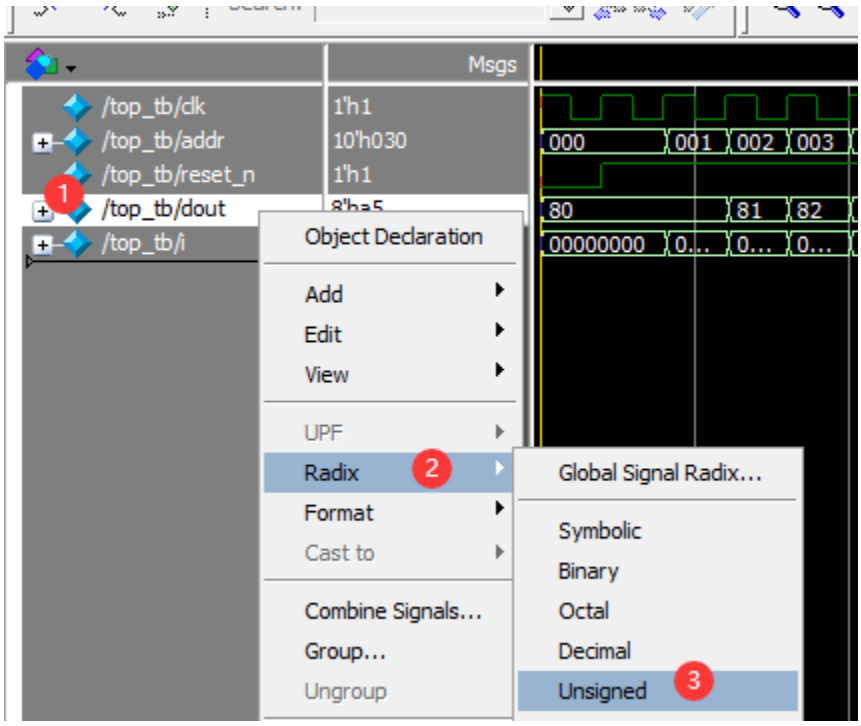


图 11-16 设置输出数据的显示格式

最后可以看到如图 11-17 所示的正弦波波形。

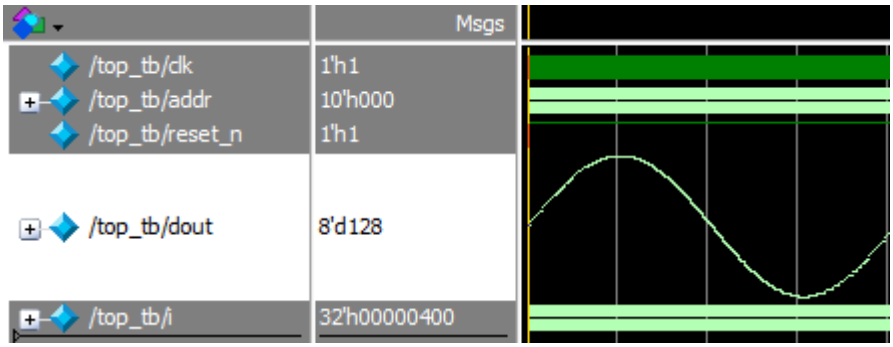


图 11-17 正弦波仿真波形

11.3常见问题

1. 对于一个完整的 FPGA 工程开发流程，仿真是一个重要且必不可少的步骤。
2. 要注意在本章实验中不设置输出寄存器时，信号“addr”是从 0 开始计数，addr[0]对应信号“dout”第一个数据。

11.4总结

本章通过演示如何创建一个 ROM IP 核，引导各位读者来创建一个 ROM IP 核。本章结束后读者可以自行设计一个 ROM IP 核并进行仿真，进一步加深对 ROM IP 核理解以及激励文件的编写能力。建议读者能够跟随本实验内容，完整的进行整个实验。