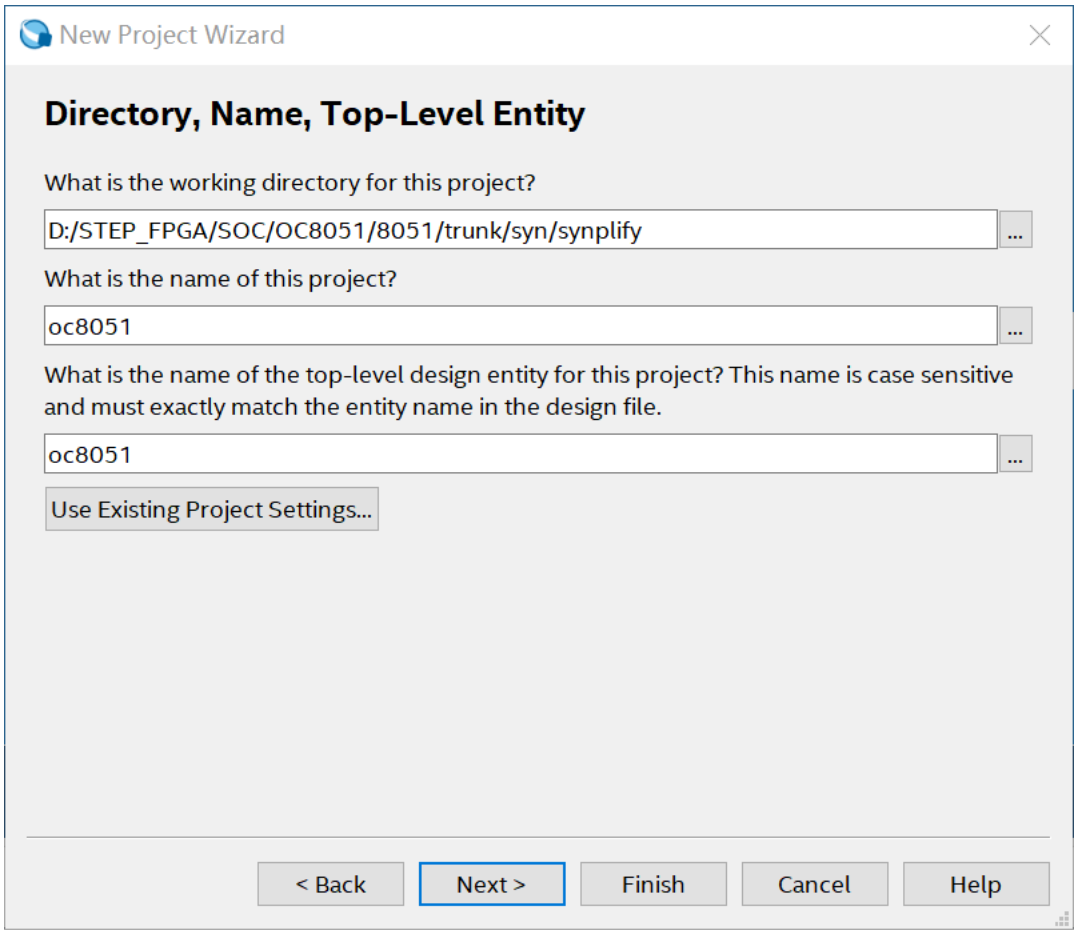
使用 Quartus 和 ModelSim 仿真 OC8051

该手册中使用软件版本 Quartus Prime Lite Edition 17.0 ， ModelSim-altera

1.新建 Quartus 工程

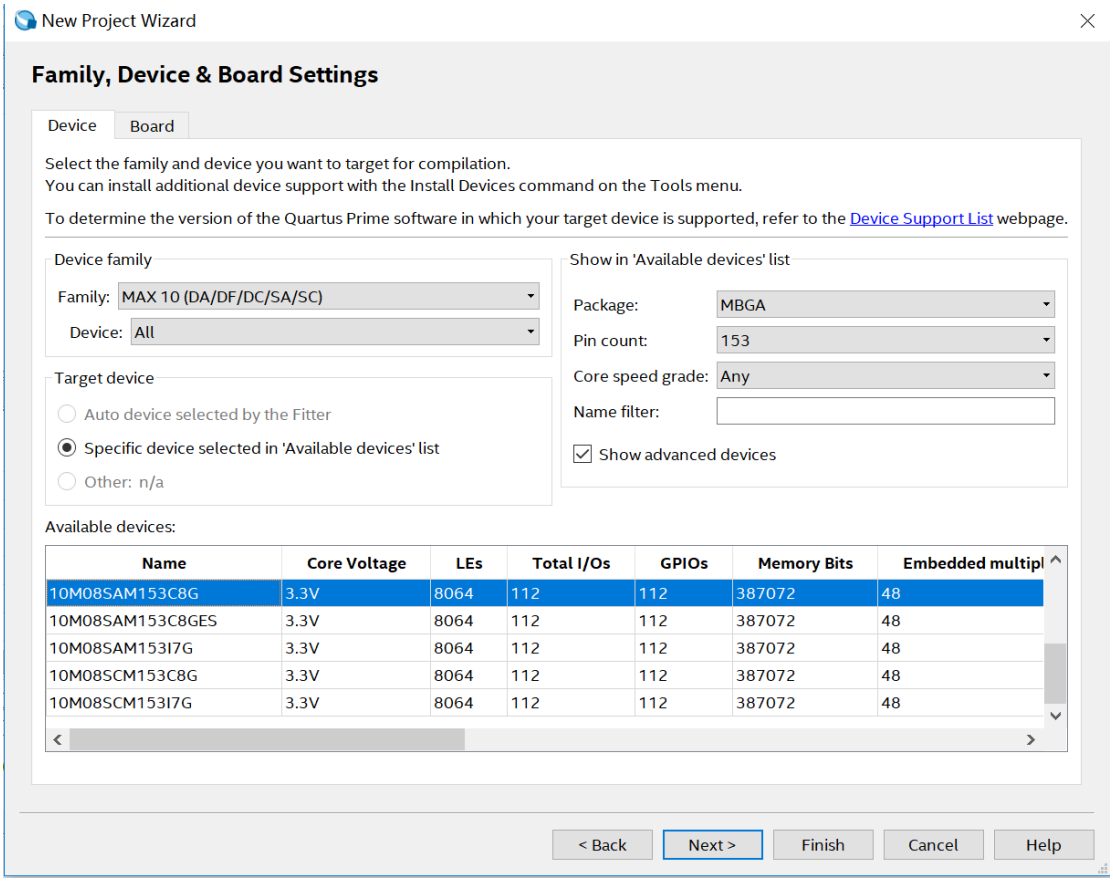
（1）新建工程名为"oc8051 "，保存路径 “..\8051\trunk\syn\synplify"



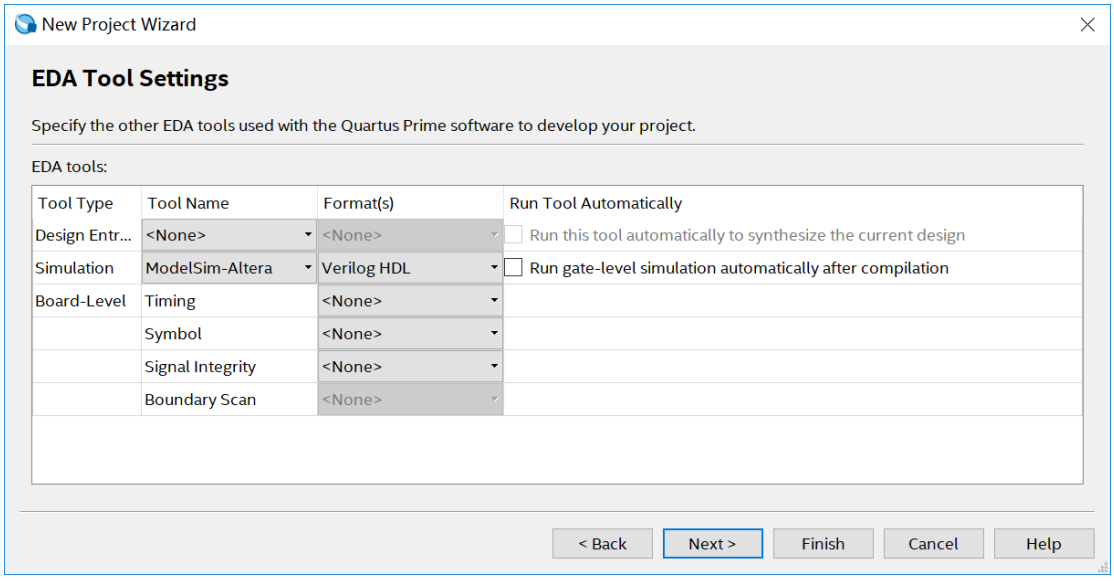
（2）将目录 “8051_latest（源码）\8051\trunk\rtl\verilog ” 下，除

oc8051_alu_test.v 文件外的所有文件添加到工程中。

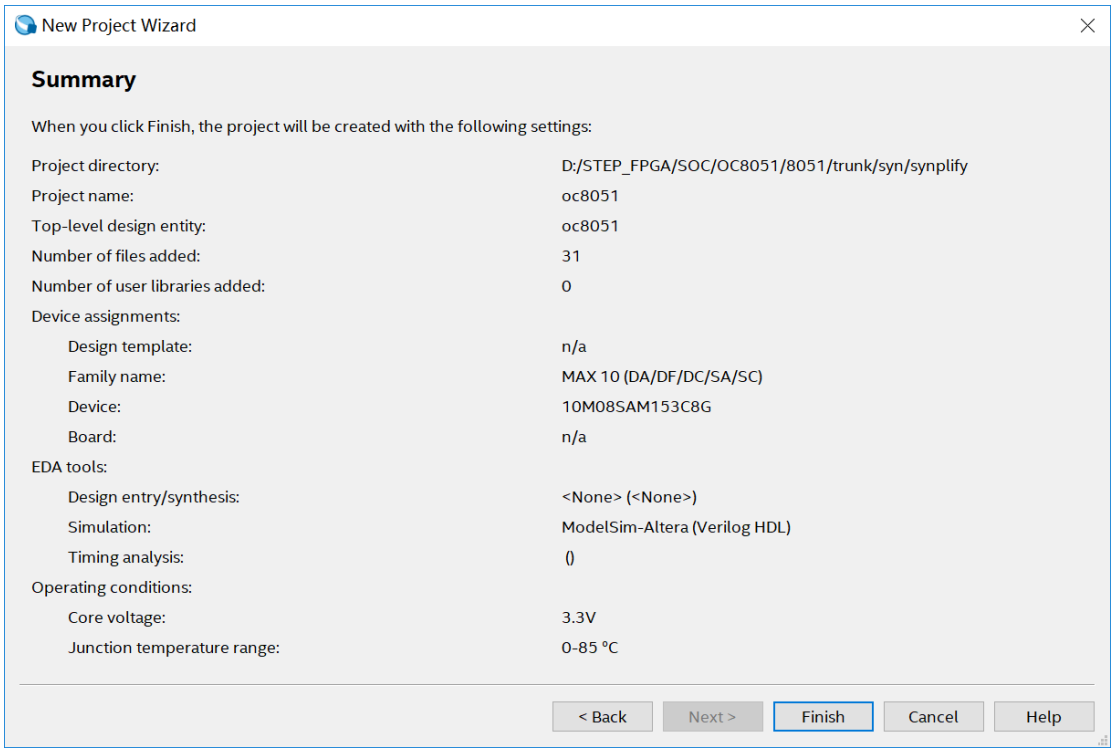
（3）选择小脚丫 MAX10 开发板芯片 10M08SAM153C8G



（4）仿真工具设置为 ModelSim-Altera，格式选择是 Verilog-HDL，这是为仿真设置的，其他默认即可，如下图所示：

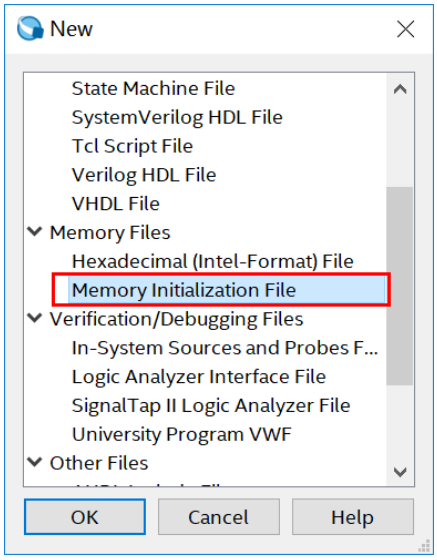


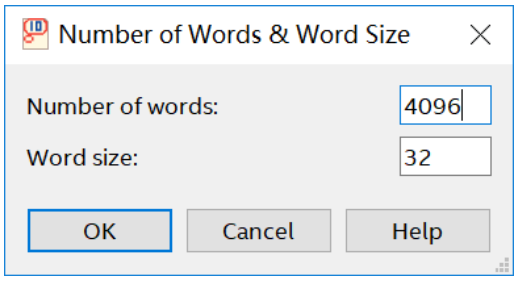
（5）确定后工程信息如下，工程创建完成



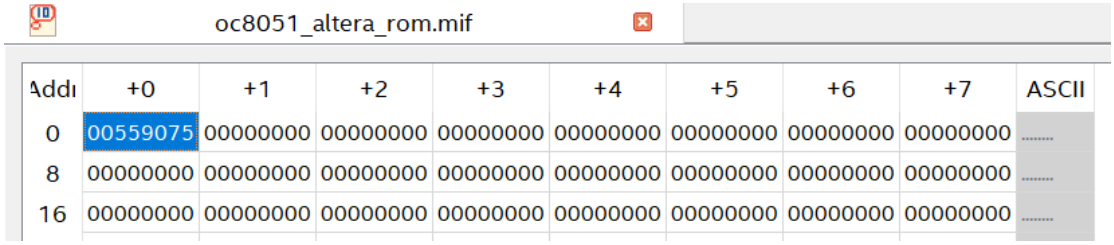
2.新建 mem 初始化文件

新建一个 Memory Init File，文件名可以设置为 oc8051_altera_rom.mif 设置如下，注意数据宽度是 32 位：





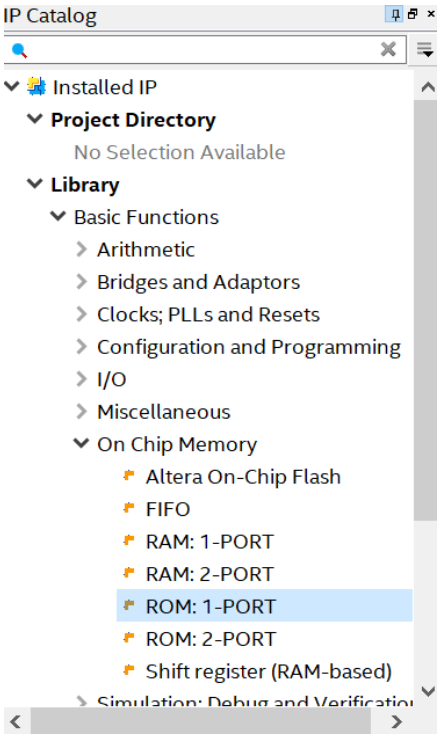
内容如下，只在第一个存储位置填入数据 00559075，这里的 75 是指令 mov，90 是 P1 的地址，55 是移入的立即数，所以该指令的作用就是向 P1 端口输出数 0x55



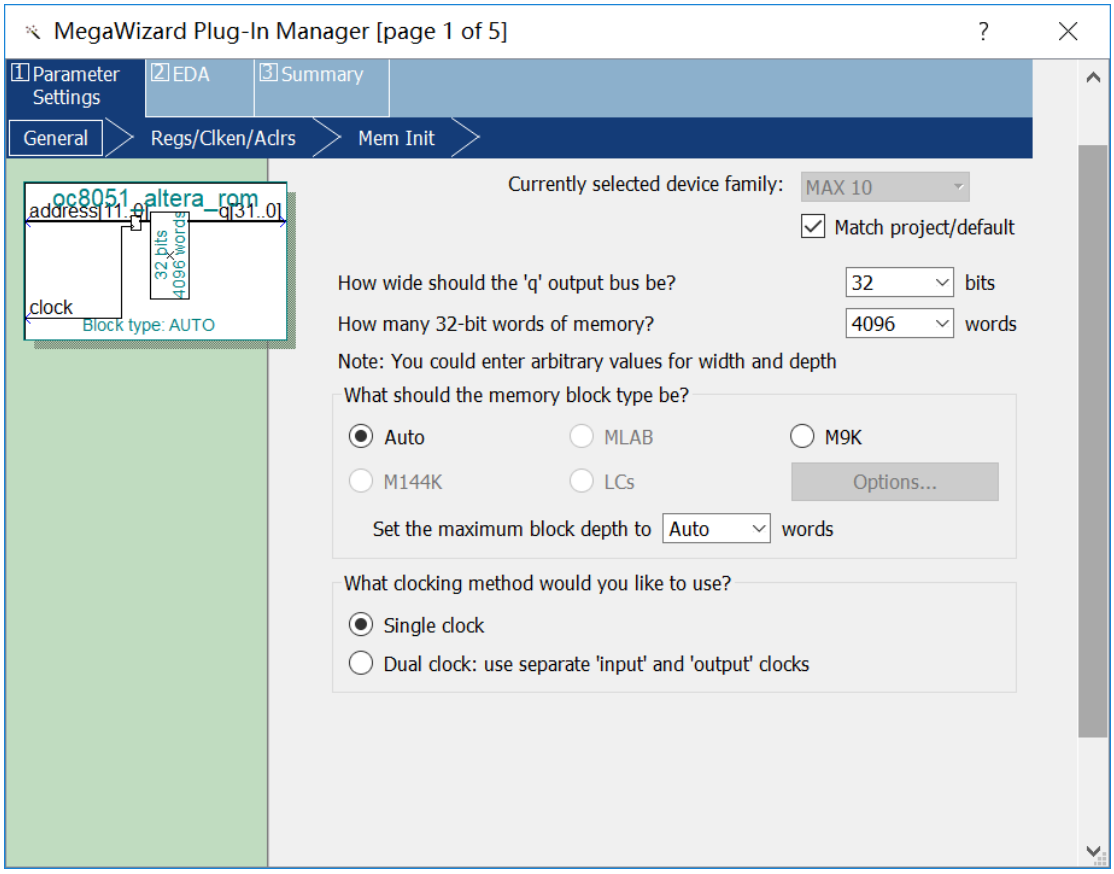
Addı	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00559075	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

3.创建 onchiprom

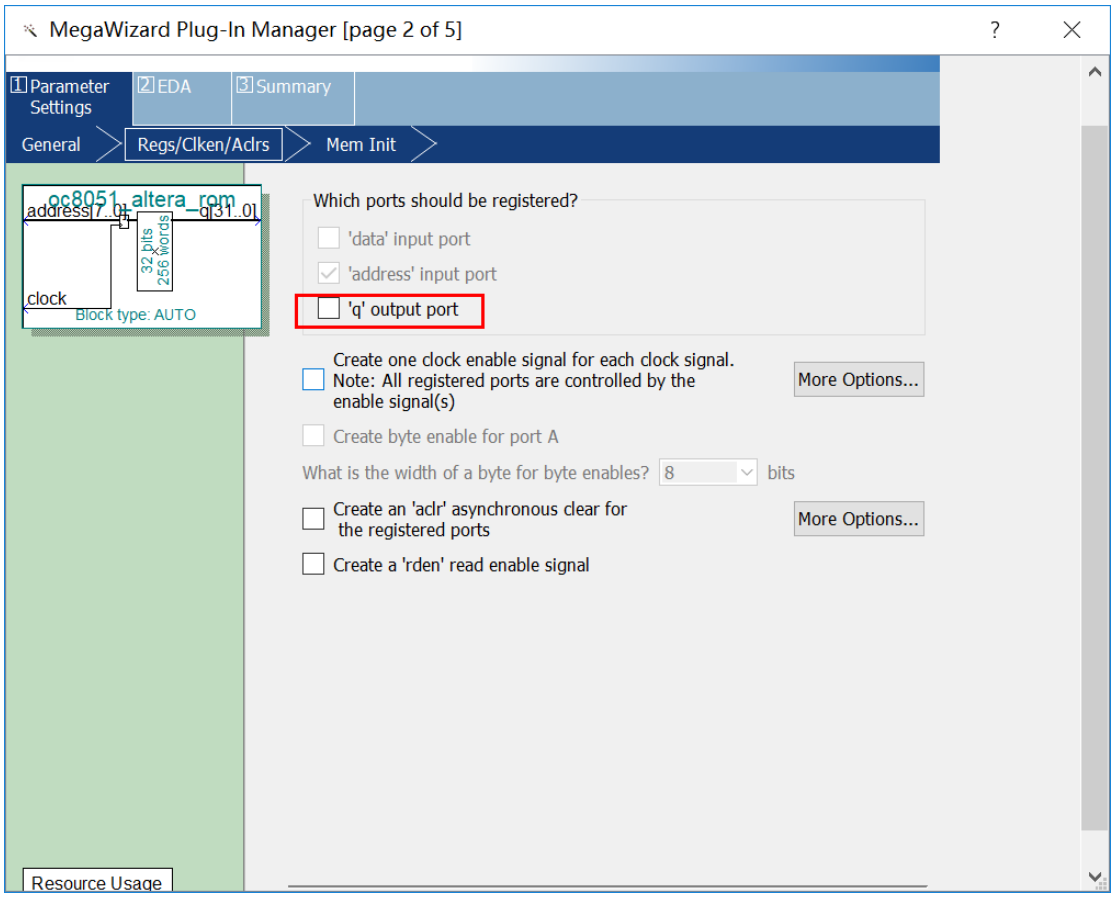
使用 Quartus 中的 IP Catalog 工具创建 ROM，配置过程如下：



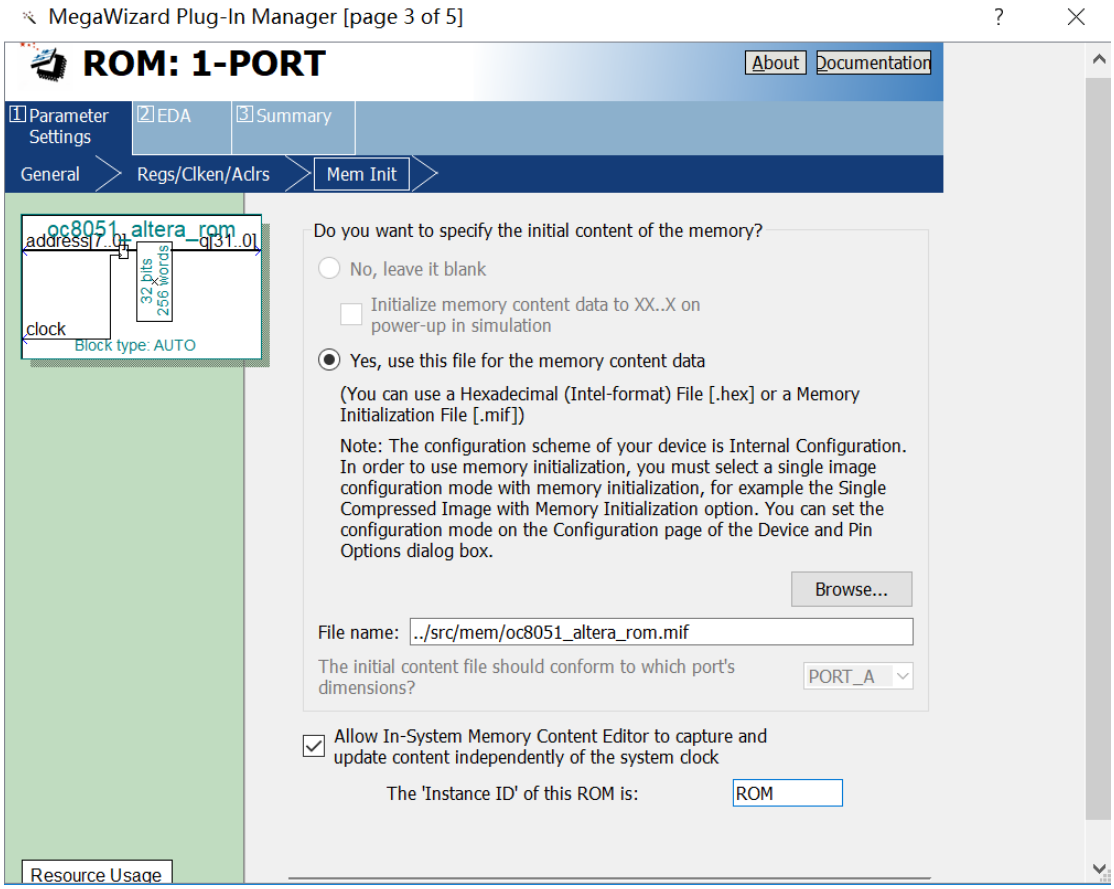
此处数据宽度是 32 位，设置如下：



下面将'q' output port 复选框中取消选中，也就是输出没有锁存



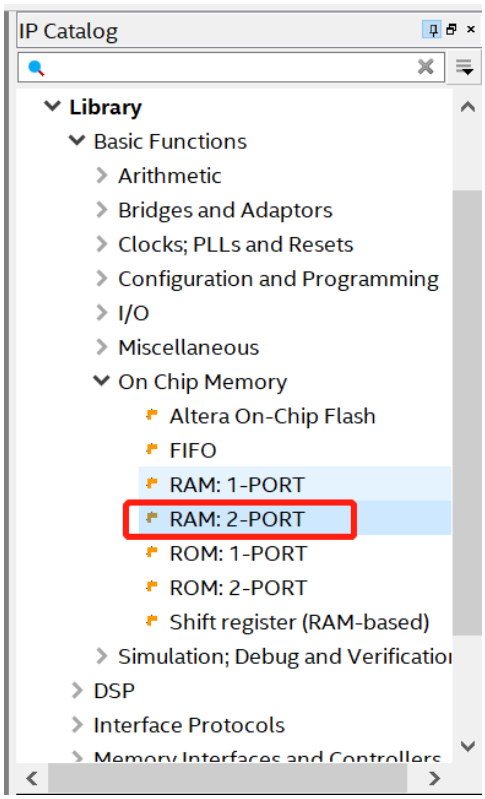
选择初始化文件是上面创建的 mif 文件，同时设置一个 Instance ID，这样就可以在代码运行时从 Quartus 中查看修改 ROM 内容。

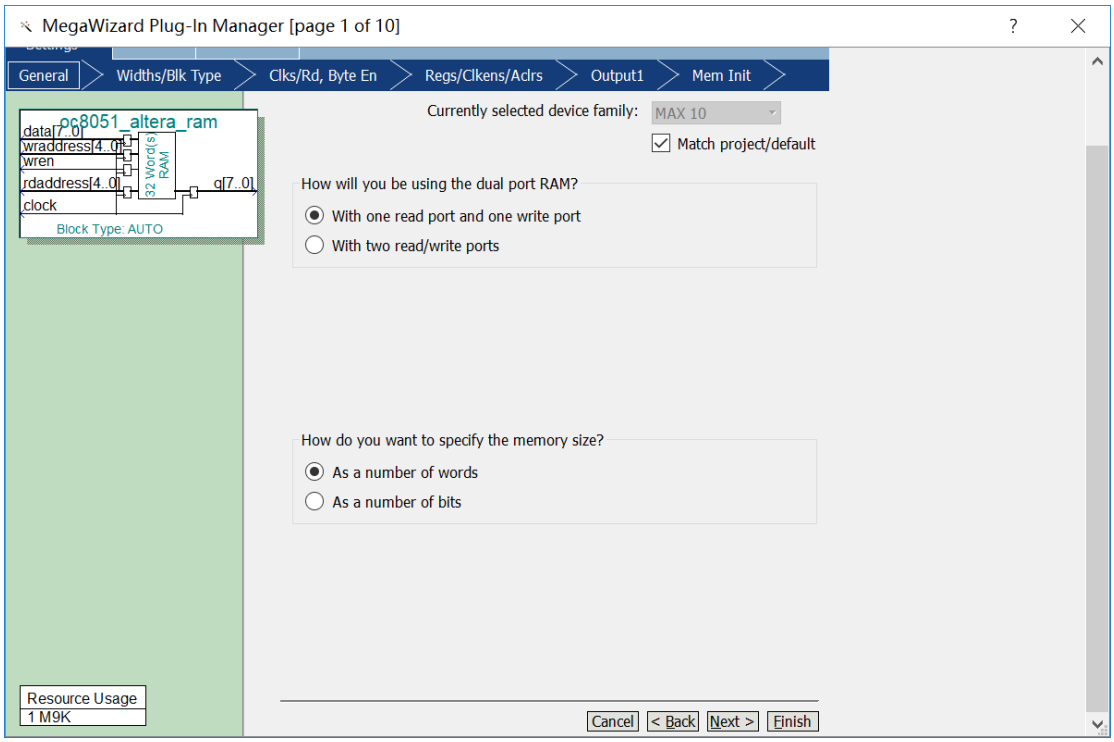


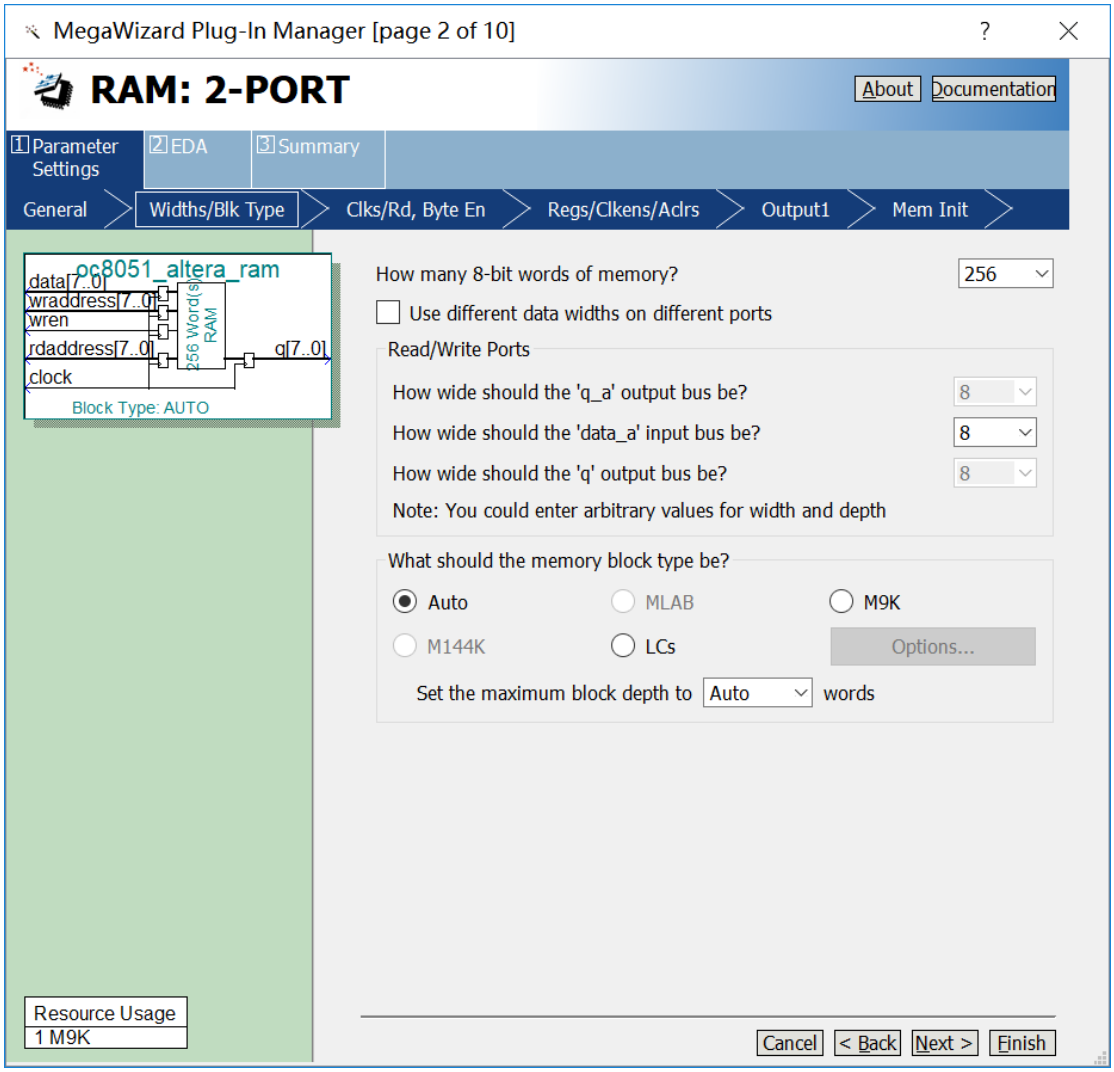
ROM 创建完成。

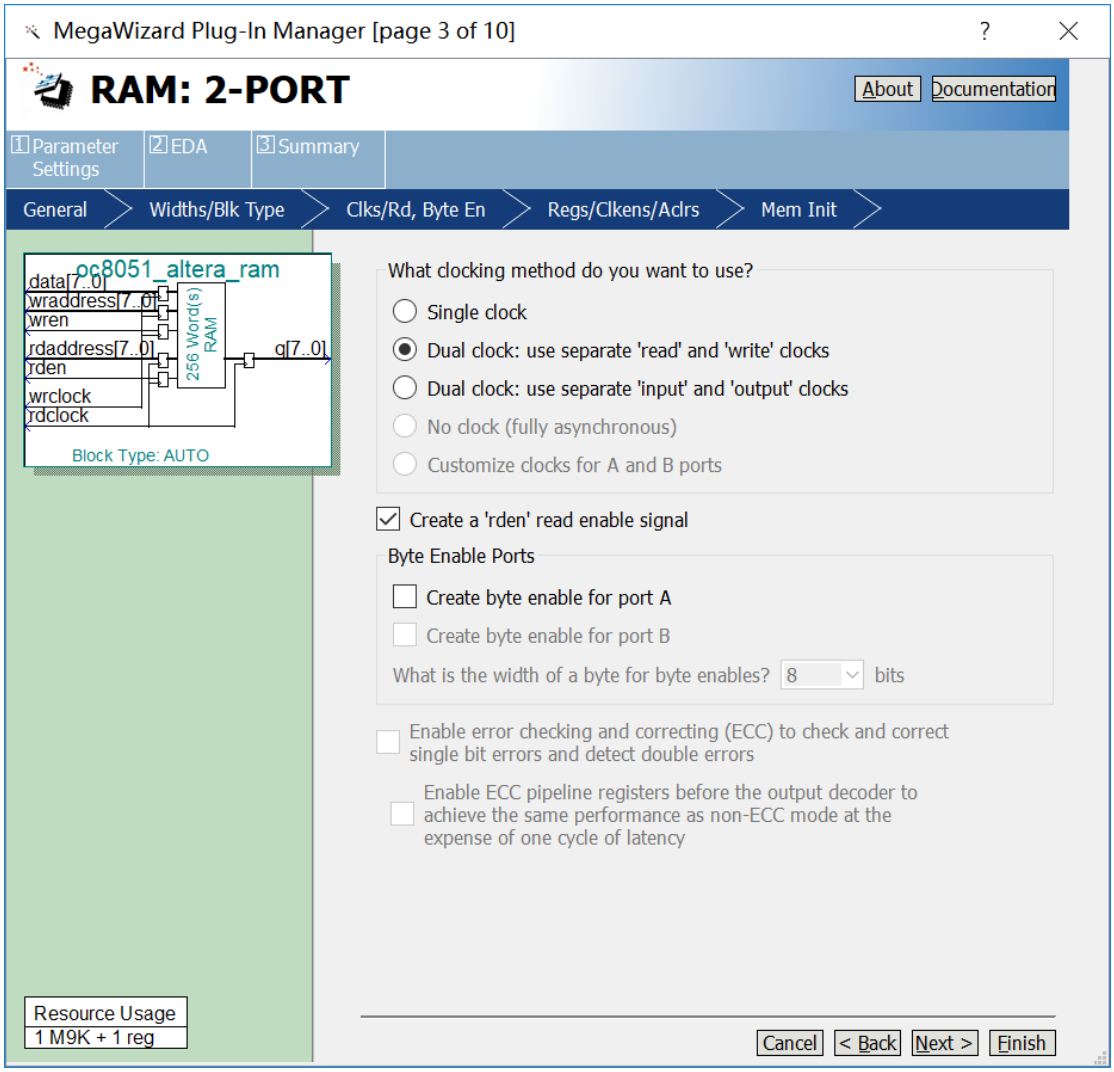
4.创建 onchipram

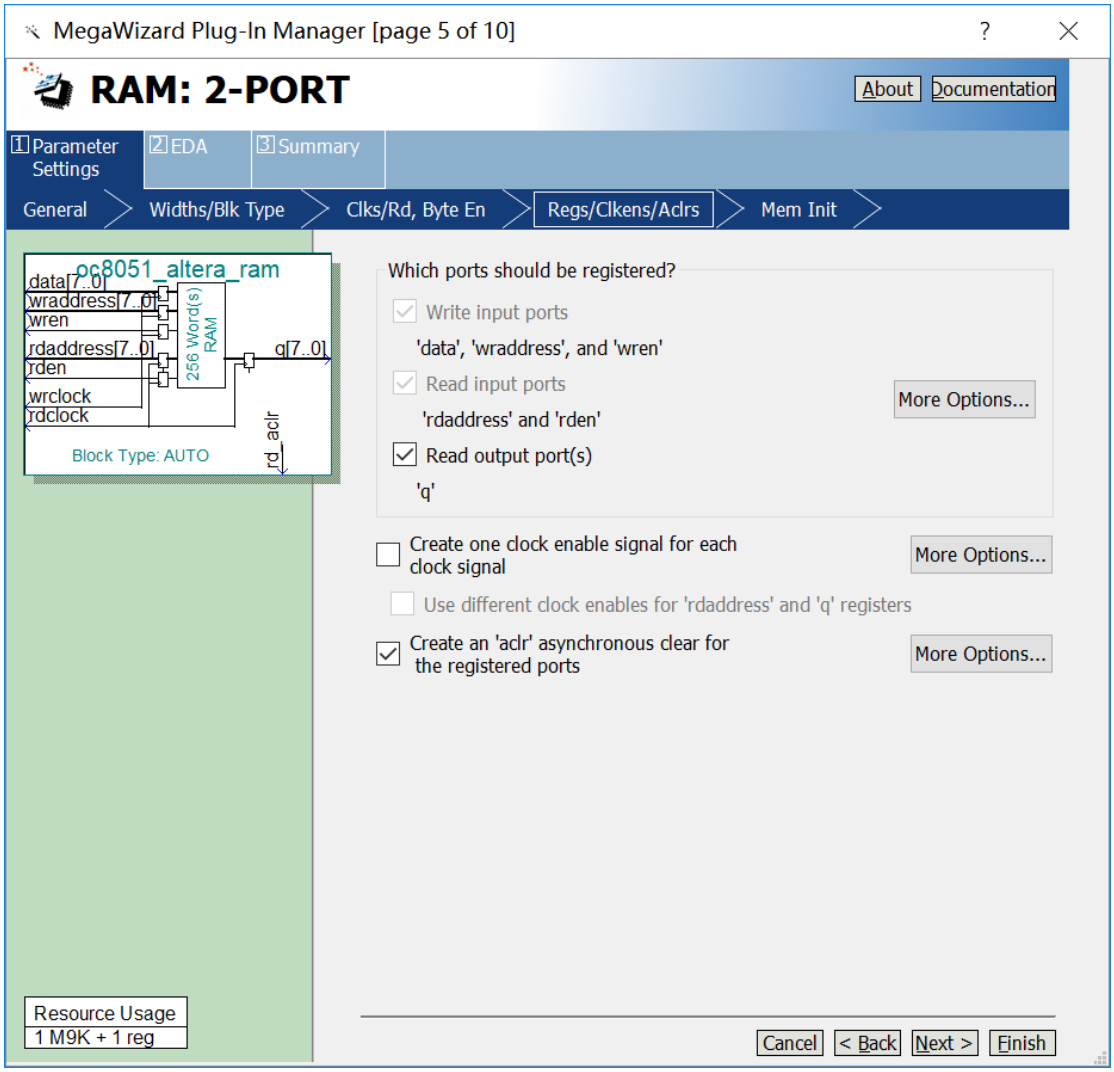
使用 Quartus 中的 IP Catalog 工具创建双端口 RAM，配置过程同上一节中 ROM 的过程类似，不过没有 mem 初始化文件，配置过程如下：

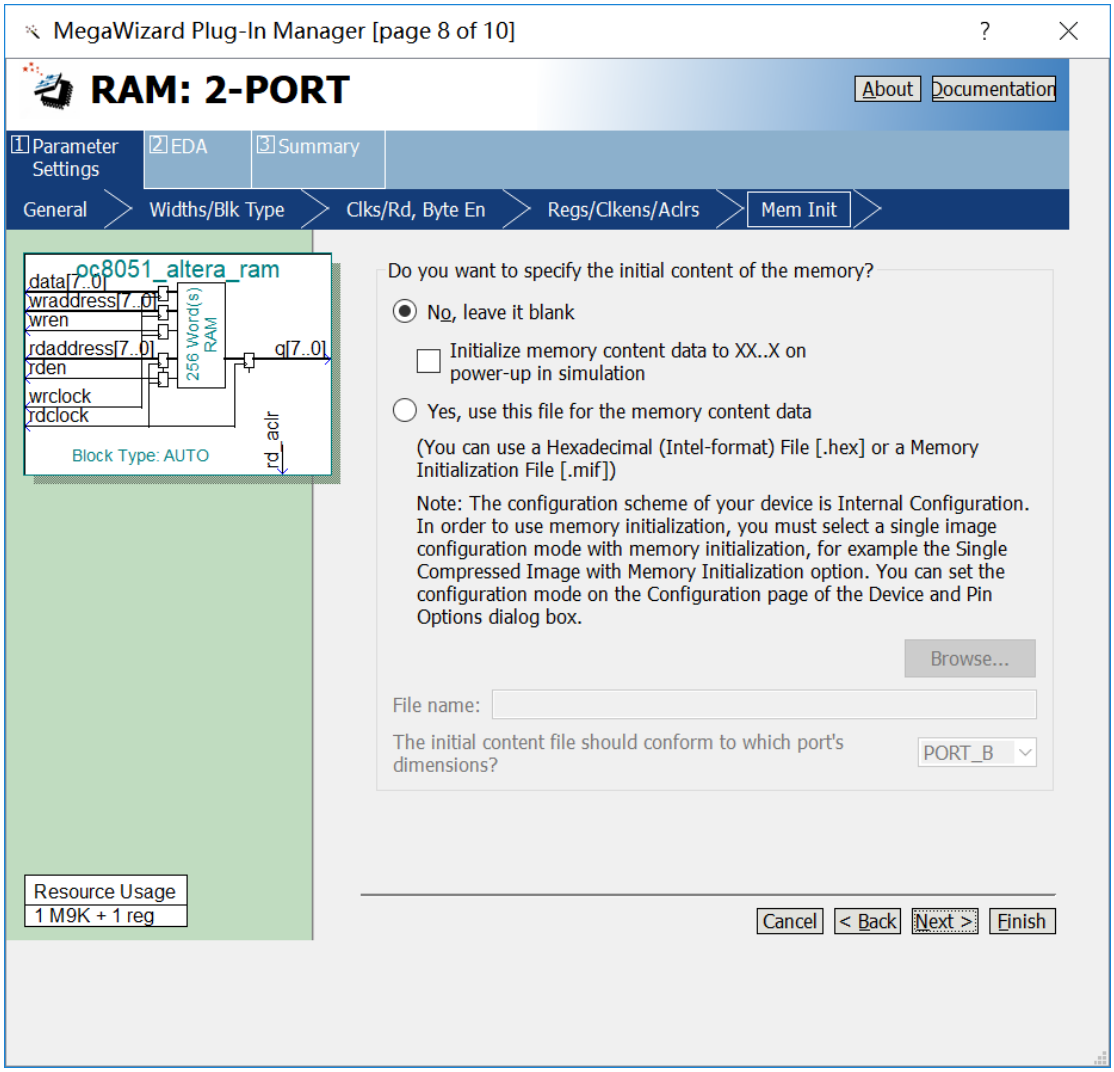

IP Catalog
Library
Basic Functions
Arithmetic
Bridges and Adaptors
Clocks; PLLs and Resets
Configuration and Programming
I/O
Miscellaneous
On Chip Memory
Altera On-Chip Flash
FIFO
RAM: 1-PORT
RAM: 2-PORT
ROM: 1-PORT
ROM: 2-PORT
Shift register (RAM-based)
Simulation; Debug and Verificatio
DSP
Interface Protocols


MegaWizard Plug-In Manager [page 1 of 10]
General
Widths/Blk Type
Clks/Rd, Byte En
Regs/Clkens/Aclrs
Output1
Mem Init
oc8051_altera_ram
data[7..0]
wraddress[4..0]
wren
rdaddress[4..0]
clock
32 Word(s) RAM
q[7..0]
Block Type: AUTO
Currently selected device family:
MAX 10
Match project/default
How will you be using the dual port RAM?
With one read port and one write port
With two read/write ports
How do you want to specify the memory size?
As a number of words
As a number of bits
Resource Usage
1 M9K
Cancel
< Back
Next >
Finish


MegaWizard Plug-In Manager [page 2 of 10]
RAM: 2-PORT
About
Documentation
1 Parameter Settings
2 EDA
3 Summary
General
Widths/Blk Type
Clks/Rd, Byte En
Regs/Clkens/Aclrs
Output1
Mem Init
pc8051_altera_ram
data[7..0]
wraddress[7..0]
wren
rdaddress[7..0]
clock
256 Word(s) RAM
q[7..0]
Block Type: AUTO
How many 8-bit words of memory?
256
Use different data widths on different ports
Read/Write Ports
How wide should the 'q_a' output bus be?
8
How wide should the 'data_a' input bus be?
8
How wide should the 'q' output bus be?
8
Note: You could enter arbitrary values for width and depth
What should the memory block type be?
Auto
MLAB
M9K
M144K
LCs
Options...
Set the maximum block depth to
Auto
words
Resource Usage
1 M9K
Cancel
< Back
Next >
Finish


MegaWizard Plug-In Manager [page 3 of 10]
RAM: 2-PORT
About
Documentation
1 Parameter Settings
2 EDA
3 Summary
General
Widths/Blk Type
Clks/Rd, Byte En
Regs/Clkens/Aclrs
Mem Init
pc8051_altera_ram
data[7..0]
wraddress[7..0]
wren
rdaddress[7..0]
rden
wrclock
rdclock
256 Word(s) RAM
q[7..0]
Block Type: AUTO
What clocking method do you want to use?
Single clock
Dual clock: use separate 'read' and 'write' clocks
Dual clock: use separate 'input' and 'output' clocks
No clock (fully asynchronous)
Customize clocks for A and B ports
Create a 'rden' read enable signal
Byte Enable Ports
Create byte enable for port A
Create byte enable for port B
What is the width of a byte for byte enables? 8 bits
Enable error checking and correcting (ECC) to check and correct single bit errors and detect double errors
Enable ECC pipeline registers before the output decoder to achieve the same performance as non-ECC mode at the expense of one cycle of latency
Resource Usage
1 M9K + 1 reg
Cancel
< Back
Next >
Finish


MegaWizard Plug-In Manager [page 5 of 10]
RAM: 2-PORT
About
Documentation
1 Parameter Settings
2 EDA
3 Summary
General
Widths/Blk Type
Clks/Rd, Byte En
Regs/Clkens/Aclrs
Mem Init
pc8051_altera_ram
data[7..0]
wraddress[7..0]
wren
rdaddress[7..0]
rden
wrclock
rdclock
256 Word(s) RAM
q[7..0]
rd_aclr
Block Type: AUTO
Which ports should be registered?
Write input ports
'data', 'wraddress', and 'wren'
Read input ports
'rdaddress' and 'rden'
Read output port(s)
'q'
More Options...
Create one clock enable signal for each clock signal
More Options...
Use different clock enables for 'rdaddress' and 'q' registers
Create an 'aclr' asynchronous clear for the registered ports
More Options...
Resource Usage
1 M9K + 1 reg
Cancel
< Back
Next >
Finish

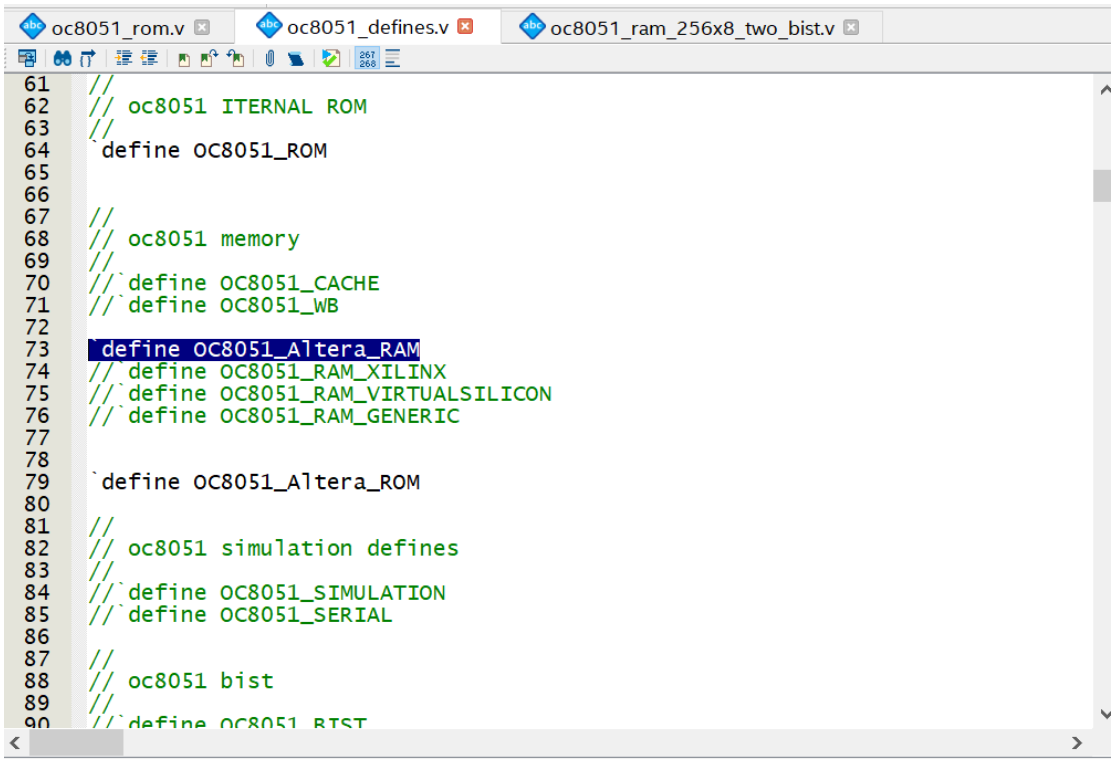


5.修改源代码

（1）修改 oc8051_rom.v，使用刚刚创建的 ROM 做为 CPU 内部 ROM。将其内容替换为如下：

```
63  `include "oc8051_timescale.v"
64  `include "oc8051_defines.v"
65
66  module oc8051_rom (rst, clk, addr, ea_int, data_o);
67
68  //parameter INT_ROM_WID= 15;
69
70  input rst, clk;
71  input [15:0] addr;
72  //input [22:0] addr;
73  output ea_int;
74  output [31:0] data_o;
75
76  wire ea;
77
78  reg ea_int;
79
80
81  `ifdef OC8051_Altera_ROM
82
83  parameter INT_ROM_WID= 12;
84
85  assign ea = | addr[15:INT_ROM_WID];
86
87
88  always @(posedge clk or posedge rst)
89   if (rst)
90     ea_int <= #1 1'b1;
91    else ea_int <= #1 !ea;
92
93  oc8051_altera_rom oc8051_altera_rom1
94  (
95  .address(addr[11:0]),
96  .clock(clk),
97  .q(data_o)
98  );
99
100
101 `else
102
```

（2）修改 defines.v 添加`define OC8051_Altera_RAM，`define OC8051_Altera_ROM，取消掉`define OC8051_ROM 前面的注释，将`define OC8051_RAM_GENERIC 注释掉。



（3）修改 oc8051_ram_256x8_two_bist，将 126-146 行使用下列代码替换：

```
98
99  `ifdef OC8051_Altera_RAM
100
101 oc8051_altera_ram OC8051_Altera_Ram_u(
102   .rdclock(clk),
103   .rd_aclr(rst),
104   .rdaddress(rd_addr),
105   .rden(rd_en),
106   .q(rd_data),
107
108   .wrclock(clk),
109   .wraddress(wr_addr),
110   .wren(wr),
111   .data(wr_data)
112    );
113
114
115 `else
```

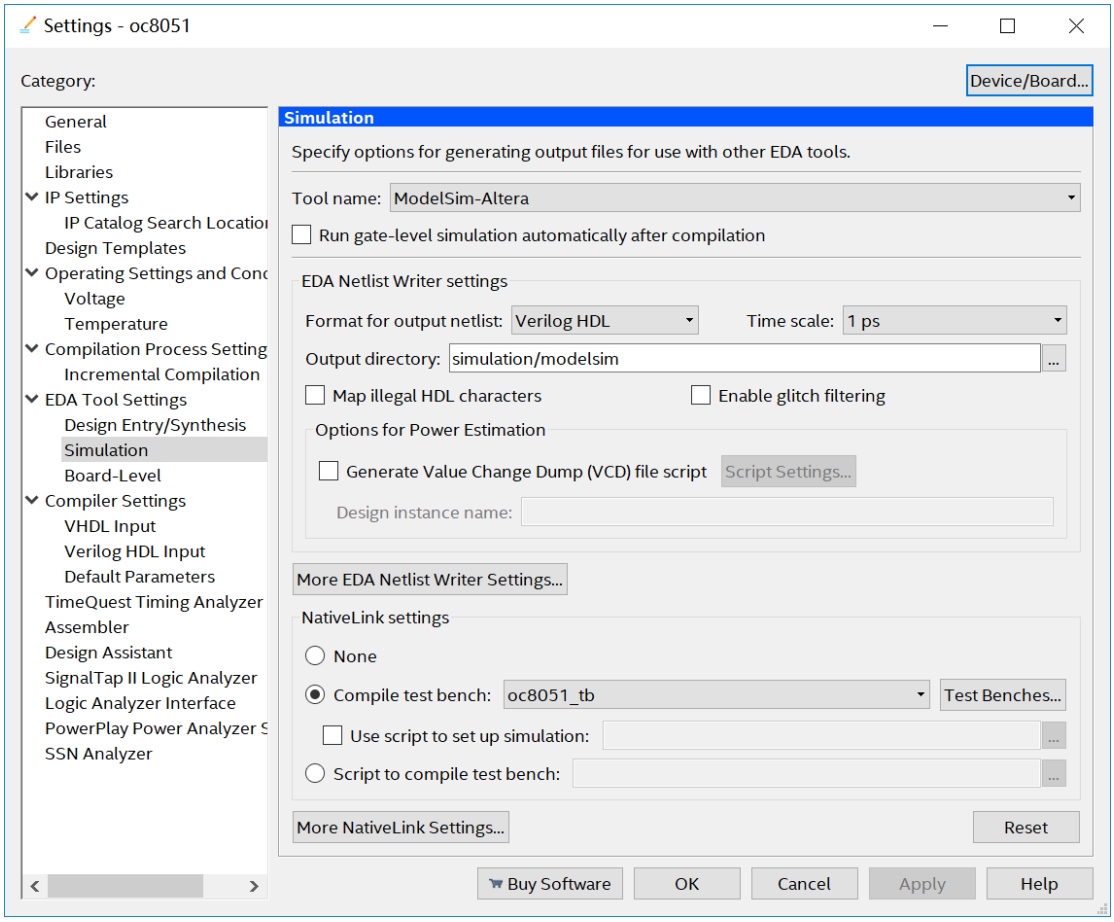
6.添加 TestBench 文件

（1）新建一个 Verilog HDL 文件，文件名为 oc8051_tb.v，这是一个测试文件，内容如下：

上面的测试代码定义了时钟周期是 20ns，注意的是要给 ack_i、iack_i、wbd_err_i、

wbi_err_i 赋初值，否则会导致时序仿真出现异常。

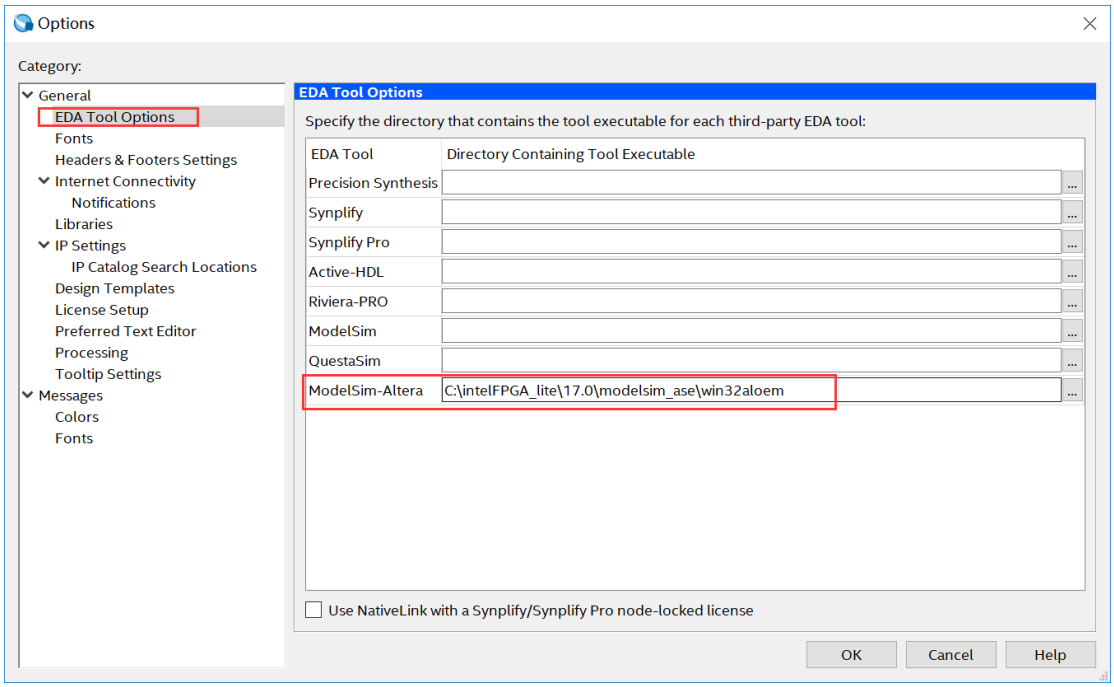
（2）点击 Assignments->Setting，选择 EDA Tools Setting->Simulation，如下设置：



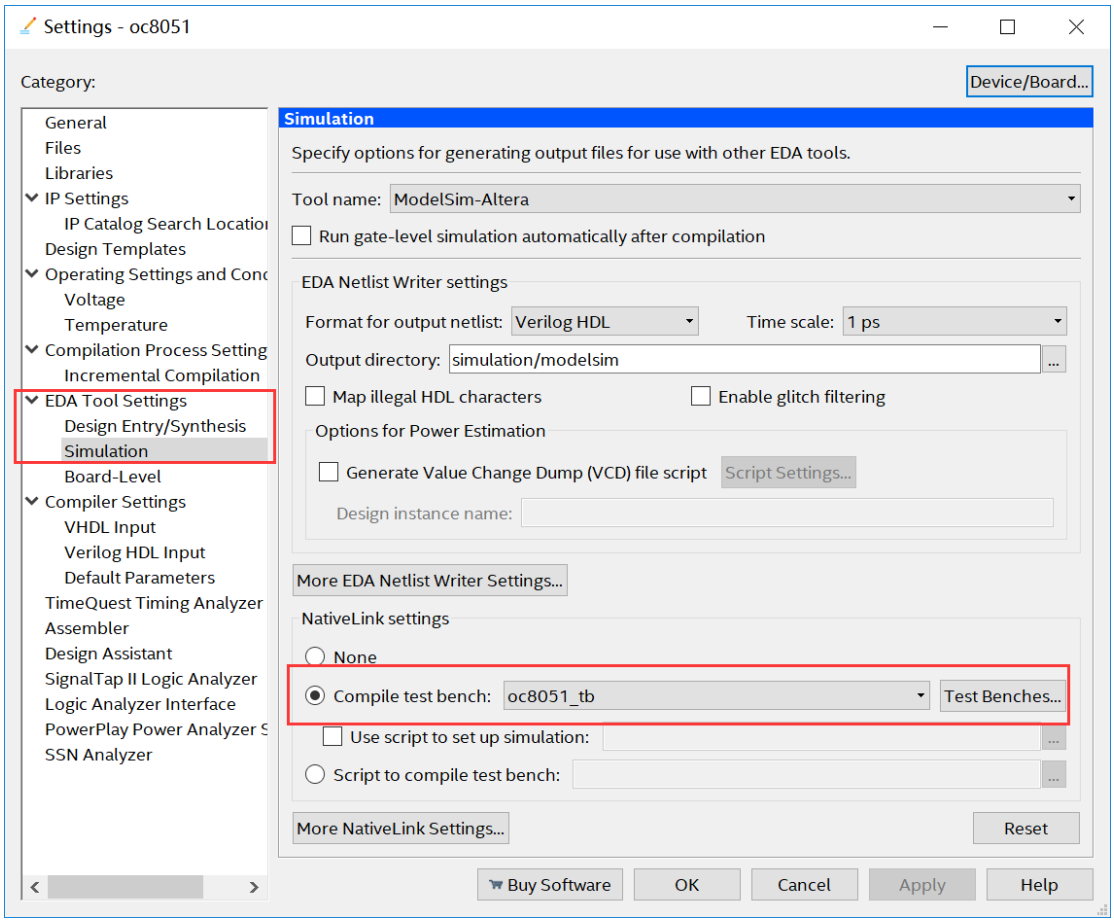
7.功能仿真

（1）仿真软件设置

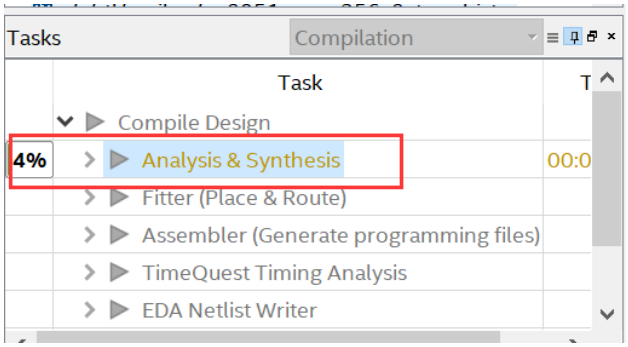
Tools ——Options



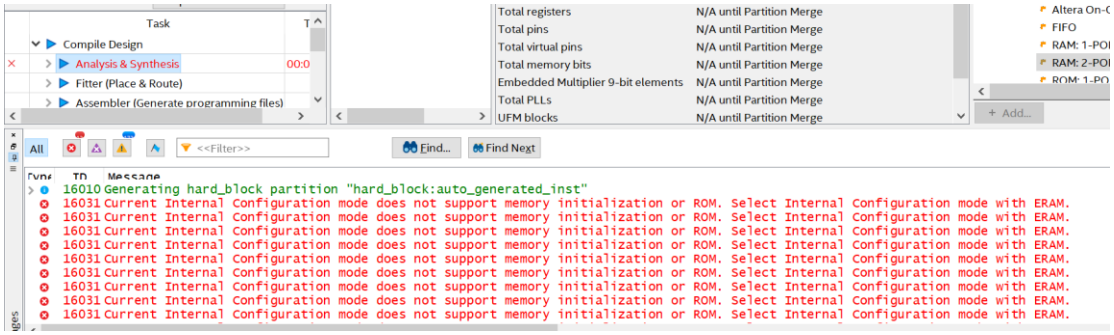
Assignments——Settings



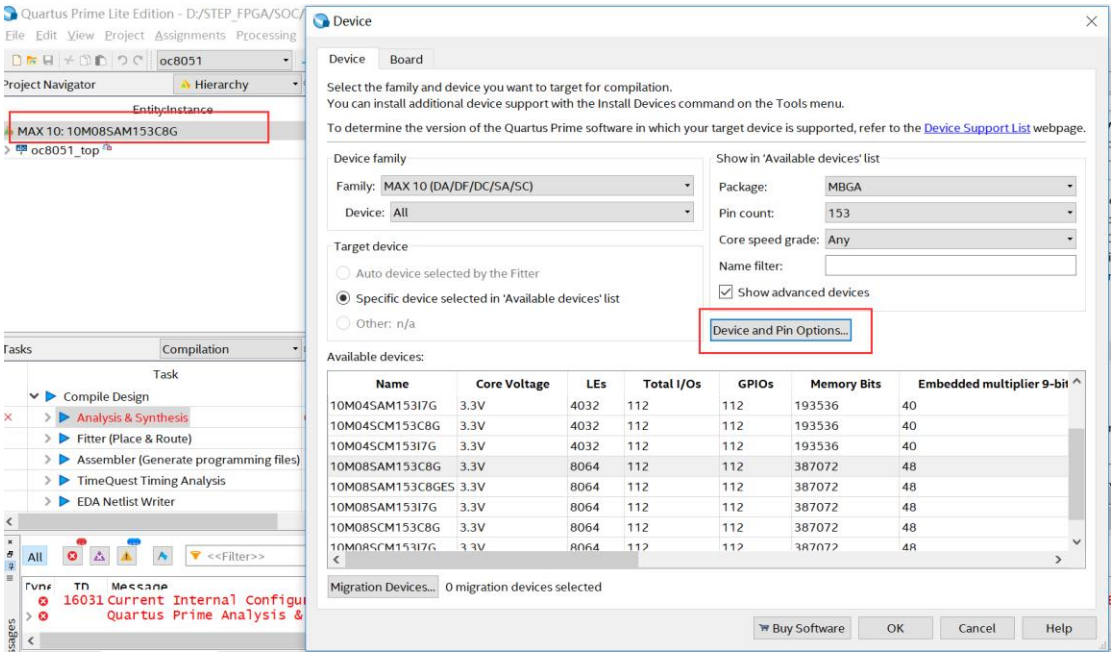
(2) Analysis&Synthesis

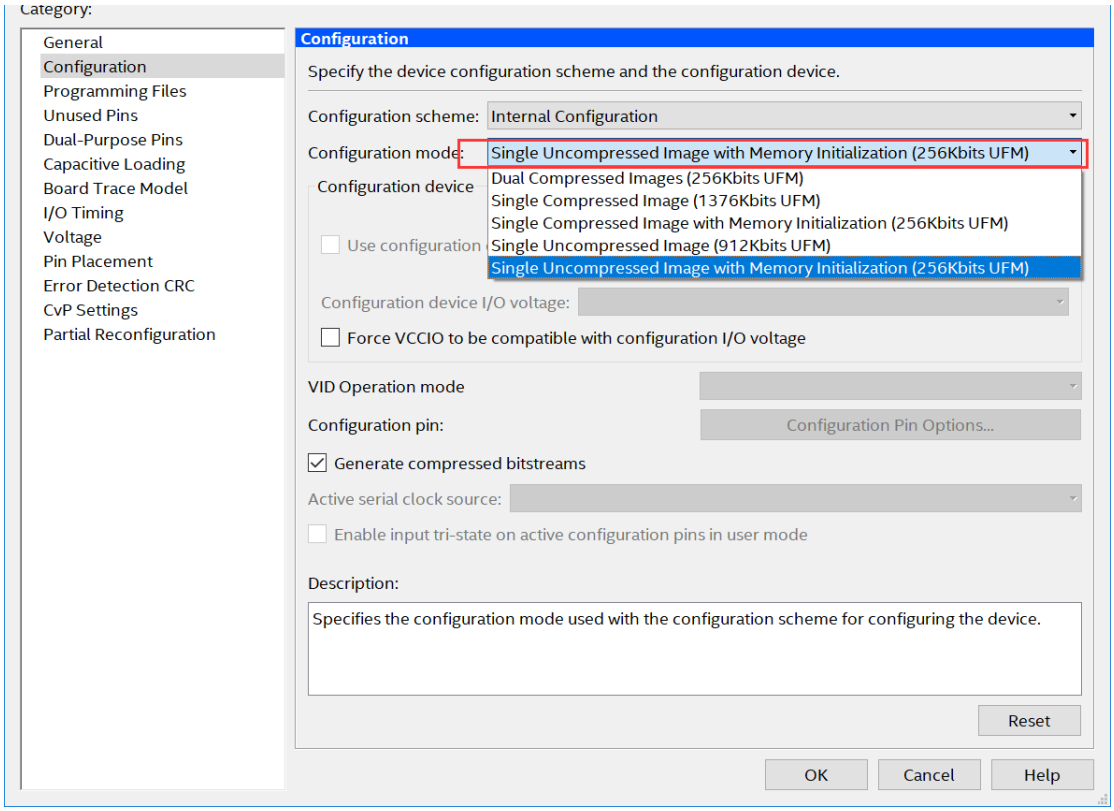


注意：如果报如下错误，需要修改 Device 属性

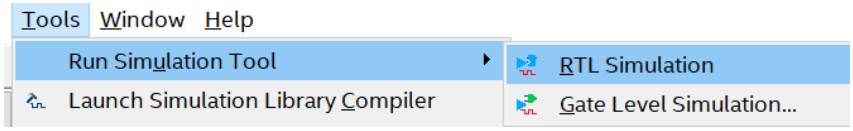


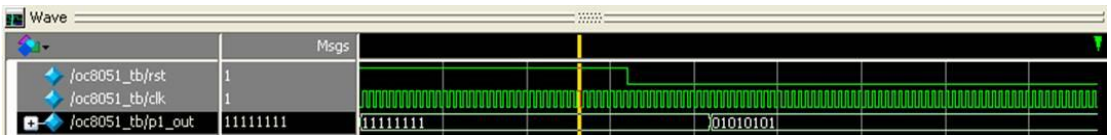
修改 Device 属性





（3）RTL 仿真





参考来源：网友 leishangwen 的《DE2 上使用 OC8051 运行点灯程序》：

https://download.csdn.net/download/leishangwen/5173363