FPGA 综合实验

PB16060240

黄骏达

实验目的

1. 熟悉 ALTERA IP 核及 Signal TAP II；
2. 了解 FPGA 基本结构中的存储器；
3. 熟悉 VHDL test benh 的设计
4. 熟练掌握 Altera FPGA 的开发环境、设计步骤和流程

实验原理

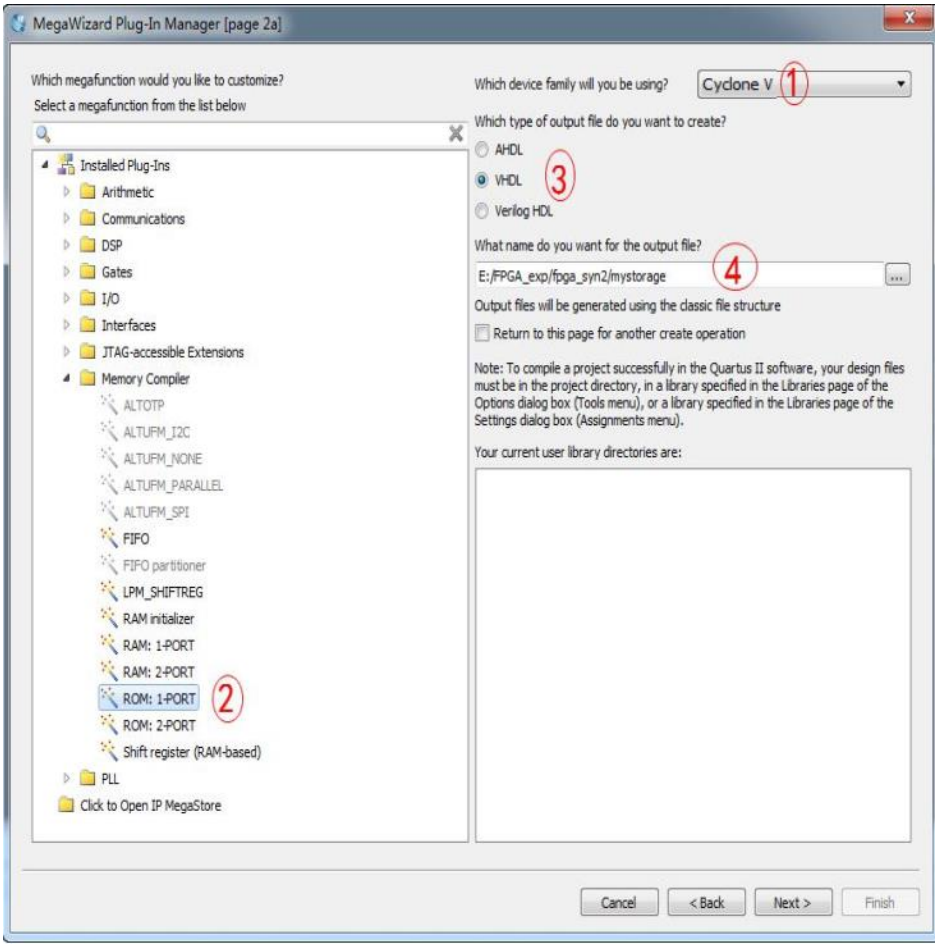
利用 QuartusII 中的 MegaWizard 能够生成 1024*8bits 的存储器（ram/rom），若将一个正弦波的 1024Bytes 数据存入存储器中，那么能够编辑 VHDL 程序，按照一定顺序将存储器中的正弦波数据读出并输出到 DA 端口上，然后进行编译和仿真。
之后通过 signaltap II 文件中观看输出到 DA 端口的数据及波形。

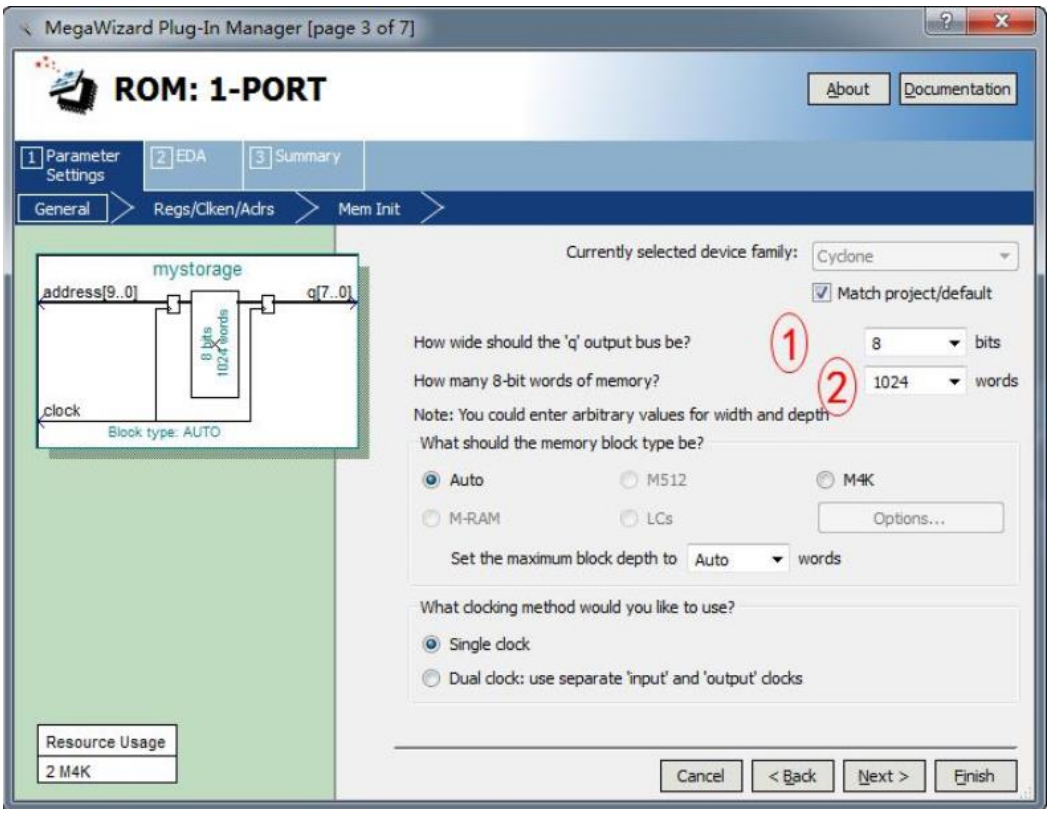
实验设计

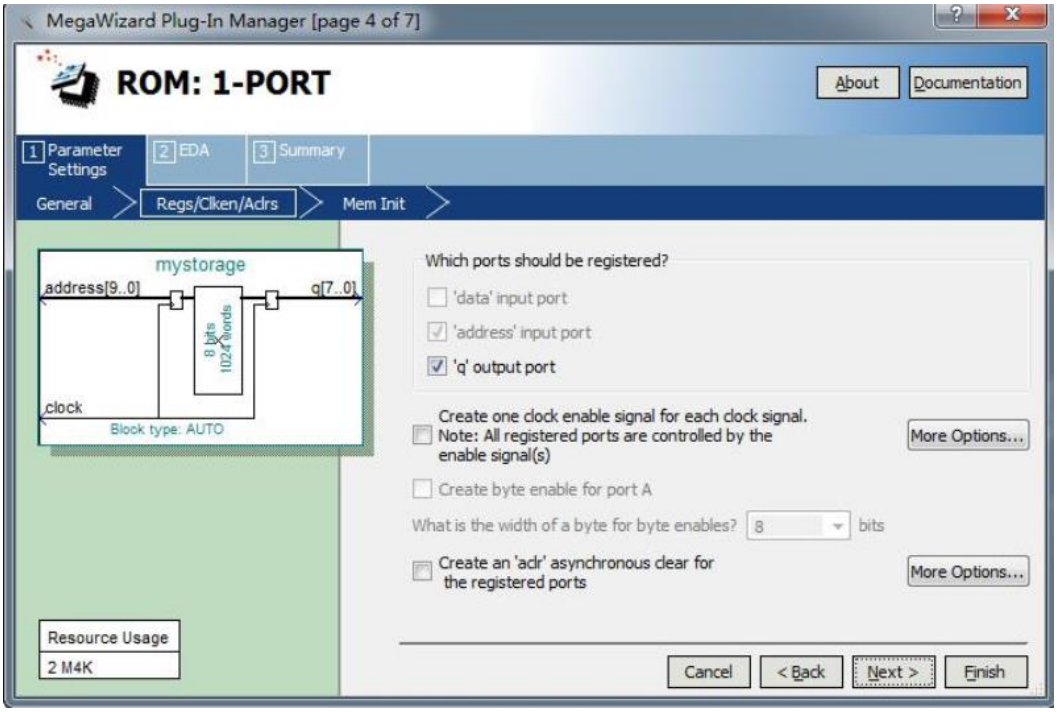
1. 利用 Matlab 生成一个正弦波并输出为.mif 文件
   a) 运行写好的 sin_generation.m 文件
   b) 生成 mystorage.mif 文件

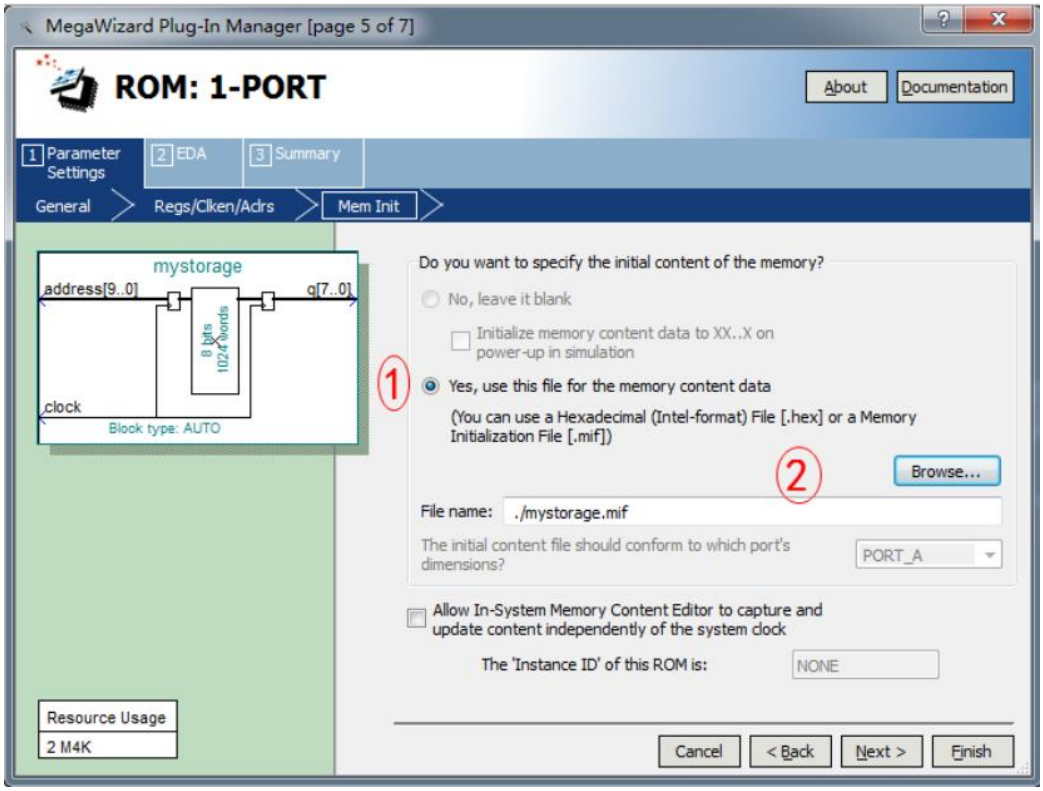
2. 创建 QuartusII 工程，并将 mystorage.mif 文件添加到工程目录下

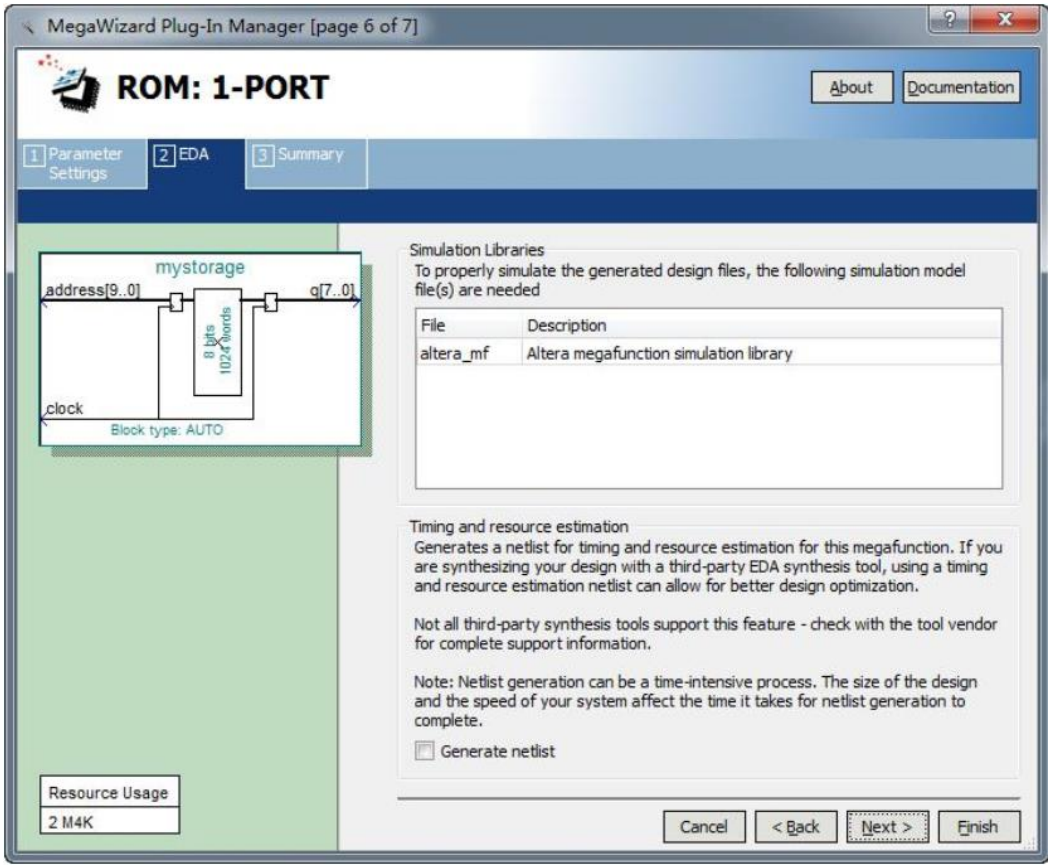
3. 使用 MegaWziard 生成 ROM 存储器：在 Tools 菜单中选择 MegaWizard Plug-In Manager 选项，操作如下所示：

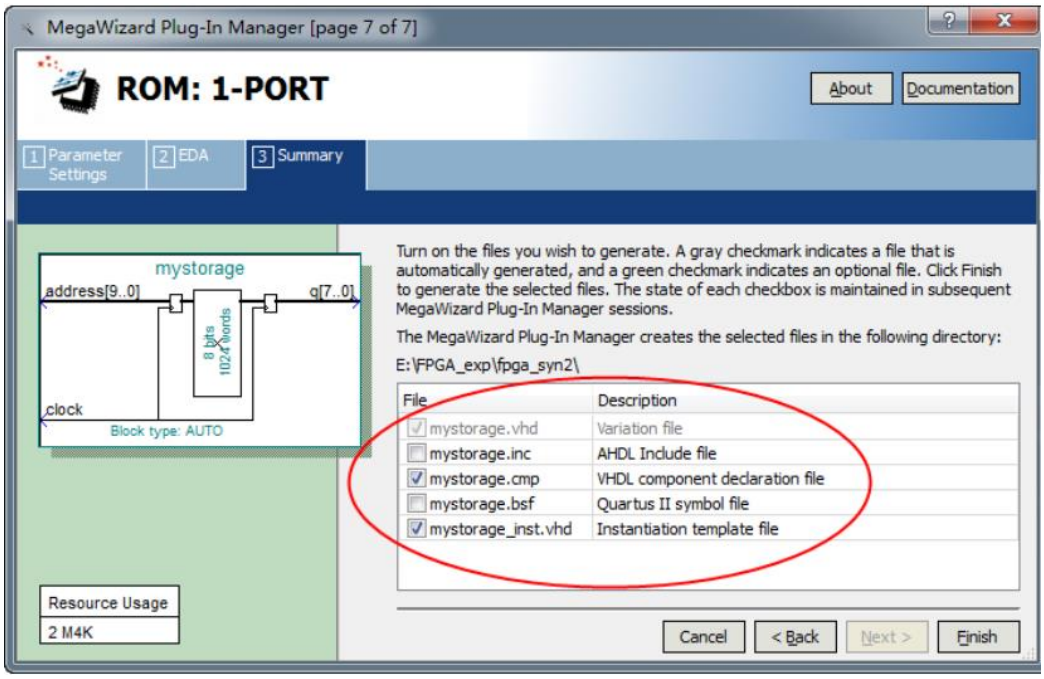

MegaWizard Plug-In Manager [page 2a]
Which megafunction would you like to customize?
Select a megafunction from the list below
Installed Plug-Ins
Arithmetic
Communications
DSP
Gates
I/O
Interfaces
JTAG-accessible Extensions
Memory Compiler
ALTOTP
ALTUFM_I2C
ALTUFM_NONE
ALTUFM_PARALLEL
ALTUFM_SPI
FIFO
FIFO partitioner
LPM_SHIFTREG
RAM initializer
RAM: 1-PORT
RAM: 2-PORT
ROM: 1-PORT
ROM: 2-PORT
Shift register (RAM-based)
PLL
Click to Open IP MegaStore
Which device family will you be using?
Cyclone V
1
Which type of output file do you want to create?
AHDL
VHDL
Verilog HDL
3
What name do you want for the output file?
E:/FPGA_exp/fpga_syn2/mystorage
4
Output files will be generated using the classic file structure
Return to this page for another create operation
Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in a library specified in the Libraries page of the Options dialog box (Tools menu), or a library specified in the Libraries page of the Settings dialog box (Assignments menu).
Your current user library directories are:
2
Cancel
< Back
Next >
Finish


MegaWizard Plug-In Manager [page 3 of 7]
ROM: 1-PORT
About
Documentation
1 Parameter Settings
2 EDA
3 Summary
General
Regs/Clken/Adrs
Mem Init
mystorage
address[9..0]
q[7..0]
8 bits
1024 words
clock
Block type: AUTO
Currently selected device family:
Cyclone
Match project/default
How wide should the 'q' output bus be?
1
8
bits
How many 8-bit words of memory?
2
1024
words
Note: You could enter arbitrary values for width and depth
What should the memory block type be?
Auto
M512
M4K
M-RAM
LCs
Options...
Set the maximum block depth to
Auto
words
What clocking method would you like to use?
Single clock
Dual clock: use separate 'input' and 'output' clocks
Resource Usage
2 M4K
Cancel
< Back
Next >
Finish


MegaWizard Plug-In Manager [page 4 of 7]
ROM: 1-PORT
About
Documentation
1 Parameter Settings
2 EDA
3 Summary
General
Regs/Clken/Aclrs
Mem Init
mystorage
address[9..0]
q[7..0]
8 bits
1024 words
clock
Block type: AUTO
Which ports should be registered?
'data' input port
'address' input port
'q' output port
Create one clock enable signal for each clock signal. Note: All registered ports are controlled by the enable signal(s)
More Options...
Create byte enable for port A
What is the width of a byte for byte enables?
8
bits
Create an 'aclr' asynchronous clear for the registered ports
More Options...
Resource Usage
2 M4K
Cancel
< Back
Next >
Finish

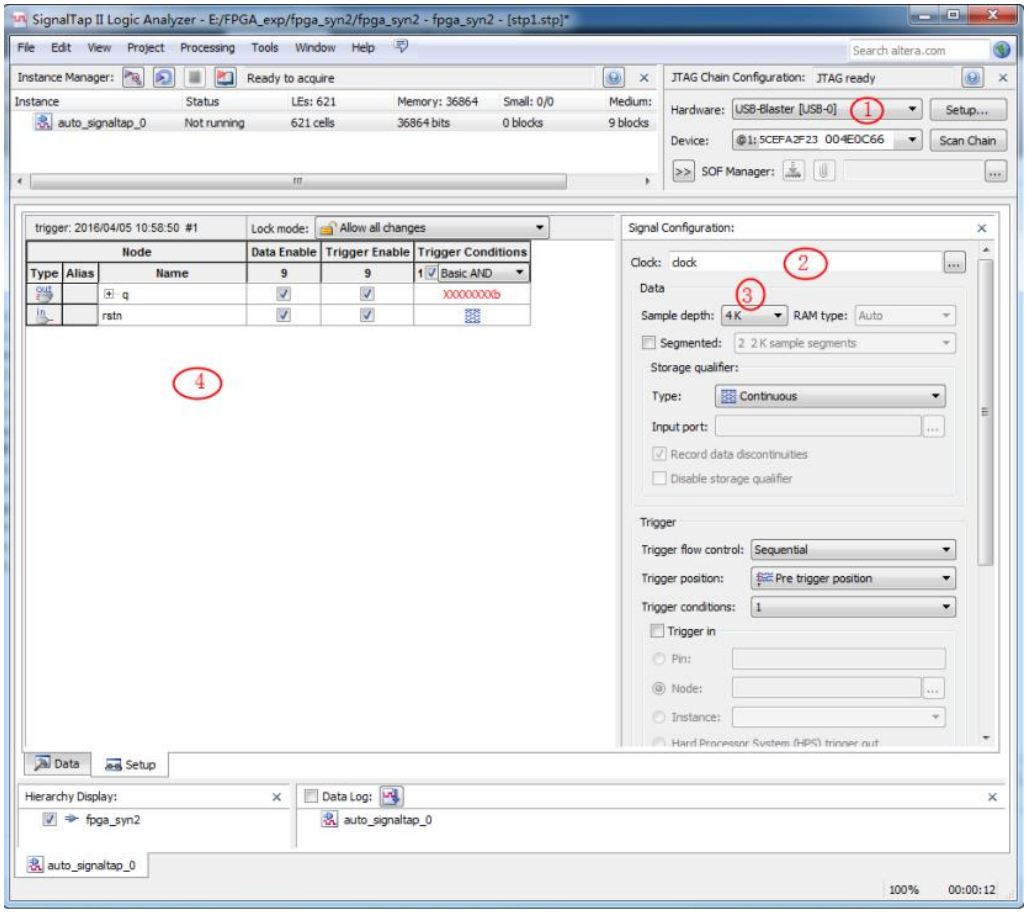

MegaWizard Plug-In Manager [page 5 of 7]
ROM: 1-PORT
About
Documentation
1 Parameter Settings
2 EDA
3 Summary
General
Regs/Clken/Aclrs
Mem Init
mystorage
address[9..0]
q[7..0]
8 bits
1024 words
clock
Block type: AUTO
Do you want to specify the initial content of the memory?
No, leave it blank
Initialize memory content data to XX..X on power-up in simulation
1
Yes, use this file for the memory content data
(You can use a Hexadecimal (Intel-format) File [.hex] or a Memory Initialization File [.mif])
2
Browse...
File name: ./mystorage.mif
The initial content file should conform to which port's dimensions?
PORT_A
Allow In-System Memory Content Editor to capture and update content independently of the system clock
The 'Instance ID' of this ROM is:
NONE
Resource Usage
2 M4K
Cancel
< Back
Next >
Finish

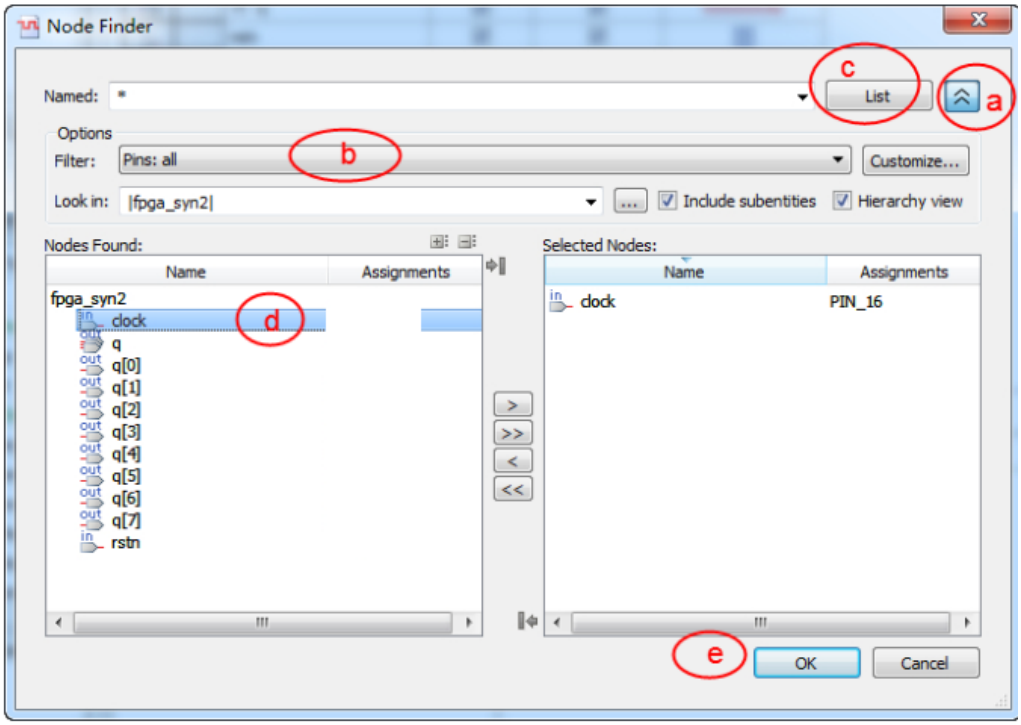




4. 编写顶层实体，顶层实体中需要包含 ieee 的 1164 库和 unsigned 库，并包含复位、时钟输入和 8 位正弦输出信号端。
5. 编写 test bench 文件，在仿真中测试输出功能。
6. 仿真完成后，可进行管脚分配，将 8 位输出分配到 EMOD0 的 EMOD[0 ~ 7]端口。

7. 向工程中添加 Signal Tap II 文件，文件打开后，设置如下：
其中，输出引脚 q 和 reset 信号线的设置是双击空白面板进行设置。

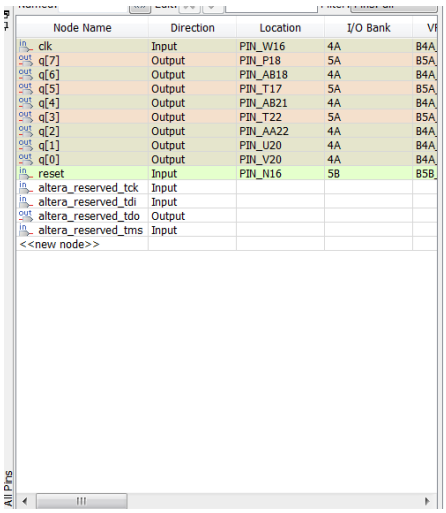




8. 将 SignalTap II 文件保存，重新全编译工程

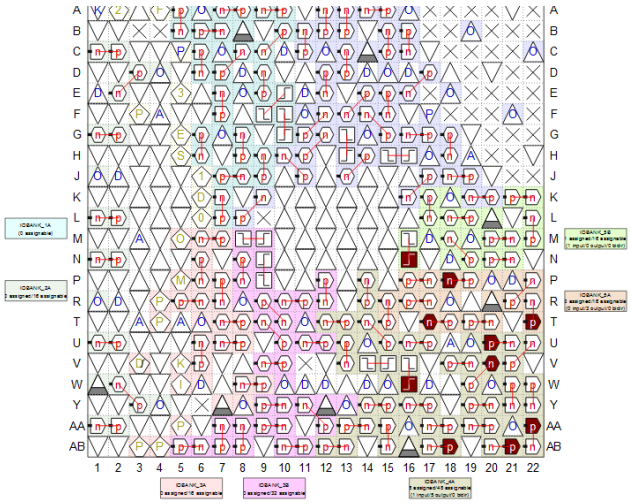
9. 将实体下载至 FPGA 中，之后在 SignalTap 文件中点击 Autorun 选项，即可看见正弦波的输出。

引脚分布



Node Name	Direction	Location	I/O Bank	VI
clk	Input	PIN_W16	4A	B4A
q[7]	Output	PIN_P18	5A	B5A
q[6]	Output	PIN_AB18	4A	B4A
q[5]	Output	PIN_T17	5A	B5A
q[4]	Output	PIN_AB21	4A	B4A
q[3]	Output	PIN_T22	5A	B5A
q[2]	Output	PIN_AA22	4A	B4A
q[1]	Output	PIN_U20	4A	B4A
q[0]	Output	PIN_V20	4A	B4A
reset	Input	PIN_N16	5B	B5B
altera_reserved_tck	Input			
altera_reserved_tdi	Input			
altera_reserved_tdo	Output			
altera_reserved_tms	Input			
<<new node>>				

All Pins



VHDL 源代码

(sin_generation.m) sin 生成函数

```
width=8;   %宽度是
depth=1024;   %深度是 1024
index = linspace(0,2*pi,depth);

sin_value = sin(index);

sin_value = sin_value * (2^(width-1) -1 )+2^(width-1);

sin_value = fix(sin_value);

sin_value =abs(sin_value);
%==============================开始写 mif 文件==============================
addr=0:depth-1;
str_width=strcat('WIDTH=',num2str(width));
str_depth=strcat('DEPTH=',num2str(depth));

fid=fopen('data.mif','w');   %打开或者新建 mif，存放位置和文件名任意
                                        %如果只写文件名，则在当前目录下建立此文件
fprintf(fid,str_width);
fprintf(fid,';\n');
```

```
fprintf(fid,str_depth);
fprintf(fid,';\n\n');
fprintf(fid,'ADDRESS_RADIX=HEX;\n');   %因为下面的数据输入我选的是 16 进制，

fprintf(fid,'DATA_RADIX=HEX;\n\n');
fprintf(fid,'CONTENT BEGIN\n');
fprintf(fid,'\t%X : %X;\n',[addr;sin_value])    %开始写数据了
fprintf(fid,'END;\n');
fclose(fid);
```

(Macro.vhd) 顶层实体 Macro

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;


entity Macro is
port(reset : in std_logic;
        clk : in std_logic := '1';
        q    : out std_logic_vector(7 downto 0));
end Macro;

architecture behavior of Macro is

component mystorage is
    PORT
    (
        address         : IN STD_LOGIC_VECTOR (9 DOWNTO 0);
        clock           : IN STD_LOGIC    := '1';
        q           : OUT STD_LOGIC_VECTOR (7 DOWNTO 0)
    );
end component;

signal address:std_LOGIC_VECTOR(9 downto 0);

begin
ROM : mystorage port map(address, clk, q);
    process(clk,reset)
    variable count : std_LOGIC_VECTOR(9 downto 0) := "0000000000";
    begin

        if(reset = '0') then count := "0000000000";
        elsif(rising_edge(clk)) then
            count := count + 1;
```

```
            end if;

            address <= count;

            if(count = "1111111111")then
                  count := "0000000000";
            end if;

      end process;

end behavior;
```

(Macro_tb.vhd) test bench 文件

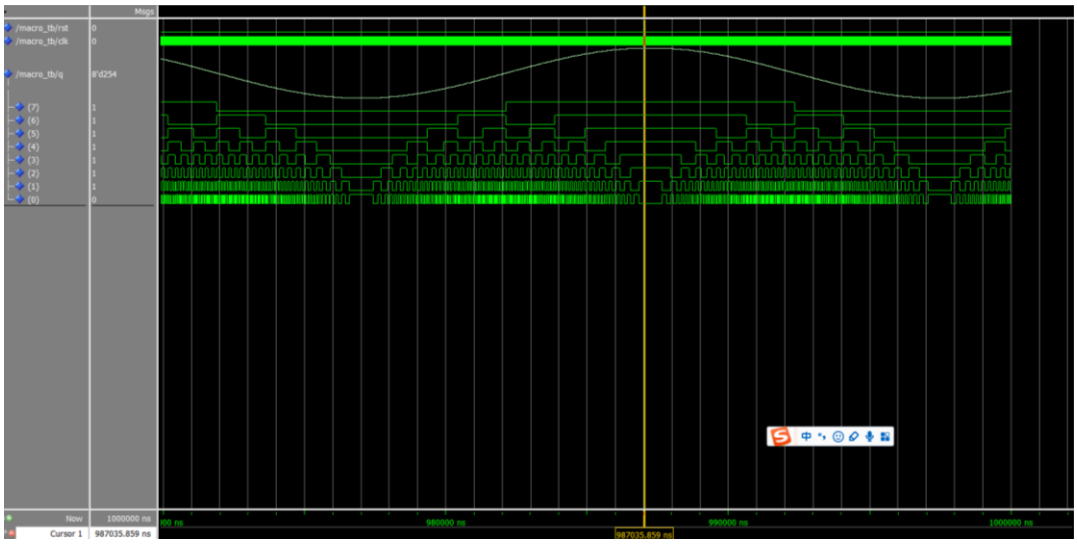
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity Macro_tb is
end Macro_tb;

architecture behav of Macro_tb is

component Macro is
port(reset : in std_logic;
            clk : in std_logic := '1';
            q    : out std_logic_vector(7 downto 0));
end component;

signal rst:std_logic;
signal clk:std_logic;
signal q:std_logic_vector(7 downto 0);

begin
      Xacro : Macro port map(rst,clk,q);

      process
            begin
            rst <= '1';
            clk <= '1';
            wait for 10ns;
            clk <= '0';
            wait for 10ns;
      end process;
end behav;
```

(mystorage.vhd) 自动生成 **mystorage** 文件

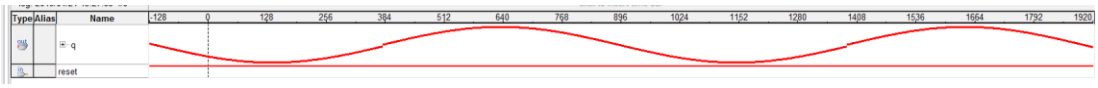
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

LIBRARY altera_mf;
USE altera_mf.all;

ENTITY mystorage IS
	PORT
	(
		address		: IN STD_LOGIC_VECTOR (9 DOWNTO 0);
		clock		: IN STD_LOGIC	:= '1';
		q		: OUT STD_LOGIC_VECTOR (7 DOWNTO 0)
	);
END mystorage;


ARCHITECTURE SYN OF mystorage IS

	SIGNAL sub_wire0	: STD_LOGIC_VECTOR (7 DOWNTO 0);



	COMPONENT altsyncram
	GENERIC (
		address_aclr_a		: STRING;
		clock_enable_input_a		: STRING;
		clock_enable_output_a		: STRING;
		init_file		: STRING;
		intended_device_family		: STRING;
		lpm_hint		: STRING;
		lpm_type		: STRING;
		numwords_a		: NATURAL;
		operation_mode		: STRING;
		outdata_aclr_a		: STRING;
		outdata_reg_a		: STRING;
		widthad_a		: NATURAL;
		width_a		: NATURAL;
		width_byteena_a		: NATURAL
	);
	PORT (
			address_a	: IN STD_LOGIC_VECTOR (9 DOWNTO 0);
```

```
            clock0      : IN STD_LOGIC ;
            q_a  : OUT STD_LOGIC_VECTOR (7 DOWNTO 0)
    );
    END COMPONENT;

BEGIN
    q       <= sub_wire0(7 DOWNTO 0);

    altsyncram_component : altsyncram
    GENERIC MAP (
        address_aclr_a => "NONE",
        clock_enable_input_a => "BYPASS",
        clock_enable_output_a => "BYPASS",
        init_file => "mystorage.mif",
        intended_device_family => "Cyclone V",
        lpm_hint => "ENABLE_RUNTIME_MOD=NO",
        lpm_type => "altsyncram",
        numwords_a => 1024,
        operation_mode => "ROM",
        outdata_aclr_a => "NONE",
        outdata_reg_a => "CLOCK0",
        widthad_a => 10,
        width_a => 8,
        width_byteena_a => 1
    )
    PORT MAP (
        address_a => address,
        clock0 => clock,
        q_a => sub_wire0
    );
END SYN;
```

仿真结果记录



FPGA 验证结果



实验总结

1. 试验箱上 reset 案件按下是'1', 第一次 FPGA 验证时未出现输出图像，但是按下 reset 键后出现了图像输出，于是将顶层实体的 reset 判断从‘1’更改成‘0’，于是成功了。
2. 在知道使用宏之后，相当于知道了用 FPGA 快速调用 ROM 和 RAM 的方法，对于图像等大量存在的并行数据读入和输出有了很大的优势。