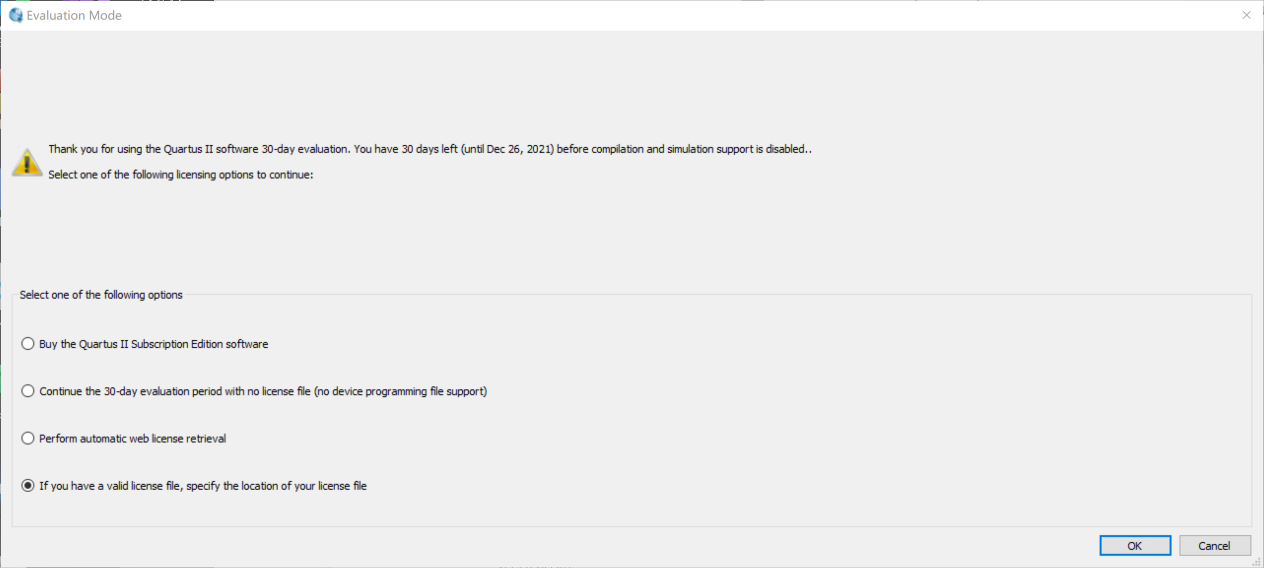
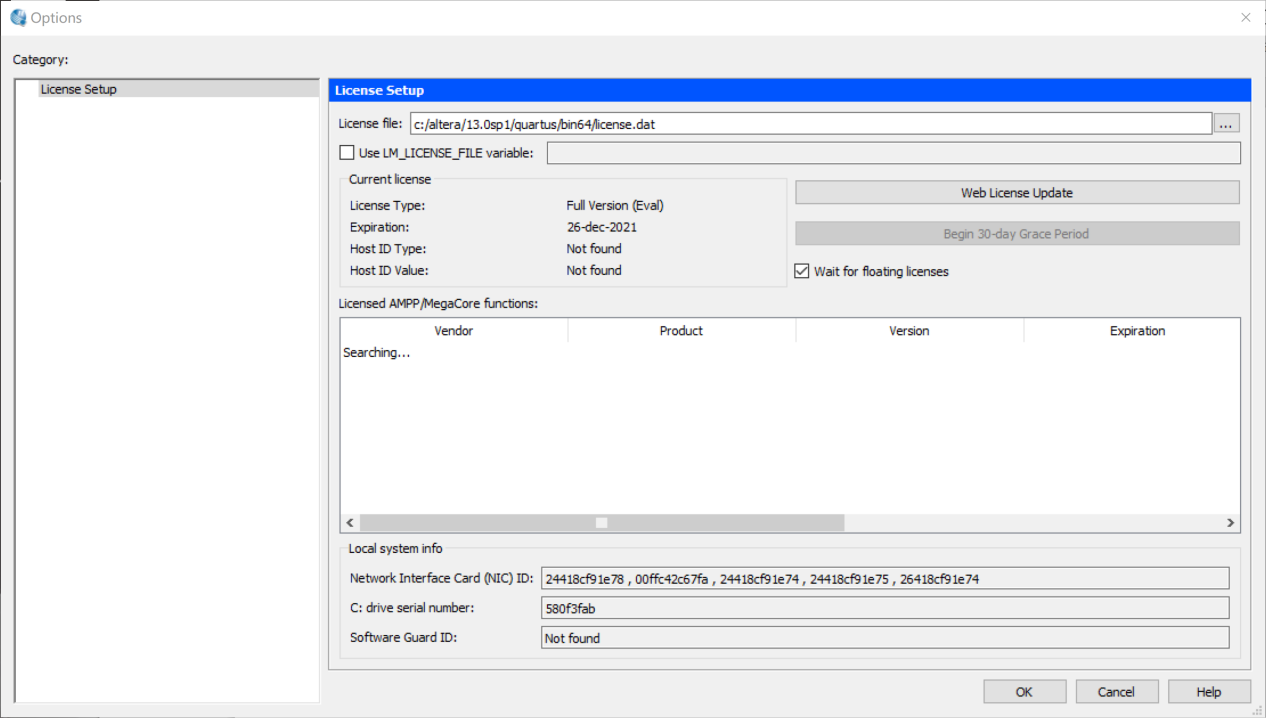
# 数字电路实验三报告

1. QuartusII软件基本使用步骤
2. 用文本编辑器正确编写源文件，并经modelsim仿真确认该电路设计正确
3. 双击打开QuartusII

弹出



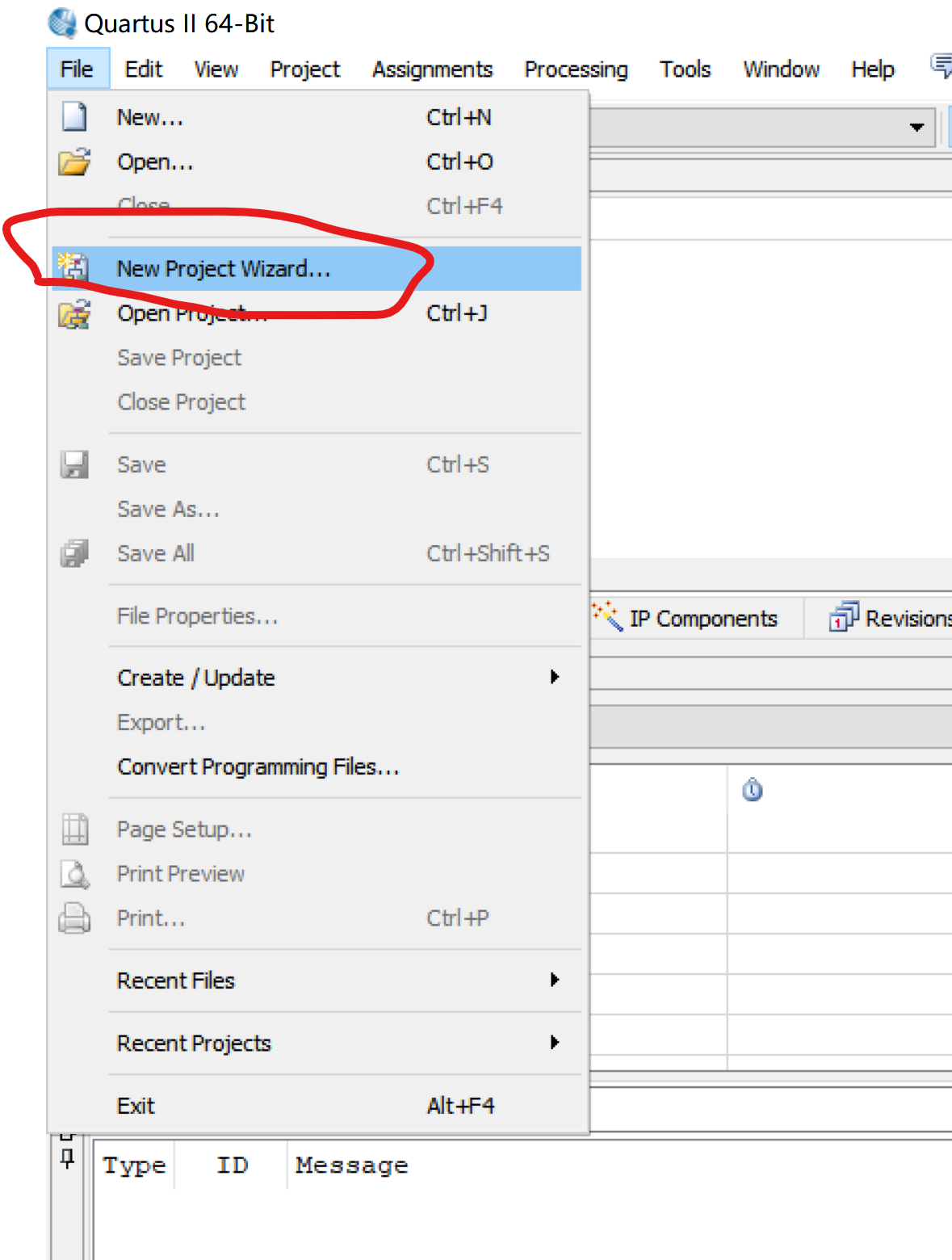
选择第四个选项，点击【OK】，弹出



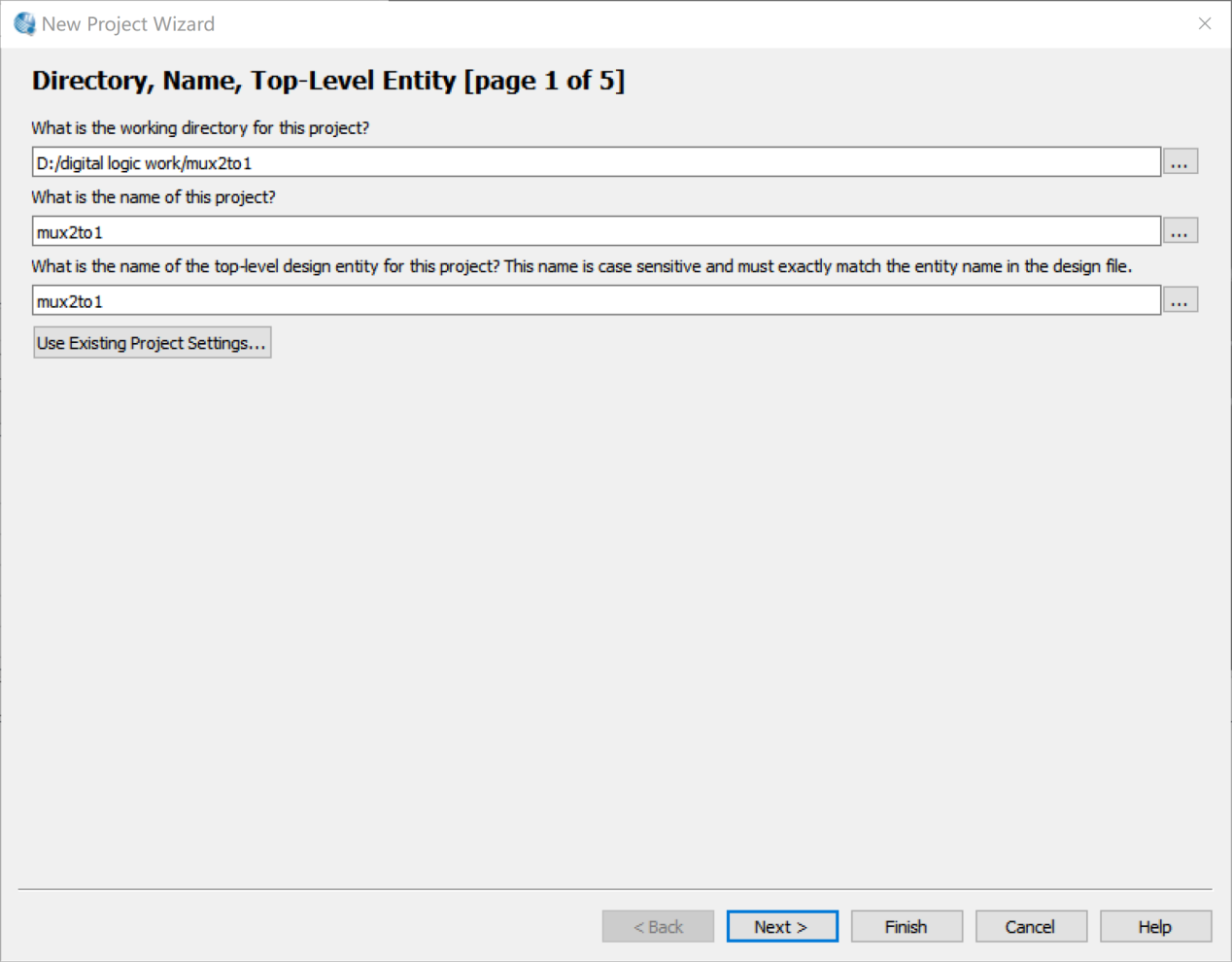
点击【Cancle】

1. 新建工程项目

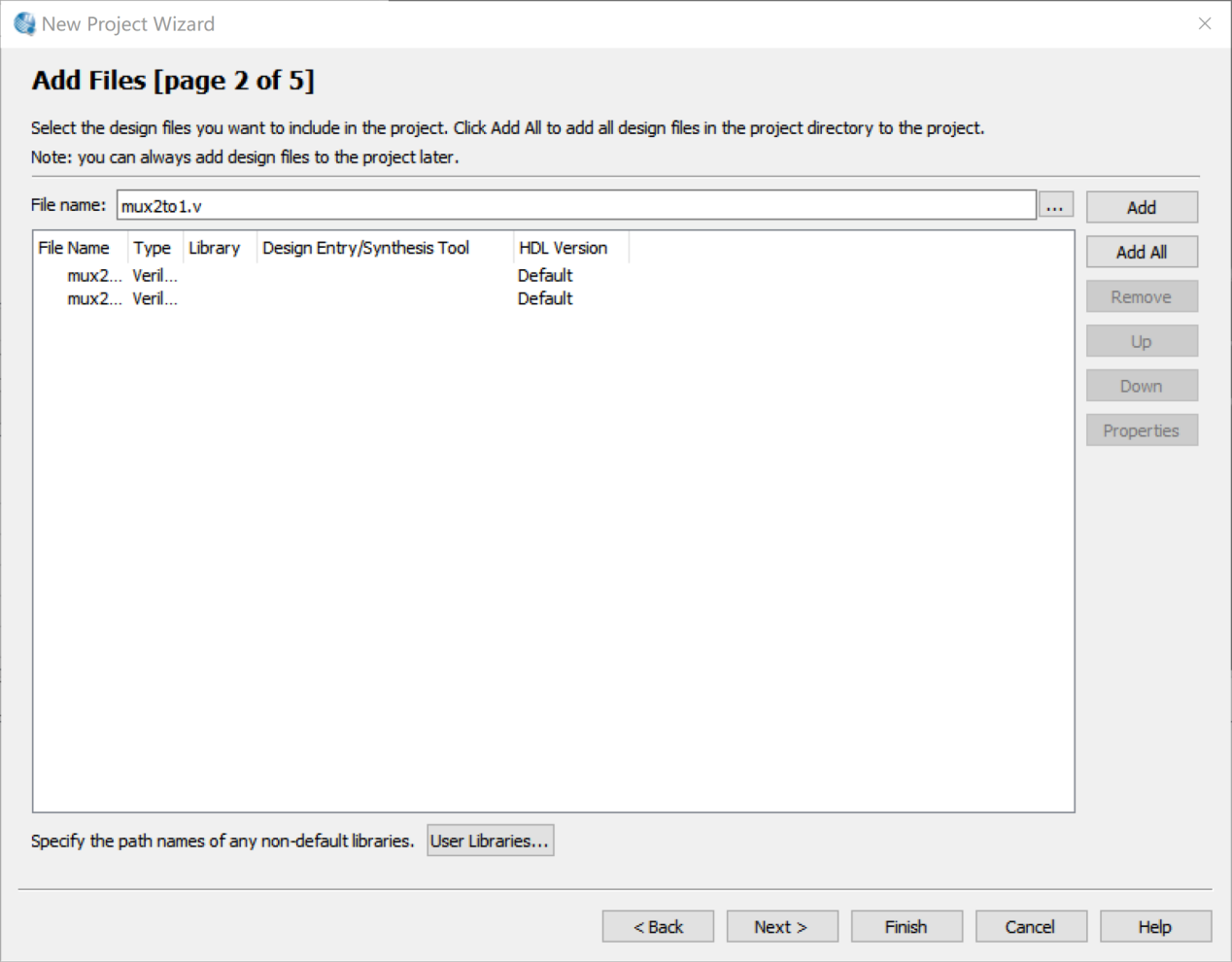
点击左上角【File】，选择【New Project Wizard...】



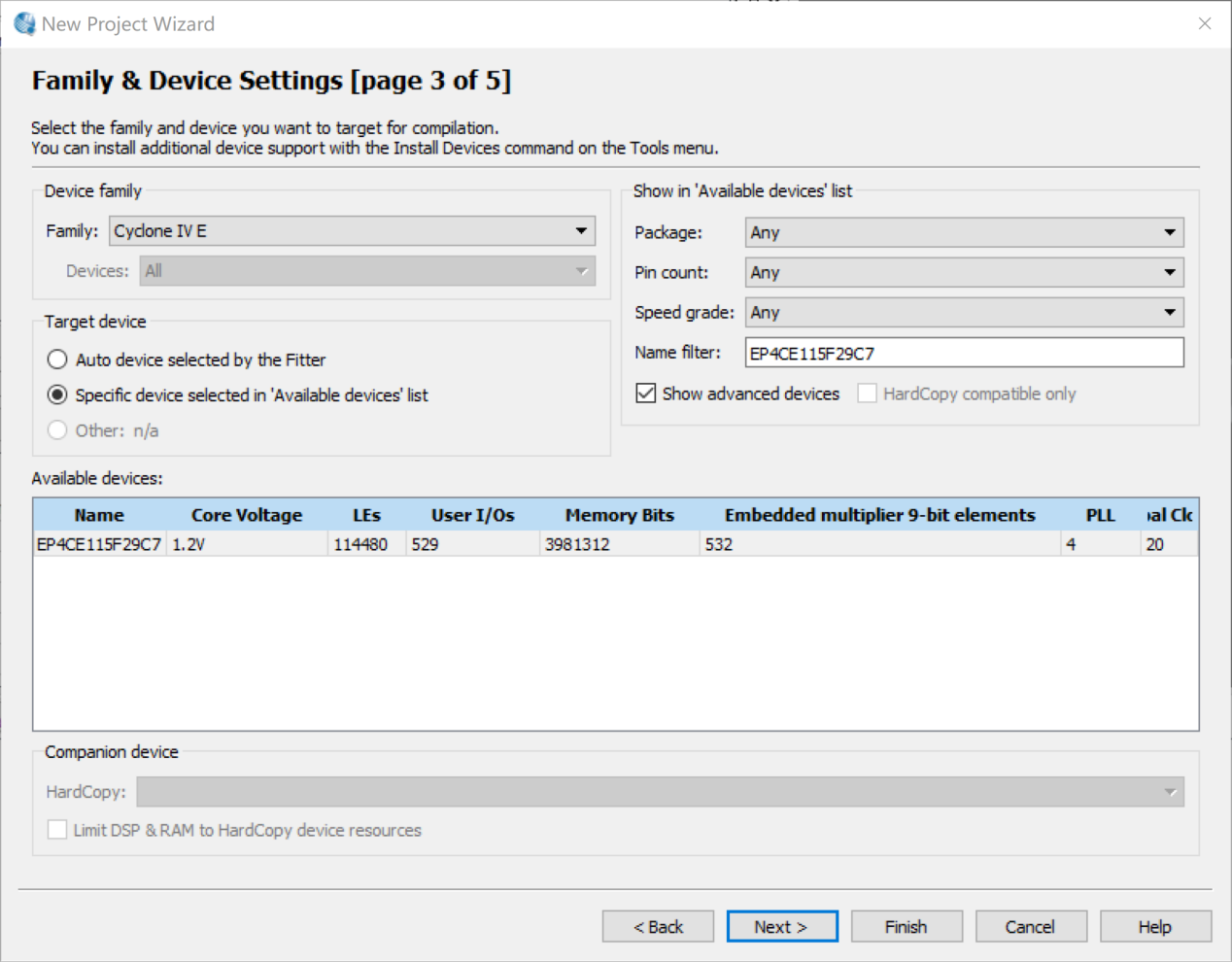
填写项目存储路径和工程名，不要出现中文路径



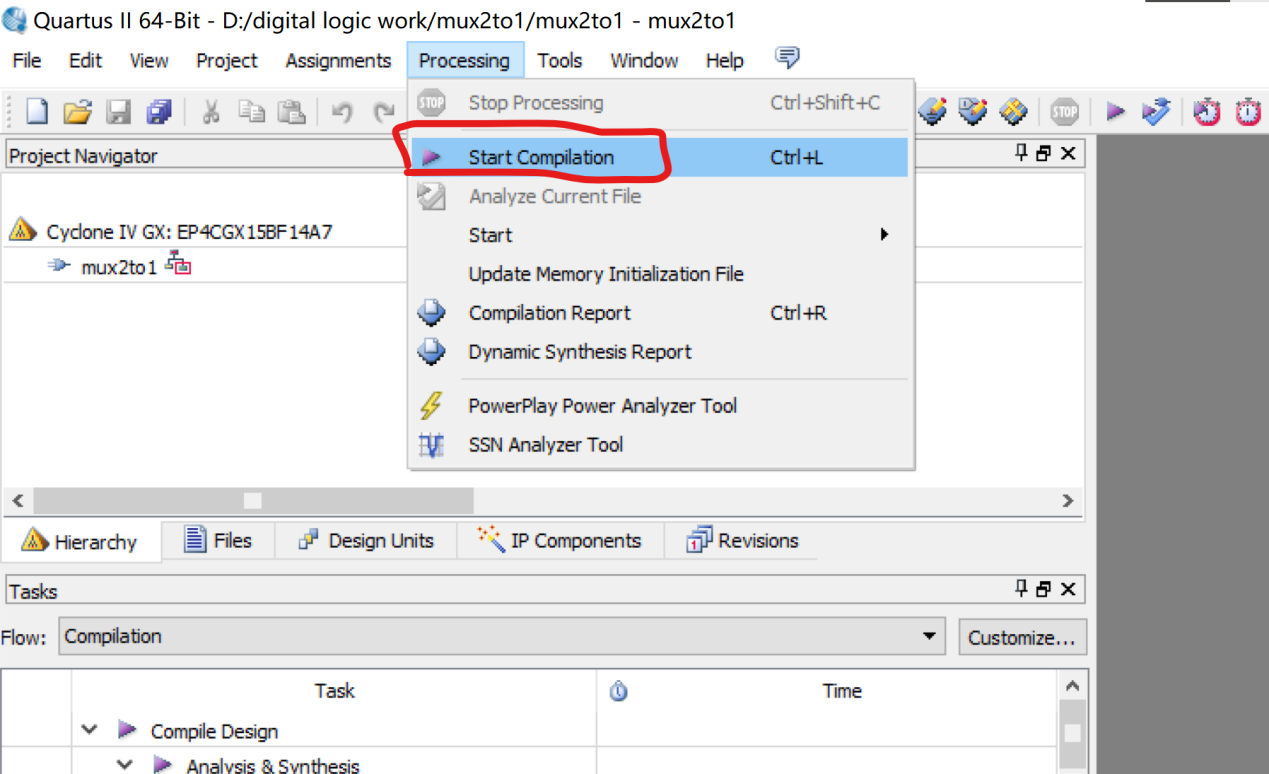
1. 添加已存在文件（可选），在【File name】下选择已经存在的工程项目，利用【Add】或【Add all】命令添加文件到新工程，点击【Next】



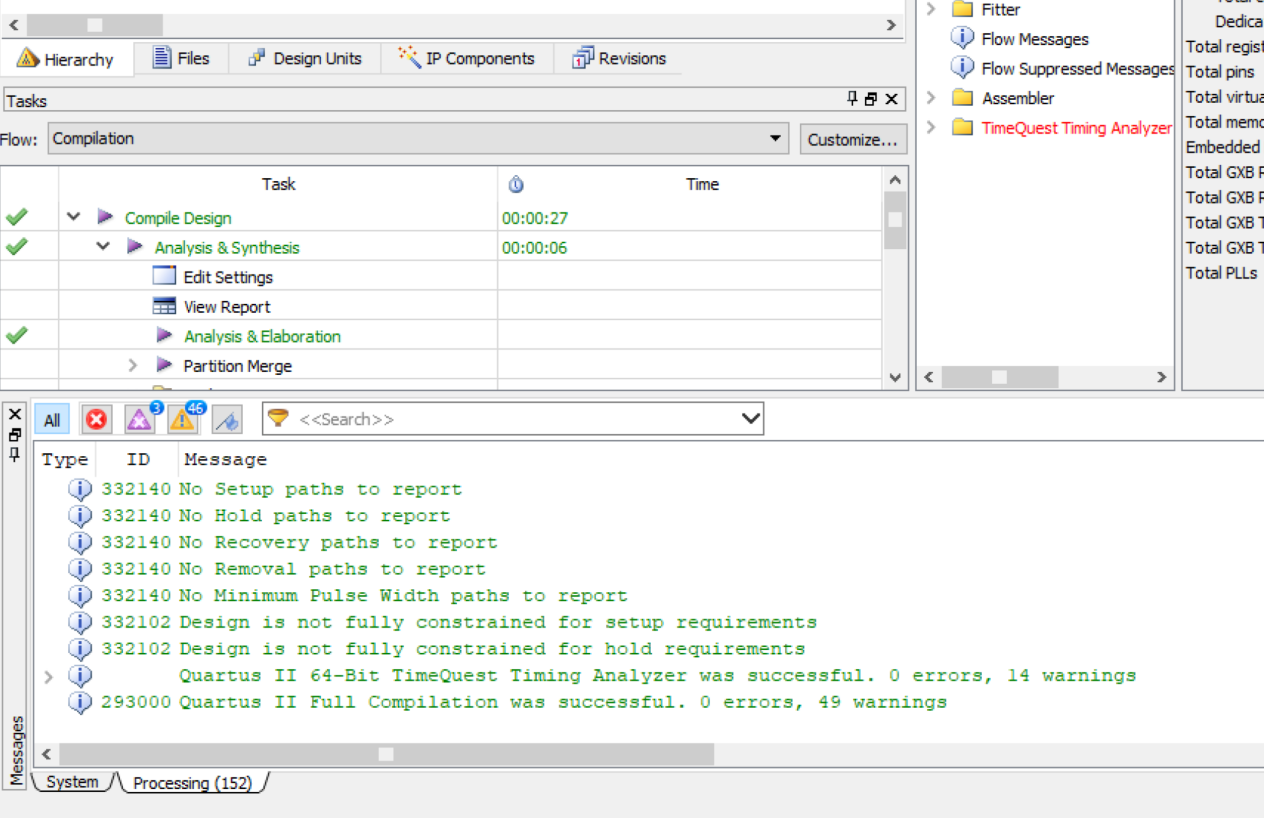
选择设备系列，并在【devices】下选择具体设备型号，这里采用Altera公司的Cyclone IV系列的EP4CE115F29C7型FPGA作为建立工程、代码编译、综合库时使用的库器件，点击【Next】



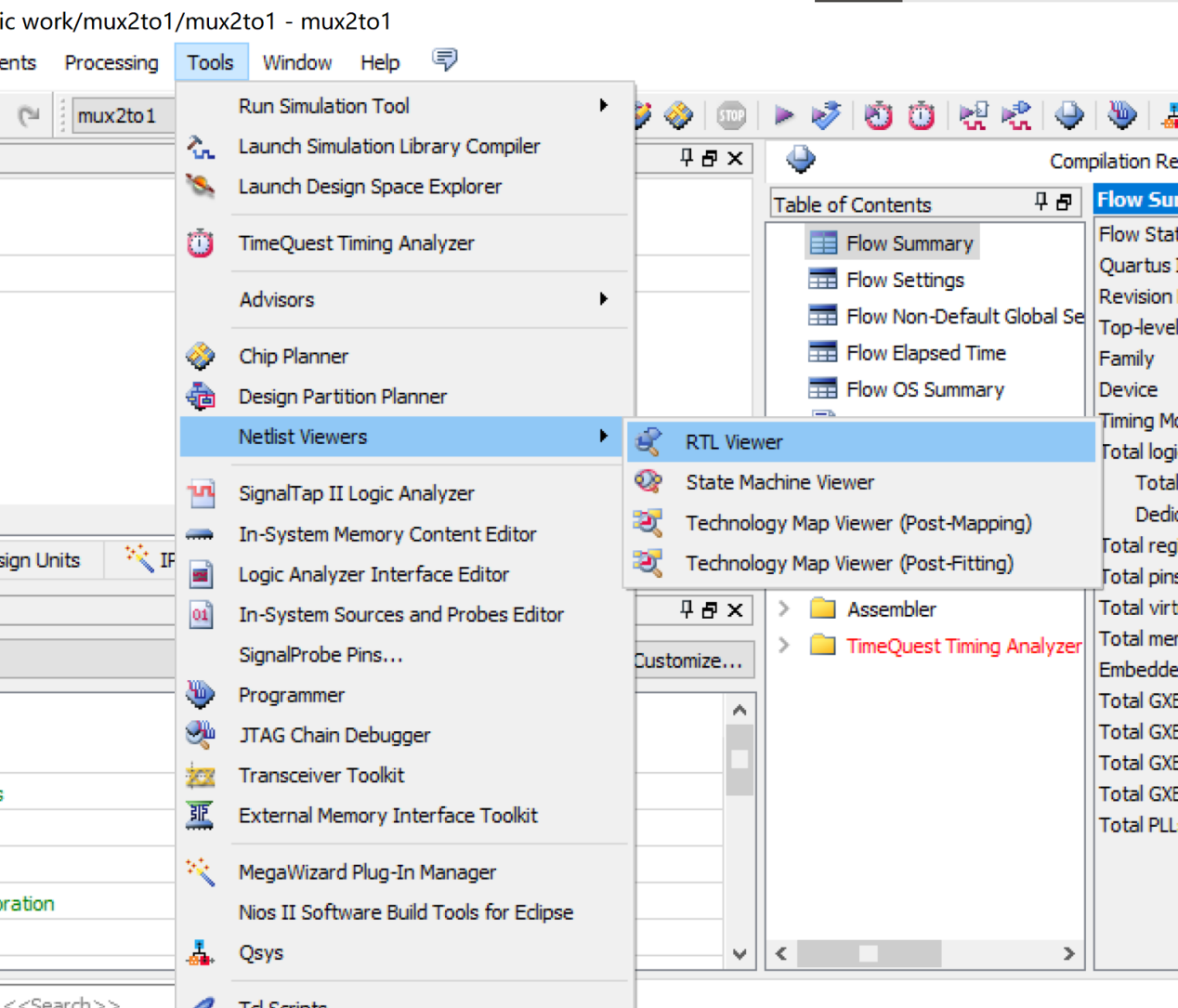
1. 点击【Processing】，选择【Start Compilation】进行编译

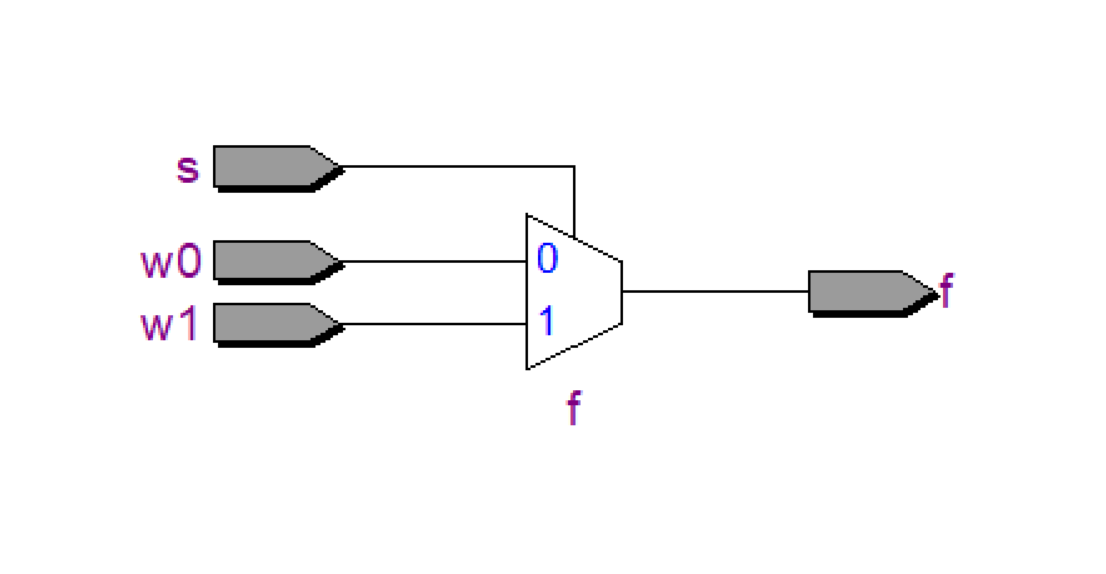


等待编译完成



1. 点击【Tool】，选择【RTL viewer】工具查看电路图结构是否和预期设计一致





1. 实验内容
2. D\_Flip Flop\_latch

verilog代码如下：

module D\_latch(D,Clock,Q);

    input D,Clock;

    output reg Q;

    always @(posedge Clock)

        Q=D;

endmodule

testbench测试文件如下：

`timescale 1ns/1ps

module tb\_D\_larch;

reg D\_test;

reg Clock\_test;

wire Q\_test;

initial

begin

Clock\_test=0;

D\_test=0;

#10 D\_test=1;

#40 D\_test=0;

#40 D\_test=1;

#40 D\_test=0;

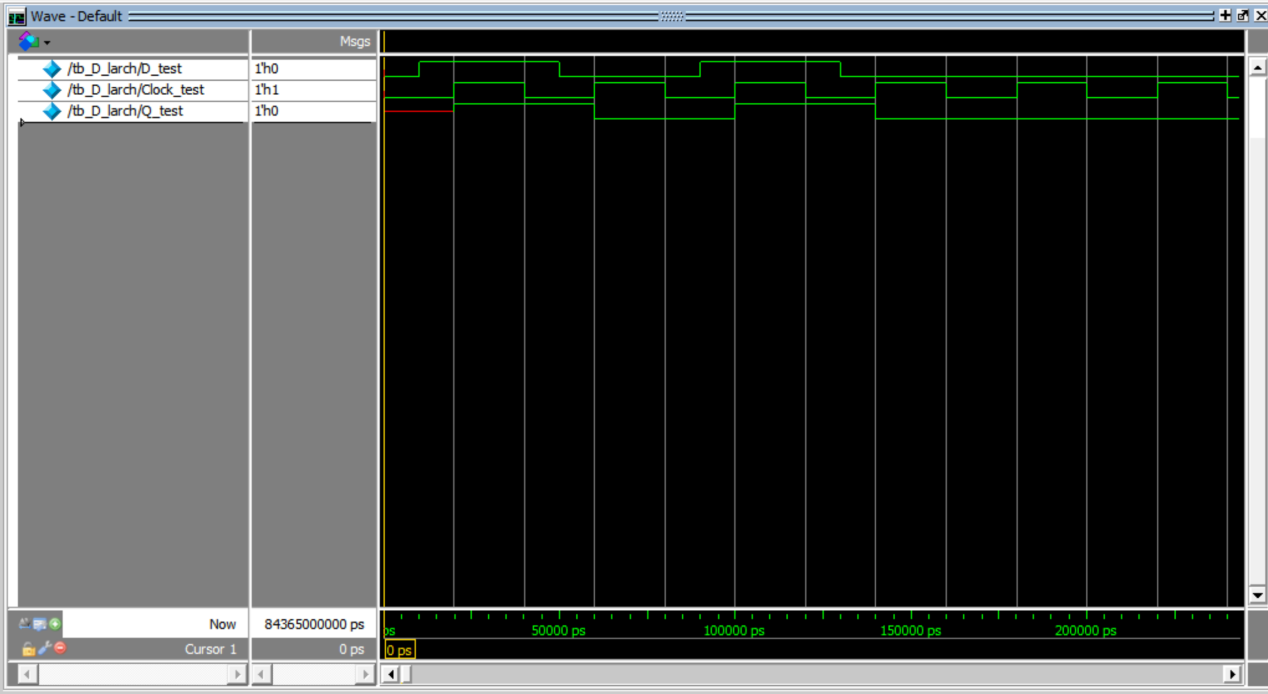
end

always #20 Clock\_test=~Clock\_test;

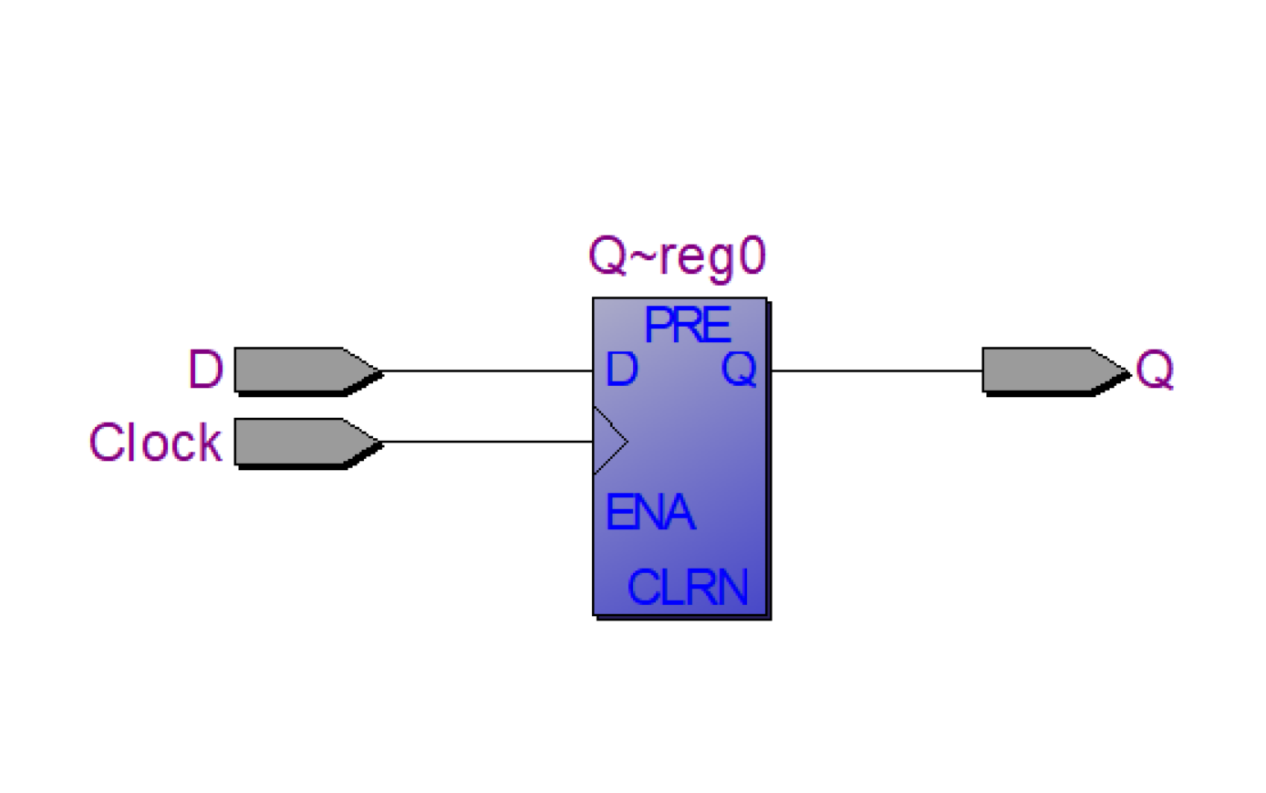
D\_latch UUT\_D\_latch(.D(D\_test),.Clock(Clock\_test),.Q(Q\_test));

endmodule

仿真波形如下：



综合电路图如下：



波形分析：

0~20ns内，Clock\_test=0，无论D\_test怎么变化，都不影响Q\_test的值，此时Q\_test的值是不确定的，因而为红色；

20ns时刻，Clock\_test由0变为1，时钟处于上升沿状态，此时D\_test=1，故Q\_test变为1；

20~40ns内，Clock\_test=1，Q\_test=1保持不变；

40ns时刻，Clock\_test由1变为0，时钟处于下降沿状态，Q\_test=1，保持不变；

40~60ns内，Clock\_test=0，Q\_test=1保持不变；

60ns时刻，Clock\_test由0变为1，时钟处于上升沿状态，此时D\_test=0，故Q\_test变为0；

因此从波形图可以看出，只有当Clock处于上升沿状态时，Q=D，即将D的值保存起来，其他时刻Q保持不变，因此该电路实现了一款时钟上升沿触发的D寄存器

1. 4bBit具有并行加载功能的移位寄存器

verilog代码如下：

module shift4(R,L,w,Clock,Q);

    input [3:0]R;

    input L,w,Clock;

    output reg [3:0]Q;

    always @(posedge Clock)

        if(L)

            Q<=R;

        else

        begin

            Q[0]<=Q[1];

            Q[1]<=Q[2];

            Q[2]<=Q[3];

            Q[3]<=w;

        end

endmodule

testbench测试文件如下：

`timescale 1ns/1ps

module tb\_shift4;

reg [3:0]R\_test;

reg L\_test;

reg w\_test;

reg Clock\_test;

wire [3:0]Q\_test;

initial

begin

R\_test=4'b0000;

L\_test=1;

w\_test=1;

Clock\_test=0;

#30 L\_test=0;

#40 w\_test=0;

#40 w\_test=1;

#120 w\_test=0;R\_test=4'b1110;

#40 L\_test=1;

#40 L\_test=0;

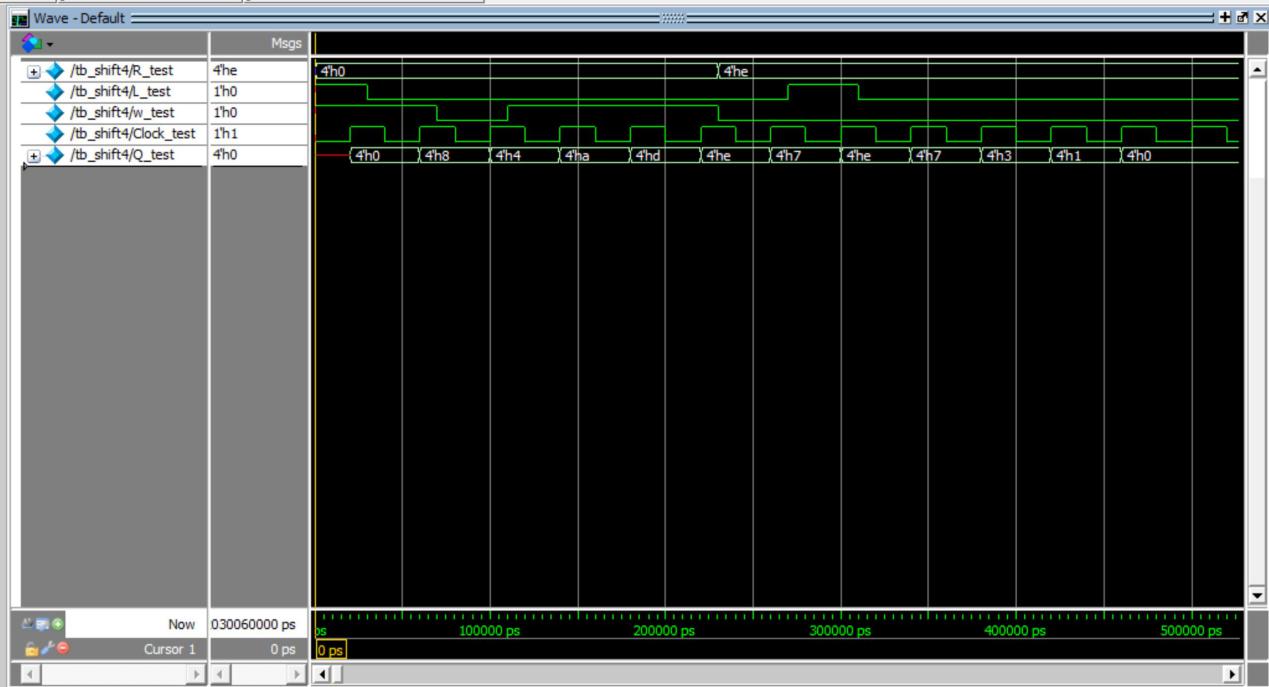
end

always #20 Clock\_test=~Clock\_test;

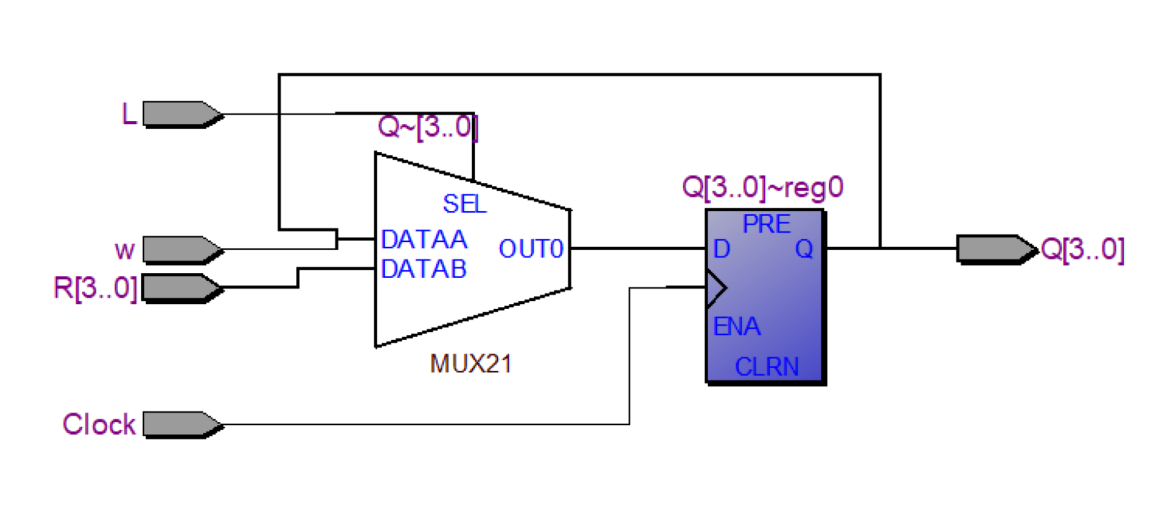
shift4 UUT\_shift4(.R(R\_test),.L(L\_test),.w(w\_test),.Clock(Clock\_test),.Q(Q\_test));

endmodule

仿真波形如下：



综合电路图如下：



波形分析：

0~20ns内，Clock\_test=0，此时Q\_test为随机值；

20ns时刻，时钟处于上升沿状态，此时L\_test=1，R\_test=0，执行并行加载功能，Q\_test=R\_test=0；

此后L\_test=0，每隔40ns时钟处于上升沿状态，执行移位功能

w\_test=1，Q\_test=4’b1000=8；

w\_test=0，Q\_test=4’b0100=4；

w\_test=1，Q\_test=4’b1010=a；

w\_test=1，Q\_test=4’b1101=d；

w\_test=1，Q\_test=4’b1110=e；

w\_test=0，Q\_test=4’b0111=7；

下一时钟上升沿，L\_test=1，R\_test=e，执行并行加载功能，Q\_test=R\_test=e；

接着L\_test=0，继续执行移位功能

综上，该电路实现了一款4bBit具有并行加载功能的移位寄存器。

1. 4 bit带复位功能的计数器

verilog代码如下：

module count(Resetn,Clock,E,Q);

    input Resetn,Clock,E;

    output reg [3:0]Q;

    always @(posedge Clock)

        if(!Resetn)

            Q<=0;

        else if(E)

            Q<=Q+1;

endmodule

testbench测试文件如下：

`timescale 1ns/1ps

module tb\_count;

reg Resetn\_test;

reg Clock\_test;

reg E\_test;

wire [3:0]Q\_test;

initial

begin

Resetn\_test=0;

Clock\_test=0;

E\_test=1;

#30 Resetn\_test=1;

#40 E\_test=0;

#40 Resetn\_test=0;

#40 E\_test=1;Resetn\_test=1;

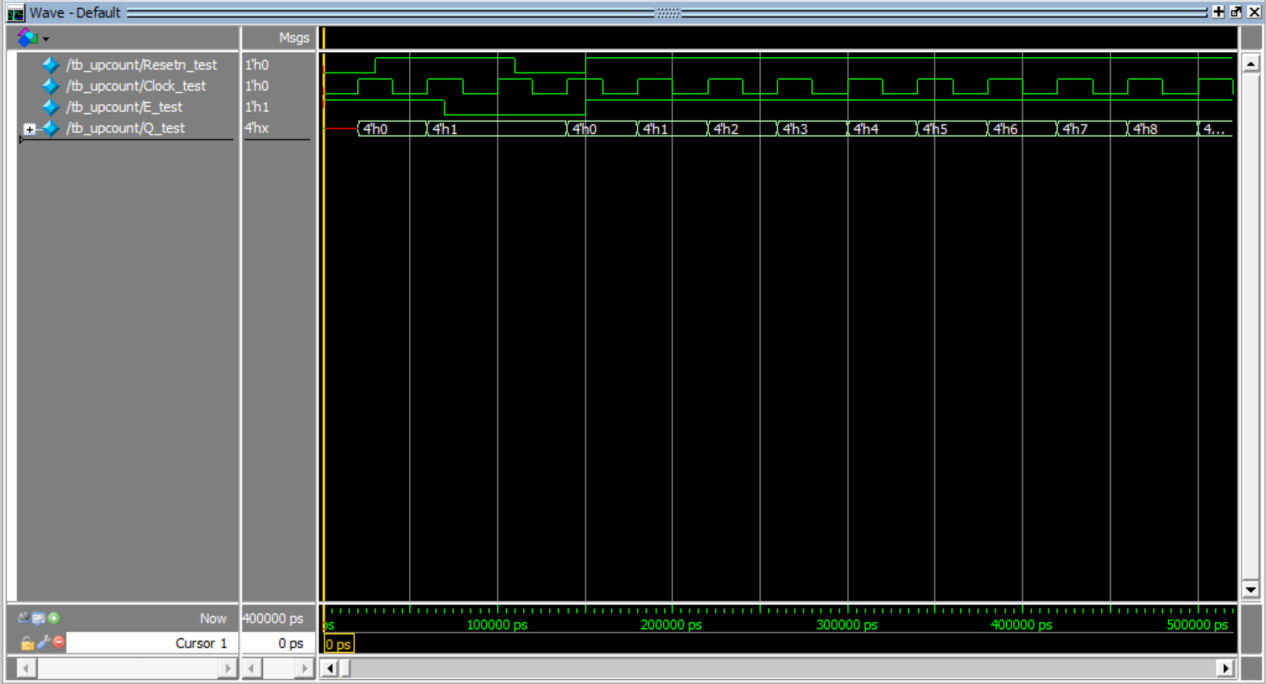
end

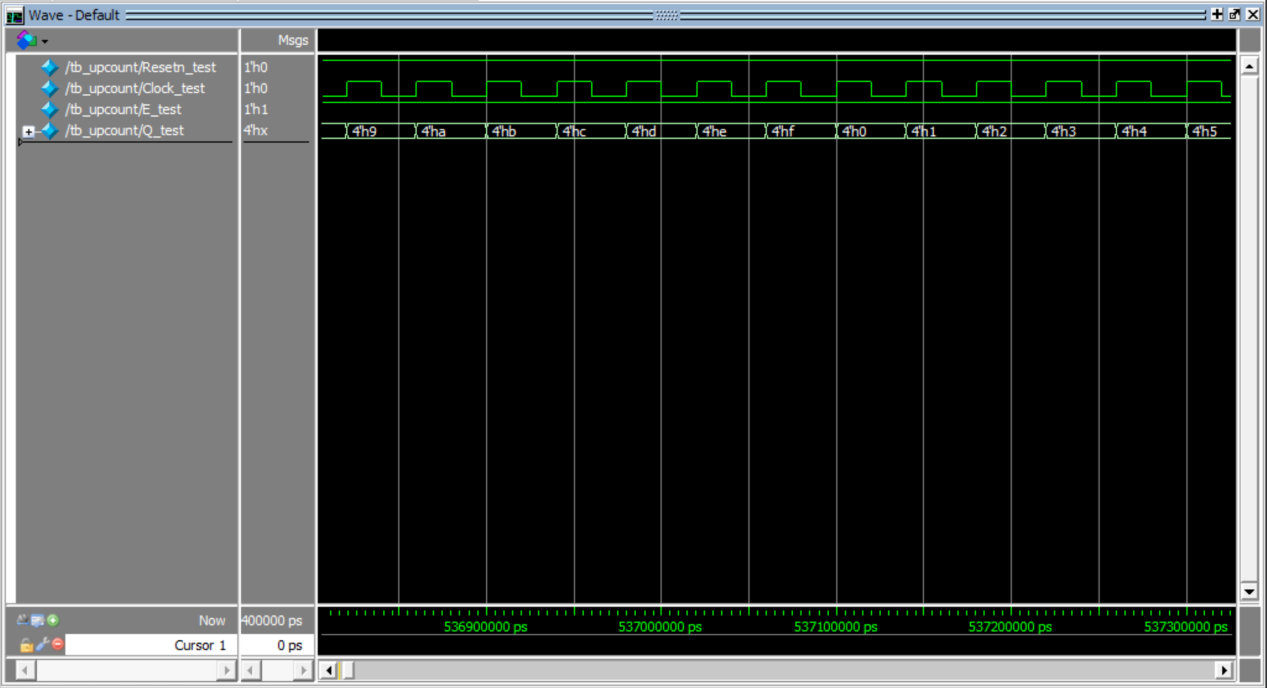
always #20 Clock\_test=~Clock\_test;

count UUT\_count(.Resetn(Resetn\_test),.Clock(Clock\_test),.E(E\_test),.Q(Q\_test));

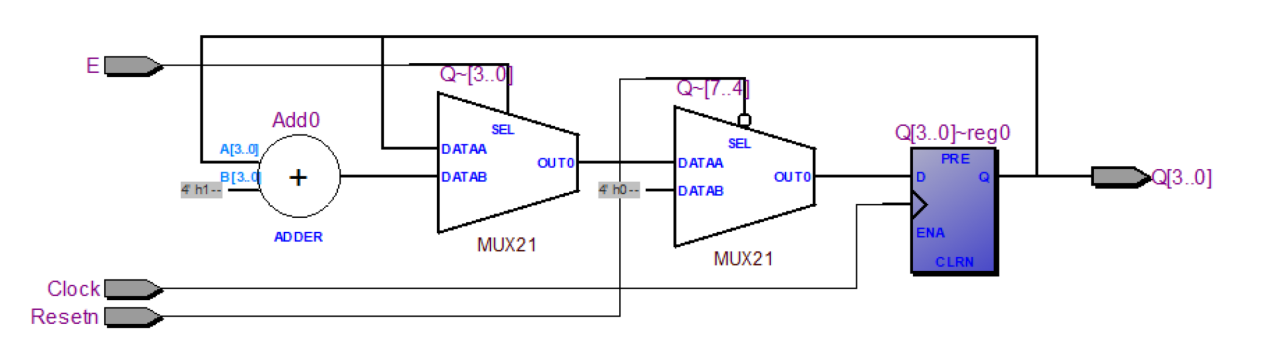
endmodule

仿真波形如下：





综合电路图如下：



波形分析：

0~20ns内，Clock\_test=0，Q\_test为随机值;

20ns时刻，时钟处于上升沿状态，此时Resetn=0，执行复位功能，Q\_test=0；

60ns时刻，时钟处于上升沿状态，此时Resetn=1，E=1，执行计数功能，Q\_test=1；

100ns时刻，时钟处于上升沿状态，此时Resetn=1，E=0，停止技术，Q\_test仍等于1；

140ns时刻，时钟处于上升沿状态，此时Resetn=0，执行复位功能，Q\_test=0；

此后Resetn=1，E=1，每隔40ns时钟处于上升沿状态，Q\_test+1，直到Q\_test=f，下一时钟上升沿到来时，Q\_test溢出，Q\_test=0，重新计数。

由此可以判定，当Resetn=0时，电路能够复位；该电路能按预期增加计数；当Q\_test=f时，继续计数可以按预定溢出，Q\_test=0。

1. 时钟频率20M,定时为1秒的定时器

verilog代码如下：

module timer(Resetn,Clock,E,flag,Q);

    input Resetn,Clock,E;

    output flag;

    output reg [31:0]Q;

    assign flag = (Q == 32'd20000000)?1:0;

    always @(posedge Clock)

        if(!Resetn)

            Q<=0;

        else if(E&&Q<32'd20000000)

            Q<=Q+1;

endmodule

testbench测试文件如下：

`timescale 1ns/1ps

module tb\_timer;

reg Resetn\_test;

reg Clock\_test;

reg E\_test;

wire flag\_test;

wire [31:0]Q\_test;

initial

begin

Resetn\_test=0;

Clock\_test=0;

E\_test=1;

#30 Resetn\_test=1;

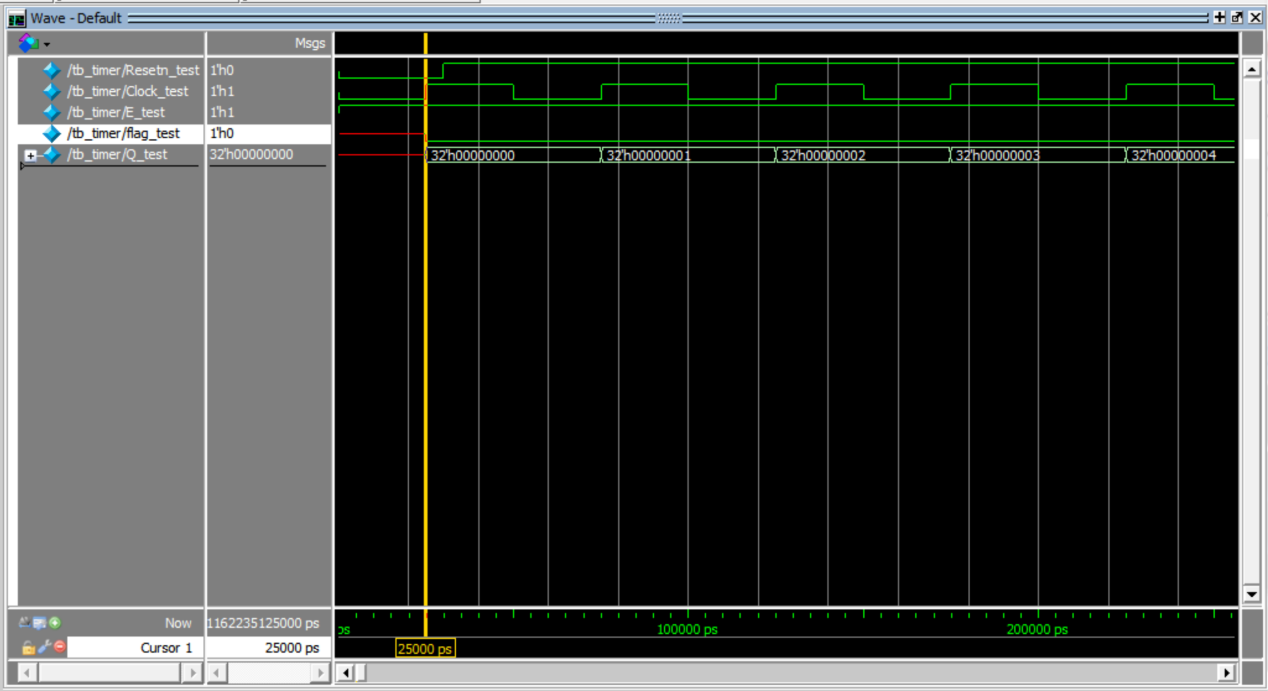
end

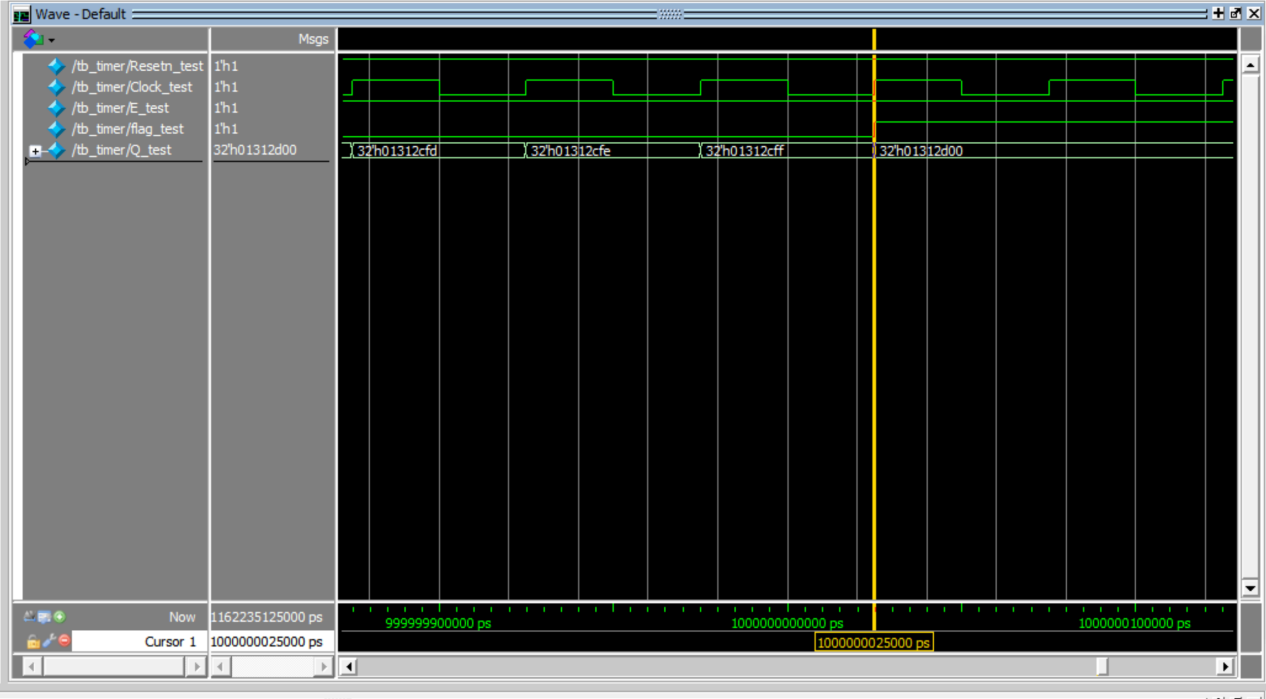
always #25 Clock\_test=~Clock\_test;

timer UUT\_timer(.Resetn(Resetn\_test),.Clock(Clock\_test),.E(E\_test),.flag(flag\_test),.Q(Q\_test));

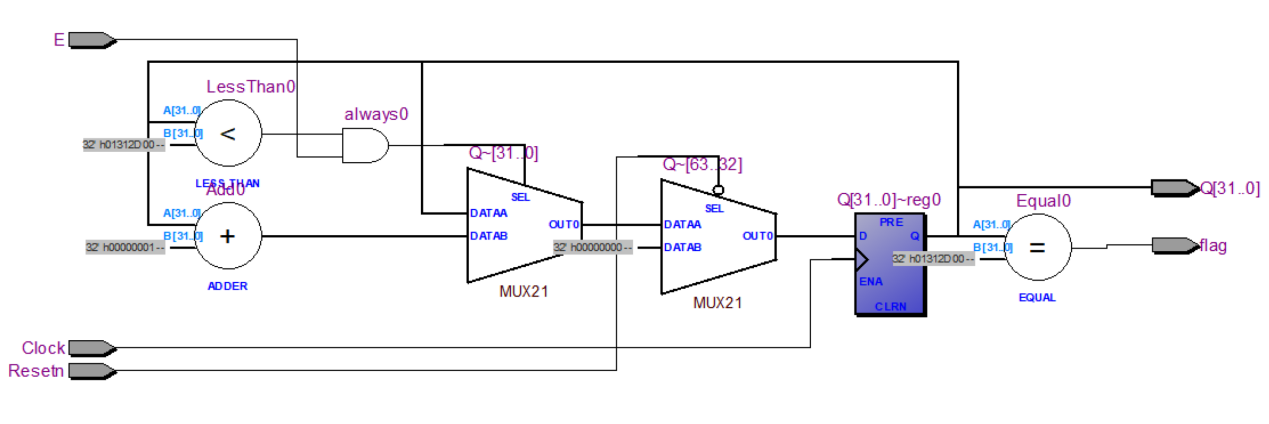
endmodule

仿真波形如下：





综合电路图如下：



波形分析：

设置Clock\_test每25ns变化一次可得到时钟周期为50ns的时钟，即时钟频率为20MHz；

0~25ns内，Clock\_test=0，Q\_test为随机值;

25ns时刻，时钟处于上升沿状态，此时Resetn\_test=0，执行复位功能，Q\_test=0，计数开始。

此后每个时钟上升沿Resetn\_test=1，执行计数功能，直到Q\_test = 32’h01312d00 = 20000000时停止计数，此时定时为1s，flag\_test=1，表示已到达定时时间。

综上，该电路实现了时钟频率20M,定时为1秒的一个定时器

1. 串并转换器

verilog代码如下：

module serial\_to\_parallel\_4bit(R,L,w,Clock,Q);

    input [3:0]R;

    input L,w,Clock;

    output wire [3:0]Q;

    muxdff Stage3 (w,R[3],L,Clock,Q[3]);

    muxdff Stage2 (Q[3],R[2],L,Clock,Q[2]);

    muxdff Stage1 (Q[2],R[1],L,Clock,Q[1]);

    muxdff Stage0 (Q[1],R[0],L,Clock,Q[0]);

endmodule

module muxdff(D0,D1,Sel,Clock,Q);

    input D0,D1,Sel,Clock;

    output reg Q;

    always @(posedge Clock)

        if(!Sel)

            Q<=D0;

        else

            Q<=D1;

endmodule

testbench测试文件如下：

`timescale 1ns/1ps

module tb\_serial\_to\_parallel\_4bit;

reg [3:0]R\_test;

reg L\_test;

reg w\_test;

reg Clock\_test;

wire [3:0]Q\_test;

initial

begin

R\_test=4'b0100;

L\_test=1;

w\_test=1;

Clock\_test=0;

#30 L\_test=0;

#80 w\_test=0;

#40 L\_test=1;R\_test=4'b1111;

#40 L\_test=0;

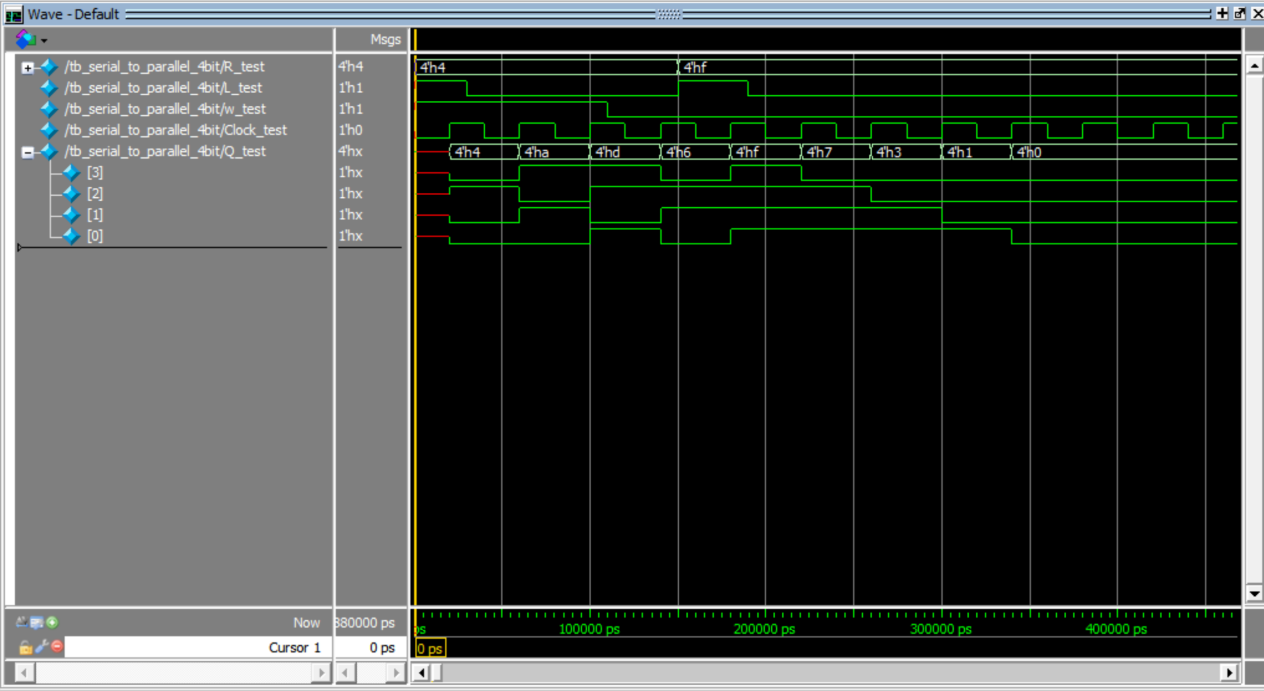
end

always #20 Clock\_test=~Clock\_test;

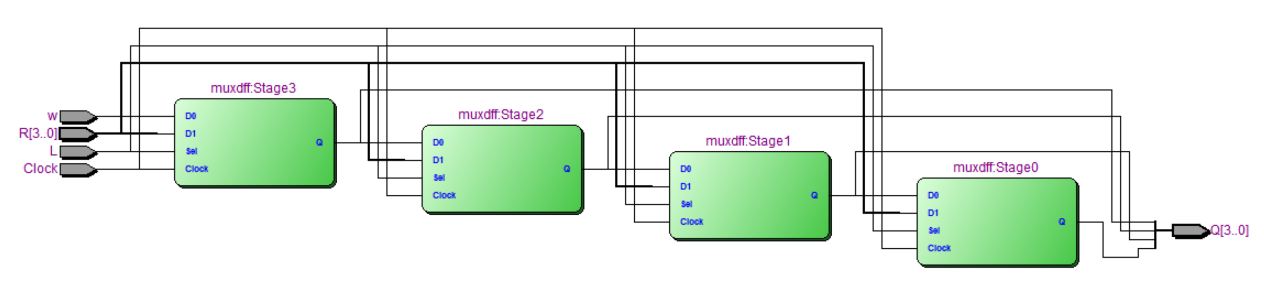
serial\_to\_parallel\_4bit UUT\_serial\_to\_parallel\_4bit(.R(R\_test),.Clock(Clock\_test),.L(L\_test),.w(w\_test),.Q(Q\_test));

endmodule

仿真波形如下：



综合电路图如下：



波形分析：

0~20ns内，Clock\_test=0，Q\_test为随机值;

20ns时刻，时钟处于上升沿状态，此时L\_test=1，处于并行接收模式，

Q\_test = R\_test = 4’b0100 = 4；

60ns时刻，时钟处于上升沿状态，此时L\_test=0，处于移位寄存器工作模式

w\_test=1，Q\_test=4’b1010=a；

100ns时刻，时钟处于上升沿状态，L\_test=0，

w\_test=1，Q\_test=4’b1101=d；

140ns时刻，时钟处于上升沿状态，L\_test=0，

w\_test=0，Q\_test=4’b0110=6；

180ns时刻，时钟处于上升沿状态，L\_test=1，处于并行接收模式，

Q\_test = R\_test = 4’b1111 = f；

此后L\_test=0，处于移位寄存器工作模式

Q\_test依次等于，4’b0111=7，4’b0011=3，4’b0001=1，4’b0000=0；

综上，该电路实现了串行加载和并行读取数据的功能，也即实现了串并转换器

1. 本次实验收获和心得

本次实验成功安装了QuartusII软件，掌握了软件的使用方法和设计流程，学习了可综合时序逻辑电路测试模块的编写、综合和不同层次的仿真。通过实验我更加体会到了时序逻辑与组合逻辑的区别，了解了不同工具之间配合使用的方法。