Terasic Technologies

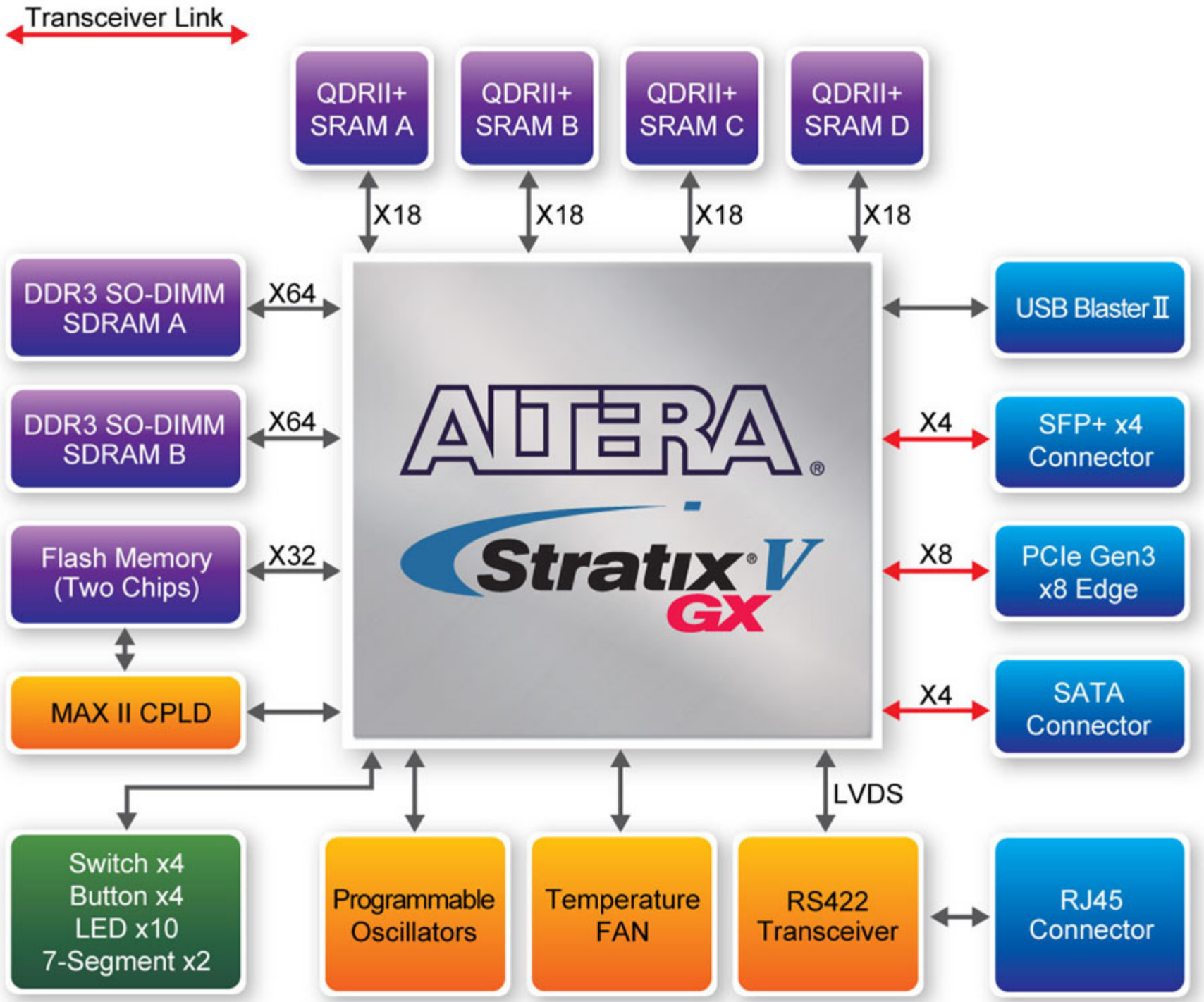
FFT design on DE5-Net Development Kit

友晶科技 李永杰

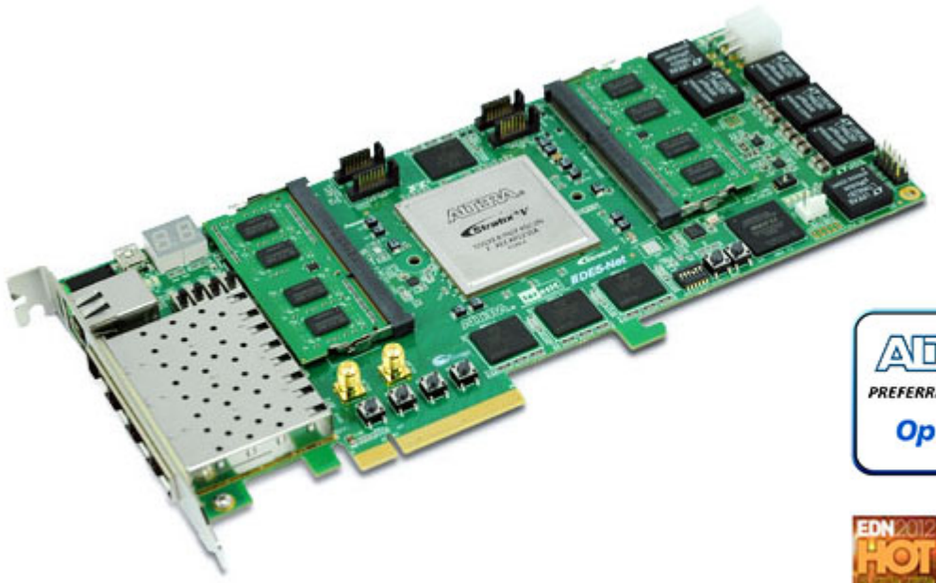
ALTERA UNIVERSITY PROGRAM

terasIC
www.terasic.com

DE5-Net 框图


Transceiver Link
QDRII+ SRAM A
QDRII+ SRAM B
QDRII+ SRAM C
QDRII+ SRAM D
X18
X18
X18
X18
DDR3 SO-DIMM SDRAM A
X64
DDR3 SO-DIMM SDRAM B
X64
Flash Memory (Two Chips)
X32
MAX II CPLD
ALTERA
Stratix V GX
USB Blaster II
X4
SFP+ x4 Connector
X8
PCIe Gen3 x8 Edge
X4
SATA Connector
LVDS
Switch x4
Button x4
LED x10
7-Segment x2
Programmable Oscillators
Temperature FAN
RS422 Transceiver
RJ45 Connector

DE5-Net 应用



EDN's
Hot Products
of 2012

1. 复杂Logic Design
2. NetFPGA
3. OpenCL

FFT on DE5-Net

实现方法：

- **FFT Logic Design**
  - 设计者根据自己算法编写代码仿真测试并在FPGA上实现
  - 开发时间，性能难以保证
- **FFT IP in Quartus® II Software (licensed is required)**
  - 设计者直接调用ALTERA FFT MegaCore function
  - 需要额外支付license 费用
  - 开发时间短，性能可以保证
- **DSP Builder**
  - 设计者使用DSP builder提供的Altera FFT simulink lib
  - 适合算法开发验证并最终在FPGA上硬件实现
- **OpenCL**
  - 设计者用OpenCL编程实现 FFT 算法
  - 适合PC或嵌入式平台系统加速运算

FFT Logic Design

参考文献:

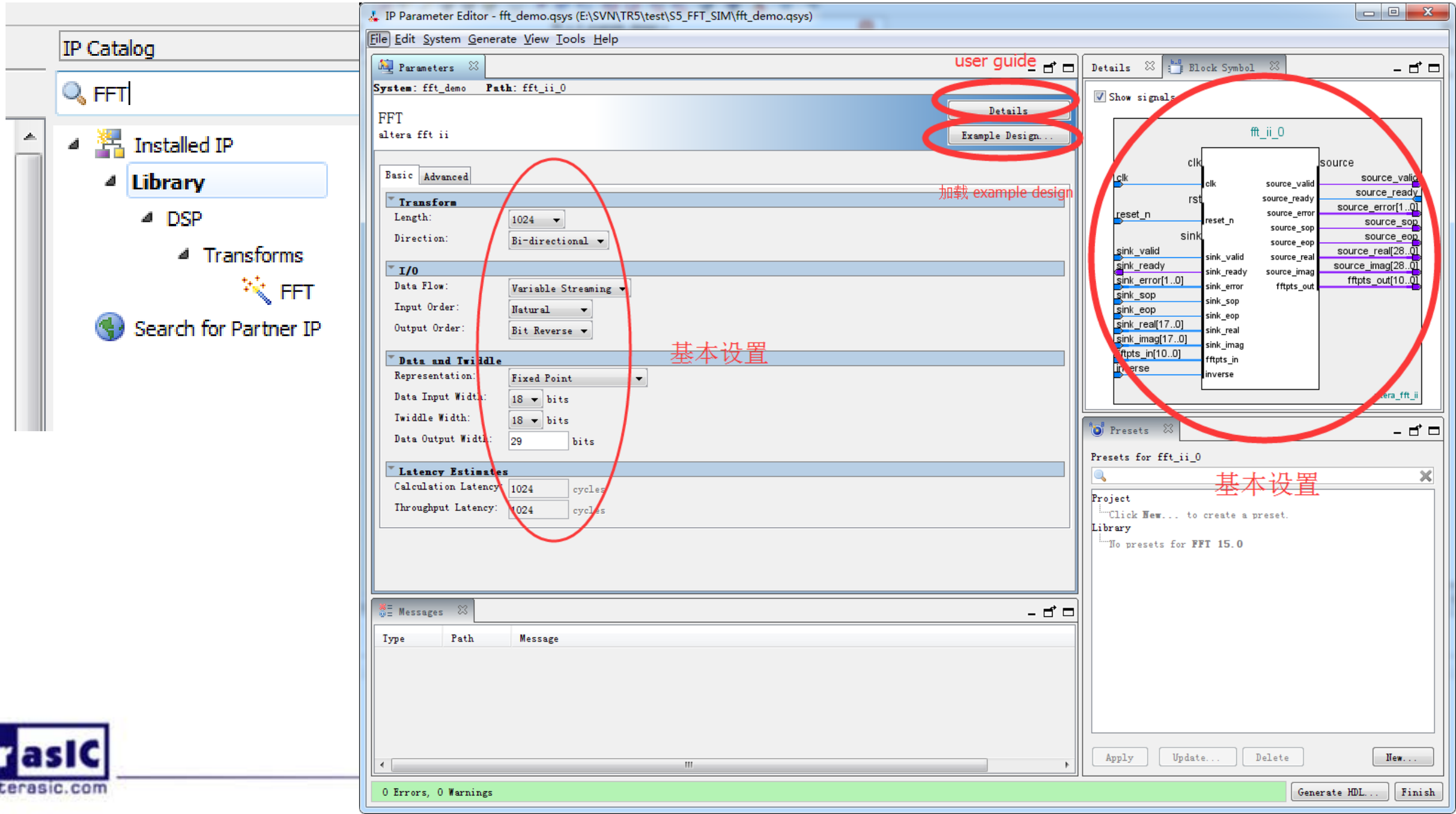
1. 基于**FPGA**的超高速**FFT**硬件实现　--- 电子科技大学学报

2. 基于CORDIC的一种高速实时定点FFT的FPGA实现 --- 微电子学与计算机

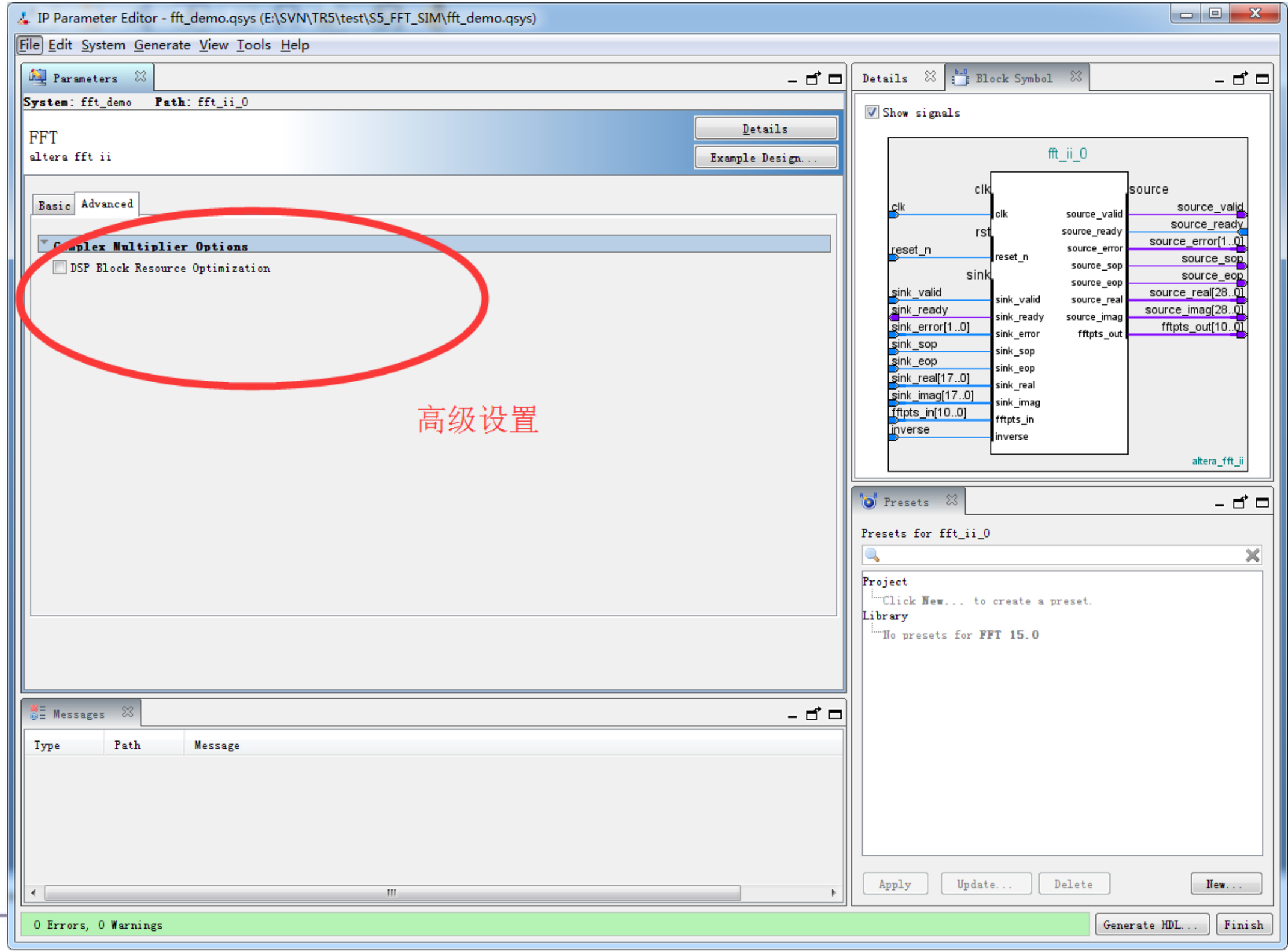
3. 基于FPGA的高速定点FFT算法的实现 --- 科学计算与信息处理

FFT IP （1）

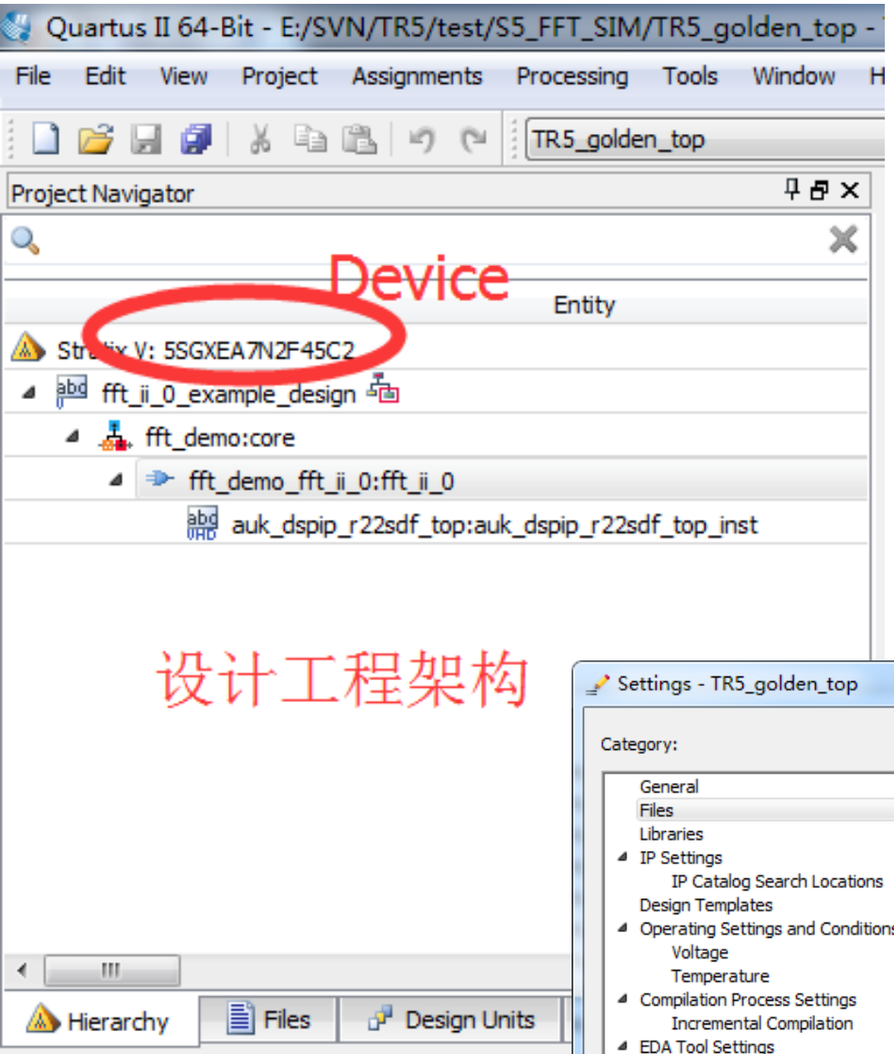
QuartusII 15.0 (IP Catalog)
Simulation with modelsim-altera



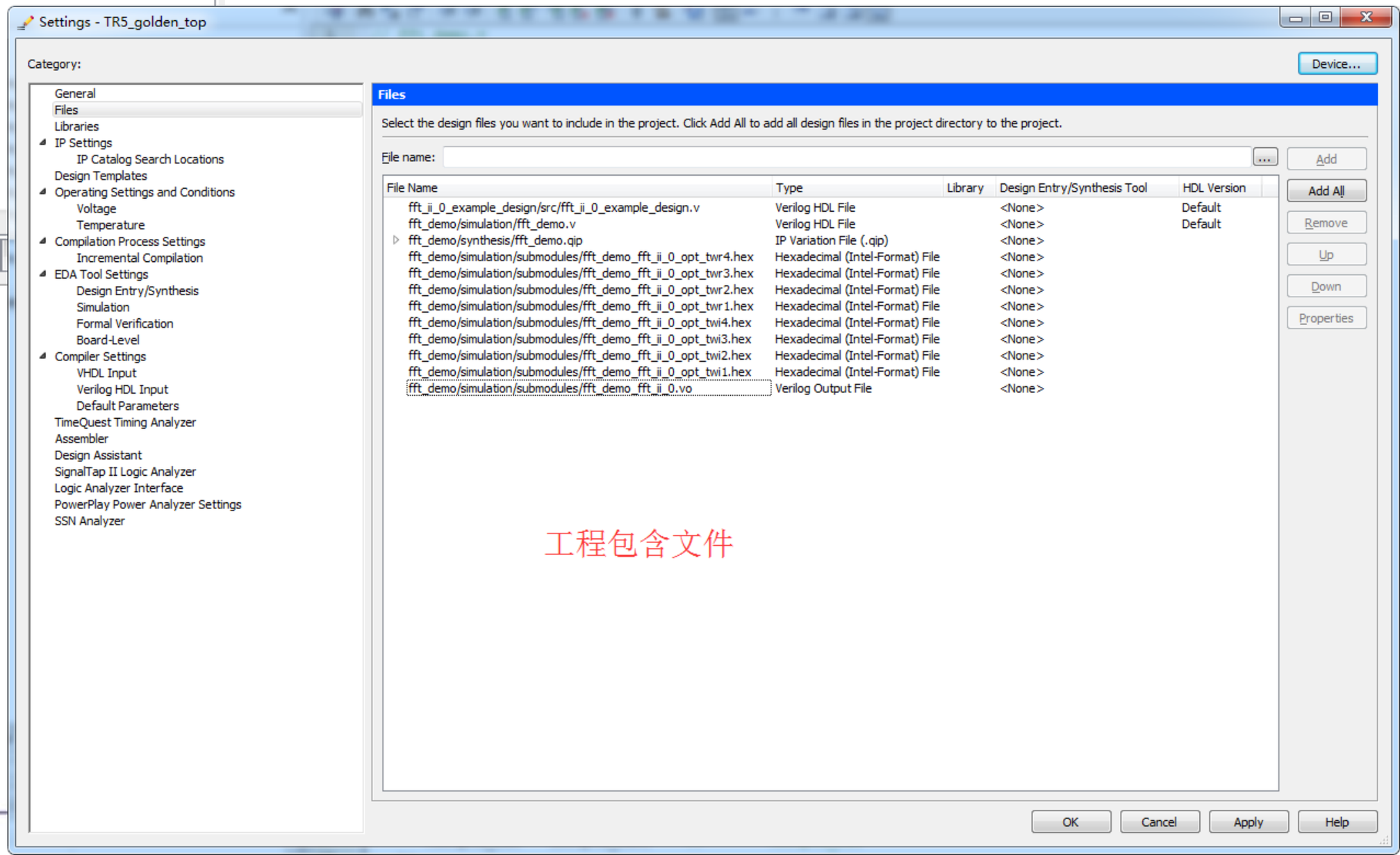
FFT IP （2）



FFT IP （3）



设计工程架构



FFT IP （4）

fft_ii_0_example_design.v | fft_demo.v

```
1  // fft_demo.v
2
3  // Generated using ACDS version 15.0 145
4
5  `timescale 1 ps / 1 ps
6  module fft_demo (
7        input  wire        clk,           //    clk.clk
8        input  wire        reset_n,       //    rst.reset_n
9        input  wire        sink_valid,    //   sink.sink_valid
10       output wire        sink_ready,    //       .sink_ready
11       input  wire [1:0]  sink_error,    //       .sink_error
12       input  wire        sink_sop,      //       .sink_sop
13       input  wire        sink_eop,      //       .sink_eop
14       input  wire [17:0] sink_real,     //       .sink_real
15       input  wire [17:0] sink_imag,     //       .sink_imag
16       input  wire [10:0] fftpts_in,     //       .fftpts_in
17       input  wire [0:0]  inverse,       //       .inverse
18       output wire        source_valid,  // source.source_valid
19       input  wire        source_ready,  //       .source_ready
20       output wire [1:0]  source_error,  //       .source_error
21       output wire        source_sop,    //       .source_sop
22       output wire        source_eop,    //       .source_eop
23       output wire [28:0] source_real,   //       .source_real
24       output wire [28:0] source_imag,   //       .source_imag
25       output wire [10:0] fftpts_out     //       .fftpts_out
26   );
27
28   fft_demo_fft_ii_0 fft_ii_0 (
29       .clk           (clk),           //    clk.clk
30       .reset_n       (reset_n),       //    rst.reset_n
31       .sink_valid    (sink_valid),    //   sink.sink_valid
32       .sink_ready    (sink_ready),    //       .sink_ready
33       .sink_error    (sink_error),    //       .sink_error
34       .sink_sop      (sink_sop),      //       .sink_sop
35       .sink_eop      (sink_eop),      //       .sink_eop
36       .sink_real     (sink_real),     //       .sink_real
37       .sink_imag     (sink_imag),     //       .sink_imag
38       .fftpts_in     (fftpts_in),     //       .fftpts_in
39       .inverse       (inverse),       //       .inverse
40       .source_valid  (source_valid),  // source.source_valid
41       .source_ready  (source_ready),  //       .source_ready
42       .source_error  (source_error),  //       .source_error
43       .source_sop    (source_sop),    //       .source_sop
44       .source_eop    (source_eop),    //       .source_eop
45       .source_real   (source_real),   //       .source_real
46       .source_imag   (source_imag),   //       .source_imag
47       .fftpts_out    (fftpts_out)     //       .fftpts_out
48   );
```

FFT IP （5）

fft_ii_0_example_design.v | fft_demo.v

DUT - design under test

```
1  // fft_ii_0_example_design.v
2
3  // Generated using ACDS version 15.0 145
4
5  `timescale 1 ps / 1 ps
6  module fft_ii_0_example_design (
7        input  wire         core_clk_clk,               //    core_clk.clk
8        input  wire         core_rst_reset_n,           //    core_rst.reset_n
9        input  wire         core_sink_valid,            //   core_sink.valid
10       output wire         core_sink_ready,            //            .ready
11       input  wire [1:0]   core_sink_error,            //            .error
12       input  wire         core_sink_startofpacket,    //            .startofpacket
13       input  wire         core_sink_endofpacket,      //            .endofpacket
14       input  wire [47:0]  core_sink_data,             //            .data
15       output wire         core_source_valid,          // core_source.valid
16       input  wire         core_source_ready,          //            .ready
17       output wire [1:0]   core_source_error,          //            .error
18       output wire         core_source_startofpacket,  //            .startofpacket
19       output wire         core_source_endofpacket,    //            .endofpacket
20       output wire [68:0]  core_source_data            //            .data
21 );
22
23 wire  [28:0] core_source_imag; // port fragment
24 wire  [28:0] core_source_real; // port fragment
25 wire  [10:0] core_fftpts_out;  // port fragment
26
27 fft_demo core (
28    .clk           (core_clk_clk),               //    clk.clk
29    .reset_n       (core_rst_reset_n),           //    rst.reset_n
30    .sink_valid    (core_sink_valid),            //   sink.valid
31    .sink_ready    (core_sink_ready),            //       .ready
32    .sink_error    (core_sink_error),            //       .error
33    .sink_sop      (core_sink_startofpacket),    //       .startofpacket
34    .sink_eop      (core_sink_endofpacket),      //       .endofpacket
35    .sink_real     ({core_sink_data[47:30]}),    //       .data
36    .sink_imag     ({core_sink_data[29:12]}),    //       .data
37    .fftpts_in     ({core_sink_data[11:1]}),     //       .data
38    .inverse       ({core_sink_data[0]}),        //       .data
39    .source_valid  (core_source_valid),          // source.valid
40    .source_ready  (core_source_ready),          //       .ready
41    .source_error  (core_source_error),          //       .error
42    .source_sop    (core_source_startofpacket),  //       .startofpacket
43    .source_eop    (core_source_endofpacket),    //       .endofpacket
44    .source_real   (core_source_real),           //       .data
45    .source_imag   (core_source_imag),           //       .data
46    .fftpts_out    (core_fftpts_out)             //       .data
47 );
48
```

FFT IP （6）

fft_ii_0_example_design.v | fft_demo.v | fft_ii_0_example_design_test_program.sv | fft_ii_0_example_design_tb.v

DUT, BFM for simulation

```
1  // fft_ii_0_example_design_tb.v
2
3  // Generated using ACDS version 15.0 145
4
5  `timescale 1 ps / 1 ps
6  module fft_ii_0_example_design_tb (
7     );
8
9     wire         fft_ii_0_example_design_inst_core_source_valid;              // fft_ii_0_example_design_inst:core_source_vali
10    wire  [68:0] fft_ii_0_example_design_inst_core_source_data;               // fft_ii_0_example_design_inst:core_source_data
11    wire         fft_ii_0_example_design_inst_core_source_ready;              // fft_ii_0_example_design_inst_core_source_bfm:
12    wire         fft_ii_0_example_design_inst_core_source_startofpacket;      // fft_ii_0_example_design_inst:core_source_sta
13    wire   [1:0] fft_ii_0_example_design_inst_core_source_error;              // fft_ii_0_example_design_inst:core_source_erro
14    wire         fft_ii_0_example_design_inst_core_source_endofpacket;        // fft_ii_0_example_design_inst:core_source_endo
15    wire   [0:0] fft_ii_0_example_design_inst_core_sink_bfm_src_valid;        // fft_ii_0_example_design_inst_core_sink_bfm:s
16    wire  [47:0] fft_ii_0_example_design_inst_core_sink_bfm_src_data;         // fft_ii_0_example_design_inst_core_sink_bfm:s
17    wire         fft_ii_0_example_design_inst_core_sink_bfm_src_ready;        // fft_ii_0_example_design_inst:core_sink_ready
18    wire   [0:0] fft_ii_0_example_design_inst_core_sink_bfm_src_startofpacket; // fft_ii_0_example_design_inst_core_sink_bfm:s
19    wire   [0:0] fft_ii_0_example_design_inst_core_sink_bfm_src_endofpacket;  // fft_ii_0_example_design_inst_core_sink_bfm:s
20    wire   [1:0] fft_ii_0_example_design_inst_core_sink_bfm_src_error;        // fft_ii_0_example_design_inst_core_sink_bfm:s
21    wire         fft_ii_0_example_design_inst_core_clk_bfm_clk_clk;           // fft_ii_0_example_design_inst_core_clk_bfm:cl
22    wire         fft_ii_0_example_design_inst_core_rst_bfm_reset_reset;       // fft_ii_0_example_design_inst_core_rst_bfm:re
23
24    fft_ii_0_example_design fft_ii_0_example_design_inst (
25       .core_clk_clk               (fft_ii_0_example_design_inst_core_clk_bfm_clk_clk),            //    core_clk.clk
26       .core_rst_reset_n           (fft_ii_0_example_design_inst_core_rst_bfm_reset_reset),        //    core_rst.reset_n
27       .core_sink_valid            (fft_ii_0_example_design_inst_core_sink_bfm_src_valid),         //   core_sink.valid
28       .core_sink_ready            (fft_ii_0_example_design_inst_core_sink_bfm_src_ready),         //            .ready
29       .core_sink_error            (fft_ii_0_example_design_inst_core_sink_bfm_src_error),         //            .error
30       .core_sink_startofpacket    (fft_ii_0_example_design_inst_core_sink_bfm_src_startofpacket), //            .startofpacket
31       .core_sink_endofpacket      (fft_ii_0_example_design_inst_core_sink_bfm_src_endofpacket),   //            .endofpacket
32       .core_sink_data             (fft_ii_0_example_design_inst_core_sink_bfm_src_data),          //            .data
33       .core_source_valid          (fft_ii_0_example_design_inst_core_source_valid),               // core_source.valid
34       .core_source_ready          (fft_ii_0_example_design_inst_core_source_ready),               //            .ready
35       .core_source_error          (fft_ii_0_example_design_inst_core_source_error),               //            .error
36       .core_source_startofpacket  (fft_ii_0_example_design_inst_core_source_startofpacket),       //            .startofpacket
37       .core_source_endofpacket    (fft_ii_0_example_design_inst_core_source_endofpacket),         //            .endofpacket
38       .core_source_data           (fft_ii_0_example_design_inst_core_source_data)                 //            .data
39    );
40
41    altera_avalon_clock_source #(
42       .CLOCK_RATE (50000000),
43       .CLOCK_UNIT (1)
44    ) fft_ii_0_example_design_inst_core_clk_bfm (
45       .clk (fft_ii_0_example_design_inst_core_clk_bfm_clk_clk)  // clk.clk
46    );
47
48    altera_avalon_reset_source #(
```

FFT IP （7）

fft_ii_0_example_design.v | fft_demo.v | fft_ii_0_example_design_test_program.sv

```
19
20      // console messaging level
21      import avalon_utilities_pkg::*;
22      import verbosity_pkg::*;
23
24      `define VERBOSITY VERBOSITY_NONE
25
26      //BFM hierachy
27      `define CLK tb.fft_ii_0_example_design_inst_core_clk_bfm
28      `define RST tb.fft_ii_0_example_design_inst_core_rst_bfm
29      `define SRC tb.fft_ii_0_example_design_inst_core_sink_bfm
30      `define SNK tb.fft_ii_0_example_design_inst_core_source_bfm
31
32      //Test parameters
33      `define BACK_PRESSURE "false"
34      `define FORWARD_PRESSURE "false"
35
36      `define IN_REAL_FILE "fft_ii_0_example_design_real_input.txt" // Real component of input data, formatted as integers (fixed po
37      `define IN_IMAG_FILE "fft_ii_0_example_design_imag_input.txt" // Imaginary component of input data, formatted as integers (fix
38      `define OUT_REAL_FILE "fft_ii_0_example_design_real_output.txt" // Real component of output data, formatted as integers (fixed
39      `define OUT_IMAG_FILE "fft_ii_0_example_design_imag_output.txt" // Imaginary component of output data, formatted as integers
40      `define OUT_EXP_FILE "fft_ii_0_example_design_exponent_output.txt" // Exponent component of output data, formatted as integers
41      `define OUT_LATENCY_FILE "fft_ii_0_example_design_latency_report.txt" // Exponent component of output data, formatted as integ
42      `define IN_BLK_FILE "fft_ii_0_example_design_blksize_report.txt" // List of block sizes used in variable sized ffts, 1 per lin
43      `define IN_INV_FILE "fft_ii_0_example_design_inverse_report.txt"// List of directions for FFT in bi-directional mode, 1 per li
44      `define DATA_REPRESENTATION "Fixed Point" //Fixed or Float or Block Float
45      `define B_IN 18 //32 for SINGLE precision Float
46      `define B_OUT 29 //32 for SINGLE precision Float
47      `define DIRECTION "Bi-directional" // Reverse or Forward or Bi-directional
48      `define DATA_FLOW "Variable Streaming" // Buffered Burst or Burst or Streaming or Variable Streaming
49      `define FFT_LENGTH 1024 //Max for variable size
50
51      `timescale 1ns/1ps
52
53
54
55
56      module test_program();
57
58      //instantiate the testbench system
59      fft_ii_0_example_design_tb tb();
60
61      localparam FFT_REP_WIDTH = $clog2(`FFT_LENGTH+1);
62
63      //BFM related parameters
64      localparam SRC_D_W    = 2*`B_IN;
65      localparam SRC_L_W    = (`DATA_FLOW == "Variable Streaming") ? FFT_REP_WIDTH : 1; //set to 1 when not used to avoid -1:0 range
66      localparam SRC_INV_W = (`DIRECTION == "Bi-directional") ? 1 : 0;
```

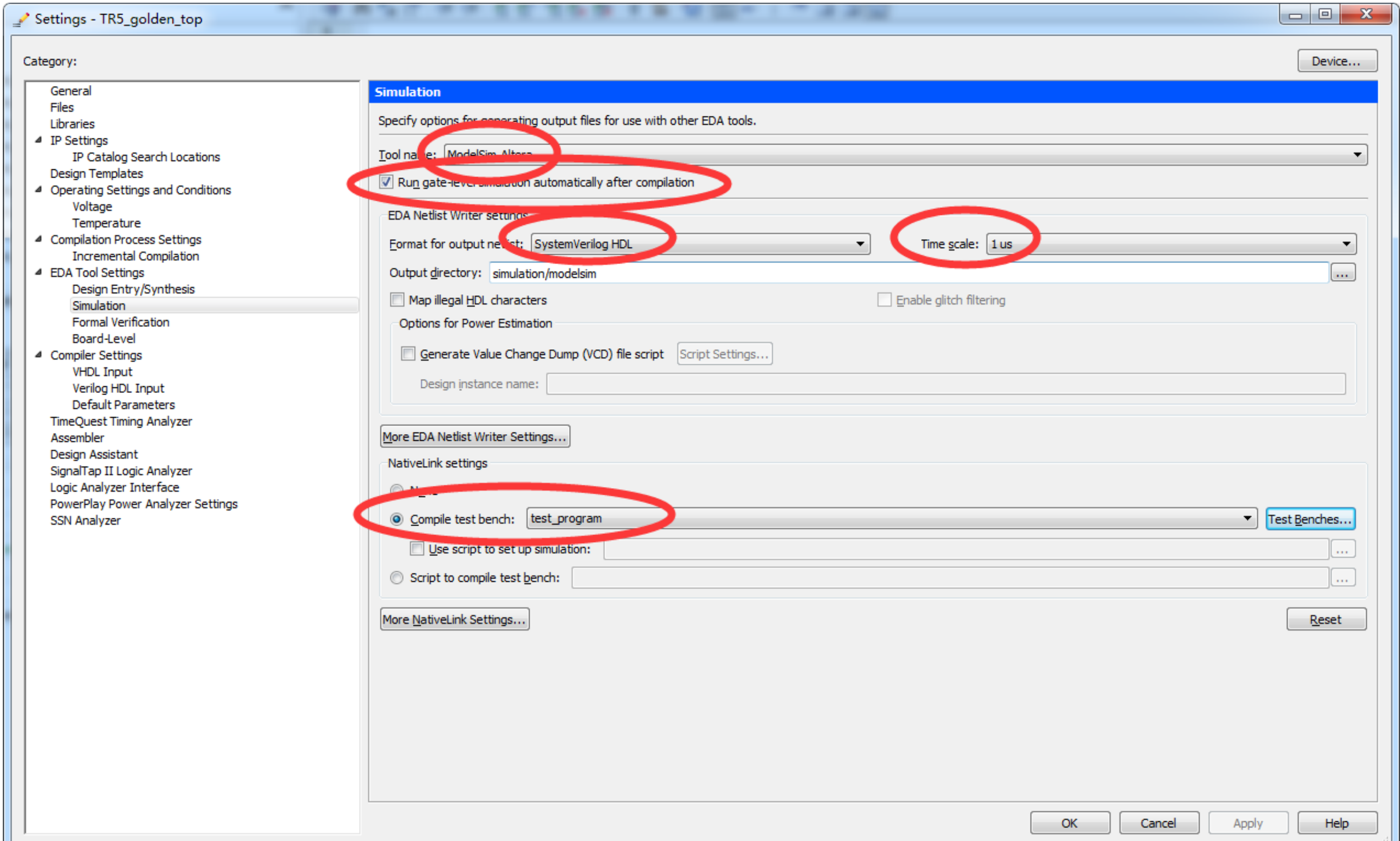
仿真包含自定义lib

仿真输入输出文件

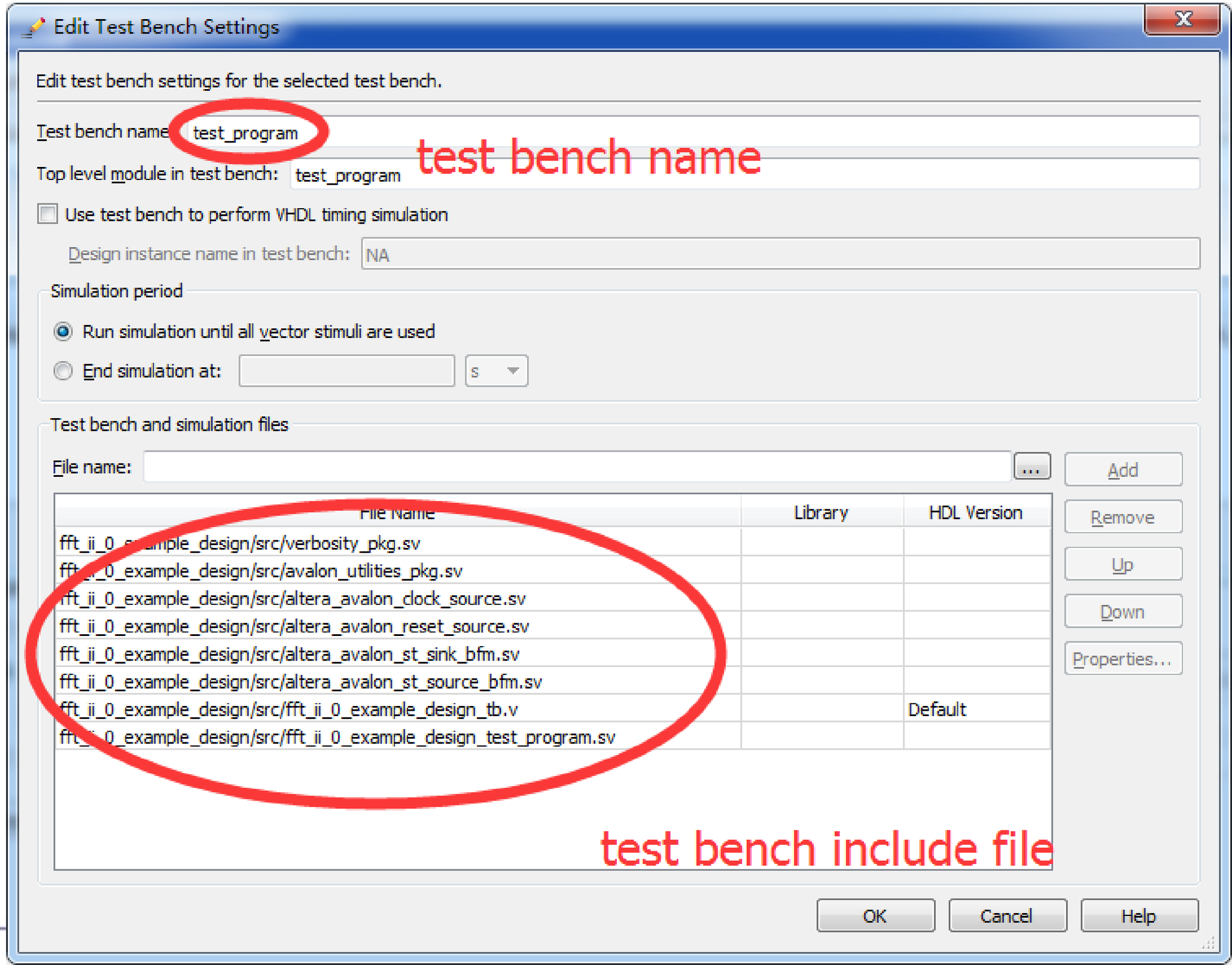
仿真时间精度

调用test bench，并运行DUT

FFT IP （8）

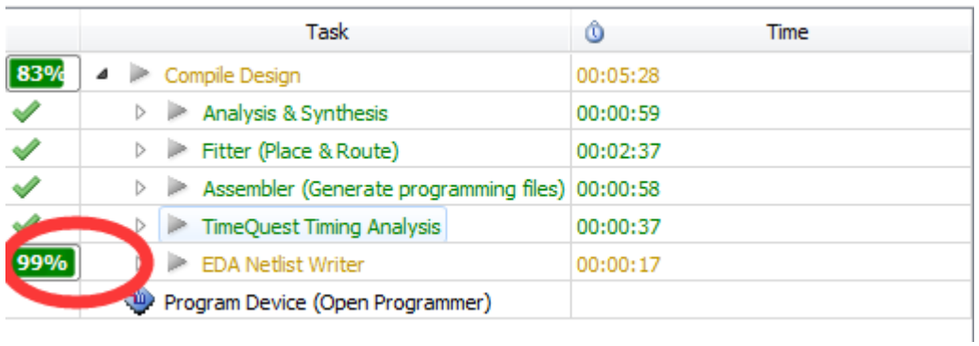

Settings - TR5_golden_top
Category:
Device...
General
Files
Libraries
IP Settings
IP Catalog Search Locations
Design Templates
Operating Settings and Conditions
Voltage
Temperature
Compilation Process Settings
Incremental Compilation
EDA Tool Settings
Design Entry/Synthesis
Simulation
Formal Verification
Board-Level
Compiler Settings
VHDL Input
Verilog HDL Input
Default Parameters
TimeQuest Timing Analyzer
Assembler
Design Assistant
SignalTap II Logic Analyzer
Logic Analyzer Interface
PowerPlay Power Analyzer Settings
SSN Analyzer
Simulation
Specify options for generating output files for use with other EDA tools.
Tool name: ModelSim-Altera
Run gate-level simulation automatically after compilation
EDA Netlist Writer settings
Format for output netlist: SystemVerilog HDL
Time scale: 1 us
Output directory: simulation/modelsim
Map illegal HDL characters
Enable glitch filtering
Options for Power Estimation
Generate Value Change Dump (VCD) file script
Script Settings...
Design instance name:
More EDA Netlist Writer Settings...
NativeLink settings
Compile test bench: test_program
Test Benches...
Use script to set up simulation:
Script to compile test bench:
More NativeLink Settings...
Reset
OK
Cancel
Apply
Help

FFT IP （9）

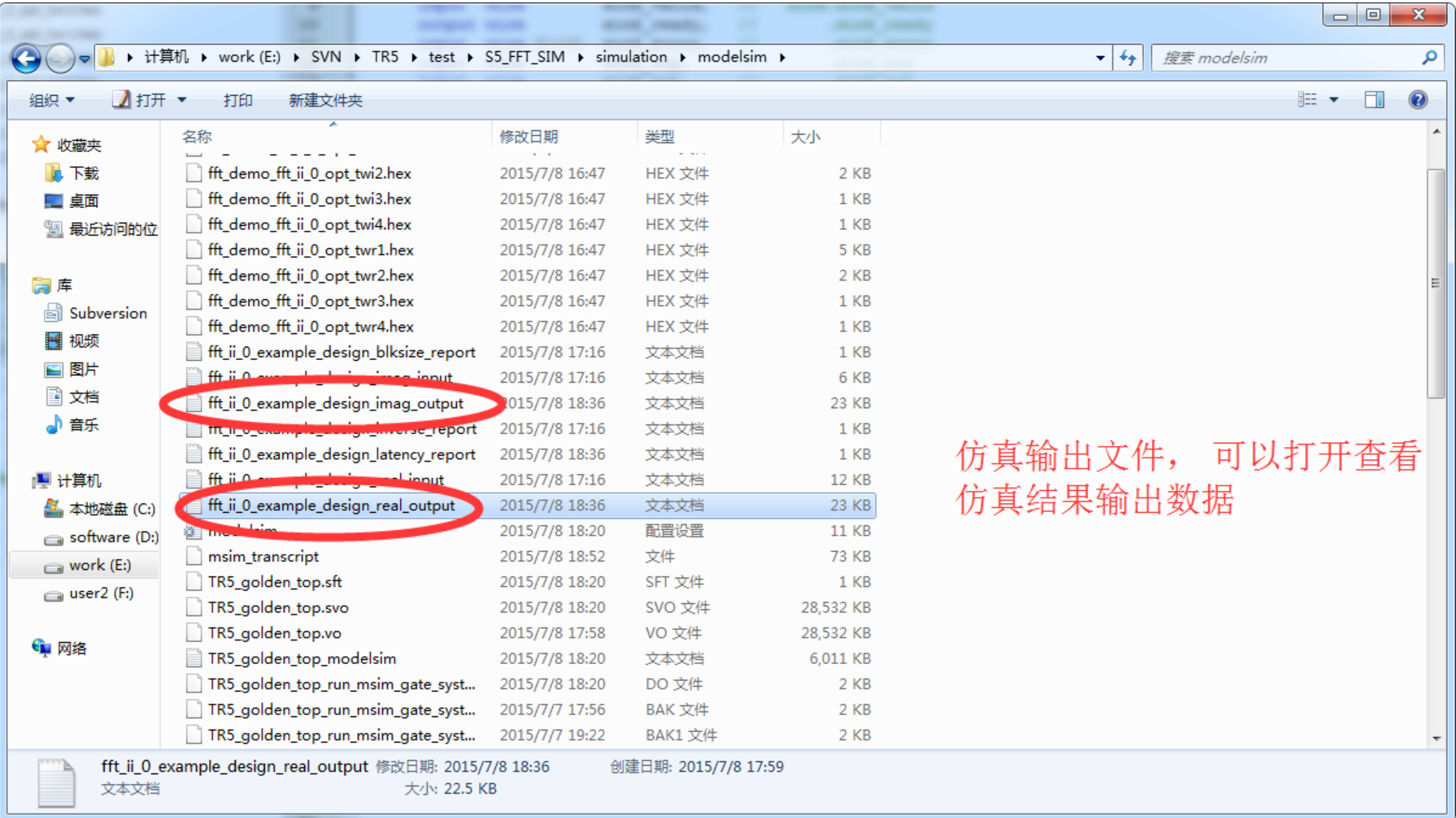

Edit Test Bench Settings
Edit test bench settings for the selected test bench.
Test bench name: test_program
Top level module in test bench: test_program
test bench name
Use test bench to perform VHDL timing simulation
Design instance name in test bench: NA
Simulation period
Run simulation until all vector stimuli are used
End simulation at:
s
Test bench and simulation files
File name:
...
Add
Remove
Up
Down
Properties...
File Name
Library
HDL Version
fft_ii_0_example_design/src/verbosity_pkg.sv
fft_ii_0_example_design/src/avalon_utilities_pkg.sv
fft_ii_0_example_design/src/altera_avalon_clock_source.sv
fft_ii_0_example_design/src/altera_avalon_reset_source.sv
fft_ii_0_example_design/src/altera_avalon_st_sink_bfm.sv
fft_ii_0_example_design/src/altera_avalon_st_source_bfm.sv
fft_ii_0_example_design/src/fft_ii_0_example_design_tb.v
Default
fft_ii_0_example_design/src/fft_ii_0_example_design_test_program.sv
test bench include file
OK
Cancel
Help

FFT IP（10）

- 完成设置后开始综合编译，在编译最后一步EDA netlist writer将要完成时，自动启动modelsim，并完成仿真，完成仿真后返回quartusII 并结束综合编译.

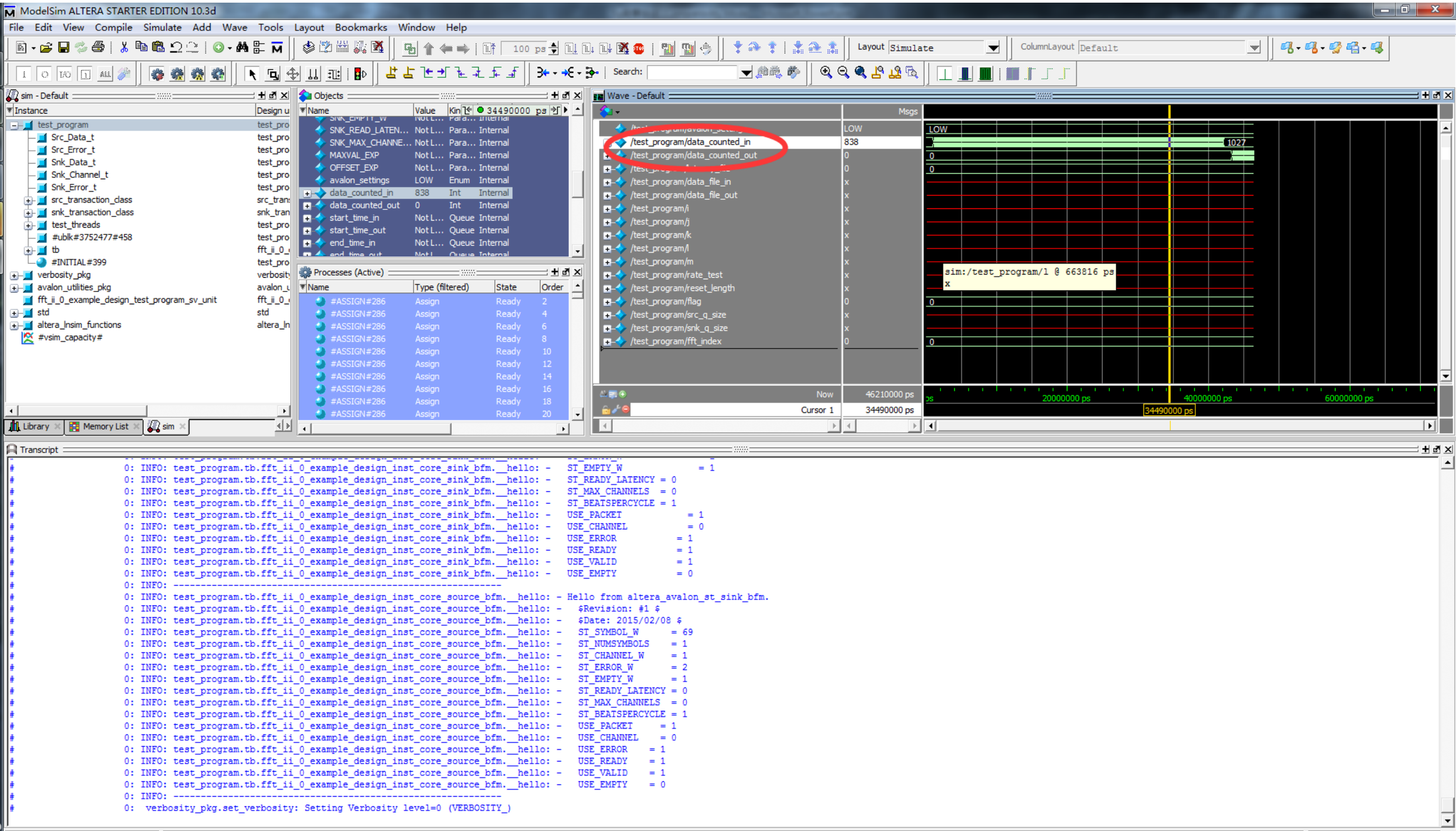


FFT IP （11）

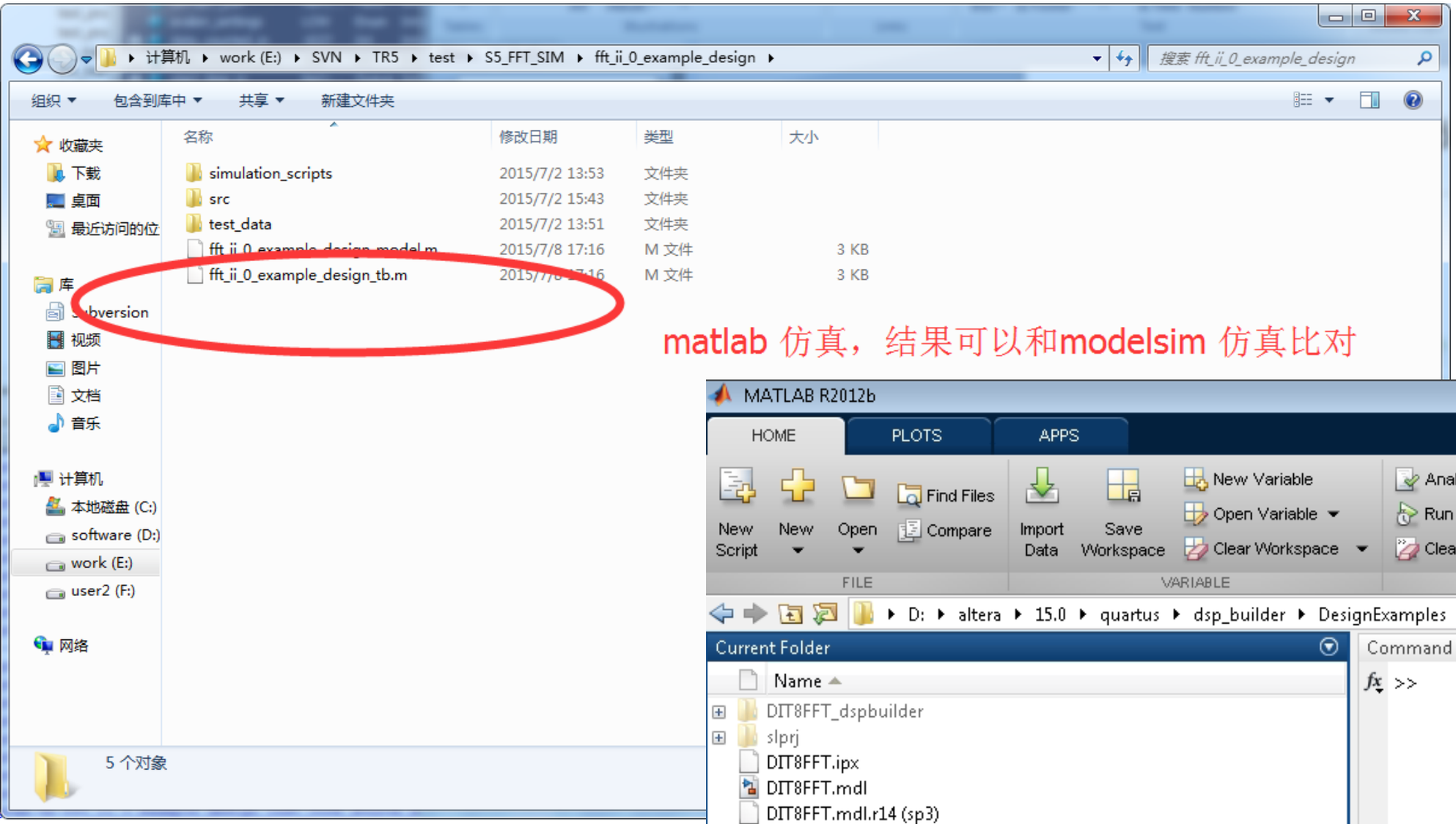


仿真输出文件， 可以打开查看
仿真结果输出数据

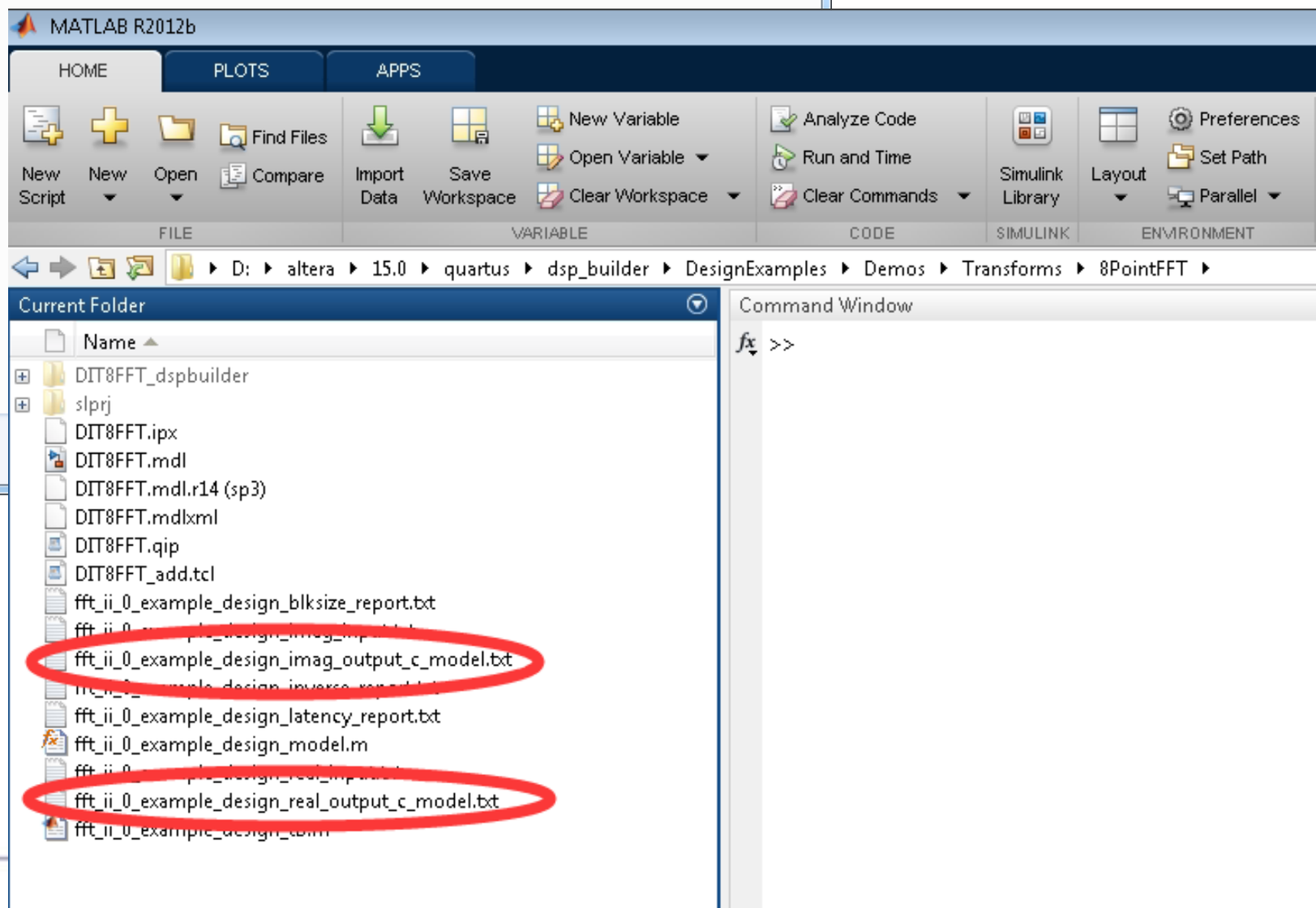
FFT IP （12）


ModelSim ALTERA STARTER EDITION 10.3d
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
sim - Default
Instance
Design u
test_program
Src_Data_t
Src_Error_t
Snk_Data_t
Snk_Channel_t
Snk_Error_t
src_transaction_class
snk_transaction_class
test_threads
#ublk#3752477#458
tb
#INITIAL#399
verbosity_pkg
avalon_utilities_pkg
fft_ii_0_example_design_test_program_sv_unit
std
altera_lnsim_functions
#vsim_capacity#
Library
Memory List
sim
Objects
Name
Value
Kin
34490000 ps
SNK_READ_LATEN...
SNK_MAX_CHANNE...
MAXVAL_EXP
OFFSET_EXP
avalon_settings
data_counted_in
data_counted_out
start_time_in
start_time_out
end_time_in
Processes (Active)
Type (filtered)
State
Order
#ASSIGN#286
Assign
Ready
Wave - Default
Msgs
/test_program/data_counted_in
/test_program/data_counted_out
/test_program/data_file_in
/test_program/data_file_out
/test_program/i
/test_program/j
/test_program/k
/test_program/l
/test_program/m
/test_program/rate_test
/test_program/reset_length
/test_program/flag
/test_program/src_q_size
/test_program/snk_q_size
/test_program/fft_index
sim:/test_program/l @ 663816 ps
Now
46210000 ps
Cursor 1
34490000 ps
20000000 ps
40000000 ps
60000000 ps
Transcript
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - ST_EMPTY_W = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - ST_READY_LATENCY = 0
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - ST_MAX_CHANNELS = 0
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - ST_BEATSPERCYCLE = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - USE_PACKET = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - USE_CHANNEL = 0
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - USE_ERROR = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - USE_READY = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - USE_VALID = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_sink_bfm.__hello: - USE_EMPTY = 0
0: INFO: ---
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - Hello from altera_avalon_st_sink_bfm.
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - $Revision: #1 $
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - $Date: 2015/02/08 $
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_SYMBOL_W = 69
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_NUMSYMBOLS = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_CHANNEL_W = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_ERROR_W = 2
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_EMPTY_W = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_READY_LATENCY = 0
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_MAX_CHANNELS = 0
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - ST_BEATSPERCYCLE = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - USE_PACKET = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - USE_CHANNEL = 0
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - USE_ERROR = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - USE_READY = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - USE_VALID = 1
0: INFO: test_program.tb.fft_ii_0_example_design_inst_core_source_bfm.__hello: - USE_EMPTY = 0
0: INFO: ---
0: verbosity_pkg.set_verbosity: Setting Verbosity level=0 (VERBOSITY_)
Now: 78,330 ns Delta: 8
sim:/test_program
0 ps to 72024064 ps

FFT IP （13）

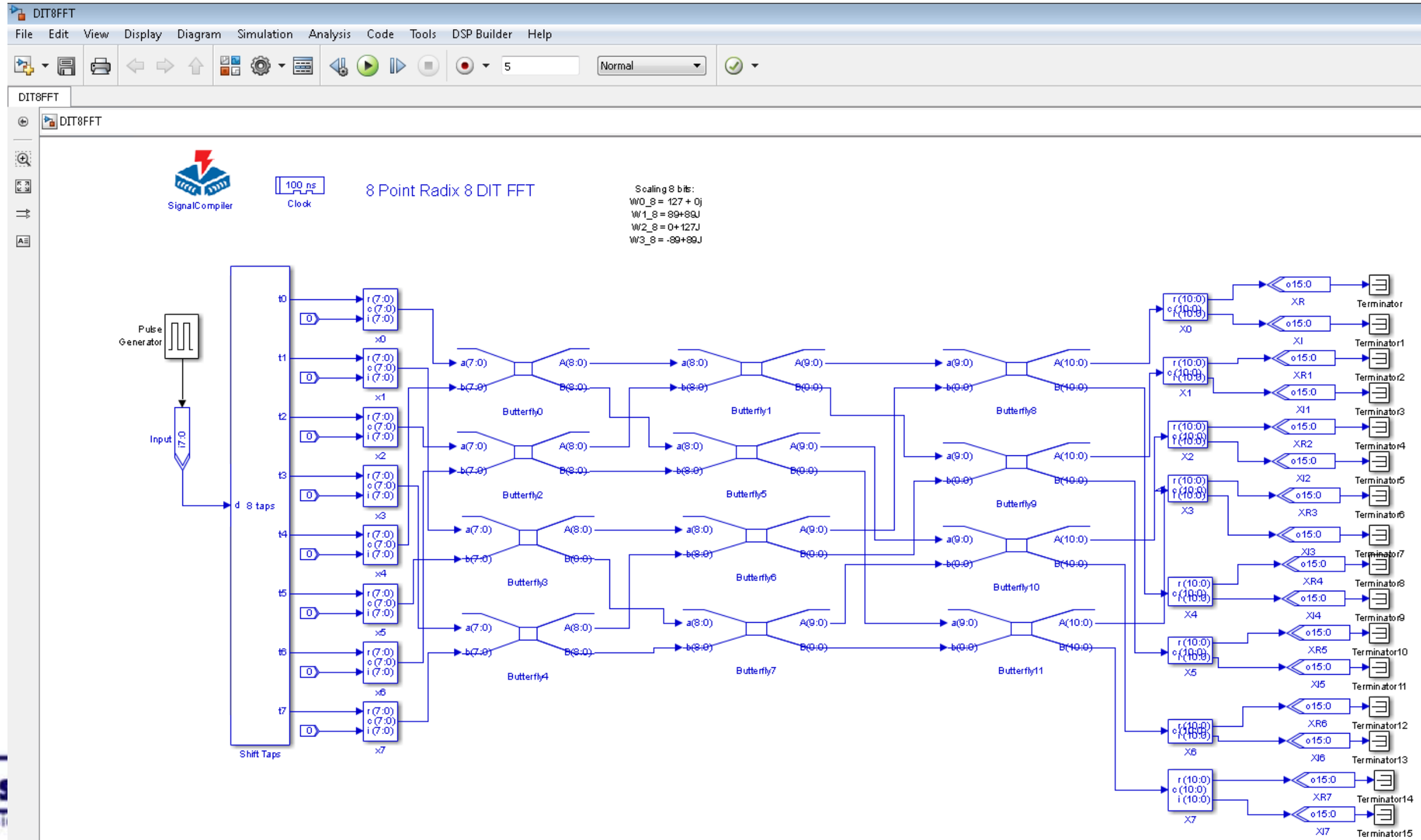


matlab 仿真，结果可以和modelsim 仿真比对



DSP builder

- 8点 基8 DIT FFT



FFT OpenCL

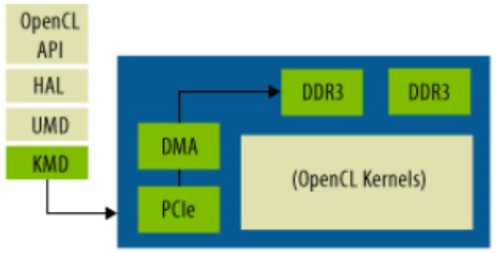
- FFT examples for OpenCL
- https://www.altera.com/products/design-software/embedded-software-developers/opencl/developer-zone.html#designexamples

FFT (1D)	• Single-precision floating-point optimizations • Single work-item kernel	• Performance	This design example demonstrates a high-performance 1D radix-4 complex fast Fourier transform (FFT) or inverse fast Fourier transform (IFFT) engine using OpenCL. This example takes advantage of the efficient sliding window data reuse pattern.
FFT Off-Chip (1D)	• Single-precision floating-point optimizations • Kernel channels • Optimized memory accesses	• Performance • Getting started with kernel channels	This design example is a high-performance implementation of a one million point FFT. Such large FFTs cannot be done completely on the FPGA and this example demonstrates how to efficiently manage the memory accesses.
FFT (2D)	• Single-precision floating-point optimizations • Kernel channels • Memory access pattern optimizations • Multiple simultaneous kernels • Mix of single work-item and NDRange kernels	• Performance • Getting started with kernel channels	This design example demonstrates a high-performance 2D radix-4 complex FFT/IFFT engine using OpenCL. This engine is targeted at large problem sizes (1024x1024 by default) and uses global memory to store the intermediate transposition. One aspect highlighted by this example is how to efficiently perform matrix transposition in global memory.

DE5-Net: Altera OpenCL 开发优选平台

HPC

The traditional OpenCL model has a host that passes data to the accelerator system over PCI Express® (PCIe). For the High-Performance Computing (HPC) platform, the system requires a large amount of local bulk storage for processing the data that the host sends to the accelerator. These applications require large amounts of memory bandwidth and are systems where computing power is of most importance. This platform is the standard platform for OpenCL accelerators.



To get started evaluating the standard HPC platform architecture, you can:

- Download a reference design that runs on the HPC platform and learn the OpenCL application development flow
- Download an HPC platform by one of our Altera Preferred Board vendor boards
- Purchase a commercial off-the-shelf (COTS) board that supports the HPC platform from one of our Altera Preferred Board vendors by clicking their logo below

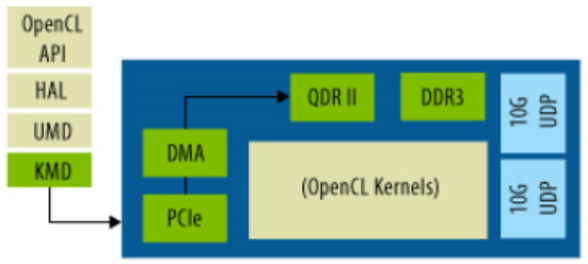
BittWare　REFLEX CES Custom Embedded Systems　pico computing

Nallatech　terasIC www.terasic.com　GIDEL

To learn how to use the reference platforms to create your own custom board, refer to the "Custom" tab below.

Network

The Network platform deviates from the traditional OpenCL model by extracting the datapath from the PCIe command and status path. Data is now streamed into the kernels using I/O channels, without host interaction over two 10 Gb user datagram protocol (UDP) ports. This streaming architecture allows the host to configure the datapath pipeline and then step out of the picture for a much lower latency data processing path that traditional FPGA developers are used to. Applications using this platform are much more concerned with achieving a lower latency result.



To get started evaluating the low-latency Network platform architecture, you can:

- Download a reference design that runs on the Network platform and learn the OpenCL application development flow
- Download a Network platform by one of our Altera Preferred Board vendors boards
- Purchase a COTS board that supports the Network platform from one of our Altera Preferred Board vendors by clicking their logo below

BittWare FPGA COMPUTING SYSTEMS　Nallatech　terasIC www.terasic.com

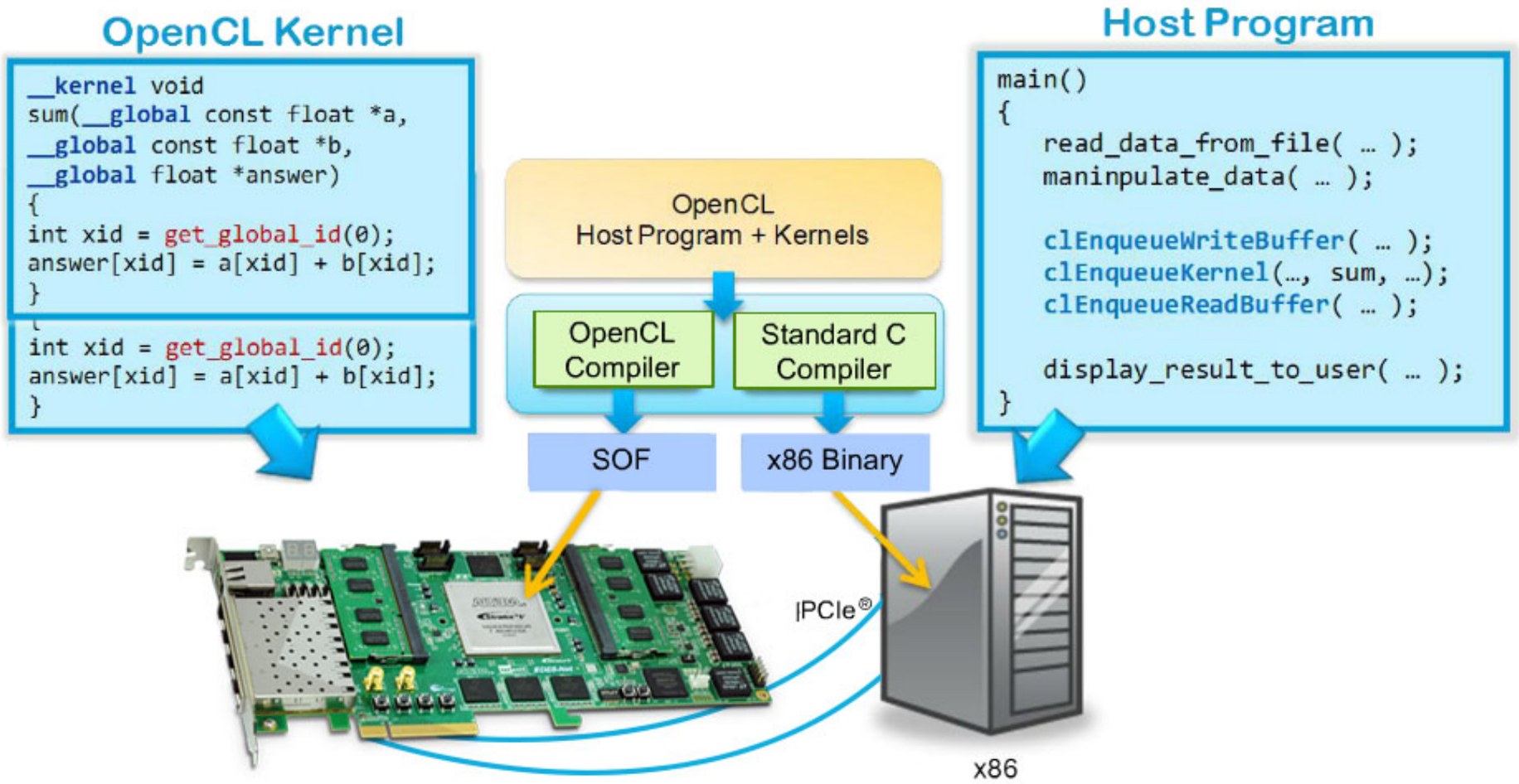
DE5-Net上实现OpenCL

系统平台实现要求：

- **主机PC**
  - 运行系统: 64位 Windows 7 或 Linux (RHEL or CentOS)系统
  - 带12V供电8/16位的PCIE卡槽）
  - 推荐32GB内存（最小24GB）
- **Quartus® II 软件(获得许可证)**
  - 64 位分期预约或开发版
  - 安装Stratix® V 器件
  - 安装相同版本的 Altera OpenCL SDK (获得许可证)
- **C 编译器**
  - Microsoft® Visual Studio编译器或者GCC编译器
  - 能编译主机程序
  - 能编译和链接64位代码
- **FPGA 开发板**
  - 安装2片2GB DDR3-SODIMM的 DE5-NET 开发板（板子标配）

DE5-Net OpenCL 架构

- **OpenCL 内核** - 实现加速单元
- **OpenCL 主机程序**–通过API函数实现主机与内核通信的纯软件程式



实现方法

下载**DE5-Net OpenCL Bsp** 包以及**manual**手册

- 官网下载**BSP**包和**manual**手册
  - *http://cd-de5-net.terasic.com*

参考**manual**描述操作

- 安装**BSP**包到宿主机（**windows**或**linux**平台）
- 设置环境变量
- 安装 **PCIE**驱动
- 验证开发平台的正确性
- 安装开发板
- 编译**OpenCL**内核
- 编译主机程序
- 测试工程

FFT OpenCL

OUTPUT

```
Launching FFT transform (ordered data layout)
Kernel initialization is complete.
        Processing time = 3.4020ms
        Throughput = 0.3082 Gpoints / sec (30.8224 Gflops)
        Signal to noise ratio on output sample: 124.236435 --> PASSED

Launching inverse FFT transform (ordered data layout)
Kernel initialization is complete.
        Processing time = 3.3848ms
        Throughput = 0.3098 Gpoints / sec (30.9787 Gflops)
        Signal to noise ratio on output sample: 124.227971 --> PASSED

Launching FFT transform (alternative data layout)
Kernel initialization is complete.
        Processing time = 2.1122ms
        Throughput = 0.4964 Gpoints / sec (49.6450 Gflops)
        Signal to noise ratio on output sample: 124.236876 --> PASSED

Launching inverse FFT transform (alternative data layout)
Kernel initialization is complete.
        Processing time = 2.1369ms
        Throughput = 0.4907 Gpoints / sec (49.0689 Gflops)
        Signal to noise ratio on output sample: 124.231472 --> PASSED
```

Technical Support

•Tel ：027-87745390

•Add：China/406, Jingfeng Bld. B, Intl Business Center,
Special No. 1, Guang Gu Rd., Wuhan, China, 430074

•Email：sales@terasic.com.cn / support@terasic.com.cn

Thank you !