Автоматизация проектирования дискретных устройств. Экзамен. Дата 27.05.2024
ФИО Симоновский Даниил Леонидович группа 5130901/10101

Вопрос 1.

С использованием типов данных и конструкций расширения System Verilog создайте описание устройства

Алгоритм работы:

- устройство принимает **потоковые** данные с трех 8 разрядных входов A, B, C;
- осуществляет: выполнение функции R= A * B +C
- формирует выходные данные R

Выводы устройства (имена выводов модуля могут быть выбраны любыми, рекомендуется выбрать имена, облегчающие интеграцию с Platform Designer (PD)):

- На входах должен быть использованы регистры.
    - Входы должны быть ориентированы на использование Stream интерфейсов в Platform Designer (PD) с поддержкой сигналов Ready и Valid (по входному сигналу Valid =1 осуществляется запись во входной регистр); Сигнал Ready постоянно равен 1
- На выходе должен быть использован регистр
    - Выход должен быть ориентирован на использование Stream интерфейса в Platform Designer (PD). Сигнал Ready не анализируем. Сигнал Valid постоянно равен 1.
- Вход: тактового сигнала – clk.
- Вход: сигнала асинхронного сброса – arst.

Verilog_labs - my_module.sv

```
module my_module (
  // clk, rst
  input  bit        csi_clk,         //  clock.clk
  input  bit        rsi_reset,       //  reset.reset
  // in_A ST
  input  bit [ 7:0] asi_in0_data,    //  asi_in0.data
  output bit        asi_in0_ready,   //         .ready
  input  bit        asi_in0_valid,   //         .valid
  // in_B ST
  input  bit [ 7:0] asi_in1_data,    //  asi_in1.data
  output bit        asi_in1_ready,   //         .ready
  input  bit        asi_in1_valid,   //         .valid
  // in_C ST
  input  bit [ 7:0] asi_in2_data,    //  asi_in2.data
  output bit        asi_in2_ready,   //         .ready
  input  bit        asi_in2_valid,   //         .valid
  // out_R ST
  output bit [15:0] aso_out0_data,   //  asi_in2.data
  input  bit        aso_out0_ready,  //         .ready
  output bit        aso_out0_valid   //         .valid
);

  assign asi_in0_ready  = 1;
  assign asi_in1_ready  = 1;
  assign asi_in2_ready  = 1;
  assign aso_out0_valid = 1;

  bit [7:0] A, B, C;

  always_ff @(posedge csi_clk, posedge rsi_reset) begin
    if (rsi_reset) begin
      A             <= 0;
      B             <= 0;
      C             <= 0;
      aso_out0_data <= 0;
    end else begin
      if (asi_in0_valid) A <= asi_in0_data;
      if (asi_in1_valid) B <= asi_in1_data;
      if (asi_in2_valid) C <= asi_in2_data;
      aso_out0_data <= A * B + C;
    end
  end

endmodule
```

На этом месте в файле с ответами приведите созданное текстовое описание.

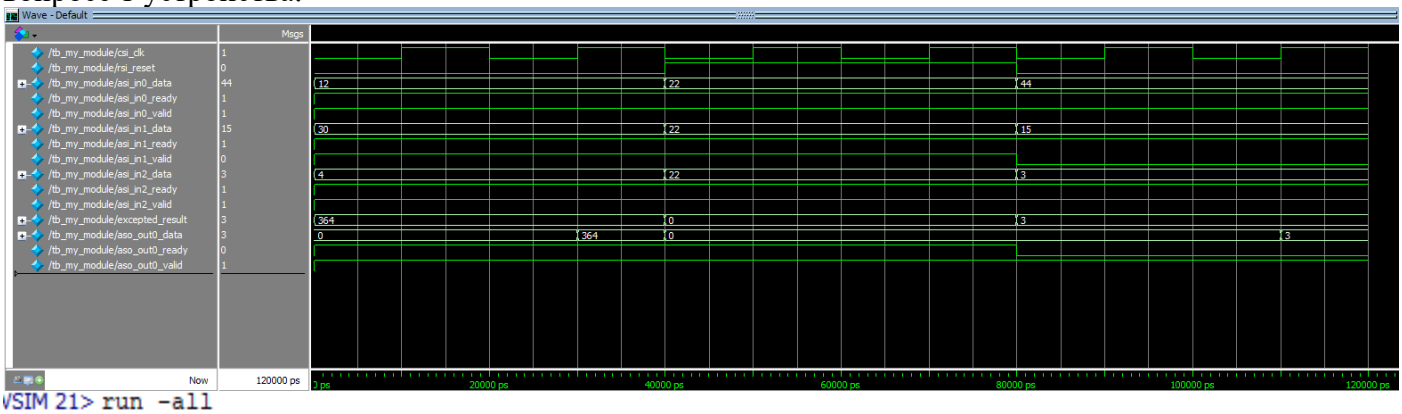
Вопрос 2.
С использованием типов данных и конструкций расширения System Verilog для устройства, созданного в вопросе 1, разработайте тест класса 2 (с самопроверкой).
Исходные данные (для A, B, C) и ожидаемые данные (для R) должны считываться из файлов (достаточно 3-х наборов данных).
Тест должен обеспечивать проверку всех режимов работы устройства (включая сброс).

Verilog_labs - tb_my_module.sv

```
`timescale 1 ns / 1 ns
module tb_my_module ();
  localparam PERIOD = 20;
  // clk, rst
  bit        csi_clk;  //  clock.clk
  bit        rsi_reset;  //  reset.reset
  // in_A ST
  bit [ 7:0] asi_in0_data;  //  asi_in0.data
  bit        asi_in0_ready;  //         .ready
  bit        asi_in0_valid;  //         .valid
  // in_B ST
  bit [ 7:0] asi_in1_data;  //  asi_in1.data
  bit        asi_in1_ready;  //         .ready
  bit        asi_in1_valid;  //         .valid
  // in_C ST
  bit [ 7:0] asi_in2_data;  //  asi_in2.data
  bit        asi_in2_ready;  //         .ready
  bit        asi_in2_valid;  //         .valid
  // out_R ST
  bit [15:0] aso_out0_data;  //  asi_in2.data
  bit        aso_out0_ready;  //         .ready
  bit        aso_out0_valid;  //         .valid

  my_module u0 (.*);

  bit [16:0] excepted_result;
  int input_file, answers_file, char_1, char_2;

  initial begin
    forever #(PERIOD / 2) csi_clk = ~csi_clk;
  end

  initial begin
    `define eof 32'hffff_ffff
    input_file   = $fopen("input_file.txt", "r");
    answers_file = $fopen("except_output.txt", "r");
    char_1       = $fgetc(input_file);
    char_2       = $fgetc(answers_file);
    while (char_1 != `eof && char_2 != `eof) begin
      $ungetc(char_1, input_file);
      $ungetc(char_2, answers_file);
      $fscanf(input_file, "%d", asi_in0_data);
      $fscanf(input_file, "%d", asi_in0_valid);
      $fscanf(input_file, "%d", asi_in1_data);
      $fscanf(input_file, "%d", asi_in1_valid);
      $fscanf(input_file, "%d", asi_in2_data);
      $fscanf(input_file, "%d", asi_in2_valid);
      $fscanf(input_file, "%d", aso_out0_ready);
      $fscanf(input_file, "%d", rsi_reset);
      $fscanf(answers_file, "%d", excepted_result);

      #(PERIOD * 2);
      if (
        excepted_result != aso_out0_data &&
        asi_in0_ready == 1 &&
        asi_in1_ready == 1 &&
        asi_in2_ready == 1
      ) begin
        $display("Incorrect output:\nExcepted: %b\nActual: %b", excepted_result, aso_out0_data);
        $display("r1: %b\nr2: %b\nr3: %b", asi_in0_ready, asi_in1_ready, asi_in2_ready);
        $stop;
      end
      char_1 = $fgetc(input_file);
      char_2 = $fgetc(answers_file);
    end
    $display("All test have been passed!");
    $fclose(input_file);
    $fclose(answers_file);
    $stop;
  end

endmodule

```

На этом месте в файле с ответами приведите созданное текстовое описание теста.
Вопрос 3.

В пакете ModelSim, используя созданный в вопросе 2 тест, проведите моделирование созданного в вопросе 1 устройства.

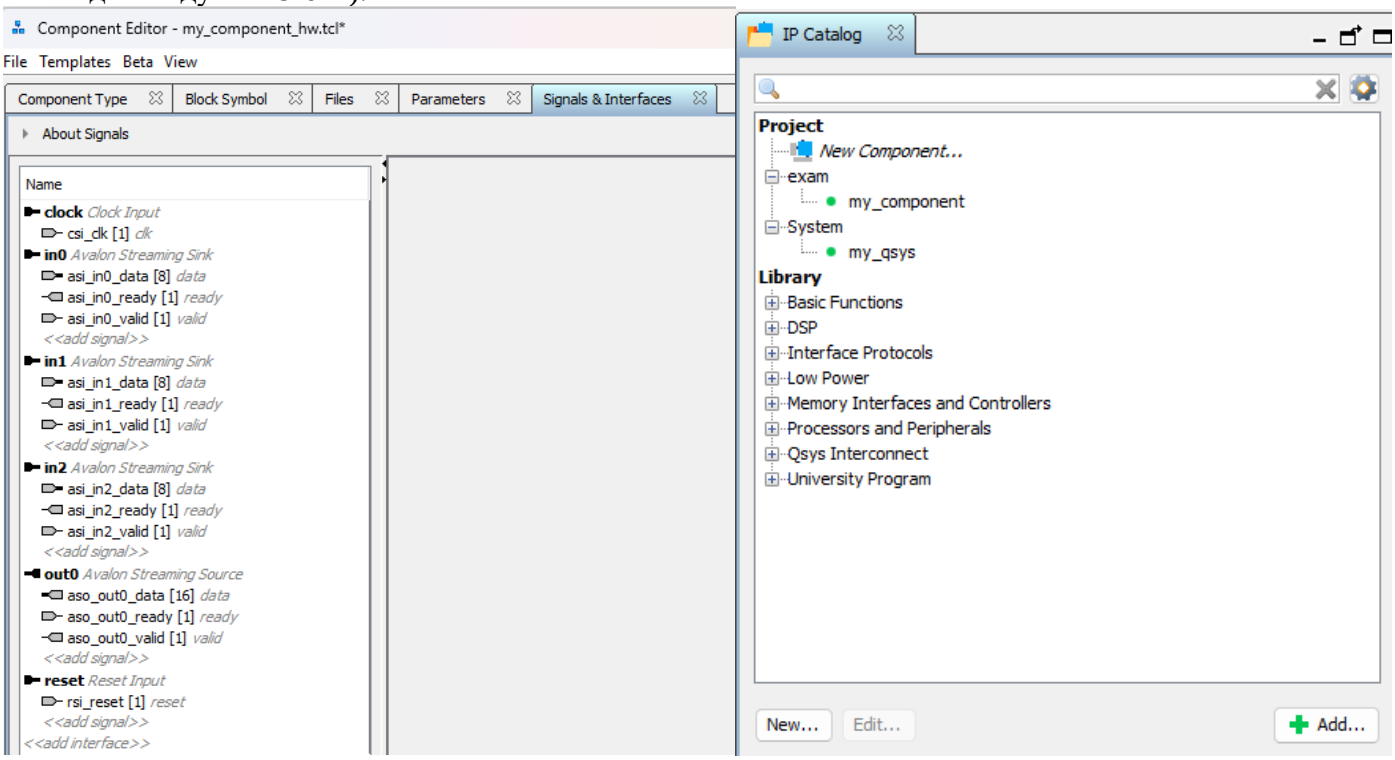



```
VSIM 21> run -all
# All test have been passed!
# ** Note: $stop    : tb_my_module.sv(69)
#    Time: 120 ns  Iteration: 0  Instance: /tb_my_module
# Break in Module tb_my_module at tb_my_module.sv line 69
```

На этом месте в файле с ответами приведите временные диаграммы результатов моделирования и результаты, представленные в консоли (т.к. тест с самопроверкой).

Вопрос 4.

Интегрируйте устройство, созданное в вопросе 1, как библиотечный компонент в PD (библиотечная папка для модуля – **exam**).

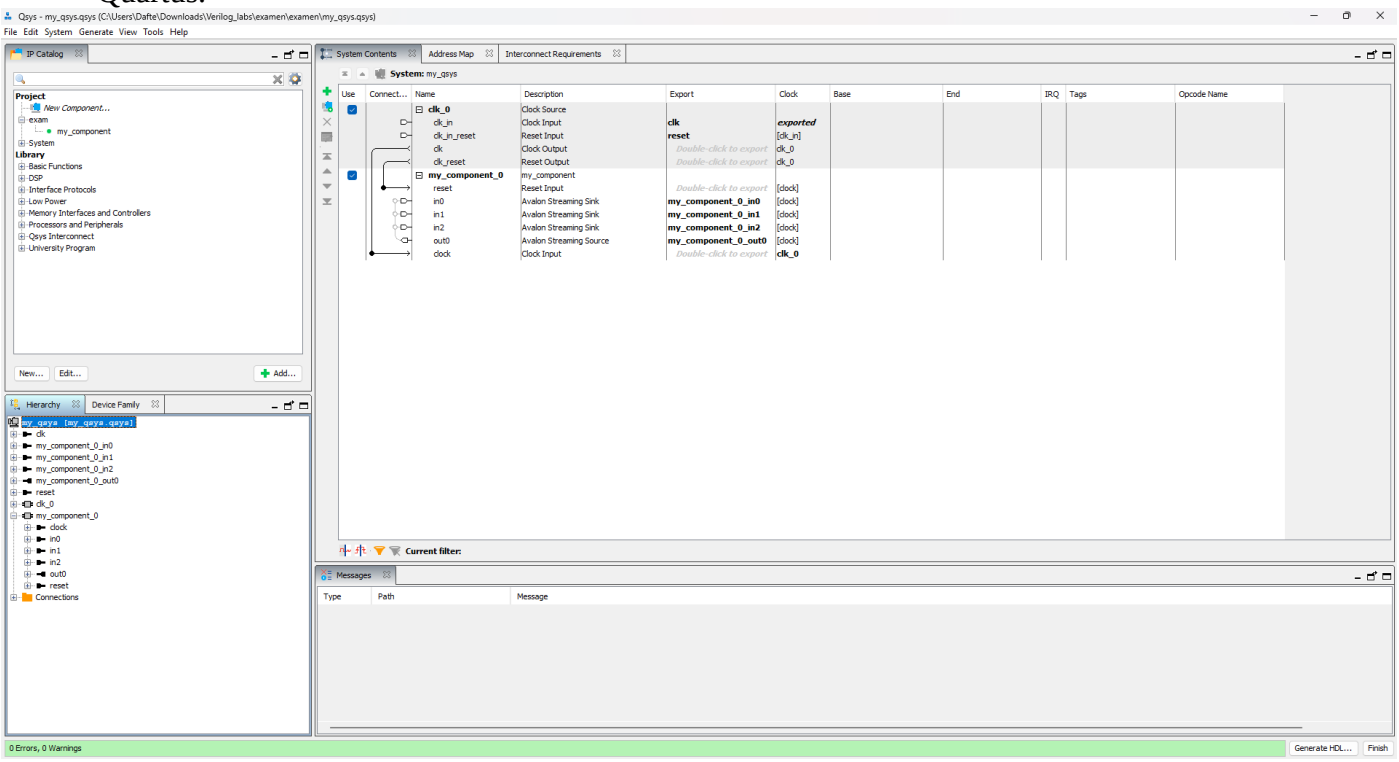


На этом месте в файле с ответами приведите снимки экрана:

- ***с библиотекой PD в которой есть папка <u>exam</u> с созданным компонентом***
- ***настройки интерфейсной части компонента.***

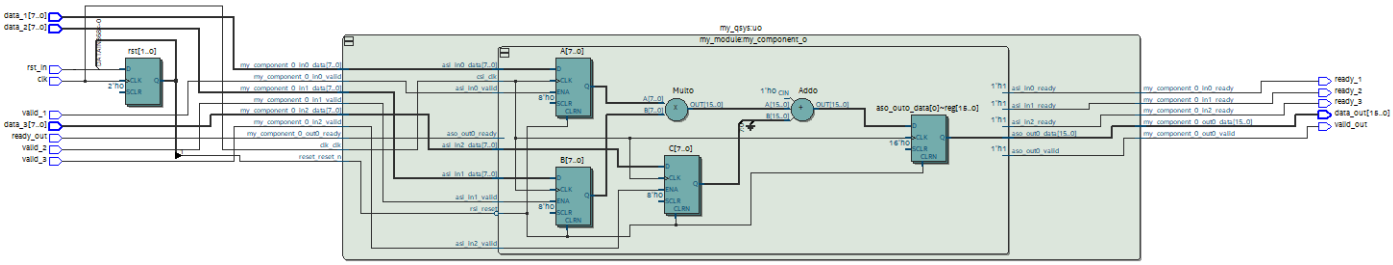
Вопрос 5.

- В PD создайте описание системы, включающей модуль тактового сигнала и компонент, созданный в вопросе 4.
- Экспортируйте выводы данных.
- Создайте HDL описание в приложении PD.
- С использованием типов данных и конструкций расширения System Verilog создайте описание верхнего уровня, в котором созданная система используется как компонент.
- **Вход сброса, в файле верхнего уровня, должен быть подключен через два триггера**.
- Осуществите компиляцию и получите структуру системы, используя RTL Viewer в пакете Quartus.



Verilog_labs - exam.sv

```systemverilog
module exam (
  input  bit        clk,
  input  bit        rst_in,
  input  bit [ 7:0] data_1,
  input  bit [ 7:0] data_2,
  input  bit [ 7:0] data_3,
  output bit [15:0] data_out,
  input  bit        valid_1,
  input  bit        valid_2,
  input  bit        valid_3,
  output bit        valid_out,
  output bit        ready_1,
  output bit        ready_2,
  output bit        ready_3,
  input  bit        ready_out
);

  bit [1:0] rst = '0;

  always_ff @(posedge clk) begin
    rst[1:0] <= {rst[0], rst_in};
  end

  my_qsys u0 (
    .clk_clk                 (clk),        //               clk.clk
    .reset_reset_n           (rst[1]),     //             reset.reset_n
    .my_component_0_in0_data  (data_1),     //  my_component_0_in0.data
    .my_component_0_in0_ready (ready_1),    //                    .ready
    .my_component_0_in0_valid (valid_1),    //                    .valid
    .my_component_0_in1_data  (data_2),     //  my_component_0_in1.data
    .my_component_0_in1_ready (ready_2),    //                    .ready
    .my_component_0_in1_valid (valid_2),    //                    .valid
    .my_component_0_in2_data  (data_3),     //  my_component_0_in2.data
    .my_component_0_in2_ready (ready_3),    //                    .ready
    .my_component_0_in2_valid (valid_3),    //                    .valid
    .my_component_0_out0_data (data_out),   // my_component_0_out0.data
    .my_component_0_out0_ready(ready_out),  //                    .ready
    .my_component_0_out0_valid(valid_out)   //                    .valid
  );

endmodule
```



На этом месте в файле с ответами приведите снимки экрана:

- ***структуры системы в PD***
- ***созданного описания верхнего уровня***
- ***структуры, полученной в RTL Viewer.***