Вопрос 1.

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

Алгоритм работы:

* устройство принимает **потоковые** данные с двух N-х разрядных входов A и B;
* осуществляет: выполнение функции R= А \* В
* формирует выходные данные R

Выводы устройства (имена выводов модуля м.б. выбраны любыми, рекомендуется выбрать имена, облегчающие интеграцию с Platform Designer (PD)):

* На входах д.б 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 д.б использован регистр
  + Выход должн быть ориентирован на использование Conduit интерфейса в Platform Designer (PD).
* Вход: тактового сигнала – clk.
* Вход: сигнала синхронного сброса – srst.

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

|  |
| --- |
| `timescale 1ns / 1ps  module exam #(  parameter N = 8)(  input bit csi\_clk, rsi\_reset\_n,    output bit asi\_pin\_a\_ready,  input bit asi\_pin\_a\_valid,  input bit [N-1:0] asi\_pin\_a\_data,  output bit asi\_pin\_b\_ready,  input bit asi\_pin\_b\_valid,  input bit [N-1:0] asi\_pin\_b\_data,    input bit coe\_pin\_r\_ready,  output bit [(N\*2)-1:0] coe\_pin\_r\_data  );    bit [N-1:0] r\_a, r\_b;  bit [(N\*2)-1:0] reg\_r;  assign asi\_pin\_a\_ready = 1;  assign asi\_pin\_b\_ready = 1;  always\_ff @(posedge csi\_clk) begin  if (!rsi\_reset\_n) begin;  if (asi\_pin\_a\_valid) r\_a <= asi\_pin\_a\_data;  if (asi\_pin\_b\_valid) r\_b <= asi\_pin\_b\_data;  if (coe\_pin\_r\_ready) coe\_pin\_r\_data <= reg\_r;  end else begin  r\_a <= #(N)'h0;  r\_b <= #(N)'h0;  coe\_pin\_r\_data <= #(N\*2)'h0;  end        end  assign reg\_r = r\_a \* r\_b;  endmodule |

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 и ожидаемые данные для проверки должны считываться из файлов. Типы команд считывания из файлов – любые.

Тест должен обеспечивать проверку всех режимов работы устройства (включая сброс).

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

|  |
| --- |
| `timescale 1ns / 1ps  module tb\_exam();    parameter N = 8;  bit csi\_clk = 1, rsi\_reset\_n;  bit asi\_pin\_a\_ready, asi\_pin\_b\_ready;  bit asi\_pin\_a\_valid, asi\_pin\_b\_valid;  bit [N-1:0] asi\_pin\_a\_data, asi\_pin\_b\_data;    bit coe\_pin\_r\_ready;  bit [(N\*2)-1:0] coe\_pin\_r\_data;  bit [(N\*2)-1:0] f\_r;  bit [N-1:0] datafile [19:0];  bit [(N\*2)-1:0] resultfile [4:0];  int i, j;  exam #(N) exam\_inst(.\*);  initial forever #10 csi\_clk = ~csi\_clk;  initial begin  $readmemh("./data.dat", datafile);  $readmemh("./result.dat", resultfile);  end  initial begin  #1;  j = 0;  for(i = 0; i < 20; i = i + 4) begin  coe\_pin\_r\_ready = 1'b1;    asi\_pin\_a\_valid = datafile[i];  asi\_pin\_b\_valid = datafile[i];    rsi\_reset\_n = datafile[i + 1];    asi\_pin\_a\_data = datafile[i + 2];  asi\_pin\_b\_data = datafile[i + 3];    f\_r = resultfile[j];  j = j + 1;  #100;  if(f\_r == coe\_pin\_r\_data)  $display ("GOOD! Time: %t\t write = %b reset = %b a = %b b = %b r = %b",  $realtime, asi\_pin\_a\_valid, rsi\_reset\_n, asi\_pin\_a\_data, asi\_pin\_b\_data, coe\_pin\_r\_data);  else  $display ("ERROR! Time: %t\t write = %b reset = %b a = %b b = %b r = %b| file r = %b ",  $realtime, asi\_pin\_a\_valid, rsi\_reset\_n, asi\_pin\_a\_data, asi\_pin\_b\_data,  coe\_pin\_r\_data, f\_r);    end  #60;  $stop;  end  endmodule |

Входные данные

|  |
| --- |
| 1 0 02 02  1 0 03 03  0 0 02 03  1 0 02 00  1 1 ff ff |

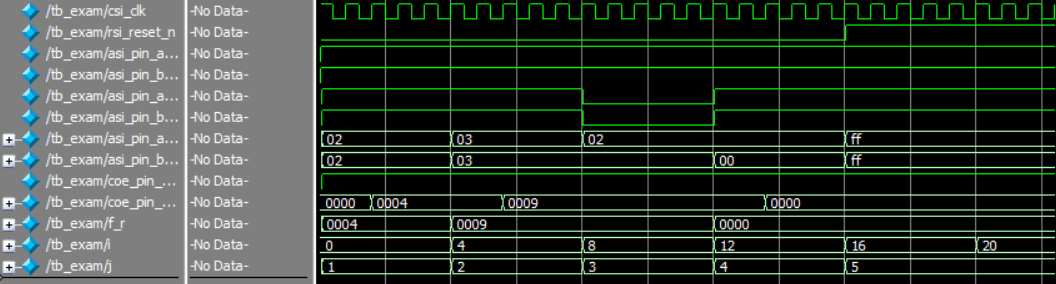
Выходные данные

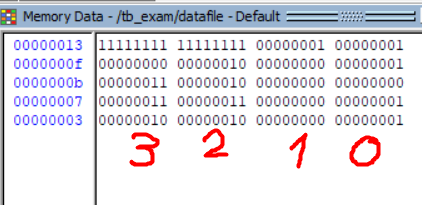
|  |
| --- |
| 0004  0009  0009  0000  0000 |

Вопрос 3.

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

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



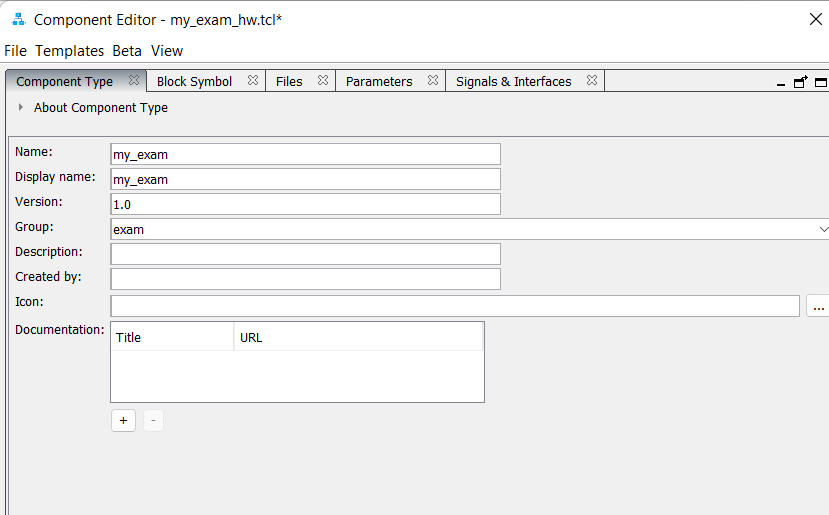


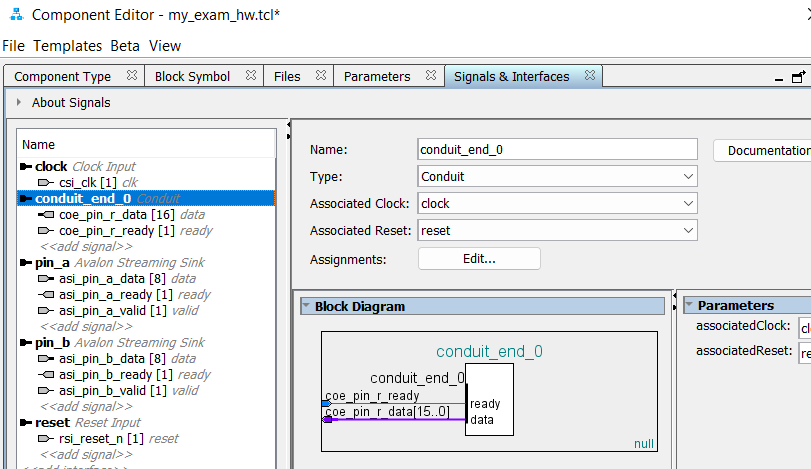
Вопрос 4.

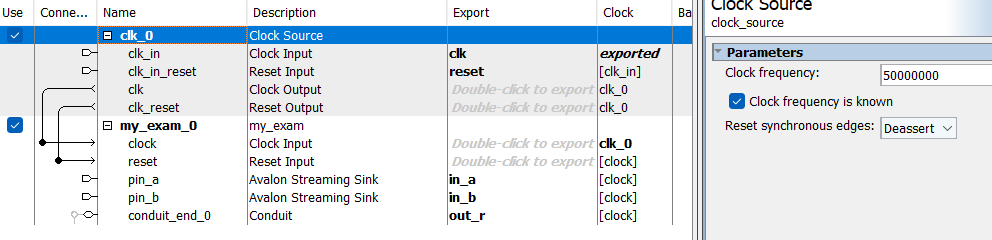
Интегрируйте устройство, созданное в вопросе 1, как библиотечный компонент в PD (библиотечная папка для модуля – exam).

***На этом месте в файле с ответами приведите снимки экрана:***

* ***с библиотекой PD в которой есть папка exam с созданным компонентом***
* ***настройки интерфейсной части компонента.***







|  |
| --- |
| module exam\_sys\_top(  input logic clk,  output logic in\_a\_ready,  input logic in\_a\_valid,  input logic [7:0] in\_a\_data,  output logic in\_b\_ready,  input logic in\_b\_valid,  input logic [7:0] in\_b\_data,  input logic out\_r\_ready,  output logic [15:0] out\_r\_data,  input logic reset  );  exam\_sys exam\_sys\_inst(  .clk\_clk (clk), // clk.clk  .in\_a\_ready (in\_a\_ready), // in\_a.ready  .in\_a\_valid (in\_a\_valid), // .valid  .in\_a\_data (in\_a\_data), // .data  .in\_b\_ready (in\_b\_ready), // in\_b.ready  .in\_b\_valid (in\_b\_valid), // .valid  .in\_b\_data (in\_b\_data), // .data  .out\_r\_ready (out\_r\_ready), // out\_r.ready  .out\_r\_data (out\_r\_data), // .data  .reset\_reset\_n (rr\_reset) // reset.reset\_n  );  bit r\_reset, rr\_reset;  always\_ff @(posedge clk) begin  r\_reset <= reset;  rr\_reset <= r\_reset;  end  endmodule |

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.

***На этом месте в файле с ответами приведите снимки экрана:***

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

