САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ПЕТРА ВЕЛИКОГО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Отчет

по заданию lab6

Дисциплина

«Технологии проектирования аппаратных средств компьютерных систем»

выполнил:

Курякин Д. А

группа:

преподаватель:

Антонов А. П.

Санкт-Петербург

2022

Оглавление

[1 Задание lab10 3](#_Toc99802188)

[1.1 Задание 3](#_Toc99802189)

[1.2 Создание макета в Platform Designer 3](#_Toc99802190)

[1.3 Анализ системы 5](#_Toc99802191)

[1.4 Выводы 7](#_Toc99802192)

[2 Задание lab11 7](#_Toc99802193)

[2.1 Задание 7](#_Toc99802194)

[2.2 Создание макета в Platform Designer 7](#_Toc99802195)

[2.3 Анализ системы 9](#_Toc99802196)

[2.1 Описание на языке System Verilog 10](#_Toc99802197)

[2.2 Результат синтеза (RTL) 11](#_Toc99802198)

[2.3 Моделирование 11](#_Toc99802199)

[2.4 Выводы 12](#_Toc99802200)

[3 Задание lab12 12](#_Toc99802201)

[3.1 Задание 12](#_Toc99802202)

[3.2 Создание макета в Platform Designer 12](#_Toc99802203)

[3.3 Анализ системы 13](#_Toc99802204)

[3.4 Описание на языке System Verilog 14](#_Toc99802205)

[3.5 Результат синтеза (RTL) 14](#_Toc99802206)

[3.6 Моделирование 15](#_Toc99802207)

[3.7 Выводы 16](#_Toc99802208)

# Задание lab3\_2

## Задание

Самостоятельно реализовать проект в среде Vivado.

## Описание на языке Verilog

Описание разрабатываемого устройства на языке Verilog приведено ниже в листингах 1-1 – 1-7.

Листинг 1-1. Lab3\_2.v

|  |
| --- |
| `timescale 1ns / 1ps  module lab3\_2 #(  parameter div = 100000000)(  input clk,  input dir,  input ena,  input rst,  output reg [6:0] digit,  output reg [7:0] an  );  wire [6:0] out0, out1;  reg [15:0] num\_indicate = 32'b0;  wire int\_rst, int\_ena, int\_dir;  CYNCin SYNCin\_RST(.clk(clk), .in(rst), .out(int\_rst));  CYNCin SYNCin\_ENA(.clk(clk), .in(ena), .out(int\_ena));  CYNCin SYNCin\_DIR(.clk(clk), .in(dir), .out(int\_dir));  wire cout;  CNT\_DIV #(div) CNT\_DIV\_inst(.clk(clk), .rst(int\_rst), .ena(int\_ena), .ena\_out(cout));  wire [5:0] cnt\_val;  CNT CNT\_inst(.clk(clk), .rst(int\_rst), .ena(cout & int\_ena), .dir(int\_dir), .out(cnt\_val));    wire [6:0] hex\_int [1:0];  CODER CODER\_inst(.clk(clk), .addr(cnt\_val), .hex\_data0(hex\_int[0]), .hex\_data1(hex\_int[1]));  CYNCout CYNCout\_inst(.clk(clk), .reset(int\_rst), .ena(int\_ena),  .in0(hex\_int[0]), .in1(hex\_int[1]),  .out0(out0), .out1(out1));  always @(posedge clk) begin  if (num\_indicate > 16'h8000) begin  digit <= out0;  an <= ~8'h01;  end else begin  digit <= out1;  an <= ~8'h02;  end  num\_indicate = num\_indicate + 16'h01;  end  endmodule |

Листинг 1-2. CYNCin.v

|  |
| --- |
| module CYNCin(  input clk, in,  output reg out  );  reg data;  always @(posedge clk) begin  data <= in;  out <= data;  end  endmodule |

Листинг 1-3. CNT\_DIV.v

|  |
| --- |
| `timescale 1ns / 1ps  module CNT\_DIV #(  parameter div\_cnt = 50000000)(  input clk, rst, ena,  output reg ena\_out  );    reg [26:0] count;  initial begin  count = 10'b0;  ena\_out = 0'b0;  end  always @(posedge clk)  begin  if(ena) begin  if(rst) begin  if(count == div\_cnt-1) begin  count = 26'b0;  ena\_out = 1'b1;  end else begin  count = count + 1'b1;  ena\_out = 1'b0;  end  end else begin  count = 26'b0;  ena\_out = 1'b0;  end  end  end  endmodule |

Листинг 1-4. CNT.v

|  |
| --- |
| module CNT (  input clk, rst, ena, dir,  output [5:0] out  );  integer MAX\_VALUE = 17;  reg [5:0] count;  initial begin  count = 6'b0;  end  assign out = count;  always @(posedge clk)  begin  if(ena) begin  if(rst) begin  if(dir)  if(count != MAX\_VALUE)  count = count + 1'b1;  else  count = 6'b0;  else  if(count != 6'b0)  count = count - 1'b1;  else  count = MAX\_VALUE;  end else begin  count = 6'b0;  end  end  end  endmodule |

Листинг 1-5. CODER.v

|  |
| --- |
| `timescale 1ns / 1ps  module CODER (  input clk,  input [5:0] addr,  output [6:0] hex\_data0, hex\_data1  );  wire [3:0] addr0, addr1;  blk\_mem\_gen\_0 ROM(.clka(clk), .clkb(clk), .addra(addr0), .douta(hex\_data0), .addrb(addr1), .doutb(hex\_data1));  assign addr1 = (addr==0) ? 0: addr/8'ha;  assign addr0 = (addr==0) ? 0: addr%8'ha;  endmodule |

Листинг 1-6. CYNCout.v

|  |
| --- |
| module CYNCout (  input clk, reset, ena,  input [6:0] in0, in1,  output reg [6:0] out0, out1  );  always @(posedge clk, posedge reset) begin  if(reset)  if(ena) begin  out0 = in0;  out1 = in1;  end  else begin  out0 = 7'b0;  out1 = 7'b0;  end  end  endmodule |

## Результат синтеза

Результат синтеза описания на языке Verilog в пакете Quartus приведен ниже, на Рис. 1‑1.

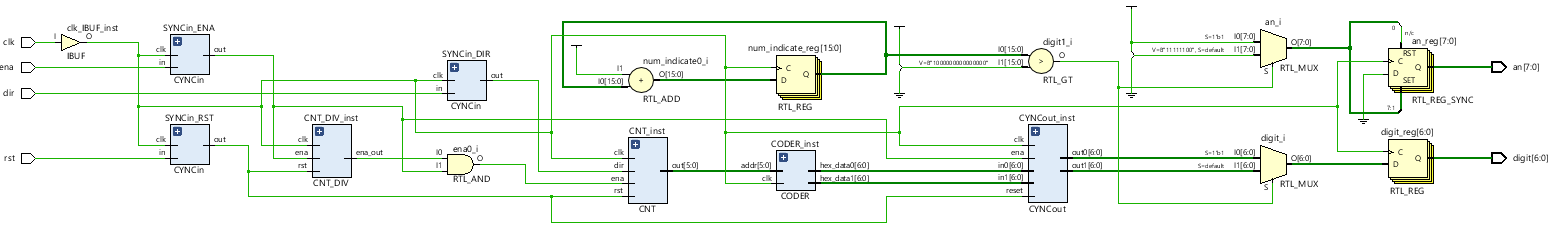


Рис. 1‑1 Синтезированная схема

## Моделирование

Для проверки правильности работы созданного Verilog описания использовались тесты:

Листинг 1-7. tb\_lab3\_2.v

|  |
| --- |
| `timescale 1ns / 1ps  module tb\_lab3\_2();  localparam div = 4;  reg clk = 1'b1, dir = 1'b1, ena = 1'b1, rst = 1'b1;  wire [6:0] digit;  wire [7:0] an;  lab3\_2 #(div) lab(.clk(clk), .dir(dir), .ena(ena), .rst(rst), .digit(digit), .an(an));  initial forever #10 clk = ~clk;  initial begin  #1600  rst = 1'b0;  #200  rst = 1'b0;  #1000  ena = 1'b0;  #400  $stop;  end  endmodule  initial begin: init\_data  dir <= 1'b0;  aclr <= 1'b0;  end    initial begin:assert\_  forever begin  for (j = 0; j < 2; j = j + 1) begin  for (i = 0; i < 32; i = i + 1) begin  @ (posedge clk) begin  $fdisplay(file, "time: %t res: dir: %b aclr: %b data: %b",  $realtime, dir, aclr, q);  $display("time: %t res: dir: %b aclr: %b data: %b",  $realtime, dir, aclr, q);  end  end  aclr = ~aclr; #20; aclr = ~aclr;  dir = ~dir;  if(j != 1) begin  $fdisplay(file, "\*\*\*CHANGE DIRECTION\*\*\*");  $display("\*\*\*CHANGE DIRECTION\*\*\*");  end  end  $fdisplay(file, "\*\*\*TEST SUCESSFULL PASSED\*\*\*");  $display("\*\*\*TEST SUCESSFULL PASSED\*\*\*");  $fclose(file);  $stop;  end  end  endmodule |

Результаты моделирования приведены на Рис. 1‑2

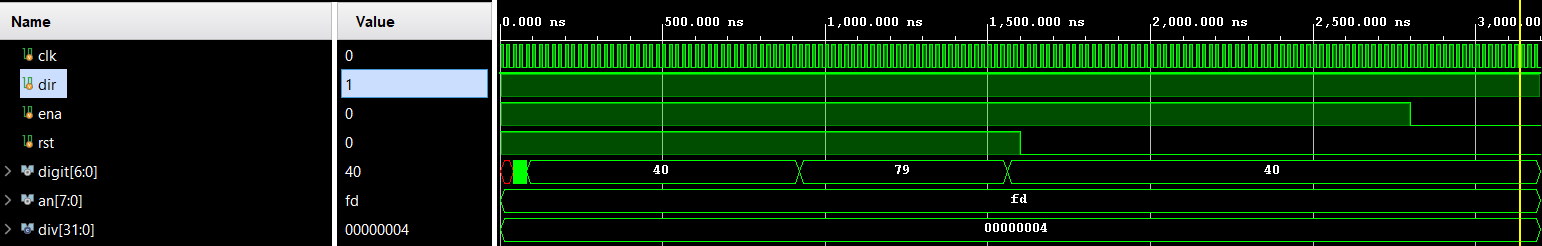


Рис. 1‑2 Результат моделирования

## Тестирование логическим анализатором

Для проверки правильности работы созданного Verilog описания были созданы ip-ядра vio, ila, clocking wizard. На листинге 1-8 приведен файл верхнего уровня:

Листинг 1-8. db\_Lab3\_2.v

|  |
| --- |
| `timescale 1ns / 1ps  module db\_Lab3\_2(input clk);  localparam div = 4;  wire clk\_2x, dir, ena, rst;  wire [6:0] digit;  wire [7:0] an;  clk\_wiz\_0 wiz\_inst(.clk\_in1(clk), .clk\_2x(clk\_2x));  lab3\_2 #(div) lab\_inst(.clk(clk\_2x), .dir(dir), .ena(ena), .rst(rst), .digit(digit), .an(an));  vio\_0 vio\_inst(.clk(clk\_2x), .probe\_in0(digit), .probe\_in1(an), .probe\_in2(clk), .probe\_out0(dir), .probe\_out1(ena), .probe\_out2(rst));  ila\_0 ila\_inst(.clk(clk\_2x), .probe0(digit), .probe1(an), .probe2(clk));  endmodule |

В проверки проект работал в соответствии с ожиданиями.

## Назначение выводов СБИС

Назначение выводов СБИС и стандартов ввода вывода приведено на Рис. 1‑9.

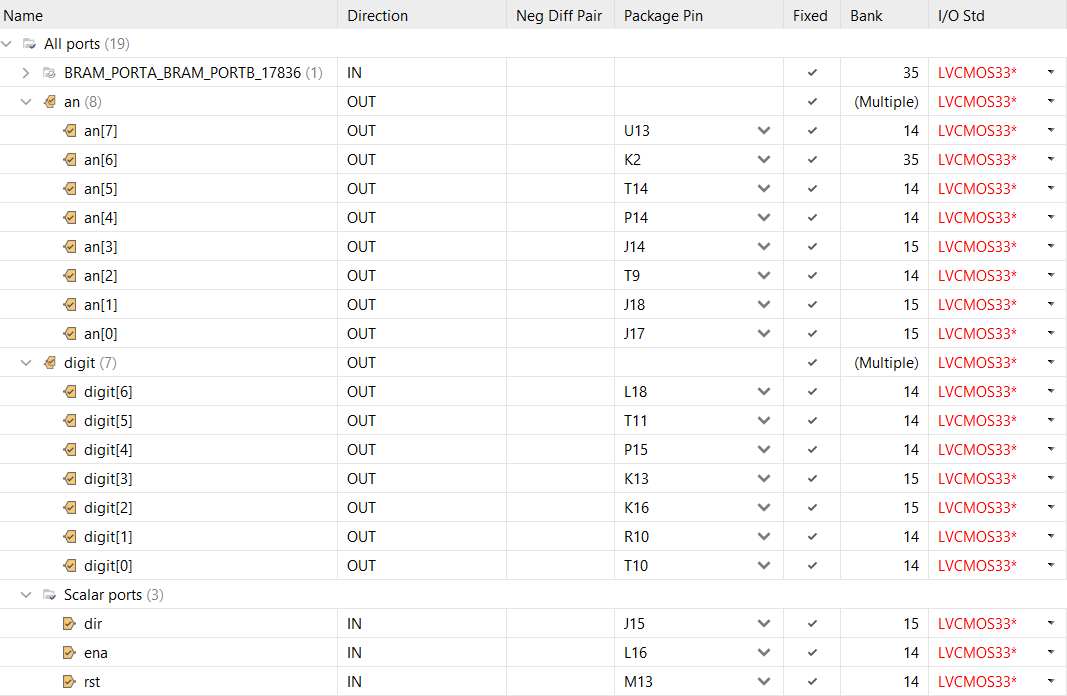


Рис. 1‑5 Назначение выводов

## Тестирование на плате Nexys4 DDR

Для тестирования проекта на плате использовались тесты, описанные в разделе 1.4. Проведенное на плате Nexys4 DDR тестирование разработанного устройства показало: результаты совпадают с ожидаемыми, устройство работает в соответствии с заданием.

## Выводы

В результате было реализовано устройство, описанное в задании, и были написаны тесты, требуемые в задании.