|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА - Российский технологический университет»**  **РТУ МИРЭА** |

Институт Информационных Технологий

Кафедра Вычислительной Техники (ВТ)

**ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №** **4**

«Основы языка команд Tcl»

по дисциплине

«Схемотехника устройств компьютерных систем»

|  |  |
| --- | --- |
| Выполнил студент группы  ИВБО-08-22 | Стецюк В.В. |
| Принял ассистент кафедры ВТ | Дуксин Н. А. |
| Практическая работа выполнена | « \_\_ » \_\_\_\_\_\_\_ 2024 г. |
| «Зачтено» | « \_\_ » \_\_\_\_\_\_\_ 2024 г. |

Москва 2024

АННОТАЦИЯ

Данная работа включает в себя 9 рисунков, 16 листингов. Количество страниц в работе — 33.

СОДЕРЖАНИЕ

[1 ПОСТАНОВКА ЗАДАЧИ 4](#_Toc163689954)

[2 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ 5](#_Toc163689955)

[2.1 Исходный код файлов проекта 5](#_Toc163689956)

[2.2 Файл с описанной программой на языке Tcl и результаты работы 21](#_Toc163689960)

[ЗАКЛЮЧЕНИЕ 30](#_Toc163689962)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 31](#_Toc163689963)

1. ПОСТАНОВКА ЗАДАЧИ

Сформировать набор файлов для создания проекта: Файлы на языке Verilog, содержащие модули для описания устройства, заданного вариантом, а также модули для верификации RTL-модели, файлы проектных ограничений для размещения проекта на ПЛИС, в набор файлов должны входить: конфигурация для работы устройства на частоте 100 МГц и для работы устройства на частоте 200 МГц. Сформировать файл с описанной программой на языке TCL, в которой: создать новый проект для чипа «xc7a100tcsg324-1». создать в рамках проекта наборы файлов «designs» для каждого отдельно оформленного теста, создать два набора файлов «constrs» проектных ограничений, добавить в проект в набор файлов «source set» с именем «sources\_1» файлы, содержащие модули для описания устройства, заданного вариантом, добавить в проект в каждый набор файлов «simulation set» необходимые файлы для тестирования в рамках набора, добавить в проект в каждый набор файлов «constraints set» требуемые файлы проектных ограничений, запуск процесса симуляции последовательно для каждого из набора «simulation set», запустить синтез и имплементацию последовательно для каждого набора «constraints set», для каждого варианта имплементации получить отчёт о временных задержках («Timing Summary Report»), сохранить данные отчёта в соответствующие файлы. Запустить на исполнение разработанный скрипт. Составить отчёт.

# ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ

## 2.1 Исходный код файлов проекта

Индивидуальный вариант: a - b << c \* d . Исходный код проекта был взят из третьей практической работы, где и проведено его описание. Результат представлен на Листингах 2.1 – 2.4.

*Листинг 2.1 – Модуль basis*

`timescale 1ns / 1ps

module basis(

input clk,

input btn\_c,

input btn\_rst,

input [15:0] SW,

output reg [31:0] shift\_register,

output reg [7:0] an\_mask,

output reg error

);

reg [3:0] state;

reg [31:0] a\_reg, b\_reg, d\_reg;

reg [15:0] c\_reg;

initial begin

shift\_register = 0;

state <= 0;

an\_mask = 8'b00000000;

end

always @(posedge clk) begin

if (btn\_rst)

state <= 0;

else

case (state)

4'b0: begin

a\_reg <= 0;

b\_reg <= 0;

c\_reg <= 0;

an\_mask <= 8'b11110000;

shift\_register <= 0;

error <= 0; state <= 1;

end

4'd1: begin

shift\_register <= {SW};

if (btn\_c) begin

a\_reg <= {SW};

state <= 2;

end

end

4'd2: begin

shift\_register <= {SW};

if (btn\_c) begin

*Продолжение Листинга 2.1*

b\_reg <= {SW};

state <= 3;

end

end

4'd3: begin

shift\_register <= {SW};

if (btn\_c) begin

c\_reg <= {SW};

state <= 4;

end

end

4'd4: begin

shift\_register <= {SW};

if (btn\_c) begin

d\_reg <= {SW};

an\_mask <= 8’b00000000;

state <= 5;

end

end

4'd5: begin

shift\_register <= a\_reg - b\_reg << c\_reg \* d\_reg

state <= 6;

end

end

endcase

end

endmodule

*Листинг 2.2 – Модуль SevenSegLED*

`timescale 1ns / 1ps

`timescale 1ns / 1ps

module SevenSegLED (

input clk,

input RST,

input [31:0] NUMBER,

input [7:0] AN\_MASK,

output [7:0] AN,

output reg [6:0] SEG

);

reg [7:0] AN\_REG = 0;

assign AN = AN\_REG | AN\_MASK;

reg [2:0] digit\_counter;

wire [3:0] NUMBER\_SPLITTER [0:7];

genvar i;

generate

for (i = 0; i < 8; i = i + 1)

begin

assign NUMBER\_SPLITTER[i] = NUMBER[((i+1)\*4-1)-:4];

end

endgenerate

initial begin

digit\_counter = 0;

end

always @(posedge clk or posedge RST)

*Продолжение Листинга 2.2*

digit\_counter <= RST ? 0 : digit\_counter + 4'b1;

always @(digit\_counter)

begin

case (NUMBER\_SPLITTER[digit\_counter])

4'h0: SEG <= 7'b1000000;

4'h1: SEG <= 7'b1111001;

4'h2: SEG <= 7'b0100100;

4'h3: SEG <= 7'b0110000;

4'h4: SEG <= 7'b0011001;

4'h5: SEG <= 7'b0010010;

4'h6: SEG <= 7'b0000010;

4'h7: SEG <= 7'b1111000;

4'h8: SEG <= 7'b0000000;

4'h9: SEG <= 7'b0010000;

4'ha: SEG <= 7'b0001000;

4'hb: SEG <= 7'b0000011;

4'hc: SEG <= 7'b1000110;

4'hd: SEG <= 7'b0100001;

4'he: SEG <= 7'b0000110;

4'hf: SEG <= 7'b0001110;

default: SEG <= 7'b1111111;

endcase

case (digit\_counter[2:0])

3'd0: AN\_REG <= 8'b11111110;

3'd1: AN\_REG <= 8'b11111101;

3'd2: AN\_REG <= 8'b11111011;

3'd3: AN\_REG <= 8'b11110111;

3'd4: AN\_REG <= 8'b11101111;

3'd5: AN\_REG <= 8'b11011111;

3'd6: AN\_REG <= 8'b10111111;

3'd7: AN\_REG <= 8'b01111111;

default: AN\_REG <= 8'b11111111;

endcase

end

endmodule

*Листинг 2.3 – Модуль фильтра дребезга контактов - filter*

`timescale 1ns / 1ps

module filter #(size = 3) (

input CLK, CLOCK\_ENABLE, IN\_SIGNAL,

output reg OUT\_SIGNAL, OUT\_SIGNAL\_ENABLE

);

reg [1:0] IN\_SIGNAL\_SYNC;

reg [size-1:0] counter;

initial begin

IN\_SIGNAL\_SYNC = 0; counter = 0;

OUT\_SIGNAL = 0; OUT\_SIGNAL\_ENABLE = 0;

end

always @(posedge CLK) begin

IN\_SIGNAL\_SYNC <= {IN\_SIGNAL\_SYNC[0], IN\_SIGNAL};

counter <= (IN\_SIGNAL\_SYNC[1] ~^ OUT\_SIGNAL) ?

{size{1'd0}} : (CLOCK\_ENABLE ? counter + 1 : counter);

if (&(counter) & CLOCK\_ENABLE)

OUT\_SIGNAL <= IN\_SIGNAL\_SYNC[1];

OUT\_SIGNAL\_ENABLE <= &(counter) & CLOCK\_ENABLE & IN\_SIGNAL\_SYNC[1];

end

endmodule

*Листинг 2.4 – Модуль делителя частоты – clk\_div*

`timescale 1ns / 1ps

module clk\_div #(DIV = 2) (

input clk,

output reg clk\_d);

wire [$clog2(DIV/2)-1:0] cnt;

counter #(.STEP(1), .MODULE(DIV/2)) cntr(

.clk(clk),

.reset(1'd0),

.dir(1'd1),

.enable(1'd1),

.cnt(cnt));

initial clk\_d = 0;

always@(posedge clk)

if (cnt == 0)

clk\_d = ~clk\_d;

endmodule

*Листинг 2.5 – Модуль cчетчика – counter*

`timescale 1ns / 1ps

module counter #(MODULE = 4, STEP = 1) (

input clk,

input reset,

input enable,

input dir,

output reg [$clog2(MODULE)-1:0] cnt);

initial cnt = 0;

always@(posedge clk)

begin

if (enable)

begin

if (reset)

cnt = 0;

else if (dir)

cnt = (cnt + STEP) % MODULE;

else

cnt = (MODULE + (cnt - STEP)) % MODULE;

end

end

endmodule

## Далее будут представлены тестовые модули, разработанные в третьей практической работе, где они также были описаны и верифицированы, на Листингах 2.6 - 2.9.

*Листинг 2.6 – Тестовый модуль делителя частоты – test\_clk\_div*

`timescale 1ns / 1ps

module test\_clk\_div();

reg clk;

initial clk = 0;

always #5 clk <= ~clk;

*Продолжение Листинга 2.6*

wire clk\_div\_out;

clk\_div clk\_div1 (

.clk(clk),

.clk\_div(clk\_div\_out)

);

integer clk\_div\_period = 200\_000;

realtime t\_begin, t\_end;

initial

begin

@(posedge clk\_div\_out);

t\_begin = $realtime;

@(posedge clk\_div\_out);

t\_end = $realtime;

$display("Ожидаемый период сигнала: %0d", clk\_div\_period);

$write("Фактический период сигнала: ");

$write((t\_end - t\_begin));

if ( (t\_end - t\_begin) == clk\_div\_period )

$display("\nТест пройден.");

else

$display("\nТест НЕ пройден.");

end

endmodule

*Листинг 2.7 – Тестовый модуль фильтра дребезга – test\_filter*

`timescale 1ns / 1ps

module test\_filter();

reg clk;

initial clk = 0;

always #5 clk <= ~clk;

localparam PRESS = 1, RELEASE = 0;

reg IN\_SIGNAL; initial IN\_SIGNAL = 0;

wire OUT\_SIGNAL\_ENABLE;

filter #(5) btn\_c\_filter(

.CLK(clk),

.CLOCK\_ENABLE(1),

.IN\_SIGNAL(IN\_SIGNAL),

.OUT\_SIGNAL\_ENABLE(OUT\_SIGNAL\_ENABLE)

);

localparam TEST\_COUNT = 3;

reg [0:TEST\_COUNT-1] test\_register;

initial

begin

test\_register = {TEST\_COUNT{1'b0}};

test\_filter\_1();

test\_filter\_2();

test\_filter\_3();

test\_show\_stats();

end

task test\_filter\_1;

*Продолжение Листинга 2.7*

reg test\_result;

begin

$display("\n[%0t]: Тест 1. Реакция фильтра дребезга на сигнал высокого уровня на шине физ. манипулятора.", $time);

$display("[%0t]: (время удержания сигнала соответствует требуемому)", $time);

send\_signal\_to\_filter(PRESS, 32);

@(posedge clk) test\_result <= (OUT\_SIGNAL\_ENABLE == 1'b1);

send\_signal\_to\_filter(RELEASE, 32);

test\_info(1, test\_result);

end

endtask

task test\_filter\_2;

reg test\_result;

begin

$display("\n[%0t]: Тест 2. Реакция фильтра дребезга на сигнал высокого уровня на шине физ. манипулятора.", $time);

$display("[%0t]: (время удержания сигнала меньше требуемого)", $time);

send\_signal\_to\_filter(PRESS, 16);

@(posedge clk); test\_result = (OUT\_SIGNAL\_ENABLE == 1'b0);

test\_info(2, test\_result);

end

endtask

task test\_filter\_3;

reg test\_result;

begin

$display("\n[%0t]: Тест 3. Реакция фильтра дребезга на сигнал низкого уровня на шине физ. манипулятора.", $time);

send\_signal\_to\_filter(RELEASE, 32);

@(posedge clk); test\_result = (OUT\_SIGNAL\_ENABLE == 1'b0);

test\_info(3, test\_result);

end

endtask

task test\_info;

input integer test\_number;

input test\_result;

begin

test\_register[test\_number-1] = test\_result;

if (test\_result)

$display("[%0t]: Тест %0d пройден.", $time, test\_number);

else

$display("[%0t]: Тест %0d НЕ пройден.", $time, test\_number);

end

endtask

task test\_show\_stats;

integer i, test\_counter;

begin

$display("\nРезультаты тестирования:");

test\_counter = 0;

for (i = 0; i < TEST\_COUNT; i = i + 1)

begin

if (test\_register[i])

$display("Тест %2d пройден.", i+1);

else

$display("Тест %2d НЕ пройден.", i+1);

test\_counter = test\_counter + (test\_register[i] ? 1 : 0);

end

*Продолжение Листинга 2.7*

$display("Пройдено тестов: %0d/%0d", test\_counter, TEST\_COUNT);

end

endtask

task send\_signal\_to\_filter;

input signal\_in;

input [6:0] ticks;

begin

@(posedge clk);

IN\_SIGNAL <= signal\_in;

$display("[%0t]: Сигнал %b подан на линию.", $time, signal\_in);

repeat(ticks + 2)

@(posedge clk);

IN\_SIGNAL <= 0;

$display("[%0t]: Сигнал %b убран с линии, подан сигнал 0", $time, signal\_in);

end

endtask

task test\_filter;

input signal\_in;

input [6:0] ticks;

output reg test\_result;

begin

end

endtask

task push\_c\_button;

begin

//$display("Дребезг на кнопке C");

$srandom(33985);

repeat($urandom\_range(150,0))

begin

IN\_SIGNAL = $random;

#3;

end

IN\_SIGNAL = 1;

#200;

repeat($urandom\_range(150,0))

begin

IN\_SIGNAL = $random;

#3;

end

IN\_SIGNAL = 0;

#200;

end

endtask

endmodule

*Листинг 2.8 – Тестовый модуль индикатора – test\_seven\_seg\_led*

`timescale 1ns / 1ps

module test\_sev\_seg\_led();

reg clk; initial clk = 0;

always #10 clk <= ~clk;

localparam AN\_COUNT = 8;

localparam CATH\_COUNT = 7;

localparam DIGIT\_SIZE = 4;

localparam DIGIT\_COUNT = 16;

reg CE, RESET;

reg [AN\_COUNT\*DIGIT\_SIZE-1:0] NUMBER;

reg [AN\_COUNT-1:0] AN\_MASK;

initial

begin

CE = 1;

RESET = 0;

NUMBER = {(AN\_COUNT\*DIGIT\_SIZE){1'b0}};

end

wire [AN\_COUNT-1:0] AN;

wire [CATH\_COUNT-1:0] CATH;

SevenSegLED uut (

.clk(clk),

.RST(RESET),

.NUMBER(NUMBER),

.AN\_MASK(AN\_MASK),

.AN(AN),

.SEG(CATH)

);

initial

begin

test\_seven\_segments(8'b00101100);

test\_show\_stats();

end

function [6:0] get\_cath\_mask;

input [3:0] number;

begin

case (number)

4'h0: get\_cath\_mask = 7'b1000000;

4'h1: get\_cath\_mask = 7'b1111001;

4'h2: get\_cath\_mask = 7'b0100100;

4'h3: get\_cath\_mask = 7'b0110000;

4'h4: get\_cath\_mask = 7'b0011001;

4'h5: get\_cath\_mask = 7'b0010010;

4'h6: get\_cath\_mask = 7'b0000010;

4'h7: get\_cath\_mask = 7'b1111000;

4'h8: get\_cath\_mask = 7'b0000000;

4'h9: get\_cath\_mask = 7'b0010000;

4'ha: get\_cath\_mask = 7'b0001000;

4'hb: get\_cath\_mask = 7'b0000011;

4'hc: get\_cath\_mask = 7'b1000110;

4'hd: get\_cath\_mask = 7'b0100001;

4'he: get\_cath\_mask = 7'b0000110;

4'hf: get\_cath\_mask = 7'b0001110;

default: get\_cath\_mask = 7'b1111111;

*Продолжение Листинга 2.8*

endcase

end

endfunction

function [7:0] get\_an\_mask;

input [2:0] an\_number;

begin

case (an\_number)

3'd0: get\_an\_mask = 8'b11111110;

3'd1: get\_an\_mask = 8'b11111101;

3'd2: get\_an\_mask = 8'b11111011;

3'd3: get\_an\_mask = 8'b11110111;

3'd4: get\_an\_mask = 8'b11101111;

3'd5: get\_an\_mask = 8'b11011111;

3'd6: get\_an\_mask = 8'b10111111;

3'd7: get\_an\_mask = 8'b01111111;

default: get\_an\_mask = 8'b11111111;

endcase

end

endfunction

reg [AN\_COUNT-1:0] test\_an\_register;

reg [DIGIT\_COUNT-1:0] test\_digit\_register;

reg test\_an\_mask\_register;

task test\_seven\_segments;

input [AN\_COUNT-1:0] mask\_value;

reg [3:0] i;

reg [3:0] number;

begin

$display("\n[%0t]: Тест отображения цифр на индикаторах, принципа работы динамической индикации и анодной маски.", $time);

test\_an\_register = {AN\_COUNT{1'b1}};

test\_digit\_register = {DIGIT\_COUNT{1'b1}};

test\_an\_mask\_register = 1'b1;

AN\_MASK = mask\_value;

$display("Битовая маска (AN\_MASK): %b", AN\_MASK);

wait(uut.digit\_counter == AN\_COUNT-1);

@(posedge clk);

number = 0;

repeat(DIGIT\_COUNT)

begin

// Подача числа на входную шину

for (i = 0; i < AN\_COUNT; i = i + 1)

NUMBER[ ((i+1)\*4)-1 -: 4 ] <= number;

@(posedge clk);

$display("\n[%0t]: Тест для цифры: %h", $time, number);

for (i = 0; i < AN\_COUNT; i = i + 1)

begin

$display("Текущий анод: %d", i);

test\_digit\_register[number] <= CATH == get\_cath\_mask(number);

$display("Ожидаемые сигналы на линии катодов (CATH): %b", get\_cath\_mask(number));

$display("Фактические сигналы на линии катодов (CATH): %b", CATH);

test\_an\_register[number] <= uut.AN\_REG == get\_an\_mask(i);

*Продолжение Листинга 2.8*

$display("Ожидаемые сигналы на линии анодов (ДО применения анодной маски): %b", get\_an\_mask(i));

$display("Фактические сигналы на линии анодов (ДО применения анодной маски): %b", uut.AN\_REG);

function [7:0] get\_an\_mask;

test\_an\_mask\_register <= AN == (get\_an\_mask(i) | AN\_MASK);

$display("Ожидаемые сигналы на линии анодов (ПОСЛЕ применения анодной маски): %b", get\_an\_mask(i) | AN\_MASK);

$display("Фактические сигналы на линии анодов (ПОСЛЕ применения анодной маски): %b", AN);

if (i != AN\_COUNT-1)

@(posedge clk);

end

number = number + 1;

end

end

endtask

task test\_show\_stats;

localparam TEST\_COUNT = 3;

integer test\_counter, i;

begin

test\_counter = 0;

$display("\n[%0t]: Результаты тестирования:", $time);

// Отображение цифры

if (&(test\_digit\_register))

begin

$display("1. Тест на отображение пройден успешно для всех возможных вариантов цифр.");

test\_counter = test\_counter + 1;

end

else begin

$display("1. Тест на отображение цифр НЕ пройден");

for (i = 0; i < DIGIT\_COUNT; i = i + 1)

if (!test\_digit\_register[i])

$display("Ошибка отображения цифры %d", i);

end

// Динамическая индикация

if (&(test\_an\_register))

begin

test\_counter = test\_counter + 1;

$display("2. Тест работы динамической индикации пройден успешно.");

end

else begin

$display("2. Тест работы динамической индикации НЕ пройден.", i);

for (i = 0; i < AN\_COUNT; i = i + 1)

if (!test\_an\_register[i])

$display("Ошибка на индикаторе %0d.", i);

end

// Анодная маска

if (test\_an\_mask\_register)

begin

$display("3. Тест анодной маски пройден успешно.");

test\_counter = test\_counter + 1;

end

else

$display("3. Тест анодной маски НЕ пройден.");

$display("Пройдено тестов: %0d/%0d.", test\_counter, TEST\_COUNT);

end

endtask

endmodule

*Листинг 2.9 – Тестовый модуль конечного автомата – test\_basis*

`timescale 1ns / 1ps

module test\_basis();

reg clk;

initial clk = 0;

always #5 clk <= ~clk;

reg btn\_c, btn\_rst;

reg [31:0] SW;

wire [31:0] shift\_register;

wire [7:0] an\_mask;

wire error;

initial begin

btn\_c = 0;

btn\_rst = 0;

end

basis uut(

.clk(clk),

.btn\_c(btn\_c),

.btn\_rst(btn\_rst),

.SW(SW),

.shift\_register(shift\_register),

.an\_mask(an\_mask),

.error(error)

);

localparam TEST\_COUNT = 3;

reg [0:TEST\_COUNT-1] test\_register;

initial

begin

test\_register = {TEST\_COUNT{1'b0}};test\_op\_1();

send\_rst();

test\_op\_2();send\_rst();

test\_op\_3();

test\_show\_stats();

end

task test\_op\_1;

reg test\_result;

begin

$display("\n[%0t]: Тест 1. a = 1001, b = 0001, c = 0003, d = 0001.", $time);

send\_signal\_to\_op(16'h1001);

send\_signal\_to\_op(16'h0001);

send\_signal\_to\_op(16'h0003);

send\_signal\_to\_op(16'h0001);

repeat(5)

@(posedge clk);

test\_result = ((shift\_register == 32'h000037cd) && (error == 1'b0));

test\_info(1, test\_result);end

endtask

task test\_op\_2;

reg test\_result;

begin

$display("\n[%0t]: Тест 2. a = 5400, b = 4400, c = 0002, d = 0001.", $time);

*Продолжение Листинга 2.9*

send\_signal\_to\_op(16'h5400);

send\_signal\_to\_op(16'h4400);

send\_signal\_to\_op(16'h0002);

send\_signal\_to\_op(16'h0001);

repeat(5)

@(posedge clk);

test\_result = (shift\_register == 16'hffff && error == 1'b1);

test\_info(2, test\_result);

end

endtask

task test\_op\_3;

reg test\_result;

begin

$display("\n[%0t]: Тест 3. a = 3219, b = 1119, c = 0001, d = 0002.", $time);

send\_signal\_to\_op(16'h3219);

send\_signal\_to\_op(16'h1119);

send\_signal\_to\_op(16'h0001);

send\_signal\_to\_op(16'h0002);

repeat(5)

@(posedge clk);

test\_result = (shift\_register == 32'h000000c8 && error == 1'b0);

test\_info(3, test\_result);

end

endtask

task test\_info;

input integer test\_number;

input test\_result;

begin

test\_register[test\_number-1] = test\_result;

if (test\_result)

$display("[%0t]: Тест %0d пройден.", $time, test\_number);

else

$display("[%0t]: Тест %0d НЕ пройден.", $time, test\_number);

end

endtask

task test\_show\_stats;integer i, test\_counter;

begin

$display("\nРезультаты тестирования:");

test\_counter = 0;

for (i = 0; i < TEST\_COUNT; i = i + 1)

begin

if (test\_register[i])

$display("Тест %2d пройден.", i+1);

else

$display("Тест %2d НЕ пройден.", i+1);

test\_counter = test\_counter + (test\_register[i] ? 1 : 0);

end

$display("Пройдено тестов: %0d/%0d", test\_counter, TEST\_COUNT);

end

endtask

*Продолжение Листинга 2.9*

task send\_rst;

begin

@(posedge clk);

@(posedge clk) btn\_rst <= 1;

@(posedge clk) btn\_rst <= 0;

end

endtask

task send\_signal\_to\_op;

input [15:0] in\_signal;

begin

@(posedge clk);

SW <= in\_signal;

$display("[%0t]: Сигнал %h подан на линию.", $time, in\_signal);@(posedge clk) btn\_c <= 1;

@(posedge clk) btn\_c <= 0;

end

endtask

endmodule

## Далее представлен модуль верхнего уровня на Листинге 2.10

*Листинг 2.10 – Модуль верхнего уровня – upper*

`timescale 1ns / 1ps

module upper(

input clk,

input btn\_c,

input btn\_rst,

input [15:0] SW,

output [7:0] AN,

output [6:0] SEG

);

wire [31:0] shift\_register;

wire [7:0] an\_mask;

wire error;

reg CLOCK\_ENABLE = 0;

always @(posedge clk)

CLOCK\_ENABLE <= ~CLOCK\_ENABLE;

wire btn\_c\_out, btn\_c\_out\_enable;

filter #(.size(7)) btn\_c\_filter (

.CLK(clk),

.CLOCK\_ENABLE(CLOCK\_ENABLE),

.IN\_SIGNAL(btn\_c),

.OUT\_SIGNAL(btn\_c\_out),

.OUT\_SIGNAL\_ENABLE(btn\_c\_out\_enable)

);

wire btn\_rst\_out, btn\_rst\_out\_enable;

filter #(.size(7)) btn\_RST\_filter (

.CLK(clk),

.CLOCK\_ENABLE(CLOCK\_ENABLE),

.IN\_SIGNAL(btn\_rst),

.OUT\_SIGNAL(btn\_rst\_out),

.OUT\_SIGNAL\_ENABLE(btn\_rst\_out\_enable)

*Продолжение Листинга 2.10*

);

basis basis1(

.clk(clk),

.btn\_c(btn\_c\_out\_enable),

.btn\_rst(btn\_rst\_out\_enable),

.SW(SW),

.shift\_register(shift\_register),

.an\_mask(an\_mask),

.error(error)

);

clk\_div #(.DIV(20)) clk\_div1 (

.clk(clk),

.clk\_d(clk\_div\_out)

);

clk\_div #(.DIV(10)) clk\_div2 (

.clk(clk),

.clk\_d(clk\_div\_out\_10)

);

SevenSegLED seg (

.clk(clk\_div\_out),

.RST(btn\_rst\_out\_enable),

.NUMBER(shift\_register),

.AN\_MASK(an\_mask),

.AN(AN),

.SEG(SEG)

);

endmodule

## На Листинге 2.11 представлен модуль для взаимодействия с vio, также разработанный в предыдущей работе

*Листинг 2.11 – Модуль для взаимодействия с vio – main*

`timescale 1ns / 1ps

module main(

input clk,

output [7:0] AN,

output [6:0] SEG

);

wire btn\_c, btn\_rst;

wire [15:0] SW;

upper uut(

.clk(clk),

.btn\_c(btn\_c),

.btn\_rst(btn\_rst),

.SW(SW),

.AN(AN),

.SEG(SEG)

);

vio\_0 vio(

.clk(clk),

.probe\_in0(AN),

.probe\_in1(SEG),

.probe\_out0(btn\_c),

.probe\_out1(btn\_res),

.probe\_out2(SW)

);

endmodule

Далее представлены файлы проектных ограничений, два из которых предназначены для работы без vio и два для работы с vio. Также учтено, что в работу должны входить две конфигурации для работы устройства на частоте 100 и 200 МГц. Файлы проектных ограничений представлены на Листингах 2.12 – 2.15.

*Листинг 2.12 – Конфигурация файла проектных ограничений с частотой 100 и работы с vio*

create\_clock -add -name sys\_clk\_pin -period 10.00 -waveform {0 5} [get\_ports {clk}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN E3} [get\_ports {clk}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J17} [get\_ports {AN[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J18} [get\_ports {AN[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T9} [get\_ports {AN[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J14} [get\_ports {AN[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P14} [get\_ports {AN[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T14} [get\_ports {AN[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K2} [get\_ports {AN[6]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN U13} [get\_ports {AN[7]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T10} [get\_ports {SEG[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN R10} [get\_ports {SEG[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K16} [get\_ports {SEG[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K13} [get\_ports {SEG[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P15} [get\_ports {SEG[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T11} [get\_ports {SEG[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN L18} [get\_ports {SEG[6]}]

*Листинг 2.13 – Конфигурация файла проектных ограничений с частотой 200 и работы с vio*

create\_clock -add -name sys\_clk\_pin -period 5.00 -waveform {0 4} [get\_ports {clk}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN E3} [get\_ports {clk}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J17} [get\_ports {AN[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J18} [get\_ports {AN[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T9} [get\_ports {AN[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J14} [get\_ports {AN[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P14} [get\_ports {AN[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T14} [get\_ports {AN[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K2} [get\_ports {AN[6]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN U13} [get\_ports {AN[7]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T10} [get\_ports {SEG[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN R10} [get\_ports {SEG[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K16} [get\_ports {SEG[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K13} [get\_ports {SEG[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P15} [get\_ports {SEG[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T11} [get\_ports {SEG[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN L18} [get\_ports {SEG[6]}]

*Листинг 2.14–Конфигурация файла проектных ограничений с частотой 100 и работы без vio*

create\_clock -add -name sys\_clk\_pin -period 10.00 -waveform {0 5} [get\_ports {clk}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN E3} [get\_ports {clk}]

set\_property -dict {PACKAGE\_PIN J15 IOSTANDARD LVCMOS33} [get\_ports {SW[0]}]

set\_property -dict {PACKAGE\_PIN L16 IOSTANDARD LVCMOS33} [get\_ports {SW[1]}]

set\_property -dict {PACKAGE\_PIN M13 IOSTANDARD LVCMOS33} [get\_ports {SW[2]}]

set\_property -dict {PACKAGE\_PIN R15 IOSTANDARD LVCMOS33} [get\_ports {SW[3]}]

set\_property -dict {PACKAGE\_PIN R17 IOSTANDARD LVCMOS33} [get\_ports {SW[4]}]

set\_property -dict {PACKAGE\_PIN T18 IOSTANDARD LVCMOS33} [get\_ports {SW[5]}]

set\_property -dict {PACKAGE\_PIN U18 IOSTANDARD LVCMOS33} [get\_ports {SW[6]}]

set\_property -dict {PACKAGE\_PIN R13 IOSTANDARD LVCMOS33} [get\_ports {SW[7]}]

set\_property -dict {PACKAGE\_PIN T8 IOSTANDARD LVCMOS33} [get\_ports {SW[8]}]

set\_property -dict {PACKAGE\_PIN U8 IOSTANDARD LVCMOS33} [get\_ports {SW[9]}]

set\_property -dict {PACKAGE\_PIN R16 IOSTANDARD LVCMOS33} [get\_ports {SW[10]}]

set\_property -dict {PACKAGE\_PIN T13 IOSTANDARD LVCMOS33} [get\_ports {SW[11]}]

set\_property -dict {PACKAGE\_PIN H6 IOSTANDARD LVCMOS33} [get\_ports {SW[12]}]

set\_property -dict {PACKAGE\_PIN U12 IOSTANDARD LVCMOS33} [get\_ports {SW[13]}]

set\_property -dict {PACKAGE\_PIN U11 IOSTANDARD LVCMOS33} [get\_ports {SW[14]}]

set\_property -dict {PACKAGE\_PIN V10 IOSTANDARD LVCMOS33} [get\_ports {SW[15]}]

set\_property -dict {PACKAGE\_PIN N17 IOSTANDARD LVCMOS33} [get\_ports {btn\_c}]

set\_property -dict {PACKAGE\_PIN M17 IOSTANDARD LVCMOS33} [get\_ports {btn\_rst}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J17} [get\_ports {AN[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J18} [get\_ports {AN[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T9} [get\_ports {AN[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J14} [get\_ports {AN[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P14} [get\_ports {AN[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T14} [get\_ports {AN[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K2} [get\_ports {AN[6]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN U13} [get\_ports {AN[7]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T10} [get\_ports {SEG[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN R10} [get\_ports {SEG[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K16} [get\_ports {SEG[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K13} [get\_ports {SEG[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P15} [get\_ports {SEG[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T11} [get\_ports {SEG[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN L18} [get\_ports {SEG[6]}]

*Листинг2.15–Конфигурация файла проектных ограничений с частотой 200 и работы без vio*

create\_clock -add -name sys\_clk\_pin -period 5.00 -waveform {0 4} [get\_ports {clk}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN E3} [get\_ports {clk}]

set\_property -dict {PACKAGE\_PIN J15 IOSTANDARD LVCMOS33} [get\_ports {SW[0]}]

set\_property -dict {PACKAGE\_PIN L16 IOSTANDARD LVCMOS33} [get\_ports {SW[1]}]

set\_property -dict {PACKAGE\_PIN M13 IOSTANDARD LVCMOS33} [get\_ports {SW[2]}]

set\_property -dict {PACKAGE\_PIN R15 IOSTANDARD LVCMOS33} [get\_ports {SW[3]}]

set\_property -dict {PACKAGE\_PIN R17 IOSTANDARD LVCMOS33} [get\_ports {SW[4]}]

set\_property -dict {PACKAGE\_PIN T18 IOSTANDARD LVCMOS33} [get\_ports {SW[5]}]

set\_property -dict {PACKAGE\_PIN U18 IOSTANDARD LVCMOS33} [get\_ports {SW[6]}]

set\_property -dict {PACKAGE\_PIN R13 IOSTANDARD LVCMOS33} [get\_ports {SW[7]}]

set\_property -dict {PACKAGE\_PIN T8 IOSTANDARD LVCMOS33} [get\_ports {SW[8]}]

set\_property -dict {PACKAGE\_PIN U8 IOSTANDARD LVCMOS33} [get\_ports {SW[9]}]

set\_property -dict {PACKAGE\_PIN R16 IOSTANDARD LVCMOS33} [get\_ports {SW[10]}]

set\_property -dict {PACKAGE\_PIN T13 IOSTANDARD LVCMOS33} [get\_ports {SW[11]}]

set\_property -dict {PACKAGE\_PIN H6 IOSTANDARD LVCMOS33} [get\_ports {SW[12]}]

*Продолжение Листинга 2.15*

set\_property -dict {PACKAGE\_PIN U12 IOSTANDARD LVCMOS33} [get\_ports {SW[13]}]

set\_property -dict {PACKAGE\_PIN U11 IOSTANDARD LVCMOS33} [get\_ports {SW[14]}]

set\_property -dict {PACKAGE\_PIN V10 IOSTANDARD LVCMOS33} [get\_ports {SW[15]}]

set\_property -dict {PACKAGE\_PIN N17 IOSTANDARD LVCMOS33} [get\_ports {btn\_c}]

set\_property -dict {PACKAGE\_PIN M17 IOSTANDARD LVCMOS33} [get\_ports {btn\_rst}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J17} [get\_ports {AN[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J18} [get\_ports {AN[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T9} [get\_ports {AN[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN J14} [get\_ports {AN[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P14} [get\_ports {AN[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T14} [get\_ports {AN[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K2} [get\_ports {AN[6]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN U13} [get\_ports {AN[7]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T10} [get\_ports {SEG[0]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN R10} [get\_ports {SEG[1]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K16} [get\_ports {SEG[2]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN K13} [get\_ports {SEG[3]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN P15} [get\_ports {SEG[4]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN T11} [get\_ports {SEG[5]}]

set\_property -dict {IOSTANDARD LVCMOS33 PACKAGE\_PIN L18} [get\_ports {SEG[6]}]

## 2.2 Файл с описанной программой на языке Tcl и результаты работы

Если проект с именем Praс\_4 открыт, то он закрывается. Далее создается новый проект с этим же именем, в который копируются все исходные модули, включая тестовые файлы и файлы проектных ограничений. Создание и настройка сета симуляции для каждого тестового модуля, которая включает создание сетов симуляций, добавление модулей дизайна и тестовых модулей в наборы. Удаление сета симуляции по умолчанию. Создание сетов для файлов проектных ограничений. Удаление сета проектных ограничений по умолчанию и добавление файлов проектных ограничений в проект. Создание IP-ядра VIO. Симуляция на разных наборах тестов и имплементация на разных наборах файлов проектных ограничений. Код программы на языке Tcl представлен на Листинге 2.16.

*Листинг 2.16 – Код программы на языке Tcl*

set project\_name Prac\_4

set project\_found [llength [get\_projects $project\_name] ]

if {$project\_found > 0} close\_project

set origin\_dir [file dirname [info script]]

cd $origin\_dir

*Продолжение Листинга 2.16*

set path $origin\_dir/$project\_name/$project\_name

# Создание проекта

create\_project $project\_name $project\_name -force -part xc7a100tcsg324-1

# Добавление файлов дизайна

file mkdir $path.srcs/designs

set design\_file\_list [list seven\_seg\_light.v basis.v clk\_div.v upper.v filter.v main.v counter.v ]

foreach s $design\_file\_list {

file copy $origin\_dir/${s} ${path}.srcs/designs/${s}

add\_files -fileset sources\_1 -norecurse ${path}.srcs/designs/${s}

}

# Добавление файлов тестового окружения

file mkdir $path.srcs/testbenches

set test\_name\_list [ list test\_light test\_basis test\_clk\_div test\_filter ]

# Формирование ассоциативного массива формата (тестовый модуль - список файлов дизайнов для тестирования)

set test\_set\_designs(test\_light) [ list seven\_seg\_light.v ]

set test\_set\_designs(test\_basis) [ list basis.v ]

set test\_set\_designs(test\_clk\_div) [ list clk\_div.v counter.v ]

set test\_set\_designs(test\_filter) [ list filter.v ]

# Создание и настройка сета симуляции для каждого тестового модуля

foreach s $test\_name\_list {

# Создание сета симуляции

set set\_name ${s}\_set

file mkdir ${path}.srcs/testbenches/${set\_name}

create\_fileset -simset $set\_name

set\_property SOURCE\_SET {} [ get\_filesets $set\_name ]

# Добавления модулей дизайна в сет

set designs $test\_set\_designs(${s})

foreach design $designs {

add\_files -fileset $set\_name -norecurse ${path}.srcs/designs/$design

}

# Добавление тестового модуля в сет

file copy $origin\_dir/${s}.v ${path}.srcs/testbenches/${set\_name}/${s}.v

add\_files -fileset $set\_name ${path}.srcs/testbenches/${set\_name}/${s}.v

set\_property TOP ${s} [get\_filesets ${set\_name}]

}

# Удаление сета симуляции по умолчанию

current\_fileset -simset [ get\_filesets test\_light\_set ]

delete\_fileset [ get\_filesets sim\_1 ]

file delete -force $path.srcs/sim\_1

# Создание сетов для файлов проектных ограничений

create\_fileset -constrset fsm\_with\_vio\_100\_set

create\_fileset -constrset fsm\_with\_vio\_200\_set

create\_fileset -constrset fsm\_100\_set

create\_fileset -constrset fsm\_200\_set

# Ассоциативный массив с указанием модуля верхнего уровня

# для каждого сета проектных ограничений

array set constrset\_upper\_module {

fsm\_with\_vio\_100\_set main

fsm\_with\_vio\_200\_set main

*Продолжение Листинга 2.16*

fsm\_100\_set upper

fsm\_200\_set upper

}

# Удаление сета проектных ограничений по умолчанию

set\_property constrset fsm\_with\_vio\_100\_set [get\_runs synth\_1]

set\_property constrset fsm\_with\_vio\_100\_set [get\_runs impl\_1]

delete\_fileset [ get\_filesets constrs\_1 ]

file delete -force $path.srcs/constrs\_1

# Добавление файлов проектных ограничений в проект

file mkdir $path.srcs/constrs/fsm\_with\_vio

file mkdir $path.srcs/constrs/fsm

file copy $origin\_dir/c\_fsm\_100.xdc $path.srcs/constrs/fsm/c\_fsm\_100.xdc

file copy $origin\_dir/c\_fsm\_200.xdc $path.srcs/constrs/fsm/c\_fsm\_200.xdc

file copy $origin\_dir/c\_vio\_100.xdc $path.srcs/constrs/fsm\_with\_vio/c\_vio\_100.xdc

file copy $origin\_dir/c\_vio\_200.xdc $path.srcs/constrs/fsm\_with\_vio/c\_vio\_200.xdc

add\_files -fileset fsm\_100\_set -norecurse $path.srcs/constrs/fsm/c\_fsm\_100.xdc

add\_files -fileset fsm\_200\_set -norecurse $path.srcs/constrs/fsm/c\_fsm\_200.xdc

add\_files -fileset fsm\_with\_vio\_100\_set -norecurse $path.srcs/constrs/fsm\_with\_vio/c\_vio\_100.xdc

add\_files -fileset fsm\_with\_vio\_200\_set -norecurse $path.srcs/constrs/fsm\_with\_vio/c\_vio\_200.xdc

# Создание IP-ядра VIO

create\_ip -name vio -vendor xilinx.com -library ip -version 3.0 -module\_name vio\_0

set\_property -dict [list \

CONFIG.C\_NUM\_PROBE\_IN {2} \

CONFIG.C\_NUM\_PROBE\_OUT {3} \

CONFIG.C\_PROBE\_IN0\_WIDTH {8} \

CONFIG.C\_PROBE\_IN1\_WIDTH {7} \

CONFIG.C\_PROBE\_OUT2\_WIDTH {16} \

] [get\_ips vio\_0]

generate\_target {instantiation\_template} [get\_files "$path.srcs/sources\_1/ip/vio\_0/vio\_0.xci"]

update\_compile\_order -fileset sources\_1

generate\_target all [get\_files "$path.srcs/sources\_1/ip/vio\_0/vio\_0.xci"]

catch { config\_ip\_cache -export [get\_ips -all vio\_0] }

export\_ip\_user\_files -of\_objects [get\_files "$path.srcs/sources\_1/ip/vio\_0/vio\_0.xci"] -no\_script -sync -force -quiet

create\_ip\_run [get\_files -of\_objects [get\_fileset sources\_1] "$path.srcs/sources\_1/ip/vio\_0/vio\_0.xci"]

launch\_runs vio\_0\_synth\_1 -jobs 16

wait\_on\_runs vio\_0\_synth\_1

export\_simulation -of\_objects [get\_files "$path.srcs/sources\_1/ip/vio\_0/vio\_0.xci"] -directory "$path.ip\_user\_files/sim\_scripts" -ip\_user\_files\_dir "$path.ip\_user\_files" -ipstatic\_source\_dir "$path.ip\_user\_files/ipstatic" -lib\_map\_path [list {modelsim="$path.cache/compile\_simlib/modelsim"} {questa="$path.cache/compile\_simlib/questa"} {riviera="$path.cache/compile\_simlib/riviera"} {activehdl="$path.cache/compile\_simlib/activehdl"}] -use\_ip\_compiled\_libs -force -quiet

# Симуляция на разных наборах (simulation sets)

foreach t\_set [ get\_filesets test\* ] {

current\_fileset -simset $t\_set

*Продолжение Листинга 2.16*

file mkdir $origin\_dir/$project\_name/sim\_output/${t\_set}

# Команда сбрасывает время симуляции

set\_property -name xsim.simulate.runtime -value 0 -objects [get\_filesets ${t\_set}]

launch\_simulation

# Запуск симуляции на 210000ns и перенаправление вывода из TCL-консоли в файл

restart

run 210000ns > "$origin\_dir/$project\_name/sim\_output/${t\_set}/sim\_output.txt"

close\_sim

}

# Имплементация на разных наборах (constraints set)

foreach c\_set [ get\_filesets fsm\* ] {

set\_property constrset $c\_set [get\_runs synth\_1]

set\_property constrset $c\_set [get\_runs impl\_1]

set\_property TOP $constrset\_upper\_module($c\_set) [get\_fileset sources\_1]

reset\_run synth\_1

launch\_runs synth\_1 -jobs 16

wait\_on\_run synth\_1

reset\_run impl\_1

launch\_runs impl\_1 -jobs 16

wait\_on\_run impl\_1

open\_run impl\_1

report\_timing\_summary -file "$origin\_dir/$project\_name/timing\_summary\_${c\_set}"

}

## Результатом работы данного кода стало создание проекта, также программой были проведены тесты, результаты которых были записаны в директории sim\_output (Рисунок 2.1). Программа провела все тесты и записала итоговый результат в соответствующие файлы (Рисунок 2.2 – 2.5). Были проведены синтез и имплементация на каждом наборе файлов проектных ограничений. Результаты были записаны во временную сводку (Рисунок 2.6 – 2.9).

## 

**Рисунок 2.1 – Директория sim\_output**

## 

**Рисунок 2.2 – Файл результата симуляции делителя частоты**

## 

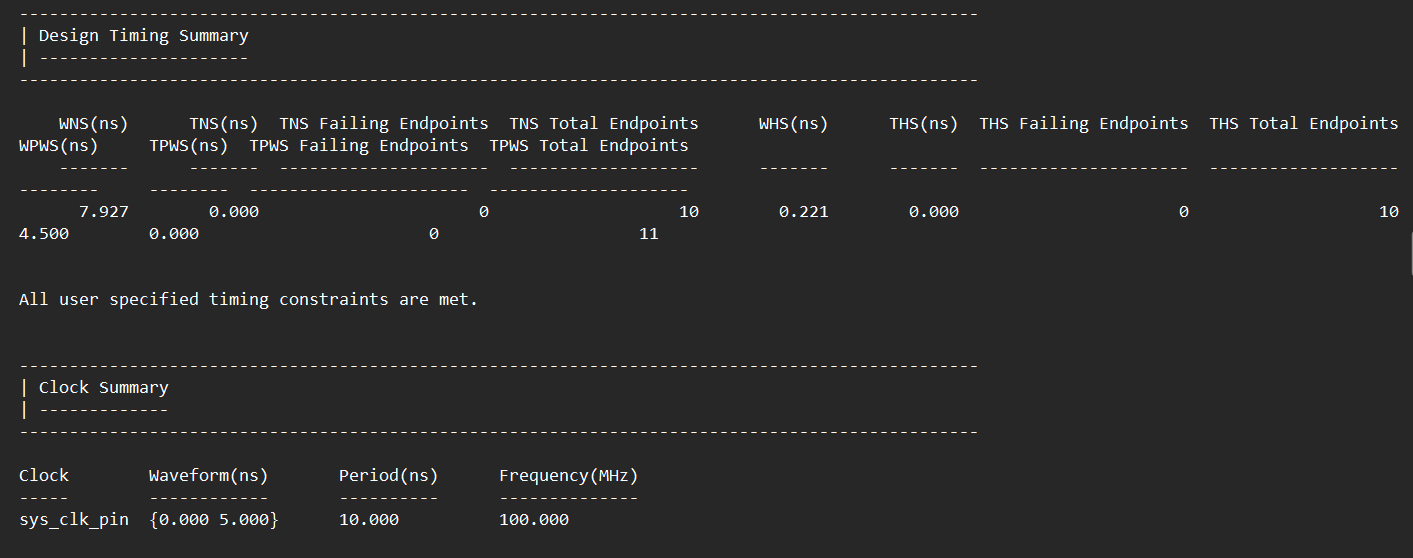
**Рисунок 2.3 – Файл результата симуляции фильтра дребезга**

## 

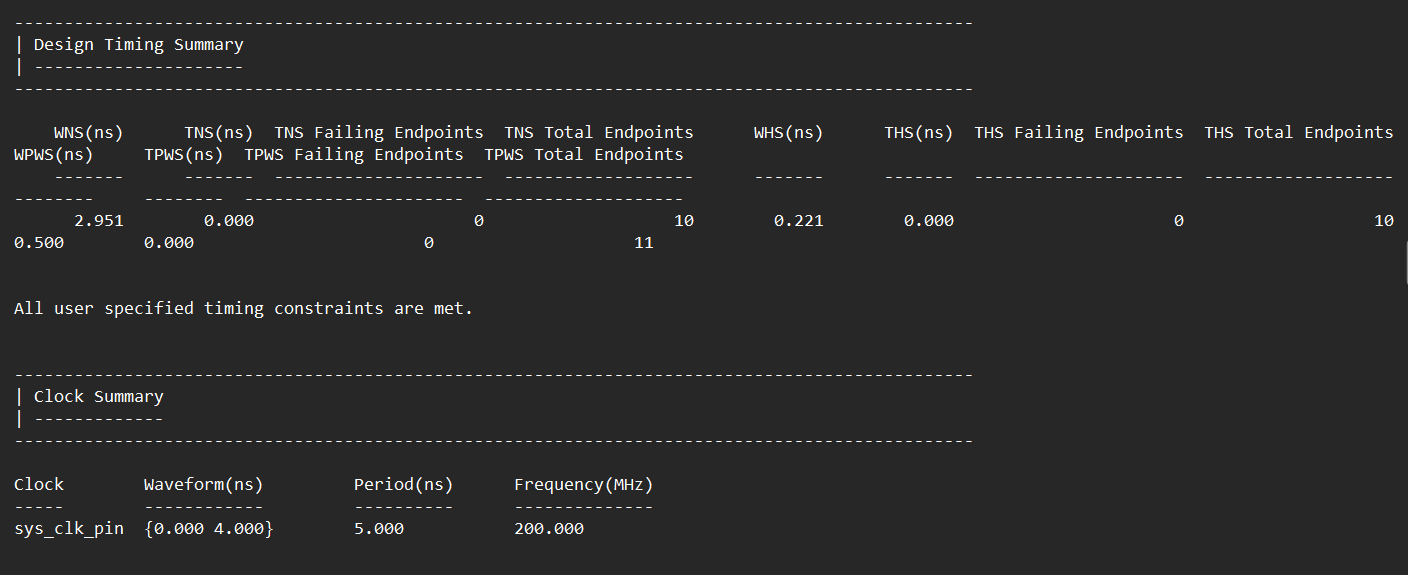
**Рисунок 2.4 – Файл результата симуляции конечного автомата**

## 

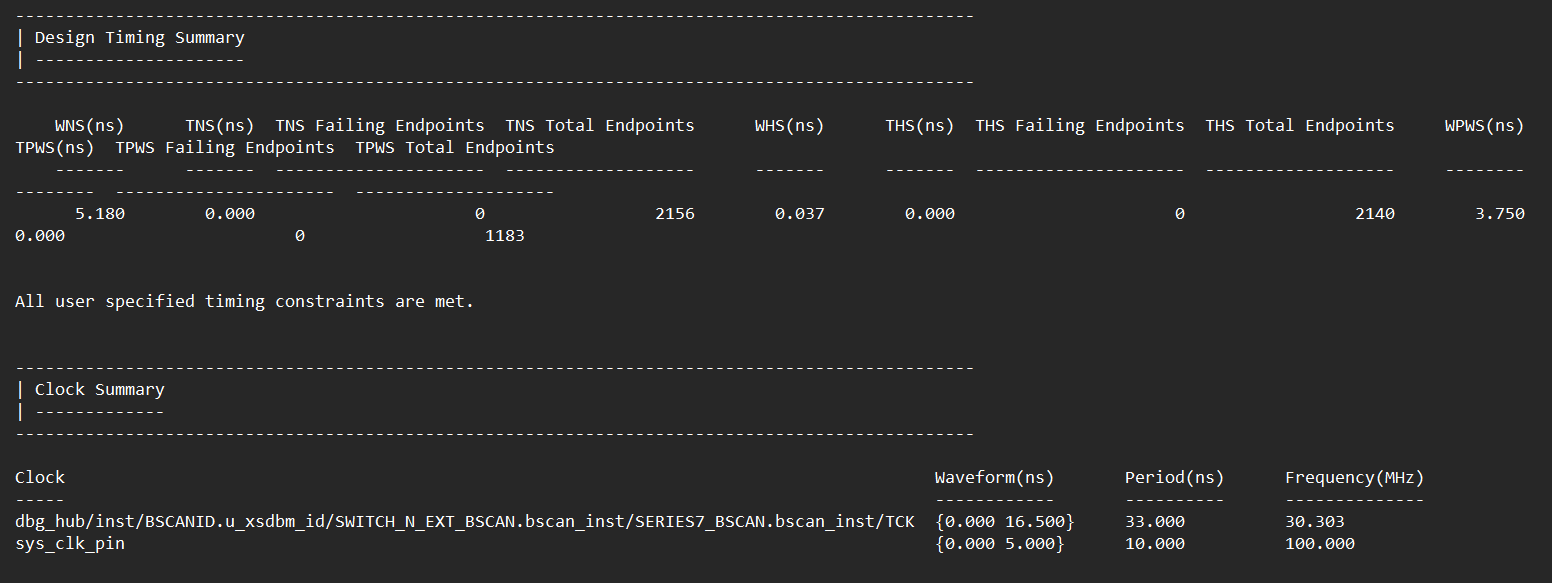
**Рисунок 2.5 – Файл результата симуляции модуля управления индикаторами**

****

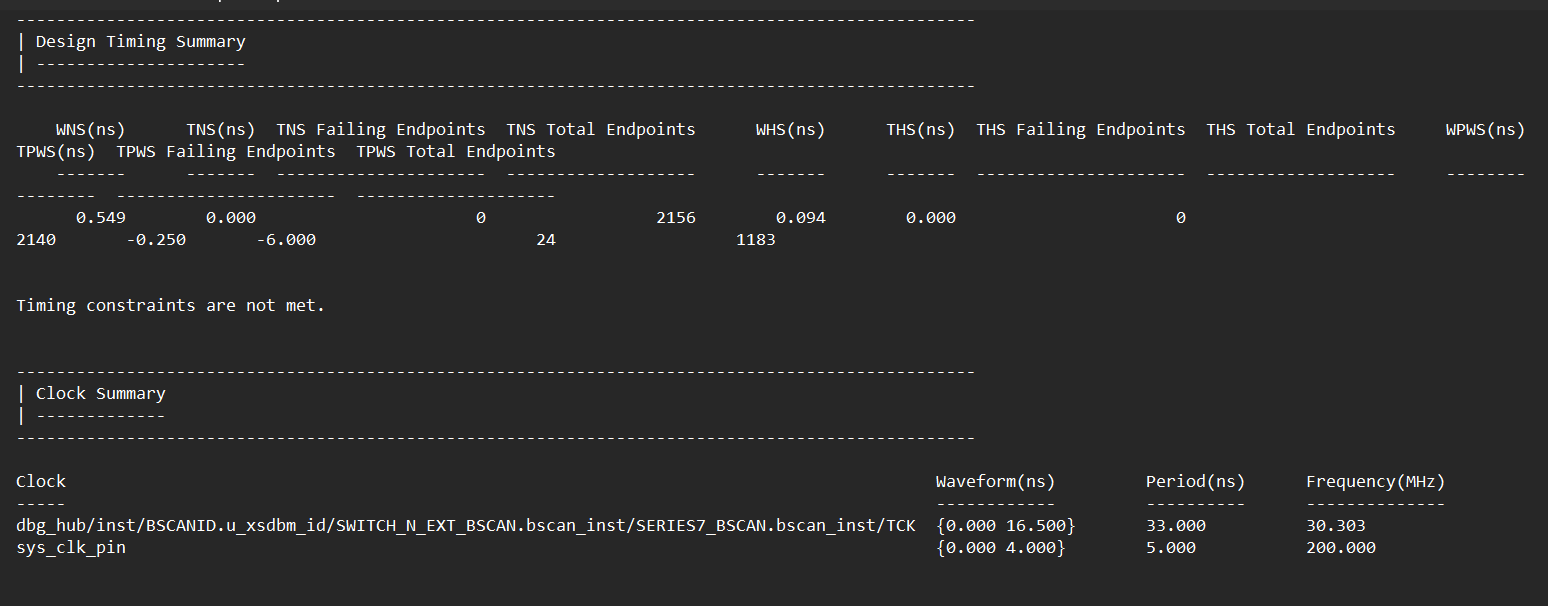
**Рисунок 2.6 – Значения задержек по Setup и Hold для набора проектных ограничений без vio с частотой 100 МГц**

****

**Рисунок 2.7 – Значения задержек по Setup и Hold для набора проектных ограничений без vio с частотой 200 МГц**

****

**Рисунок 2.8 – Значения задержек по Setup и Hold для набора проектных ограничений с vio с частотой 100 МГц**

****

**Рисунок 2.9 – Значения задержек по Setup и Hold для набора проектных ограничений с vio с частотой 200 МГц**

Программа успешно завершила свою работу, результаты временных задержек были выведены в соответствующие файлы.

ЗАКЛЮЧЕНИЕ

Таким образом, в данной практической работе был создан файл Tcl, в котором присутствует реализация создания нового проекта с исходными модулями, тестовыми модулями и файлом проектных ограничений, запуск симуляции, синтеза и имплементации на наборах тестов и файлов проектных ограничений. Синтез и имплементация была проведена без и с использованием IP-ядра VIO, с вариациями частот, равными 100 и 200 МГц. Полученный результаты были сохранены в соответствующие файлы.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Методические указания по ПР № 4 — URL: https://online-edu.mirea.ru/mod/resource/view.php?id=405132.

2. Смирнов С.С. Информатика [Электронный ресурс]: Методические указания по выполнению практических и лабораторных работ / С.С. Смирнов — М., МИРЭА — Российский технологический университет, 2018. — 1 электрон. опт. диск (CD-ROM).

3. Тарасов И.Е. ПЛИС Xilinx. Языки описания аппаратуры VHDL и Verilog, САПР, приемы проектирования. — М.: Горячая линия — Телеком, 2021. — 538 с.: ил.

4. Антик М.И. Дискретная математика [Электронный ресурс]: Учебное пособие / Антик М.И., Казанцева Л.В. — М.: МИРЭА — Российский технологический университет, 2018 — 1 электрон. опт. диск (CD-ROM).

5. Практическая работа № 3 – URL: https://online-edu.mirea.ru/pluginfile.php?file=%2F1225652%2Fassignsubmission\_file%2Fsubmission\_files%2F3439663%2FПрактика-3.pdf&amp;forcedownload=1