

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»

Институт информатики и кибернетики

Кафедра радиотехники

Отчет по индивидуальному домашнему заданию:

”Разработка цифровых устройств на базе ПЛИС”

Вариант №15

Студенты: Согонов Е.А., Третьяков А. В.

Преподаватель: Корнилин Д.В.

Группа: 6364-120304D

Самара 2022

## СОДЕРЖАНИЕ

1	Формализация текстового задания. . . . .	2
2	Описание устройства с помощью VHDL . . . . .	2
3	Симуляция устройства . . . . .	4
4	Синтез цифрового устройства . . . . .	6
5	Реализация . . . . .	9
6	Программирование ПЛИС . . . . .	10
7	Проверка функционирования устройства на отладочной плате . . .	10

## 1. Формализация текстового задания.

Исходное задание звучит так:

*Часы с индикацией минут и секунд на четырехзначном семисегментном индикаторе*

Схематически цифровое устройство, реализующее данный функционал можно изобразить с помощью блок-схемы, изображенной на рисунке 1.

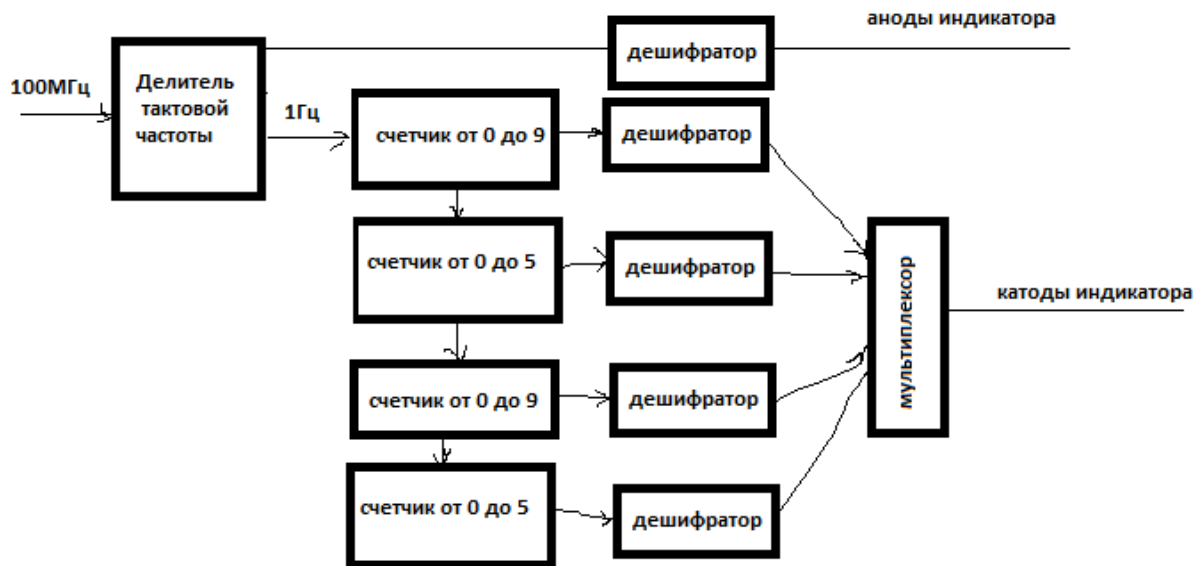


Рисунок 1 – Блок-схема часов

Первый счетчик делит частоту встроенного в отладочную плату осциллятора со 100 МГц до 1 Гц, что соответствует периоду в одну секунду, следующий счетчик считает количество этих импульсов, считая от 0 до 9. Следующий счетчик считает от 0 до 6, что соответствует десяткам секунд. Аналогично следующие два счетчика соответствуют единицам и десяткам минут. Текущее значение каждого из счетчиков поступает на дешифратор, и затем на мультиплексор, который управляет катодами семисегментного индикатора. На аноды каждого из 4 индикаторов напряжение подается поочередно, с частотой незаметной глазу. Реализовать это можно с помощью автомата с конечным числом состояний, но в текущей ситуации проще взять 2 меняющихся бита какого-то из счетчиков, и подать их на дешифратор и затем мультиплексор. Это и даст поочередное переключение 4 индикаторов. В качестве дополнения, не требуемого в задании, были реализованы клавиши для настройки (прибавляющие при нажатии минуту или секунду).

## 2. Описание устройства с помощью VHDL

Описанный в предыдущем пункте функционал реализуется с помощью следующего описания устройства на языке VHDL:

```
library IEEE;--подключение библиотек
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;
```

```
entity digital_clock is --Декларация entity и описание портов
```

```
Port (
    clk : in STD_LOGIC;          --описываем входы и выходы блока
    reset : in STD_LOGIC;
    plus_sec : in STD_LOGIC;
    plus_min : in STD_LOGIC;
    seg : out STD_LOGIC_VECTOR (6 downto 0);
    dig : out STD_LOGIC_VECTOR (3 downto 0)
);
```

```
end digital_clock;
```

```
-- архитектурное тело
```

```
architecture Behavioral of digital_clock is --Архитектура Behavioral для интерфейса digital_clock
```

```
signal cnt: unsigned(27 downto 0); --внутренние сигналы
```

```
signal sec_e: unsigned(3 downto 0);
```

```
signal sec_d: unsigned(3 downto 0);
```

```
signal min_e: unsigned(3 downto 0);
```

```
signal min_d: unsigned(3 downto 0);
```

```
-- signal Q1, Q2, Q3 : std_logic;
```

```
function f_num_2_7seg (num : in unsigned(3 downto 0))
```

```
--функция, получающая на вход значение счетчика типа unsigned
```

```
return std_logic_vector is variable seg7 : std_logic_vector(6 downto 0);
```

```
--возвращающая значение типа std_logic_vector,
```

```
--необходимое для управления катодами семисегментного индикатора
```

```
begin
```

```
    if num = X"0" then seg7 := b"1000000";
    elsif num = X"1" then seg7 := b"1111001";
    elsif num = X"2" then seg7 := b"0100100";
    elsif num = X"3" then seg7 := b"0110000";
    elsif num = X"4" then seg7 := b"0011001";
    elsif num = X"5" then seg7 := b"0010010";
    elsif num = X"6" then seg7 := b"0000010";
    elsif num = X"7" then seg7 := b"1111000";
    elsif num = X"8" then seg7 := b"0000000";
    elsif num = X"9" then seg7 := b"0010000";
    end if;
    return std_logic_vector(seg7);
```

```
end;
```

```
begin
```

```
process(clk, reset, plus_sec, plus_min) --процесс чувствителен к изменению сигналов clk, reset, plus_sec, plus_min
```

```
begin
```

```
if reset='1' then --проверка сброса
```

```
    cnt <= (others => '0');
    sec_e <= (others => '0');
    sec_d <= (others => '0');
    min_e <= (others => '0');
    min_d <= (others => '0');
    Dig <= b"1111";
```

```
elsif rising_edge(CLK) then --если пришел передний фронт тактового сигнала
```

```
    if cnt=to_unsigned(100_000_000,28) then cnt <= (others => '0');--считаем до 100 миллионов()
```

```
--if cnt=to_unsigned(100,26) then cnt <= (others => '0'); --считаем до 100 миллионов
```

```
    if sec_e=9 then sec_e <= x"0";--счетчик единиц секунд, если уже имеет значение 9, сброс и передача импульса следу
```

```
    if sec_d=5 then sec_d <= x"0";
```

```
    if min_e=9 then min_e <= x"0";
```

```
    if min_d=5 then min_d <= x"0";
```

```

        else min_d <= min_d + 1; end if;
        else min_e <= min_e + 1; end if;
        else sec_d <= sec_d + 1; end if;
        else sec_e <= sec_e + 1; end if; --иначе (если на счетчике единиц не 9), увеличиваем значение счетчика на 1.
    else cnt <= cnt + 1; end if; --аналогично
    case to_integer(cnt(14 downto 13)) is --в качестве переменной, которая будет меняться и быть аргументом для case,
        when 0 => Dig <= b"0111"; seg <= (f_num_2_7seg(sec_e)); --вывод единиц секунд
        when 1 => Dig <= b"1011"; seg <= (f_num_2_7seg(sec_d)); -- вывод десятков секунд
        when 2 => Dig <= b"1101"; seg <= (f_num_2_7seg(min_e)); --аналогично единицы минут
        when 3 => Dig <= b"1110"; seg <= (f_num_2_7seg(min_d)); -- десятки минут
        when others => Dig <= b"1111"; seg <= (others => '1');
    end case;
end if;
if plus_sec='1' then sec_e <= sec_e + 1; end if; --если нажали кнопку plus_sec, увеличить значение счетчика секунд
if plus_min='1' then min_e <= min_e + 1; end if; --аналогично
end process;
end Behavioral;

```

Данный код полностью описывает логику требуемого устройства.

### 3. Симуляция устройства

Чтобы проверить, насколько полученное описание хорошо работает, нужно запустить симуляцию. Для ускорения симуляции был изменен кусок кода со счетчиком, считающим до 100 миллионов.

```

--if cnt=to_unsigned(100000000,26) then cnt <= (others => '0'); --считаем до 100 миллионов
if cnt=to_unsigned(100,26) then cnt <= (others => '0'); --считаем до 100

```

Так же для симуляции необходимо создать файл симуляции (Test bench) SIMM.vhd на языке VHDL, содержимое которого можно увидеть ниже:

```

--тестовый файл для симуляции устройства
--подключение библиотек
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;

entity tb_sim is
end tb_sim;

architecture Behavioral of tb_sim is
    COMPONENT delitel -- декларация компонента для UUT
    PORT(
        clk : in STD_LOGIC;          --описываем входы и выходы блока
        reset : in STD_LOGIC;
        plus_sec : in STD_LOGIC;
        plus_min : in STD_LOGIC
    );
    END COMPONENT;

    signal clk : std_logic := '0'; --декларация сигналов и присвоение им значения "0"
    signal reset : std_logic := '0'; --декларация сигналов и присвоение им значения "0"
    signal plus_sec : std_logic := '0';
    signal plus_min : std_logic := '0';

begin

```

```

uut: delitel --конкретизация компонента для Unit Under Test (UUT)
PORT MAP ( --подключение выводов
    clk => clk, reset => reset, plus_sec => plus_sec, plus_min => plus_min
);

clock: process --создание тактового сигнала
begin
    clk <= '0'; wait for 5 ns;
    clk <= '1'; wait for 5 ns;
end process;

plussec : process --создание тестового сигнала(эмуляция четырех нажатий на кнопку plus_sec)
begin
    plus_sec <= '0'; wait for 50 ms;
    plus_sec <= '1'; wait for 5 ns;
    plus_sec <= '0'; wait for 5 ns;
    plus_sec <= '1'; wait for 5 ns;
    plus_sec <= '0'; wait for 5 ns;
    plus_sec <= '1'; wait for 5 ns;
    plus_sec <= '0'; wait for 5 ns;
    plus_sec <= '1'; wait for 5 ns;
end process;

plusmin : process --создание тестового сигнала(эмуляция четырех нажатий на кнопку plus_min)
begin
    plus_min <= '0'; wait for 50 ms;
    plus_min <= '1'; wait for 5 ns;
    plus_min <= '0'; wait for 5 ns;
    plus_min <= '1'; wait for 5 ns;
    plus_min <= '0'; wait for 5 ns;
    plus_min <= '1'; wait for 5 ns;
    plus_min <= '0'; wait for 5 ns;
    plus_min <= '1'; wait for 5 ns;
end process;

reseting : process--создание тестового сигнала(эмуляция нажатия на кнопку reset)
begin
    reset <= '0'; wait for 150 ns;
    reset <= '1'; wait for 5 ns;
    reset <= '0'; wait for 100 ms;
end process;

end Behavioral;

```

После запуска симуляции были получены временные диаграммы, изображенные на рисунках 2, 3, 4, 5, 6, 7

В начальный момент времени значения всех счетчиков не определены. При сбросе с помощью сигнала reset значения счетчиков обнуляются, после чего начинается счет посредством счетчика cnt. Досчитав до 100(вообще он должен до 100 миллионов считать, но для симуляции ждать такое значение бессмысленно), счетчик сбрасывается, передавая импульс следующему счетчику, считающего единицы секунд. Он в свою очередь считает до 9, и тактует счетчик десятков секунд, считающего до 5. Так же, можно просмотреть момент, когда при нажатии кнопок, прибавляющих по секунде и минуте соответственно, происходит изменение значения счетчиков.

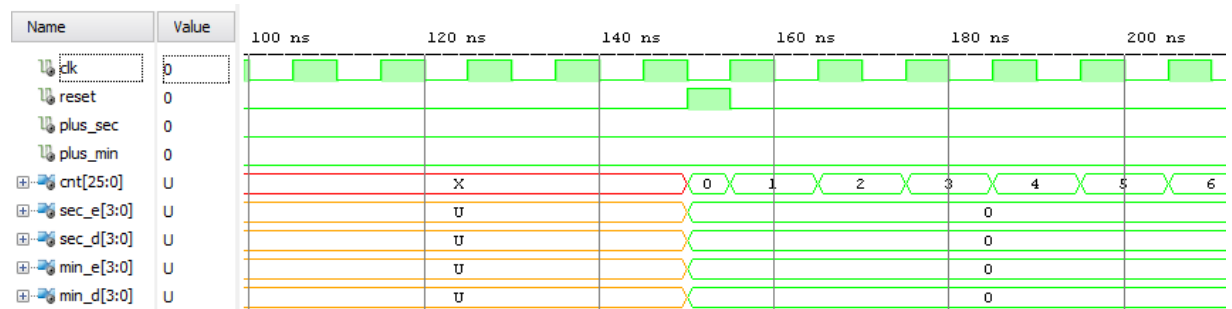


Рисунок 2 – Нулевой момент времени

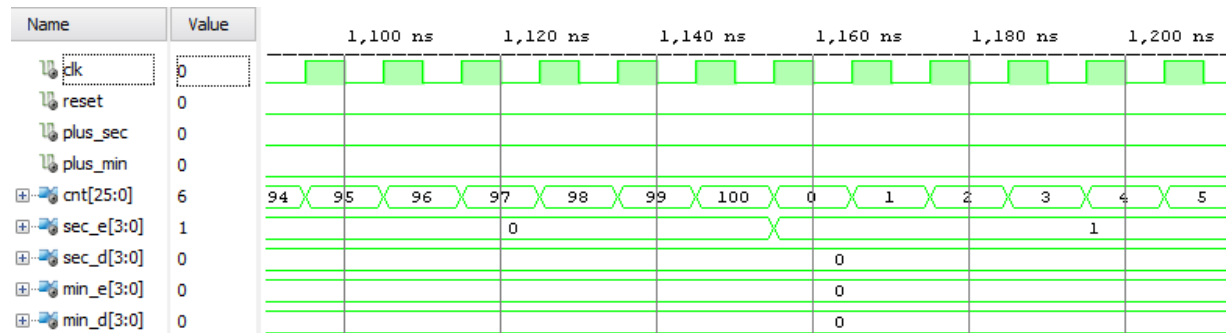


Рисунок 3 – Счетчик единиц секунд при приходе импульса от делителя тактовой частоты

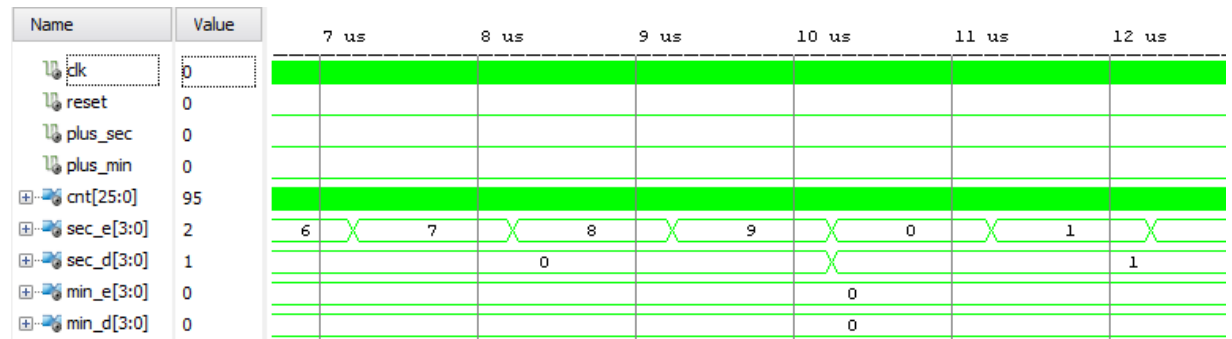


Рисунок 4 – Переключение счетчика десятков секунд счетчиком единиц секунд

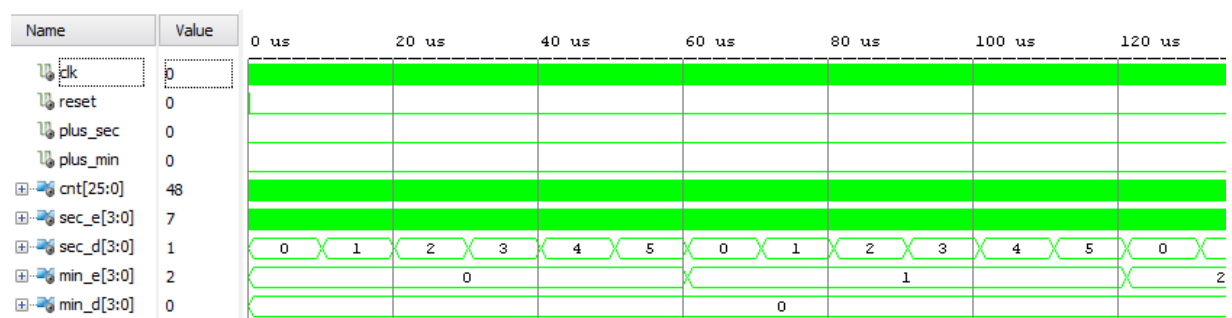


Рисунок 5 – Переключение счетчика единиц минут счетчиком десятков секунд

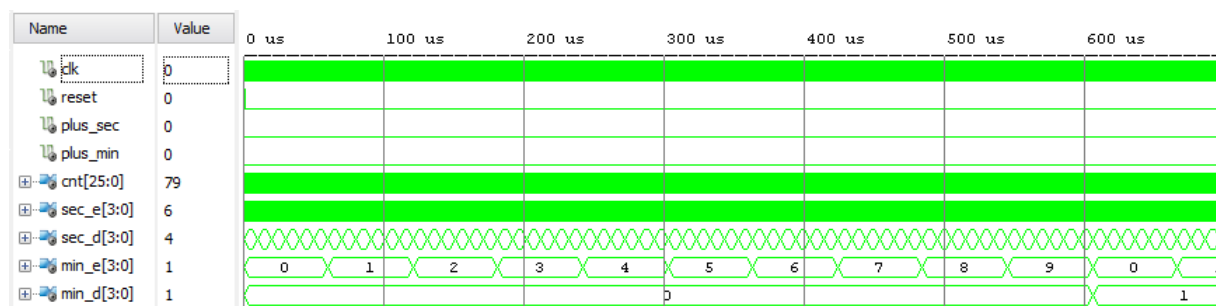


Рисунок 6 – Переключение счетчика десятков минут счетчиком единиц минут

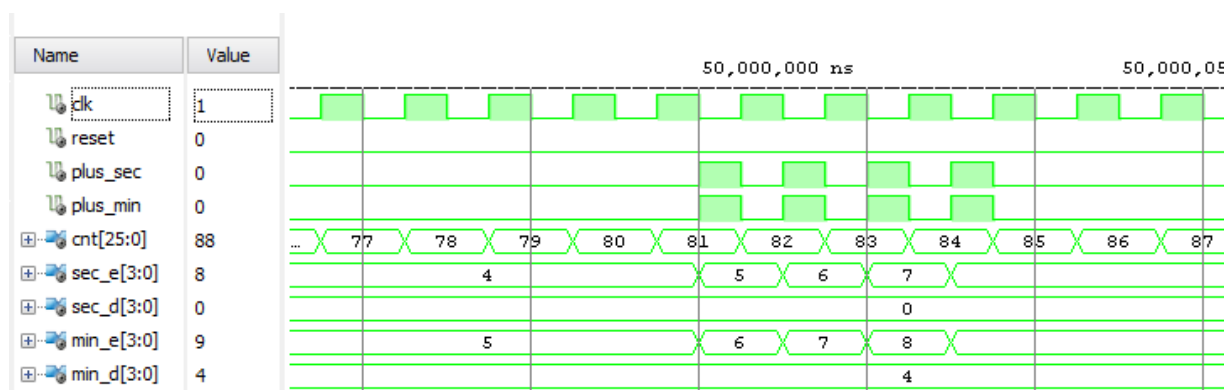


Рисунок 7 – Нажатие plus\_sec и plus\_min

Таким образом, симуляцию можно считать успешной.

#### 4. Синтез цифрового устройства

Для синтеза необходимо задать временные ограничения. Создаем файл временных ограничений, в котором нужно задать частоту тактового сигнала, в окне Primary Clocks. Так же можно задать параметры и для Generated Clocks, но особого смысла в этом нет.

В результате синтеза была получена следующая схема(рисунок 8)



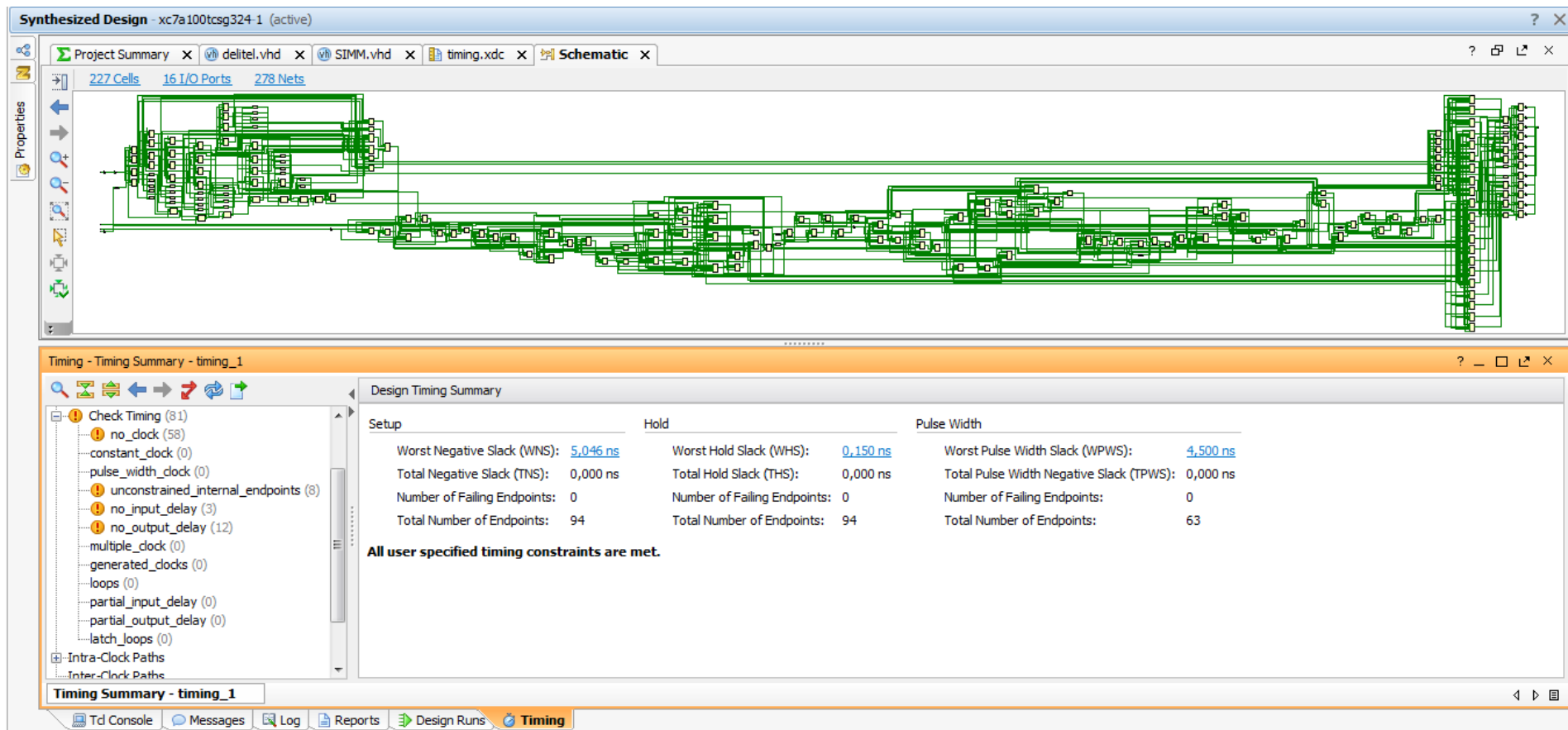


Рисунок 8 – Синтезированная схема и сводка по времени

Система отмечает отсутствие ограничений Input и Output, а также группы no clock и unconstrained elements. Причина этого в том, что все счетчики тактируется выходным сигналом другого счетчика, который не указан в качестве тактового.

Так же на этом этапе необходимо указать внешние порты ввода-вывода для размещения схемы внутри ПЛИС. Сделать это нужно вручную.

Name	Direction	N...	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Driv...	Slew Type	Pull T...	Off-Chip Termination	IN_TERM
All ports (16)													
dig (4)	OUT			<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
dig[3]	OUT		N5	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
dig[2]	OUT		M3	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
dig[1]	OUT		M6	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
dig[0]	OUT		N6	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg (8)	OUT			<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[7]	OUT		M4	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[6]	OUT		L6	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[5]	OUT		M2	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[4]	OUT		K3	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[3]	OUT		L4	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[2]	OUT		L5	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[1]	OUT		N1	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
seg[0]	OUT		L3	<input checked="" type="checkbox"/>		34 LVCMOS33*	3.300		12	SLOW	NONE	FP_VTT_50	
Scalar ports (4)													
clk	IN		E3	<input checked="" type="checkbox"/>		35 LVCMOS33*	3.300				NONE	NONE	
plus_min	IN		F15	<input checked="" type="checkbox"/>		15 LVCMOS33*	3.300				NONE	NONE	
plus_sec	IN		V10	<input checked="" type="checkbox"/>		14 LVCMOS33*	3.300				NONE	NONE	
reset	IN		T16	<input checked="" type="checkbox"/>		14 LVCMOS33*	3.300				NONE	NONE	

Рисунок 9 – Расположение портов ввода-вывода

## 5. Реализация

Следующий шаг разработки устройства – это реализация. После завершения процесса реализации нужно проверить выполнение временных ограничений (Report Timing Summary)(рисунок 10)

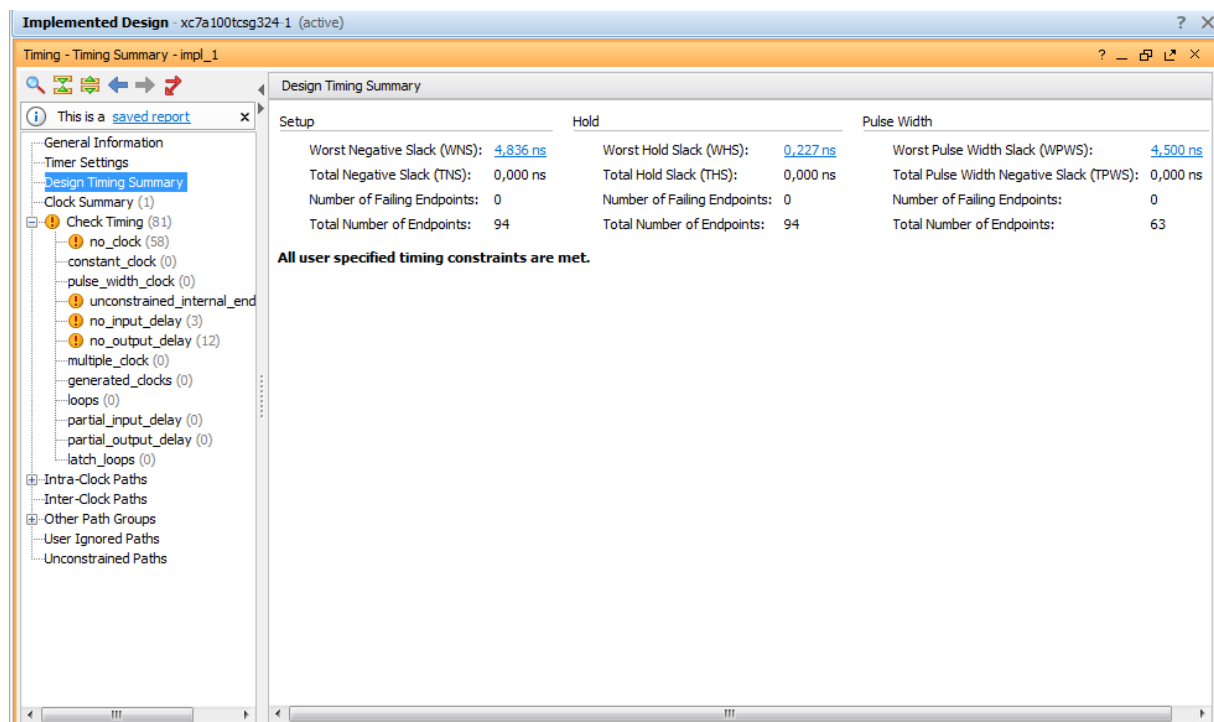


Рисунок 10 – временная сводка после реализации

## 6. Программирование ПЛИС

Для программирования ПЛИС нужно выбрать Generate Bitstream на левой панели (Flow Navigator), а после завершения процесса открыть Hardware Manager, подключить кабель USB к отладочной плате (разъем PROG) и затем к компьютеру. Включить плату (переключатель POWER на плате), а также проверить правильность установки перемычек JP1 и JP2.

## 7. Проверка функционирования устройства на отладочной плате

На отладочной плате устройство работало правильно, все описанные функции работают.