|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 2 |

**Название:**

Проектирование цифровых устройств на основе ПЛИС

**Дисциплина:**  Основы проектирования устройств ЭВМ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ6-62Б |  |  | С.В. Астахов | |
|  | (Группа) |  | (Подпись, дата) | | (И.О. Фамилия) |
|  |  |  |  | |  |
| Преподаватель |  |  |  | |  |
|  |  |  | (Подпись, дата) | | (И.О. Фамилия) |

Москва, 2022

**Вариант 1**

**Введение**

**Цель работы:** закрепление на практике теоретических сведений, полученных при изучении методики проектирования цифровых устройств на основе программируемых логических интегральных схем (ПЛИС), получение необходимых навыков работы с системой автоматизированного проектирования ISE WebPack устройств на основе ПЛИС фирмы Xilinx, изучение аппаратных и программных средств моделирования, макетирования и отладки устройств на основе ПЛИС.

**Ход работы**

Условие по варианту показаны в таблице 1:

Таблица 1 – условия по варианту

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант | Набор | State0 | State1 | State2 | State3 |
| 1 | XC3S200 | 00 | 01 | 10 | 11 |

**Задание 1**

Выполнить кодирование состояний автомата в соответствии с индивидуальным вариантом.

Функциональная схема разрабатываемого устройства показана на рисунке 1.

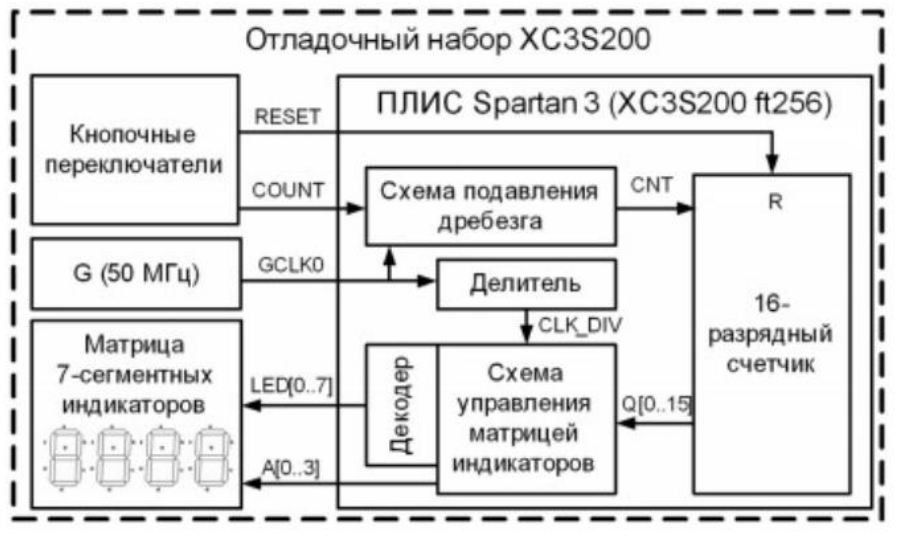
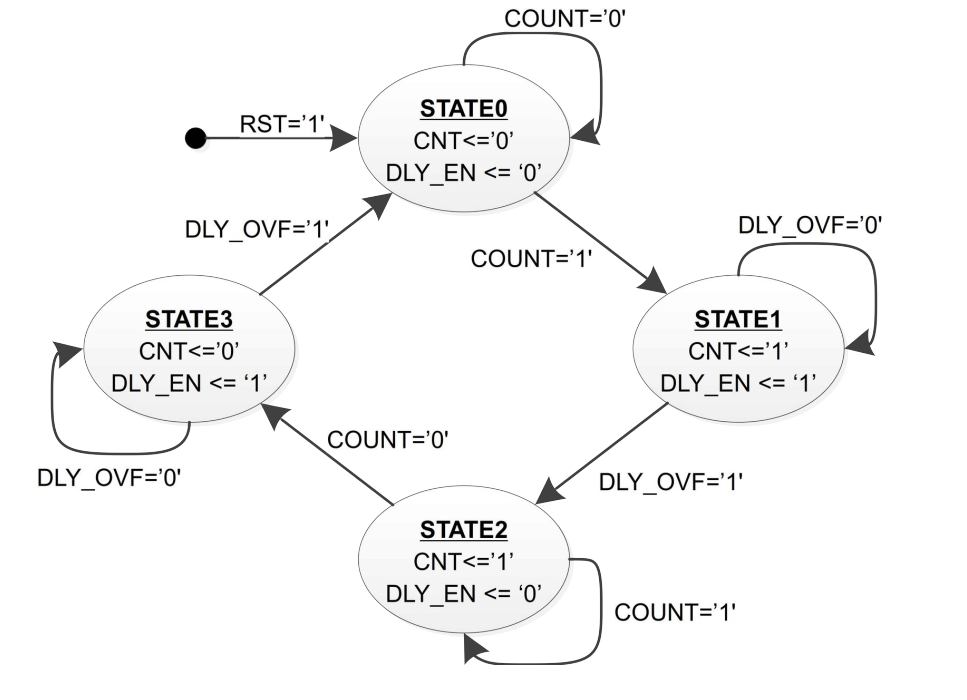


Рисунок 1 - функциональная схема разрабатываемого устройства

Диаграмма состояний автомата подавления дребезга представлена на рисунке 2.

Рисунок 2 - диаграмма состояний автомата подавления дребезга



Функциональная схема устройства показана на рисунке 3.



Рисунок 3 – функциональная схема устройства

В таблице 2 представлены состояния выходов в зависимости от состояния автомата.

Таблица 2 – таблица состояний выходов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Состояние | State0 | State1 | State2 | State3 |
| Двоичный код состояния S(1),S(0) | 00 | 01 | 10 | 11 |
| CNT | 0 | 1 | 1 | 0 |
| DLY\_EN | 0 | 1 | 0 | 1 |

Из таблицы можно получить функции, задающие CNT и DLY\_EN:

CNT = S(1) xor S(0)

DLY\_EN = S(0)

Затем составим таблицу состояний SN.

Таблица 3 – таблица состояний SN

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| COUNT | DLY\_OVF | S1(t) | S0(t) | S1(t+1) | S0(t+1) | SN(1) | SN(0) | Описание события |
| 0 | x | 0 | 0 | 0 | 0 | 0 | 0 | Ожидание нажатия кнопки |
| 1 | X | 0 | 0 | 0 | 1 | 0 | 1 | Нажатие кнопки |
| X | 0 | 0 | 1 | 0 | 1 | 0 | 1 | Ожидание окончания счета |
| X | 1 | 0 | 1 | 1 | 0 | 1 | 0 | Конец счета |
| 1 | X | 1 | 0 | 1 | 0 | 1 | 0 | Ожидание отпускания |
| 0 | X | 1 | 0 | 1 | 1 | 1 | 1 | Отпускание кнопки |
| X | 0 | 1 | 1 | 1 | 1 | 1 | 1 | Ожидание окончания счета |
| x | 1 | 1 | 1 | 0 | 0 | 0 | 0 | Конец счета |

Составим карты Карно для SN(1) и SN(0). Они представлены таблицами 3 и 4.

Таблица 3 – карта Карно для SN(1).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | S1, S0 | | | |
| C,D | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 1 | 1 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | 0 | 1 | 0 | 1 |
| 10 | 0 | 0 | 1 | 1 |

SN(1) = (S1 \* not-S0) v (S1 \* S0 \* not-DLY\_OVF) v (not-S1 \* S0 \* DLY\_OVF)

Таблица 4 – карта Карно для SN(0).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | S1, S0 | | | |
| C,D | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 1 | 1 |
| 01 | 0 | 0 | 0 | 1 |
| 11 | 1 | 0 | 0 | 0 |
| 10 | 1 | 1 | 1 | 0 |

SN(0) = (S0 \* not-DLY\_OVF) v (S1 \* not-S0 \* not-COUNT) v (not-S1 \* not-S0 \* COUNT)

**Задание 2**

Разработать текстовое описание модуля в соответствии с полученными функциями DLY\_EN, CNT, SN(0), SN(1). Собрать на основе полученного описания проект в САПР Xilinx ISE.

Код программы по варианту:

-- Пример модуля подавления дребезга 10 мс.

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

Entity lab2\_example IS

PORT (

RST: IN STD\_LOGIC; --Системный сигнал сброса

CLK: IN STD\_LOGIC; --Сигнал синхронизации

COUNT: IN STD\_LOGIC; --Сигнал кнопки с дребезгом

CNT: OUT STD\_LOGIC --Сигнал кнопки, очищенный от дребезга

);

END lab2\_example;

ARCHITECTURE behavioral OF lab2\_example IS

-- Кодируем состояния в соответствии с вариантом

CONSTANT STATE0: STD\_LOGIC\_VECTOR (1 downto 0) := "00";

CONSTANT STATE1: STD\_LOGIC\_VECTOR (1 downto 0) := "01";

CONSTANT STATE2: STD\_LOGIC\_VECTOR (1 downto 0) := "10";

CONSTANT STATE3: STD\_LOGIC\_VECTOR (1 downto 0) := "11";

-- Состояние автомата в момент времени t

SIGNAL S: STD\_LOGIC\_VECTOR (1 downto 0);

-- Состояние автомата в момент времени t+1

SIGNAL SN: STD\_LOGIC\_VECTOR (1 downto 0);

SIGNAL COUNTER: integer; -- Счетчик 2^20

SIGNAL DLY\_OVF: STD\_LOGIC; -- Сигнал "Завершение счета"

SIGNAL DLY\_EN: STD\_LOGIC; -- Сигнал разрешения работы счетчика

BEGIN

-- Память состояний

FSM\_STATE\_inst: PROCESS (CLK)

BEGIN

IF (CLK='1' and CLK'event) THEN

IF (RST='1') THEN

S <= STATE0;

ELSE

S <= SN;

END IF;

END IF;

END PROCESS;

-- Комбинационная схема для выработки сигналов CNT и DLY\_EN (по индивидуальному варианту)

CNT <= S(1) xor S(0);

DLY\_EN <= S(0);

--Комбинационные схемы для определения следующего состояния (по индивидуальному варианту)

SN(0) <= (not DLY\_OVF and S(0)) or (S(1) and not S(0) and not COUNT) or (not S(1) and not S(0) and COUNT);--пример 1

SN(1) <= (S(1) and not S(0)) or (S(1) and S(0) and not DLY\_OVF) or (not S(1) and S(0) and DLY\_OVF);

-- Описание счетчика

COUNTER\_inst: PROCESS (CLK)

BEGIN

IF (CLK='1' and CLK'event) THEN

IF (RST='1' or DLY\_EN = '0') THEN

COUNTER <= 0;

ELSE

COUNTER <= COUNTER + 1;

END IF;

END IF;

END PROCESS;

DLY\_OVF <= '1' WHEN COUNTER = 2\*\*7-1 ELSE '0'; --Длительность задержки

END Behavioral;

**Задание 3**

В интегрированном редакторе тестов САПР Xilinx ISE разработать тест для полученного устройства и выполнить моделирование его работы в симуляторе Modelsim.

На рисунке 4 показаны входные исходные для теста в графическом представлении.

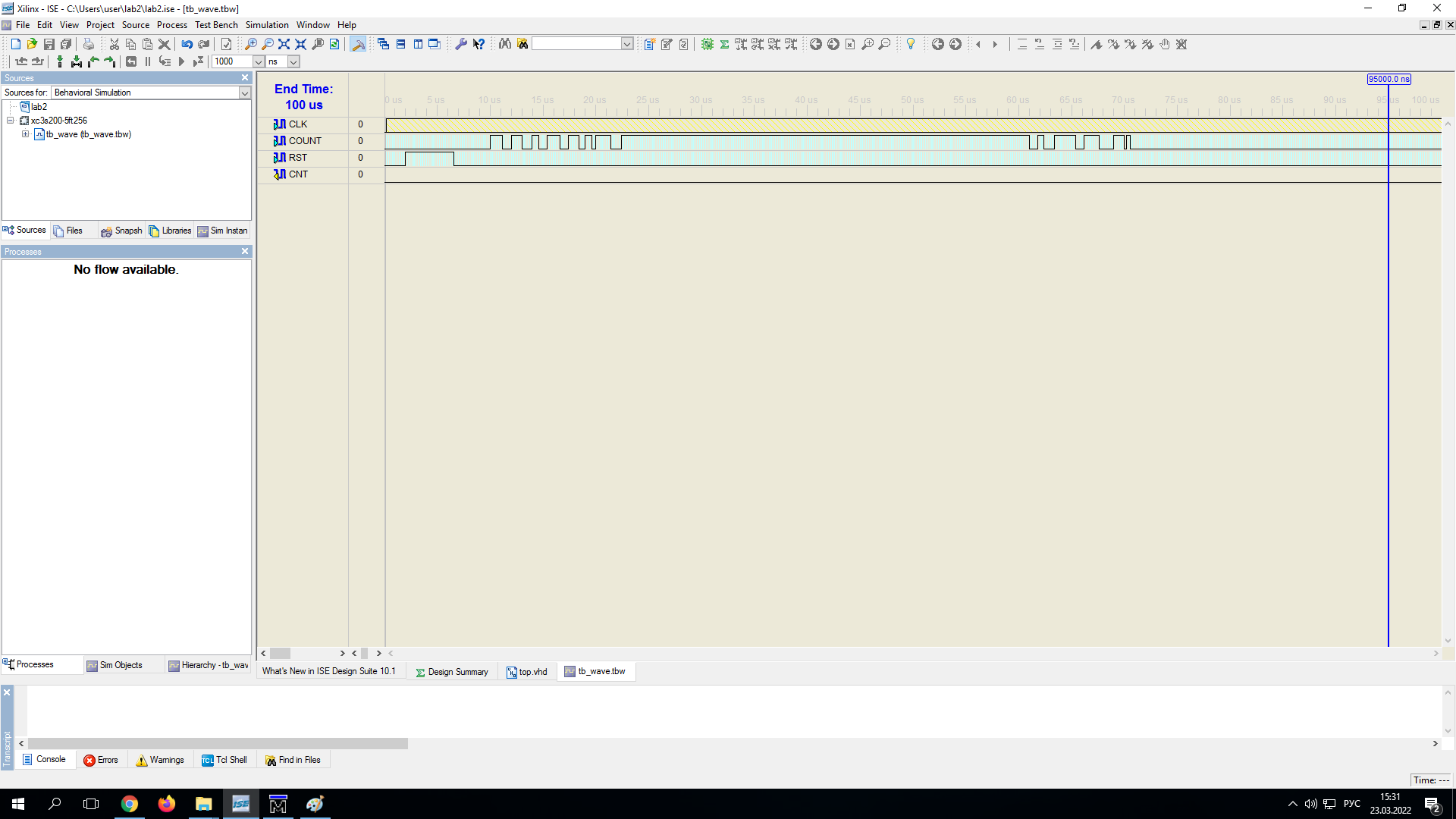


Рисунок 4 – исходные данные теста

Результаты теста показаны на рисунке 5.

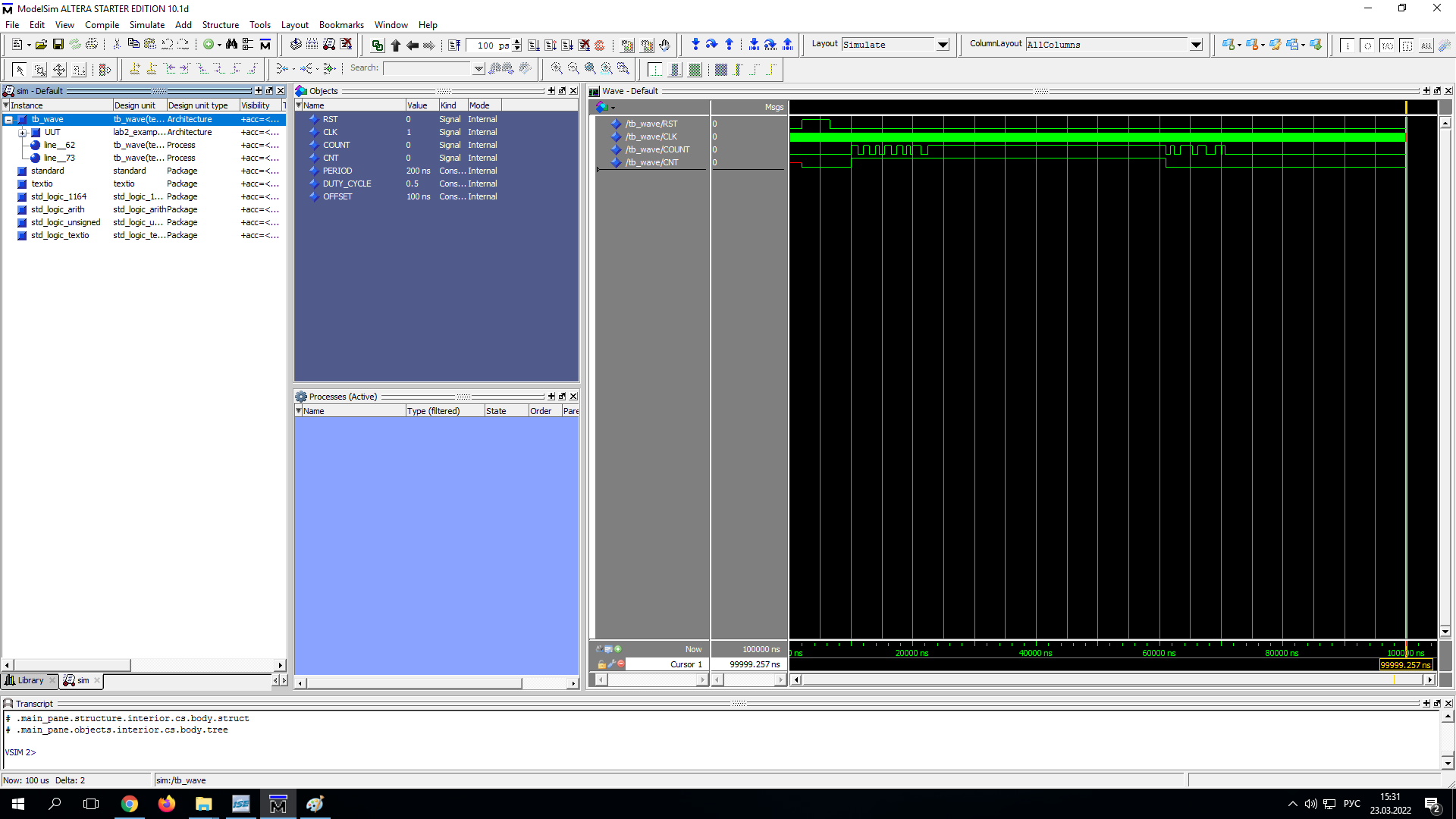


Рисунок 5 – результаты теста

Как видно из результатов теста – устройство работает корректно.

**Задание 4**

Разработать устройство управления, принимающее 16-разрядное информационное слово Q[0..15] и управляющее их последовательной выдачей по шине D[0..3] на декодер 7-сегментных индикаторов.

Исходный код модуля:

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_arith.ALL;

ENTITY Seven\_Segment\_Driver IS

PORT(

CLK : IN std\_logic;

CLK\_DIV : IN std\_logic;

Q : IN std\_logic\_vector(15 DOWNTO 0);

RST : IN std\_logic;

D : OUT std\_logic\_vector(3 DOWNTO 0);

A : OUT std\_logic\_vector(3 DOWNTO 0));

END ENTITY Seven\_Segment\_Driver;

ARCHITECTURE Struct OF Seven\_Segment\_Driver IS

--Internal Anode

SIGNAL A\_int : std\_logic\_vector(3 DOWNTO 0);

BEGIN

--Output Anode

A <= A\_int;

A\_drive: PROCESS (CLK,RST)

BEGIN

IF (RST = '1') THEN

A\_int<="1110";

ELSIF (CLK'EVENT AND CLK='1') THEN

IF (CLK\_DIV='1') THEN

A\_int(3)<=A\_int(2);

A\_int(2)<=A\_int(1);

A\_int(1)<=A\_int(0);

A\_int(0)<=A\_int(3);

END IF;

END IF;

END PROCESS A\_drive;

D(0) <= (Q(0) AND NOT(A\_int(0)))

OR (Q(4) AND NOT(A\_int(1)))

OR (Q(8) AND NOT(A\_int(2)))

OR (Q(12) AND NOT(A\_int(3)));

D(1) <= (Q(1) AND NOT(A\_int(0)))

OR (Q(5) AND NOT(A\_int(1)))

OR (Q(9) AND NOT(A\_int(2)))

OR (Q(13) AND NOT(A\_int(3)));

D(2) <= (Q(2) AND NOT(A\_int(0)))

OR (Q(6) AND NOT(A\_int(1)))

OR (Q(10) AND NOT(A\_int(2)))

OR (Q(14) AND NOT(A\_int(3)));

D(3) <= (Q(3) AND NOT(A\_int(0)))

OR (Q(7) AND NOT(A\_int(1)))

OR (Q(11) AND NOT(A\_int(2)))

OR (Q(15) AND NOT(A\_int(3)));

END ARCHITECTURE Struct;

На рисунке 6 показаны временные диаграммы, полученные при тестировании данного модуля.

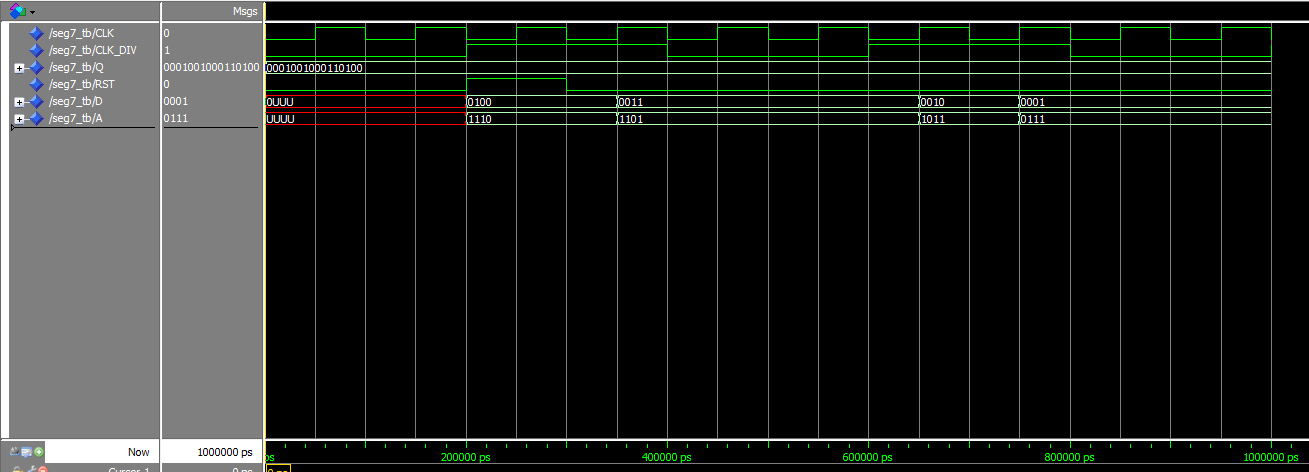


Рисунок 6 – тестирование модуля управления

**Задние 5**

Разработать поведенческое VHDL описание схемы преобразования четырехразрядного информационного кода D[0..3] в код активизации 7-сегментного индикатора LED[0..7].

Исходный код:

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY led\_decode IS

PORT (

DH: IN STD\_LOGIC\_VECTOR (3 DOWNTO 0);

SEG\_DATA: OUT STD\_LOGIC\_VECTOR (7 DOWNTO 0)

);

END led\_decode;

ARCHITECTURE Behavioral OF led\_decode IS

BEGIN

PROCESS (DH)

BEGIN

CASE DH IS

WHEN "0000" => SEG\_DATA <= "10000001";

WHEN "0001" => SEG\_DATA <= "11001111";

WHEN "0010" => SEG\_DATA <= "10010010";

WHEN "0011" => SEG\_DATA <= "10000110";

WHEN "0100" => SEG\_DATA <= "11001100";

WHEN "0101" => SEG\_DATA <= "10100100";

WHEN "0110" => SEG\_DATA <= "10100000";

WHEN "0111" => SEG\_DATA <= "10001111";

WHEN "1000" => SEG\_DATA <= "10000000";

WHEN "1001" => SEG\_DATA <= "10000100";

WHEN "1010" => SEG\_DATA <= "10001000";

WHEN "1011" => SEG\_DATA <= "11100000";

WHEN "1100" => SEG\_DATA <= "10110001";

WHEN "1101" => SEG\_DATA <= "11000010";

WHEN "1110" => SEG\_DATA <= "10110000";

WHEN "1111" => SEG\_DATA <= "10111000";

WHEN OTHERS => NULL;

END CASE;

END PROCESS;

END Behavioral;

**Задание 6**

В редакторе схем САПР ISE добавить описание основного модуля.

Исходный код:

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY main IS

PORT ( CLK : IN std\_logic;

COUNT : IN std\_logic;

RESET : IN std\_logic;

A : OUT std\_logic\_vector (3 DOWNTO 0);

LED : OUT std\_logic\_vector (7 DOWNTO 0));

END main;

ARCHITECTURE Behavioral OF main IS

COMPONENT Seven\_Segment\_Driver

PORT ( CLK : IN std\_logic;

CLK\_DIV : IN std\_logic;

RST : IN std\_logic;

Q : IN std\_logic\_vector (15 DOWNTO 0);

D : OUT std\_logic\_vector (3 DOWNTO 0);

A : OUT std\_logic\_vector (3 DOWNTO 0));

END COMPONENT;

COMPONENT led\_decode

PORT ( DH : IN std\_logic\_vector (3 DOWNTO 0);

SEG\_DATA : OUT std\_logic\_vector (7 DOWNTO 0));

END COMPONENT;

COMPONENT lab2\_example

PORT ( RST : IN std\_logic;

CLK : IN std\_logic;

COUNT : IN std\_logic;

CNT : OUT std\_logic);

END COMPONENT;

SIGNAL CNT\_int,CNT\_ff,CNT\_RISE:std\_logic;

SIGNAL COUNTER: integer;

SIGNAL COUNTER\_OVF: std\_logic;

-- Main counter

SIGNAL MAIN\_COUNTER: std\_logic\_vector(15 DOWNTO 0);

SIGNAL D\_int : std\_logic\_vector(3 DOWNTO 0);

BEGIN

ssd\_inst : Seven\_Segment\_Driver

PORT MAP (CLK=>CLK,

CLK\_DIV=> COUNTER\_OVF,

Q(15 DOWNTO 0)=>MAIN\_COUNTER,

RST=>RESET,

D(3 DOWNTO 0)=>D\_INT,

A(3 DOWNTO 0)=>A

);

led\_decode\_inst : led\_decode

PORT MAP (DH(3 DOWNTO 0)=>D\_INT,

SEG\_DATA(7 DOWNTO 0)=>LED

);

lab2\_example\_inst : lab2\_example

PORT MAP (CLK=>CLK,

COUNT=>COUNT,

RST=>RESET,

CNT=>CNT\_int);

-- Описание делителя частоты

COUNTER\_inst: PROCESS (CLK)

BEGIN

IF (CLK='1' and CLK'event) THEN

IF (RESET='1' or COUNTER\_OVF='1') THEN

COUNTER <= 0;

ELSE

COUNTER <= COUNTER + 1;

END IF;

END IF;

END PROCESS;

COUNTER\_OVF <= '1' WHEN COUNTER = 2\*\*16 ELSE '0';

--Детектор фронта сигнала CNT

CNT\_RISE <= '1' WHEN CNT\_int='1' and CNT\_ff='0' ELSE '0';

CNT\_ff\_inst: PROCESS (CLK)

BEGIN

IF (CLK='1' and CLK'event) THEN

IF (RESET='1') THEN

CNT\_ff <= '0';

ELSE

CNT\_ff <= CNT\_int;

END IF;

END IF;

END PROCESS;

--Основной счетчик

MAIN\_COUNTER\_inst: PROCESS (CLK)

BEGIN

IF (CLK='1' and CLK'event) THEN

IF (RESET='1') THEN

MAIN\_COUNTER <= (others => '0');

ELSIF (CNT\_RISE = '1') THEN

MAIN\_COUNTER <= MAIN\_COUNTER + 1;

END IF;

END IF;

END PROCESS;

END BEHAVIORAL;

**Задание 7**

В программе Xilinx PACE создать файл ограничений \*.ucf или добавьте в проект имеющийся main\_xc3s200.ucf.

Исходный код:

#PACE: Start of Constraints generated by PACE

#PACE: Start of PACE I/O Pin Assignments

NET "A<0>" LOC = "D14" ;

NET "A<1>" LOC = "G14" ;

NET "A<2>" LOC = "F14" ;

NET "A<3>" LOC = "E13" ;

NET "CLK" LOC = "T9" ;

NET "COUNT" LOC = "M13" ;

NET "LED<0>" LOC = "N16" ;

NET "LED<1>" LOC = "F13" ;

NET "LED<2>" LOC = "R16" ;

NET "LED<3>" LOC = "P15" ;

NET "LED<4>" LOC = "N15" ;

NET "LED<5>" LOC = "G13" ;

NET "LED<6>" LOC = "E14" ;

NET "LED<7>" LOC = "P16" ;

NET "RESET" LOC = "L14" ;

#PACE: Start of PACE Area Constraints

#PACE: Start of PACE Prohibit Constraints

#PACE: End of Constraints generated by PACE

**Задание 8-9**

В САПР ISE выполнить автоматический синтез технологической схемы, размещение и трассировку полученного устройства на кристалле Spartan3 XC3S200 ft256 (или для Spartan XC3S 500E/1200E, FG320), генерировать файл конфигурации ПЛИС (\*.bin).

Выполнить программирование макетной ПЛИС Spartan3 отладочного набора XC3S200 или Nexys2.

Провести тестирование разработанного устройства.

**Вывод:** Были закреплены на практике знания полученных при изучении методики проектирования цифровых устройств на основе программируемых логических интегральных схем (ПЛИС)**.** Были получены знания и навыки разработки устройства для подавления дребезга и работы с 7-сегментным индикатором (с применением динамической индикации).