Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных средств

Дисциплина: Проектирование цифровых систем на языках описания аппаратуры

ОТЧЕТ
к лабораторной работе №4
на тему

ОПИСАНИЕ И МОДЕЛИРОВАНИЕ ТРИГГЕРОВ И КОНЕЧНЫХ АВТОМАТОВ

Выполнили:
ст. гр. 850702
Турко В. Д.

Проверил:
Санько Н. С.

Минск 2020

1 МОДЕЛИРОВАНИЕ ТРИГГЕРА

1.1 Цель работы

С использованием девятизначного алфавита STD_LOGIC составить VHDL-модель и провести моделирование триггера двумя способами: по логической схеме и по таблице функционирования триггера.

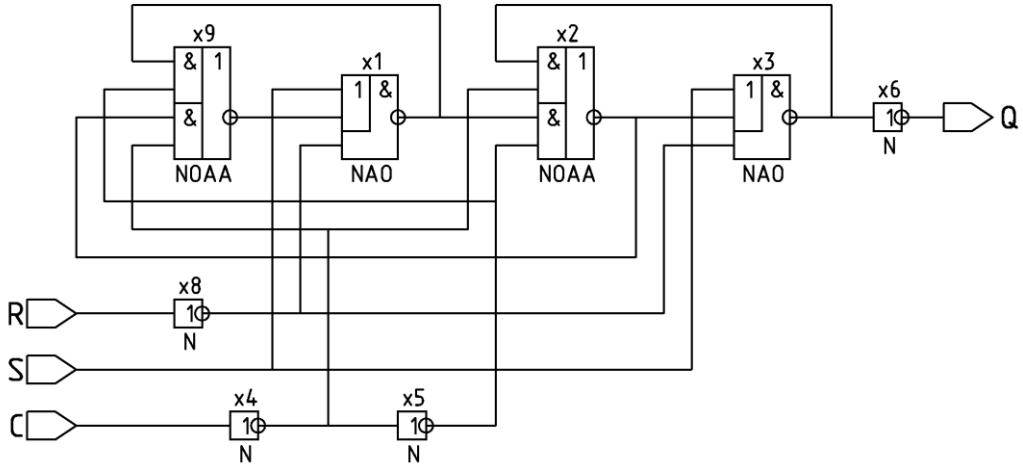


Рисунок 1.1 – Логическая схема

Таблица истинности

R	S	C	Q
1	-	[--]	0
0	1	[--]	1
0	0	[01]	^(Q)
0	0	[1-]	N
0	0	[00]	N

Рисунок 1.2 – Таблица истинности

1.2 vhdl-модель триггера

В таблице 2.1 приведена реализация моделей триггера согласно заданию.

Таблица 2.1 – модели элементов

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE work.lib_package.ALL;

ENTITY TriggerScheme IS
    PORT (
        R, S, C : IN STD_LOGIC;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY TriggerAlgorithm IS
    PORT (
        R, S, C : IN STD_LOGIC;
        Q : INOUT STD_LOGIC);
```

```
        Q : INOUT STD_LOGIC);
END TriggerScheme;
ARCHITECTURE SchemeArch OF TriggerScheme IS

   SIGNAL x : STD_LOGIC_VECTOR(9 DOWNTO 1);

BEGIN
    X9 : NOAA PORT MAP(x(1), x(5), x(2), x(4),
x(9));
    X8 : N PORT MAP(R, x(8));
    X6 : N PORT MAP(x(3), Q);
    X5 : N PORT MAP(x(4), x(5));
    X4 : N PORT MAP(C, x(4));
    X3 : NAO PORT MAP(S, x(2), x(8), x(3));
    X2 : NOAA PORT MAP(x(3), x(4), x(1),
x(5), x(2));
    X1 : NAO PORT MAP(S, x(9), x(8), x(1));
END ARCHITECTURE SchemeArch;
```

```
END TriggerAlgorithm;

ARCHITECTURE AlgorithmArch OF TriggerAlgorithm IS
    FUNCTION rising_edge (SIGNAL s : STD_LOGIC)
        RETURN BOOLEAN IS
    BEGIN
        RETURN s = '1' AND s'event;
    END;

BEGIN
    PROCESS(R, S, C)
    BEGIN
        IF R = '1' THEN
            Q <= '0' after 8 ns;
        ELSIF S = '1' THEN
            Q <= '1' after 8 ns;
        ELSE
            IF rising_edge(C) THEN
                Q <= NOT Q after 13 ns;
            ELSE
                Q <= Q after 2 ns;
            END IF;
        END IF;
    END PROCESS;
END ARCHITECTURE AlgorithmArch;
```

1.3 Моделирование триггера

Тестирующая программа:

```
ENTITY TEST_BENCH IS
END TEST_BENCH;

ARCHITECTURE MainBehavior OF TEST_BENCH IS

    COMPONENT TriggerScheme IS
        PORT (
            R, S, C : IN STD_LOGIC;
            Q : INOUT STD_LOGIC);
    END COMPONENT TriggerScheme;

    COMPONENT TriggerAlgorithm IS
        PORT (
            R, S, C : IN STD_LOGIC;
            Q : INOUT STD_LOGIC);
    END COMPONENT TriggerAlgorithm;
```

```
    SIGNAL r_test : STD_LOGIC;
    SIGNAL S_test : STD_LOGIC;
    SIGNAL c_test : STD_LOGIC;
    SIGNAL q_scheme : STD_LOGIC;
    SIGNAL q_alg : STD_LOGIC;

BEGIN
    p1 : TriggerScheme PORT MAP(r_test, S_test, c_test, q_scheme);
    p2 : TriggerAlgorithm PORT MAP(r_test, S_test, c_test, q_alg);

    PROCESS IS
    BEGIN
        r_test <= '1';
        WAIT FOR interval * 2;
        r_test <= '0';
        WAIT;
    END PROCESS;

    PROCESS IS
    BEGIN
        S_test <= '0';
        WAIT FOR interval;
        S_test <= '1';
        WAIT FOR interval * 4;
        S_test <= '0';
        WAIT;
    END PROCESS;

    PROCESS IS
    BEGIN
        c_test <= '0';
        WAIT FOR interval;
        c_test <= '1';
        WAIT FOR interval;
    END PROCESS;

    PROCESS IS
        VARIABLE count : INTEGER := 8;
    BEGIN
        IF count < 1 THEN
            stop;
        ELSE
            count := count - 1;
            WAIT FOR interval;
        END IF;
    END PROCESS;

END MainBehavior;
```

В результате моделирования получим временную диаграмму, представленную на рис. 2.1.

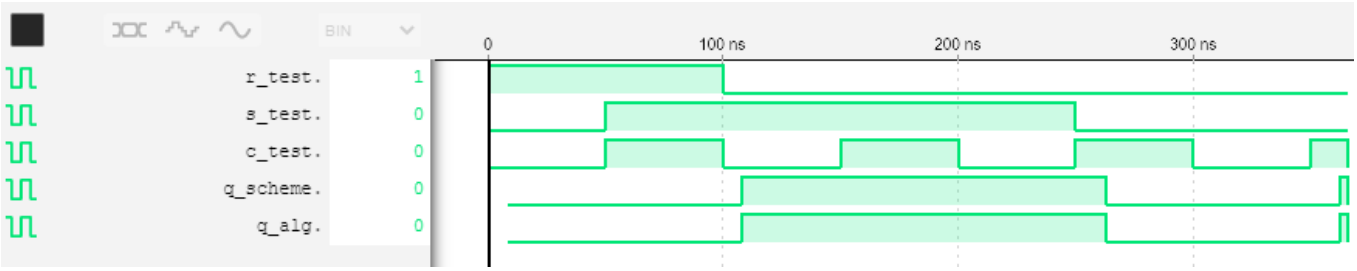


Рисунок 2.1 – Временная диаграмма

2 МОДЕЛИРОВАНИЕ АВТОМАТА МИЛИ

2.1 Цель работы

Составить VHDL-описание конечного автомата Мили, заданного совмещенной таблицей переходов (таблица 2.1).

Таблица 2.1 – модели элементов

Входные сигналы	Состояния			
	a_1	a_2	a_3	a_4
z_1	a_3/w_5	a_2/w_1	a_2/w_2	a_{11}/w_5
z_2	a_4/w_5	a_2/w_5	a_4/w_3	a_3/w_3
z_3	a_3/w_5	a_1/w_4	a_1/w_4	a_{11}/w_5

2.2 VHDL-модель автомата Мили

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY Mealy IS
    PORT (
        clk, rst : IN STD_LOGIC;
        X : IN STD_LOGIC_VECTOR(1 TO 2);
        y : OUT STD_LOGIC_VECTOR(1 TO 3));
END Mealy;

ARCHITECTURE rtl OF Mealy IS

    TYPE T_state IS (a1, a2, a3, a4);
    SIGNAL state : T_state := a1;
    SIGNAL NEXT_state : T_state;
```

```
BEGIN

    manageCase : PROCESS (state, x)
    BEGIN
        IF ((x(1) AND x(2)) = '0') THEN
            CASE state IS
                WHEN a1 =>
                    IF (x(2) = '0') THEN -- '00' || '10'
                        NEXT_state <= a3;
                    ELSE
                        NEXT_state <= a4;
                    END IF;
                    y <= "000";
                WHEN a2 =>
                    CASE x IS
                        WHEN "00" => NEXT_state <= a2; y <= "100";
                        WHEN "01" => NEXT_state <= a2; y <= "000";
                        WHEN "10" => NEXT_state <= a1; y <= "001";
                        WHEN OTHERS => NEXT_state <= state;
                    END CASE;
                WHEN a3 =>
                    CASE x IS
                        WHEN "00" => NEXT_state <= a2; y <= "011";
                        WHEN "01" => NEXT_state <= a4; y <= "010";
                        WHEN "10" => NEXT_state <= a1; y <= "001";
                        WHEN OTHERS => NEXT_state <= state;
                    END CASE;
                WHEN a4 =>
                    IF (x(2) = '0') THEN -- '00' || '10'
                        NEXT_state <= a1; y <= "000";
                    ELSE
                        NEXT_state <= a4; y <= "010";
                    END IF;
            END CASE;
        END IF;
    END PROCESS manageCase;
    state <= a1 WHEN rst = '1' ELSE
       NEXT_state WHEN clk'event AND clk = '1' ELSE
       state;
END rtl;
```

2.3 Моделирование

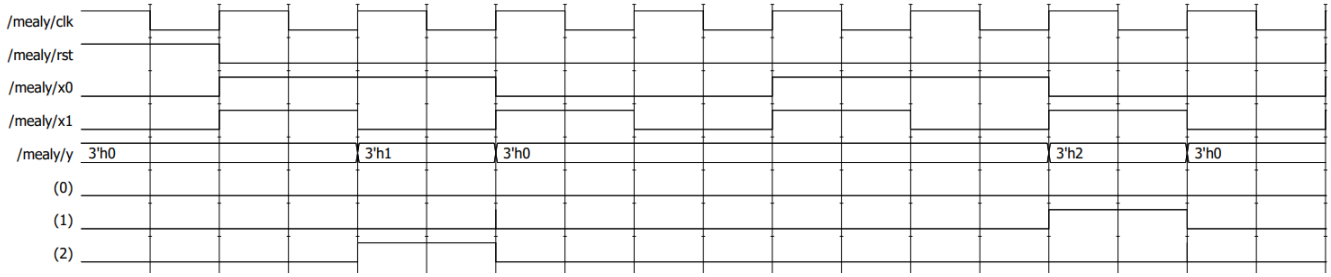
Тестирующая программа:

```
ENTITY SCHEME_TEST IS
END SCHEME_TEST;
```

```
ARCHITECTURE MainBehavior OF SCHEME_TEST IS
    COMPONENT Mealy IS
        PORT (
            clk, rst : IN STD_LOGIC;
            X : IN STD_LOGIC_VECTOR(1 TO 2);
            y : OUT STD_LOGIC_VECTOR(1 TO 3));
    END COMPONENT Mealy;

    SIGNAL clk : STD_LOGIC;
    SIGNAL rst : STD_LOGIC := '1';
    SIGNAL X : STD_LOGIC_VECTOR(1 TO 2) := (OTHERS => '0');
    SIGNAL y : STD_LOGIC_VECTOR(1 TO 3);

    CONSTANT clk_period : TIME := 20 ns;
BEGIN
    p1 : Mealy PORT MAP(clk, rst, x, y);

    clk_process : PROCESS
    BEGIN
        clk <= '1'; WAIT FOR clk_period/2;
        clk <= '0'; WAIT FOR clk_period/2;
    END PROCESS;

    rst_process : PROCESS
    BEGIN
        WAIT FOR clk_period;
        rst <= '0';
        WAIT;
    END PROCESS;

    PROCESS IS
        VARIABLE count : INTEGER := 8;
        VARIABLE tmp : STD_LOGIC_VECTOR(3 DOWNTO 0) := (OTHERS => '0');
    BEGIN
        IF count < 0 THEN
            stop;
        ELSE
            tmp := STD_LOGIC_VECTOR(to_unsigned(count, 4));
            x <= tmp(1 DOWNTO 0); WAIT FOR 50 ns;
            count := count - 1;
        END IF;
    END PROCESS;

END MainBehavior;
```

Скрипт:

```
*x0 соответствует x(1), x1 – x(2)
vsim -novopt mealy
force clk '1' 0, '0' 10 -repeat 20
force rst '1' 0, '0' 20 -repeat 180
force x0 '0' 0, '1' 20, '1' 40, '0' 60 -repeat 80
force x1 '0' 0, '1' 20, '0' 40, '1' 60 -repeat 80
add wave *
view wave
run 180 ns
quit –sim
```



3 ВЫВОД

При выполнении лабораторной работы я составил структурное и алгоритмическое VHDL-описания для двухтактного Т-триггера со сбросом и установкой и провел моделирование данных описаний, которые дали идентичный результат.

Также составил VHDL-модель конечного автомата Мили и провел моделирование с помощью тестирующей программы и скрипта, получив одинаковые результаты.