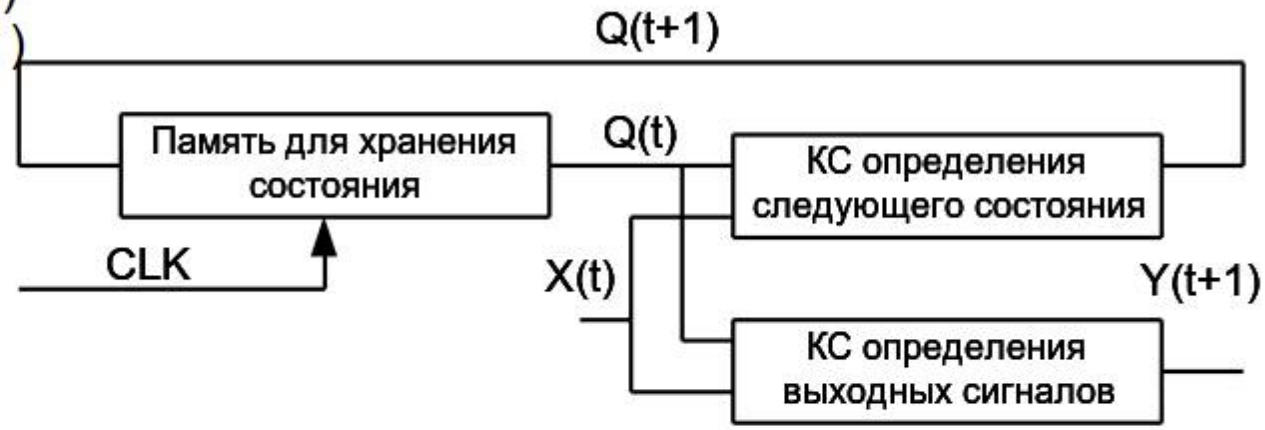
Домашнее задания

Проектирование устройств управления с жесткой логикой

Автомат Мили

$$\begin{cases} Q(t+1) = A\,(\,Q(t),\, x(t)\,) \\ Y(t+1) = B\,(\,Q(t),\, x(t)\,) \end{cases}$$

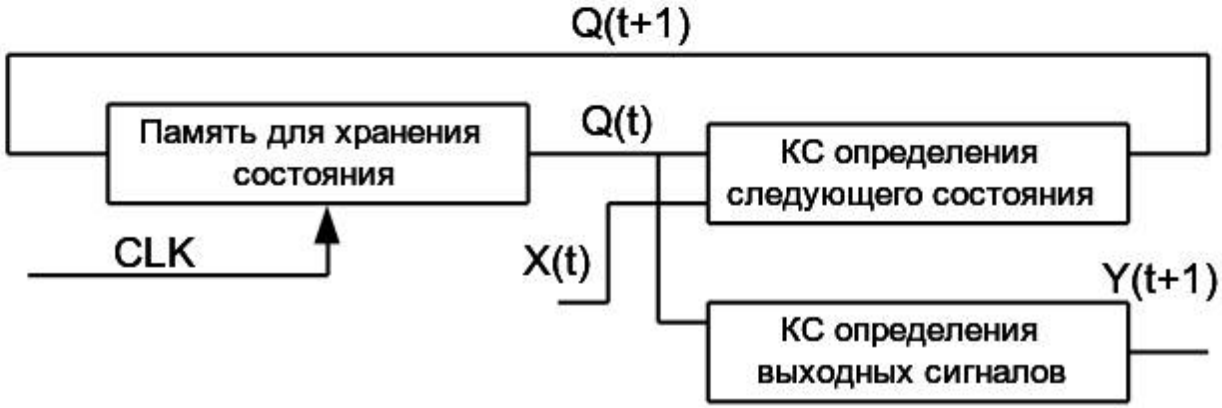
Схема автомата Мили



Автомат Мура

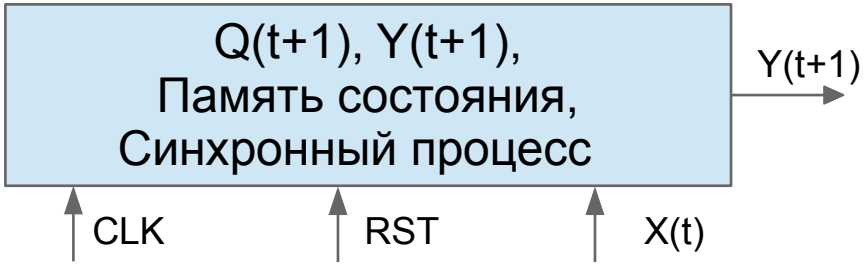
$$\begin{cases} Q(t+1) = A\,(\,Q(t),\, x(t)\,) \\ Y(t+1) = B\,(\,Q(t)\,) \end{cases}$$

Схема автомата Мура

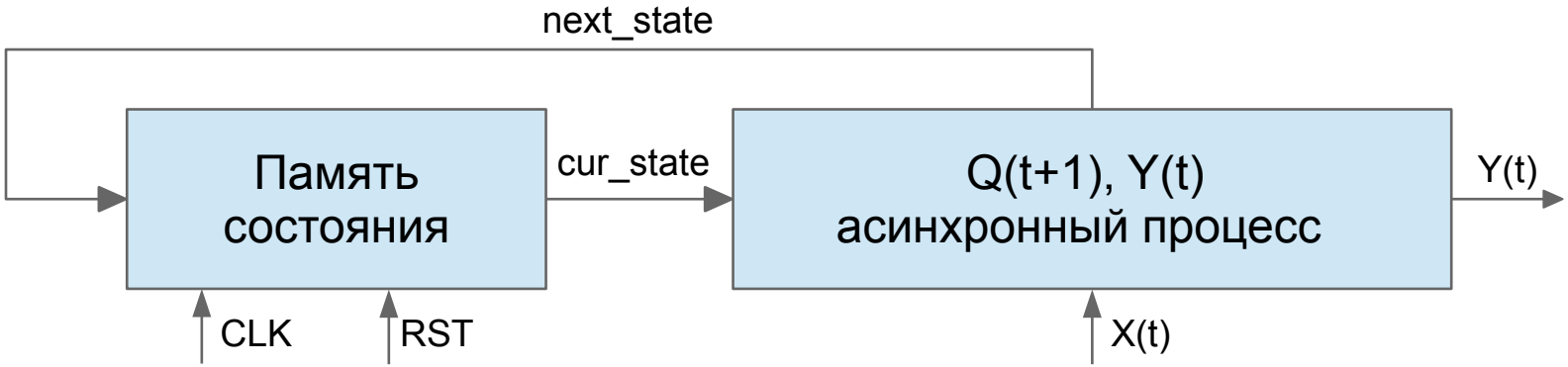


Описание автомата на языке VHDL

Описание в одном синхронном процессе



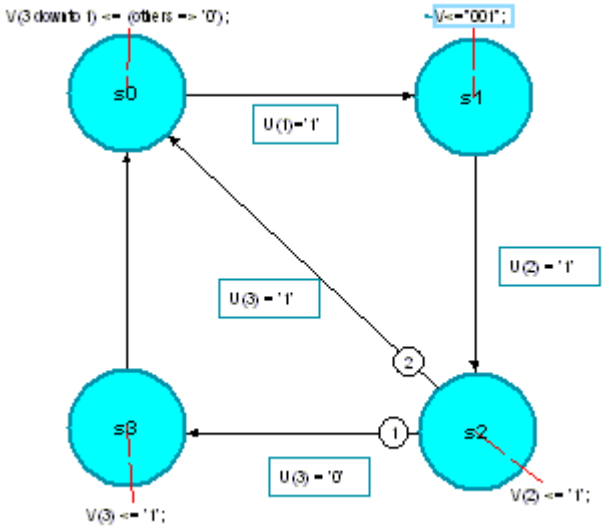
Описание в 2х процессах (синхронном и асинхронном)



Описание автомата Мура на языке VHDL

(вариант с асинхронными входами и выходами)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
ENTITY control_unit IS
  PORT(U   : IN     std_logic_vector ( 3 DOWNTO 1 );
       clk : IN     std_logic;
       rst : IN     std_logic;
       V   : OUT    std_logic_vector ( 3 DOWNTO 1 ) );
END control_unit;
ARCHITECTURE moore OF control_unit IS
  TYPE STATE_TYPE IS (s0, s1,s2,s3);
  SIGNAL current_state : STATE_TYPE;
BEGIN
  clocked_proc : PROCESS (clk, rst)
  BEGIN
    IF (rst = '0') THEN
      current_state <= s0;
    ELSIF (clk'EVENT AND clk = '1') THEN
```



```
CASE current_state IS
        WHEN s0 =>
          V(3 downto 1) <= (others => '0');
          IF (U(1)='1') THEN current_state <= s1;
          ELSE current_state <= s0; END IF;
        WHEN s1 =>
          V<= "001";
          IF (U(2) = '1') THEN current_state <= s2;
          ELSE current_state <= s1; END IF;
        WHEN s2 =>
          V <= "010";
          IF (U(3) = '0') THEN  current_state <= s3;
          ELSE current_state <= s0; END IF;
        WHEN s3 =>
          V <= "100";
          current_state <= s0;
        WHEN OTHERS =>
          current_state <= s0;
      END CASE;
    END IF;
  END PROCESS clocked_proc;
END moore;
```

Описание автомата Мура на языке VHDL
(вариант с синхронными входами и выходами)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
ENTITY control_unit IS
        PORT(       U   : IN     std_logic_vector ( 3 DOWNTO 1 );
        clk : IN     std_logic;
        rst : IN     std_logic;
        V   : OUT    std_logic_vector ( 3 DOWNTO 1 ) );
END control_unit;


ARCHITECTURE moore OF control_unit IS
        TYPE STATE_TYPE IS (s0, s1,s2,s3);
SIGNAL current_state, next_state : STATE_TYPE;
BEGIN
clocked_proc: PROCESS(clk)
BEGIN
        IF(rising_edge(clk)) THEN
                IF (reset='1') THEN
                        current_state <= s0;
                ELSE
                        current_state <= next_state;
                END IF;
        END IF;
END PROCESS;
```

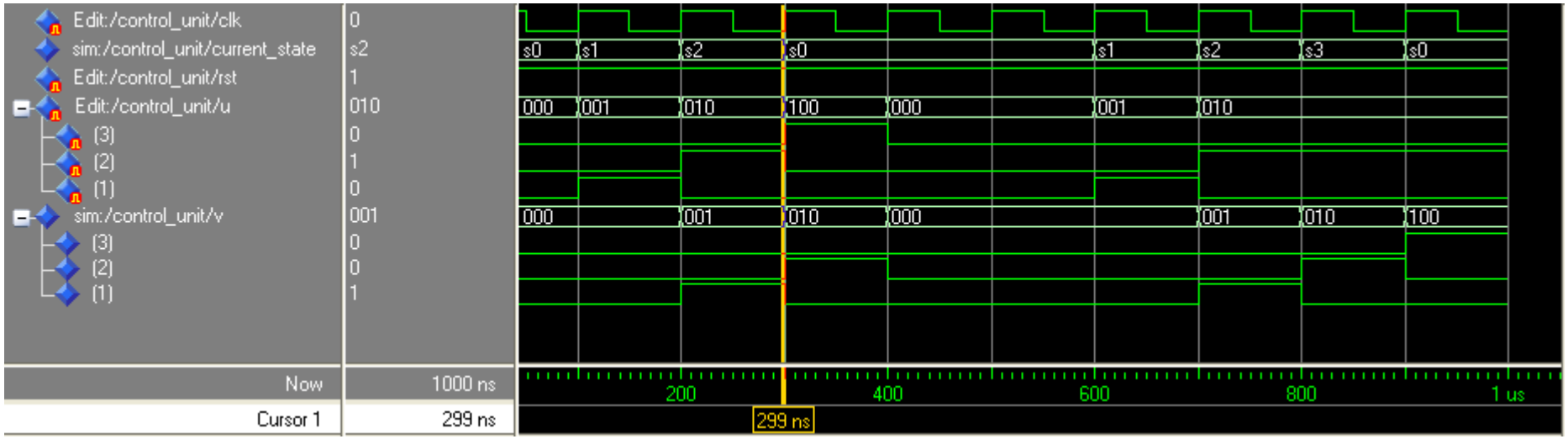
Описание автомата Мура на языке VHDL

(вариант с синхронными входами и выходами)

```
comb_proc : PROCESS (current_state,U)
BEGIN
        –– Внимание! Комбинационные сигналы
        –– должны присваиваться непрерывно.
        -- Удобно для этих целей задать
        -- значение по умолчанию:
      V(3 downto 1) <= (others => '0');
        next_state <= s0;

        CASE current_state IS
          WHEN s0 =>
              V(3 downto 1) <= (others => '0');
              IF (U(1)='1') THEN
                            next_state <= s1;
              ELSE
                            next_state <= s0;
              END IF;
          WHEN s1 =>
              V<= "001";
              IF (U(2) = '1') THEN
                            next_state <= s2;
              ELSE
                            next_state <= s1;
               END IF;
          WHEN s2 =>
                        V <= "010";
                        IF (U(3) = '0') THEN
                            nxt_state <= s3;
                        ELSE
                            next_state <= s0;
                        END IF;
           WHEN s3 =>
                        V <= "100";
                        next_state <= s0;
           WHEN OTHERS =>
                        V(3 downto 1) <= (others => '0');
                        next_state <= s0;
      END CASE;
      END PROCESS comb_proc;
      END moore;
```

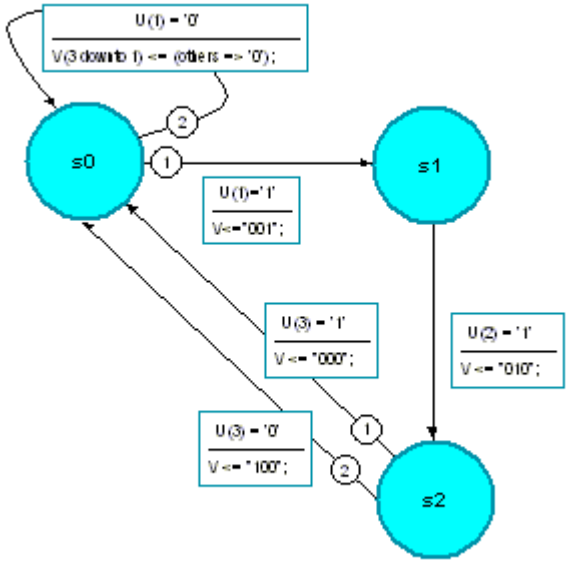
Тест автомат Мура в ModelSim 6



Описание автомата Мили на языке VHDL

(вариант с синхронными выходами)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
ENTITY control_unit IS
  PORT(U   : IN     std_logic_vector ( 3 DOWNTO 1 );
       clk : IN     std_logic;
       rst : IN     std_logic;
       V   : OUT    std_logic_vector ( 3 DOWNTO 1 ) );
END control_unit;
ARCHITECTURE mielie OF control_unit IS
  TYPE STATE_TYPE IS (s0, s1,s2);
  SIGNAL current_state : STATE_TYPE;
BEGIN
  clocked_proc : PROCESS (clk, rst)
  BEGIN
    IF (rst = '0') THEN
      current_state <= s0;
    ELSIF (clk'EVENT AND clk = '1') THEN
```



Описание автомата Мили на языке VHDL (окончание)

```
CASE current_state IS
        WHEN s0 =>
          IF (U(1)='1') THEN
            V<="001";
            current_state <= s1;
          ELSIF (U(1) = '0') THEN
            V(3 downto 1) <= (others =>
'0');
            current_state <= s0;
          ELSE
            current_state <= s0;
          END IF;
        WHEN s1 =>
          IF (U(2) = '1') THEN
            V <= "010";
            current_state <= s2;
          ELSE
            current_state <= s1;
          END IF;
        WHEN s2 =>
          IF (U(3) = '1') THEN
            V <= "000";
            current_state <= s0;
          ELSIF (U(3) = '0') THEN
            V <= "100";
            current_state <= s0;
          ELSE
            current_state <= s2;
          END IF;
        WHEN OTHERS =>
          current_state <= s0;
      END CASE;
    END IF;
  END PROCESS clocked_proc;

END mielie;
```

Описание автомата Мили на языке VHDL
(вариант с синхронными входами и выходами)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
ENTITY control_unit IS
   PORT(    U   : IN     std_logic_vector ( 3 DOWNTO 1 );
        clk : IN     std_logic;
        rst : IN     std_logic;
        V   : OUT    std_logic_vector ( 3 DOWNTO 1 ) );
END control_unit;

ARCHITECTURE mielie OF control_unit IS
   TYPE STATE_TYPE IS (s0, s1,s2,s3);
   SIGNAL current_state, next_state : STATE_TYPE;
BEGIN

clocked_proc: PROCESS(clk)
BEGIN
        IF(rising_edge(clk)) THEN
                IF (reset='1') THEN
                        current_state <= s0;
                ELSE
                        current_state <= next_state;
                END IF;
        END IF;
END PROCESS clocked_proc;
```

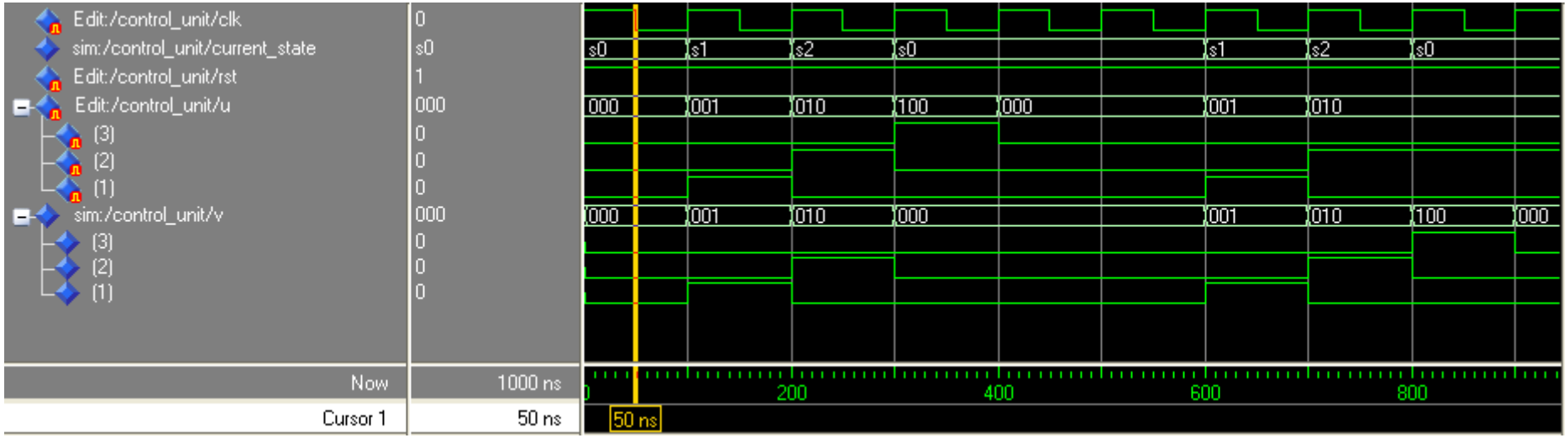
Описание автомата Мили на языке VHDL

(вариант с синхронными входами и выходами)

```
comb_proc : PROCESS (current_state,U)
BEGIN
        –- Внимание! Комбинационные сигналы
        –- должны присваиваться непрерывно.
        -- Удобно для этих целей задать
        -- значение по умолчанию:
        V(3 downto 1) <= (others => '0');
        next_state <= s0;

        CASE current_state IS
        WHEN s0 =>
                IF (U(1)='1') THEN
                        V<="001";
                        next_state <= s1;
        ELSIF (U(1) = '0') THEN
                        V(3 downto 1) <= (others => '0');
                        next_state <= s0;
        ELSE
                        next_state <= s0;
        END IF;
 WHEN s1 =>
        IF (U(2) = '1') THEN
                        V <= "010";
                        next_state <= s2;
        ELSE
                        next_state <= s1;
        END IF;
WHEN s2 =>
        IF (U(3) = '1') THEN
                        V <= "000";
                        next_state <= s0;
        ELSIF (U(3) = '0') THEN
                        V <= "100";
                        next_state <= s0;
        ELSE
                        next_state <= s2;
        END IF;
 WHEN OTHERS =>
        next_state <= s0;
        END CASE;
END PROCESS comb_proc;
END mielie;
```

Тест автомат Мили в ModelSim 6



Домашнее задание по курсу «ЭВМ»

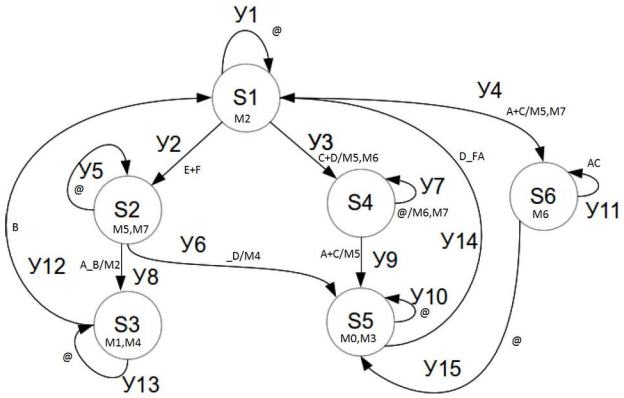
В ходе выполнения домашнего задания необходимо разработать устройство управления схемного типа, обрабатывающий входное командное слово C={ABCDEF} и выдающий сигналы управления M={M0,...,Mk-1} операционному блоку в соответствии с приведенной в индивидуальном задании логикой работы.

Этап 1.
А. По диаграмме переходов автомата (Приложение 1) и описанию условий переходов и активных сигналов (дополнительный файл варианты.pdf), определить тип управляющего автомата (автомат Мили или Мура, смешанный). Выбор обосновать.
В. Произвести кодирование состояний управляющего автомата. Составить схему переходов/состояний полученного автомата. Схему представить в отчете.

Этап 2.
Разработать описание устройства управления на языке VHDL, для чего использовать приведенные в Приложении 2 шаблоны для автоматов Мили и Мура.
Разработать тестовое описание для устройства, представляющее собой генератор входных сигналов (см. Приложение 3). Тестовое описание должно обеспечивать проверку всех ветвей автомата.

Этап 3.
А. Установить ПО ModelSim PE (или аналогичный продукт: Xilinx ISE, Altera Quartus).
Б. Выполнить моделирование полученного теста в ПО ModelSim PE. Результаты моделирования представить в отчете.

Описание устройства управления



Варианты индивидуальных заданий по курсу ЭВМ

Таблица 1. Варианты диаграмм и активных сигналов.

Вариант	Диаграмма переходов	Активные сигналы M в состоянии					
		S1	S2	S3	S4	S5	S6
1.	1	-	-	-	-	-	-
2.	2	-	1	-	4, 5, 6	0,2	3,7
3.	3	-	0,2	7	1,3	4,5	6

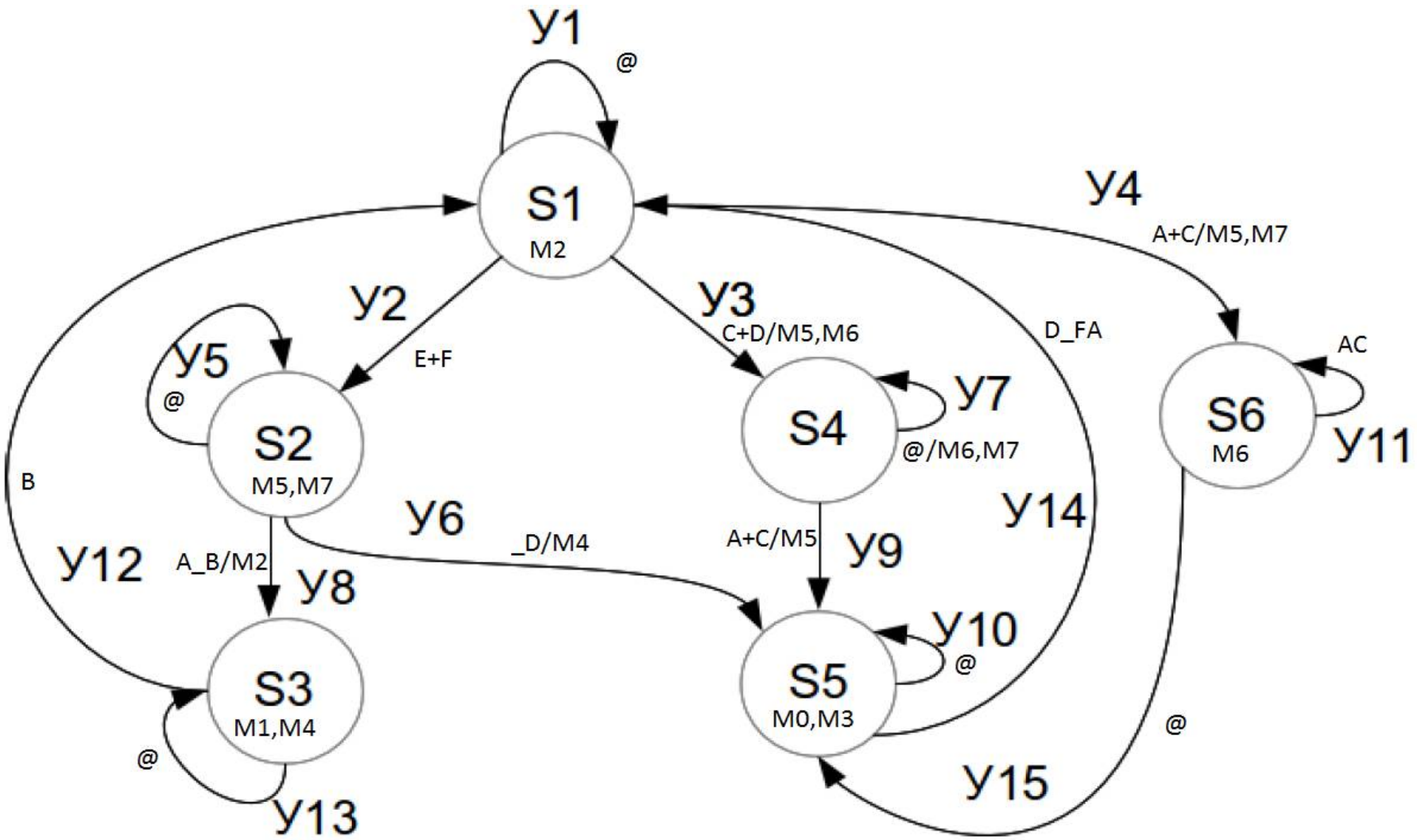
Таблица 2. Условия переходов и наименование отладочной платы («@» - иначе, «_X» - НЕ X, «+» - ИЛИ, 1- безусловный переход).

Вариант	Название отладочной платы	Активные сигналы M в состоянии														
		У1	У2	У3	У4	У5	У6	У7	У8	У9	У10	У11	У12	У13	У14	У15
1.	Spartan3	@	ABC	A_B	@	E+D	F	@	EF	@	A+_C	_A	AB	@	@	1
2.	Nexus2	@	CD_E	A+_C	@	E+C	F	@	D_F	B	@	_AC	AB	@	@	ABCF

Таблица 3. Активные сигналы для переходов.

Вариант	Активные сигналы M в состоянии														
	У1	У2	У3	У4	У5	У6	У7	У8	У9	У10	У11	У12	У13	У14	У15
1.	0	-	-	1,2	0,3	-	3	3	6,7	-	2,4	-	2	3,5	-

Диаграмма переходов состояний управляющего автомата



Листинг VHDL описания управляющего автомата

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;

entity control_unit is
    Port ( C   : in   STD_LOGIC_VECTOR (5 downto 0);
           clk : in   STD_LOGIC;
           rst : in   STD_LOGIC;
           M   : out  STD_LOGIC_VECTOR (7 downto 0));
end control_unit;

architecture hybrid of control_unit is
        type STATE_TYPE is (s1,s2,s3,s4,s5,s6);
        signal cur_state : STATE_TYPE;
begin
        clocked_proc : process (clk,rst)
        begin
                if (rst = '0') then
                        cur_state <= s1;
                        M <= "00000000";
                elsif (clk'event and clk = '1') then
                        case cur_state is
                                when s1 =>
                                        M <= "00000100";                        -- M2
                                        if (C(4) = '1' or C(5) = '1') then      -- E or F
                                                cur_state <= s2;
                                        elsif (C(2) = '1' or C(3) = '1') then   -- C or D
                                                M(5) <= '1';
                                                M(6) <= '1';
                                                cur_state <= s4;
                                        elsif (C(0) = '1' or C(2) = '1') then   -- A or C
                                                M(5) <= '1';
                                                M(7) <= '1';
                                                cur_state <= s6;
                                        else                                    -- @
                                                cur_state <= s1;
                                        end if;
                                when s2 =>
                                        M <= "10100000";                        -- M5, M7
                                        if (C(3) = '0') then                    -- not D
                                                M(4) <= '1';
                                                cur_state <= s5;
```

```
                elsif (C(0) = '1' and C(1) = '0') then -- A & _B
                        M(2) <= '1';
                        cur_state <= s3;
                else                                    -- @
                        cur_state <= s2;
                end if;
        when s3 =>
                M <= "00010010";                        -- M1, M4
                if (C(1) = '1') then                    -- B
                        cur_state <= s1;
                else                                    -- @
                        cur_state <= s3;
                end if;
        when s4 =>
                M <= "00000000";
                if (C(0) = '1' or C(2) = '1') then          -- A or C
                        M(5) <= '1';
                        cur_state <= s5;
                else                                    -- @
                        M(6) <= '1';
                        M(7) <= '1';
                        cur_state <= s4;
                end if;
        when s5 =>
                M <= "00001001";                        -- M0, M3
                if (C(3) = '1' and C(5) = '0'
                        and C(0) = '1') then          -- D & _F & A
                        cur_state <= s1;
                else                                    -- @
                        cur_state <= s5;
                end if;
        when s6 =>
                M <= "01000000";                        -- M6
                if (C(0) = '1' and C(2) = '1') then         -- A and C
                        cur_state <= s6;
                else                                    -- @
                        cur_state <= s5;
                end if;
        when others =>
                cur_state <= s1;
    end case;
end if;
```

Моделирование