

Lab 3

Part 3: Algorithmic State Machine (ASM) Charts

This project is to implement the following ASM charts and run simulations using VHDL.

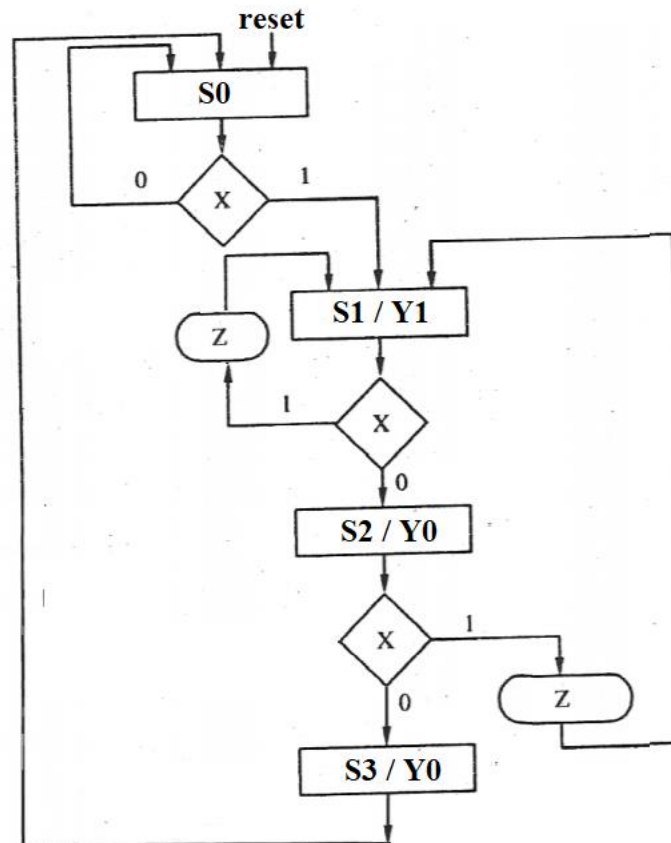


Figure 1. ASM diagram

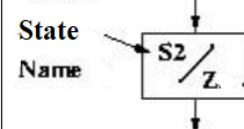
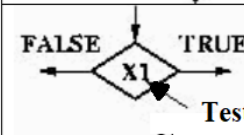
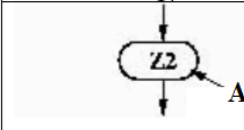
Demo Requirement

You need to demonstrate the final simulation waveform of this design to your lab instructor.

Lab Procedure

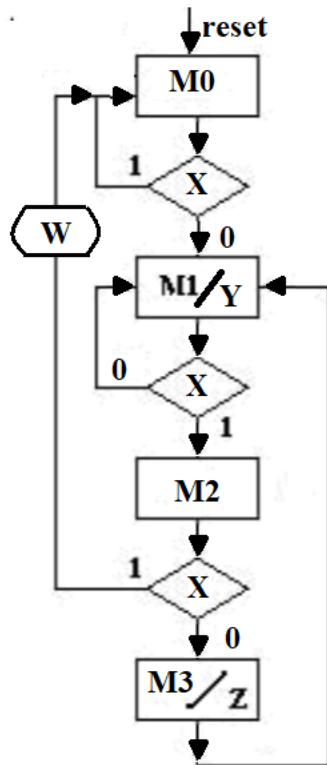
Step 1. ASM Review

First, you need to review the definition of ASM charts you have learnt in class.

 <p>State Name</p> <p>Asserted Moore Outputs</p>	<p>One state box. The state box has a name and lists outputs that are asserted when the system is in that state. These outputs are called <i>Moore</i> type outputs.</p>
 <p>FALSE</p> <p>TRUE</p> <p>Test Condition</p>	<p>Optional decision box (es). A decision box may be conditioned on a signal or a test of some kind.</p>
 <p>Asserted Mealy outputs</p>	<p>Optional conditional output box (es). Such an output box indicates outputs that are conditionally asserted. These outputs are called <i>Mealy</i> outputs.</p>

Step 2. VHDL design and testbench review

The ASM chart below has two moore outputs Y and Z. Y is asserted in the M1 state and Z is asserted in the M3 state. It has a mealy output, which is asserted when X is logic high in the M2 state.



Library ieee;

Use ieee.std_logic_1164.all;

entity chart is

port (reset, clk, x: in std_logic;

y, z, w: out std_logic ;

ckcs, ckns: out std_logic_vector (1 downto 0));

end chart;

architecture beh of chart is

constant M0: std_logic_vector(1 downto 0) := "00";

constant M1: std_logic_vector(1 downto 0) := "01";

```
constant M2: std_logic_vector(1 downto 0) := "10";
constant M3: std_logic_vector(1 downto 0) := "11";
signal cs, ns: std_logic_vector (1 downto 0);
begin
    ckcs <= cs;
    ckns <= ns;
    process(reset, clk)
    begin
        If ( reset = '1') then
            cs <= M0;
        elsif (rising_edge(clk) ) then
            cs <= ns;
        end if;
    end process;
    process(cs, x)
    begin
        case (cs) is
            when M0 => if (x='1') then
                ns <= M0;
            else
                ns <= M1;
            end if;
            when M1 => if (x='1') then
                ns <= M2;
            else
                ns <= M1;
            end if;
```

CPE166 Lab 3 Part 3

By: Prof. Pang

```

        when M2 => if (x= '0') then
            ns <= M0;
        else
            ns <= M3;
        end if;

        when M3 => ns <= M1;

        when others=> ns <= M0;
    end case;

end process;

y <= '1' when ( cs = M1 ) else '0';
z <= '1' when ( cs = M3 ) else '0';
w <= '1' when ( ( cs = M2 ) and ( x= '1' ) ) else '0';

end beh;

library IEEE;
use IEEE.std_logic_1164.all;

entity testbench is
end testbench;

architecture tb of testbench is
component chart
    port ( reset, clk, x: in std_logic;
          y, z, w: out std_logic;
          ckcs, ckns: out std_logic_vector (1 downto 0));
end component;

signal reset, clk, x, y, z, w: std_logic;
signal cs, ns: std_logic_vector (1 downto 0);

```

CPE166 Lab 3 Part 3

By: Prof. Pang

```

begin

    DUT: chart port map(reset, clk, x, y, z, w, cs, ns);

    process

    begin

        clk <= '0';

        wait for 5 ns;

        clk <= '1';

        wait for 5 ns;

    end process;

    x <= '1', '0' after 10 ns, '1' after 40 ns, '0' after 60 ns, '1' after 80ns, '0' after 120ns, '1' after
    160 ns, '0' after 200 ns, '1' after 300 ns, '0' after 350 ns;

```

```

Process
Begin

    Reset <= '1';

    Wait for 2 ns;

    Reset <= '0';

    Wait for 400 ns;

    Wait;

    End process;

End tb;

```

Step 3. Final Project Work

You need to write a VHDL design and testbench for the ASM charts shown in figure 1, and run simulations. Demonstrate your simulation waveform to your lab instructor.