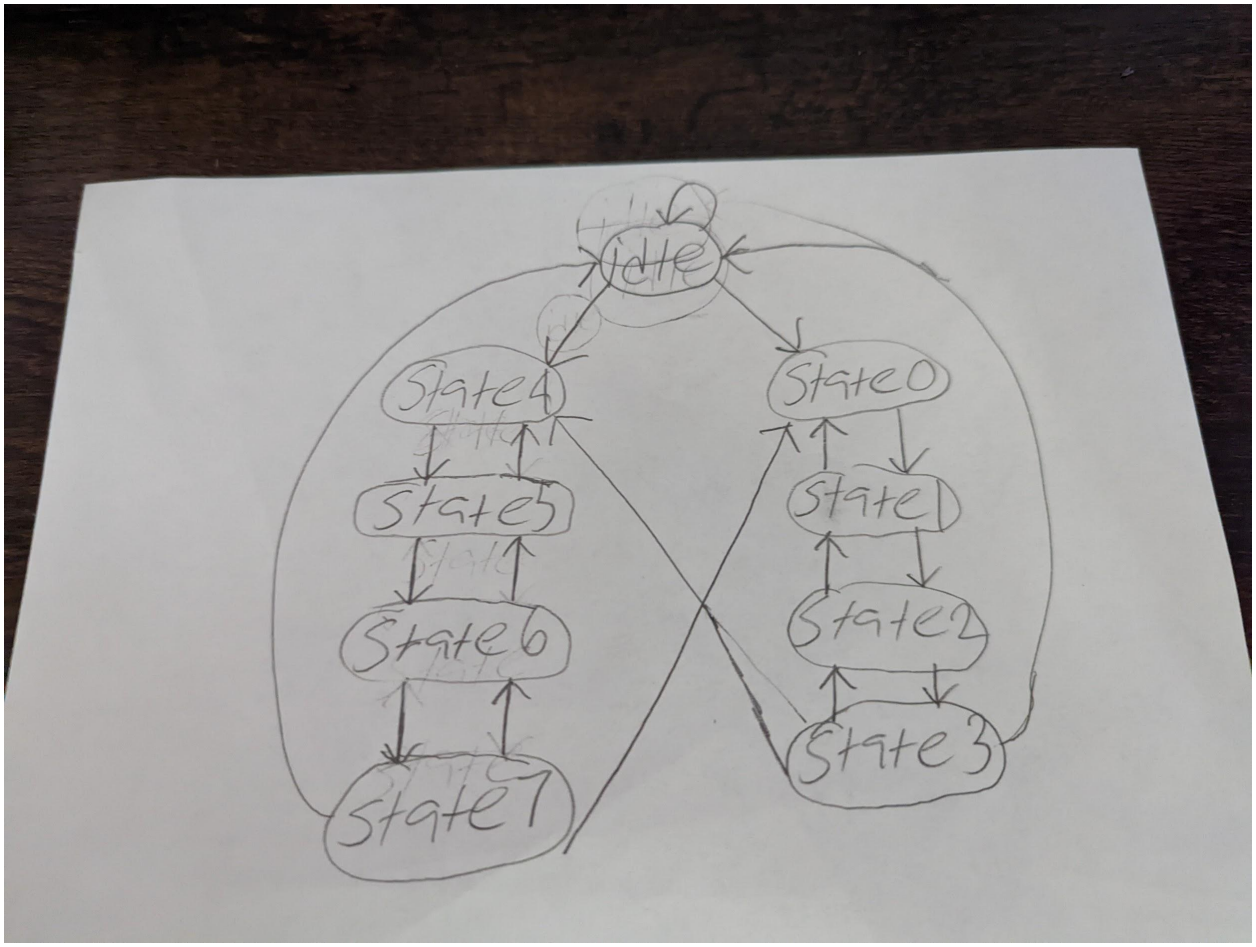
Luis Umana
801018765

ECGR 4146: Introduction to VHDL
Lab 7: Manchester Encoding

FSM for Manchester Encoder



VHDL Code for Manchester Encoder

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

entity ManchesterEncoder is
    Port ( str_input : in STD_LOGIC_VECTOR(7 downto 0);
           clk : in STD_LOGIC;
           reset : in STD_LOGIC;
           str_output : out STD_LOGIC);
end ManchesterEncoder;

architecture Behavioral of ManchesterEncoder is
```

Luis Umana
801018765

```
type state_type is (idle,state0, state1, state2, state3, state4, state5, state6,
state7,stop);
signal state_reg : state_type;
signal R_Next: state_type;
signal state : std_logic;
signal buff : std_logic;
begin

--state register
process(clk, reset)
    begin
        if (reset = '1') then
        state_reg <= idle;
      elsif (clk'event and clk='1') then
        state_reg <= R_Next;
      end if;
end process;

--output buffer
process(clk, reset)
    begin
        if (reset = '1') then
        buff <= '0';
      elsif (clk'event and clk='1') then
        buff <= state;
      end if;
end process;

--next state logic
process(state_reg, reset)
    begin
        case state_reg is
        when idle =>
        if (reset='0') then
            R_Next <= state0;
        else
           R_Next  <= idle;
        end if;

        when state0 =>
        R_Next  <= state1;

        when state1 =>
        R_Next  <= state2;

        when state2 =>
        R_Next  <= state3;

        when state3 =>
        R_Next  <= state4;

        when state4 =>
        R_Next  <= state5;

        when state5 =>
        R_Next  <= state6;

        when state6 =>
```

```
        R_Next  <= state7;

        when state7 =>
        R_Next  <= stop;

        when stop =>
        R_Next  <= stop;
        end case;
    end process;

    --look ahear moore output logic
    process(R_Next)
        begin
            case R_Next  is
            when idle =>
            state <= '0';

            when state0 =>
            state <= str_input(7);

            when state1 =>
            state <= str_input(6);

            when state2 =>
            state <= str_input(5);

            when state3 =>
            state <= str_input(4);

            when state4 =>
            state <= str_input(3);

            when state5 =>
            state <= str_input(2);

            when state6 =>
            state  <= str_input(1);

            when state7 =>
            state <= str_input(0);

            when stop =>
            state <= '0';
            end case;
    end process;

    str_output <= not(buff xor clk);

end Behavioral;
```

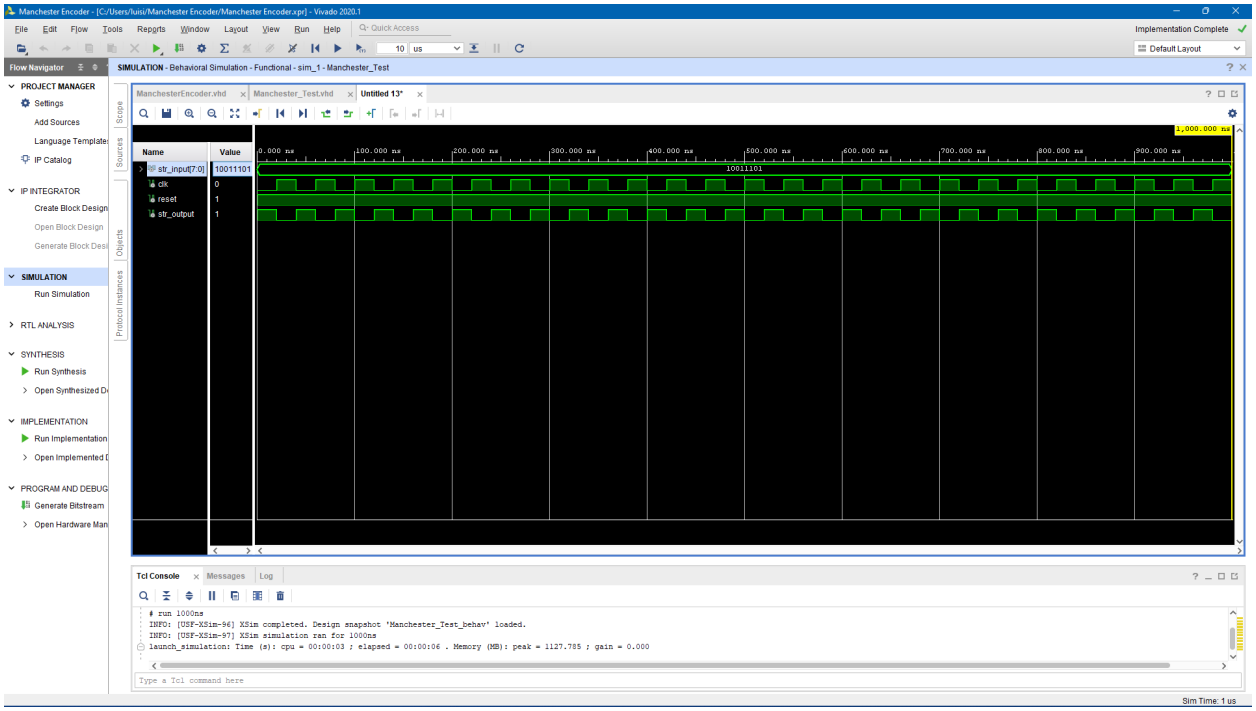
VHDL Testbench Code for Manchester Encoder

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity Manchester_Test is
--(Port)
```

Luis Umana
801018765

```
end Manchester_Test;
architecture Behavioral of Manchester_Test is
signal str_input:STD_LOGIC_VECTOR(7 downto 0);
signal clk :std_logic;
signal reset :std_logic;
signal str_output:std_logic;
begin
p0:entity work.ManchesterEncoder(Behavioral) port map
(str_input=> str_input,clk=>clk,reset=>reset,str_output=>str_output);
clock_process :process
begin
    str_input <= "01100010";
    clk <= '0';
    wait for 20 ns;
    clk <= '1';
    wait for 20 ns;
end process;
stim_proc: process
begin
    reset <= '1';
    wait for 20 ns;
    reset <= '0';
end process;
end Behavioral;
```

Simulation Waveform of Manchester Encoder



Luis Umana
801018765