Problem 1 - ASM chart

Dataflow VHDL code for the output function computing p, r, and y, as a function of the state S, and the inputs a and b:

\[
\begin{align*}
p &\leq '1' \text{ when } (S = S1) \text{ or } (S = S3) \text{ else } '0'; \\
r &\leq '1' \text{ when } ((S = S1) \text{ and } (b = '0')) \text{ or } ((S = S2) \text{ and } (b = '1')) \text{ or } ((S = S3) \text{ and } (b = '1')) \text{ or } ((S = S4) \text{ and } (b = '0')) \text{ else } '0'; \\
y &\leq '1' \text{ when } (S = S4) \text{ and } (b = '0') \text{ and } (a = '1') \text{ else } '0';
\end{align*}
\]
Problem 2 – Block Diagram of Combinational Logic
# 16 x 1 ROM

Memory Map

<table>
<thead>
<tr>
<th>ADDR</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
</tr>
</tbody>
</table>
library ieee;
use ieee.std_logic_1164.all;

entity misr is
  generic (C : std_logic_vector(7 downto 0));
  port ( -- inputs
    clk : in std_logic;
    rst : in std_logic;
    en : in std_logic;
    D : in std_logic_vector (7 downto 0);
  -- outputs
    Q_out : out std_logic_vector (7 downto 0)
  );
end misr;

architecture mixed of misr is

-- intermediate signals
  signal Q : std_logic_vector (7 downto 0);
  signal Q0_replicated : std_logic_vector (7 downto 0);
  signal d_ff_in : std_logic_vector (7 downto 0);

begin
  Q0_replicated <= (others => Q(0));
  d_ff_in <= D xor ('0' & Q(7 downto 1)) xor (C and Q0_replicated);

  -- D flip flop operation
  D_FFs: process (rst, clk)
  begin
    if (rst = '1') then
      Q <= (others => '0');
    elsif rising_edge(clk) then
      if(en = '1') then
        Q <= d_ff_in;
      end if;
    end if;
  end process;
  Q_out <= Q;
end mixed;
Problem 4 – FPGA Resources
library ieee;
use ieee.std_logic_1164.all;

entity debouncer_tb is
end debouncer_tb;

architecture behavioral of debouncer_tb is

  -- inputs
  signal clk : std_logic := '0';
  signal rst : std_logic;
  signal input: std_logic;

  -- outputs
  signal output : std_logic;

  -- constant definitions
  constant clk_period : time := 10 ns;
  constant rst_length : time := 50 ns;
  constant min_before_pulse : time := 100 ns;
  constant pulse_width : time := 500 ns;
  constant bounce_period : time := 40 ns;

begin

  debouncer_inst: entity work.debouncer
    generic map ( 
      K => 4,
      DD => 15)
    port map ( 
      clk => clk,
      rst => rst,
      input => input,
      output => output
    );

  clk <= not clk after clk_period/2;

  rst_process: process
  begin
    rst <= '1';
    wait for rst_length;
    rst <= '0';
    wait;
  end process;

input_process: process

begin

    input <= '0';
    wait for min_beforepulse;
    wait until falling_edge(clk);

    for i in 0 to 2 loop
        input <= '1';
        wait for bounce_period/2;
        input <= '0';
        wait for bounce_period/2;
    end loop;

    input <= '1';
    wait for pulse_width;

    for i in 0 to 2 loop
        input <= '0';
        wait for bounce_period/2;
        input <= '1';
        wait for bounce_period/2;
    end loop;

    input <= '0';
    wait;

end process;

end behavioral;