```
-- USN VHDL 101 course
-- template for FSM development
-- author | date

library ieee;
use ieee.std_logic_1164.all;

entity edice is
   port ( clk, rst, run, cheat: in std_logic;
          R: in std_logic_vector(2 downto 0);
          an: out std_logic_vector(3 downto 0);
          dout: out std_logic_vector(7 downto 0)
            );
end edice;

architecture arch of edice is
type state is (S0, S1, S2, S3, S4, S5, S6, S7);
signal prest, nxtst: state;  -- present state, next state

begin
-- state register
process (clk, rst)
begin
   if (rst = '1') then
          prest <= S0; -- initial state
   elsif rising_edge(clk) then
          prest <= nxtst;
   end if;
end process;

-- next-state logic
process (prest, run, cheat)
begin
   nxtst <= prest; -- stay in current state by default
```

```
   case prest is
      when S0 =>
         if run='1' then nxtst <= S1;
         end if;
      when S1 =>
         if run='1' then nxtst <= S2;
         end if;
      when S2 =>
         if run='1' then nxtst <= S3;
         end if;
      when S3 =>
         if run='1' then nxtst <= S4;
         end if;
      when S4 =>
         if run='1' then nxtst <= S5;
         end if;
      when S5 =>
         if (run='1' and cheat='0') then nxtst <= S0;
         elsif (run='1' and cheat='1' and R/="000" and R/="111") then nxtst <= S6;
         else nxtst <= S0;
         end if;
      when S6 =>
         if (run='1' and cheat='0') then nxtst <= S0;
         elsif (run='1' and cheat='1' and R/="000" and R/="111") then nxtst <= S7;
         else nxtst <= S0;
         end if;
      when S7 =>
         if run='1' then nxtst <= S0;
         end if;
   end case;
end process;

-- Moore outputs logic
process (prest)
begin
```

```
   case prest is
      when S0 =>
         dout <= "1001111" & not cheat;  -- 1
      when S1 =>
         dout <= "0010010" & not cheat;  -- 2
      when S2 =>
         dout <= "0000110" & not cheat;  -- 3
      when S3 =>
         dout <= "1001100" & not cheat;  -- 4
      when S4 =>
         dout <= "0100100" & not cheat;  -- 5
      when S5 =>
         dout <= "0100000" & not cheat;  -- 6
      when S6 =>
         if (R="001")    then dout <= "1001111" & not cheat;  -- 1
         elsif (R="010") then dout <= "0010010" & not cheat;  -- 2
         elsif (R="011") then dout <= "0000110" & not cheat;  -- 3
         elsif (R="100") then dout <= "1001100" & not cheat;  -- 4
         elsif (R="101") then dout <= "0100100" & not cheat;  -- 5
         elsif (R="110") then dout <= "0100000" & not cheat;  -- 6
         else dout <= "1111111" & not cheat;  -- all off should never happen...
         end if;
      when S7 =>
         if (R="001")    then dout <= "1001111" & not cheat;  -- 1
         elsif (R="010") then dout <= "0010010" & not cheat;  -- 2
         elsif (R="011") then dout <= "0000110" & not cheat;  -- 3
         elsif (R="100") then dout <= "1001100" & not cheat;  -- 4
         elsif (R="101") then dout <= "0100100" & not cheat;  -- 5
         elsif (R="110") then dout <= "0100000" & not cheat;  -- 6
         else dout <= "1111111" & not cheat;  -- all off should never happen...
         end if;
      when others =>
         dout <= "1111111" & not cheat;  -- all segments off
   end case;
end process;
```

```
an <= "0110";

end arch;
```

```
-- USN VHDL 101 course
-- template for test bench development
-- author | date

library ieee;
use ieee.std_logic_1164.all;

entity tb_edice is
end tb_edice;

architecture tb of tb_edice is

    component edice
        port (clk, rst : in std_logic;
              R: in std_logic_vector(2 downto 0);
              an: out std_logic_vector(3 downto 0);
              run, cheat   : in std_logic;
                           dout   : out std_logic_vector(7 downto 0)
                           );
    end component;

    signal clk, rst : std_logic;
    signal run, cheat : std_logic; -- module inputs
    signal R : std_logic_vector(2 downto 0); -- module outputs
    signal an : std_logic_vector(3 downto 0); -- module outputs
         signal dout : std_logic_vector(7 downto 0); -- module outputs

    constant clk_period : time := 10 ns;

begin

    uut : edice
    port map (clk => clk, rst => rst,
              run => run, cheat => cheat,
              R => R, an => an,
```

```
              dout => dout );

clk_process: process
   begin
      clk <= '0';
      wait for clk_period/2;
      clk <= '1';
      wait for clk_period/2;
   end process;

-- Stimuli process
   stim_proc: process
      begin
         run <= '1';
         cheat <= '0';
         R <= "010"; -- cheat on result 2 when cheat='1'
         rst <= '1';
         wait for clk_period;
         rst <= '0';
         wait for clk_period*10;
         cheat <= '1';
         wait for clk_period*10;
         run <= '0';
         wait for clk_period*10;
      end process ;

end tb;
```