```vhdl
-----------------------------------------
-- Block code:  fsm_example.vhd
-- History:     23.Nov.2017 - 1st version (dqtm)
--                  <date> - <changes>  (<author>)
-- Function: fsm_example using enumerated type declaration.
-----------------------------------------

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY fsm_example IS
    PORT (
        clk,reset :     IN std_logic;
        d, n :          IN std_logic;
        z :             OUT std_logic
    );
END fsm_example;


ARCHITECTURE rtl OF fsm_example IS
    TYPE t_state IS (s_e1, s_e2, s_e3);     -- declaration of new datatype
    SIGNAL s_state, s_nextstate :  t_state; -- 2 signals of the new datatype

BEGIN

  ---------------------------------------------------
  -- PROCESS FOR COMB-INPUT LOGIC
  ---------------------------------------------------
  fsm_drive : PROCESS (s_state, d, n)
  BEGIN
    -- Default Statement
    s_nextstate <= s_state;

    CASE s_state IS
        WHEN s_e1 =>
            IF (d = '1') THEN
                s_nextstate <= s_e3;
            ELSIF (n = '1') THEN
                s_nextstate <= s_e2;
            END IF;
        WHEN s_e2 =>
            IF (d = '1') THEN
                s_nextstate <= s_e1;
            ELSIF (n = '1') THEN
                s_nextstate <= s_e3;
            END IF;
        WHEN s_e3 =>
            IF (d = '1') THEN
                s_nextstate <= s_e2;
            ELSIF (n = '1') THEN
                s_nextstate <= s_e1;
            END IF;
        WHEN OTHERS =>
            s_nextstate <= s_e1;
    END CASE;
  END PROCESS fsm_drive;


  ---------------------------------------------------
  -- PROCESS FOR COMB-INPUT LOGIC
  ---------------------------------------------------
  fsm_output : PROCESS (s_state, d, n)
  BEGIN

    CASE s_state IS
        WHEN s_e2 | s_e3 => z <= '1';
        WHEN OTHERS =>      z <= '0';
    END CASE;
  END PROCESS fsm_output;


```

```
72      ------------------------------------------------
73      -- PROCESS FOR REGISTERS
74      ------------------------------------------------
75      PROCESS (clk, reset)
76      BEGIN
77        IF (reset = '1') THEN
78            s_state <= s_e1;
79        ELSIF rising_edge(clk) THEN
80            s_state <= s_nextstate;
81        END IF;
82      END PROCESS;
83
84      END rtl;
85
```