No : Answers **Name : Solutions**

Eskişehir Osmangazi University Faculty of Engineering and Architecture

Department of Electrical Engineering & Electronics 13.01.2026

"*Introduction To VHDL-FPGA*" Final Exam

Note : Books, notes, computers are allowed, communication of all kind is prohibited. 90 minutes.

1. A key debouncer circuit and a pulse generator generating 1 for a 1 clk cycle in every 5ms (to be used in debouncer), are given as components as shown in the following code. Complete the code that debounces `N` inputs, where `N` is given as a `generic` value.

```
entity DBN is generic (N:integer:=30)
  port ( CLK : in  STD_LOGIC;
            I : in  STD_LOGIC_VECTOR(0 to N-1);
            O : out STD_LOGIC_VECTOR(0 to N-1));
end DBN;
architecture DBN of DBN is
  component DB is port (
    CLK, EN5ms : in STD_LOGIC;
    Ki : in  STD_LOGIC;
    Ko : out STD_LOGIC);
  end component;
  component Pulse5ms is port (
    CLK : in STD_LOGIC; Pulse : out STD_LOGIC );
  end component;
  signal Pulse : STD_LOGIC;
begin
  GENLOOP: for i in 0 to N-1 generate
    DB_i: DB port map (
      CLK => CLK,
      EN5ms => Pulse,
      Ki => I(i),
      Ko => O(i));
  end generate;
  P5ms_i: Pulse5ms port map ( CLK, Pulse );
end DBN;
```

2. Complete the state-machine that traverses states A->Ax->B->Bx->C->Cx->A->Ax… with the rising and falling edges of a clean input signal I. There are 6 states. Outputs are the values, 00, 01 and 10 corresponding to states A/Ax, B/Bx and C/Cx respectively. For example, starting from state A, when I goes high, state goes to Ax. When I goes low afterwards, state goes to B. When I goes to high when in B, state changes to Bx, so on and so forth.

```
entity STM is port (
  CLK, I : in  STD_LOGIC;
       O : out STD_LOGIC_VECTOR(1 downto 0));
end STM;
architecture STM of STM is
  type state is (A, Ax, B, Bx, C, Cx);
  signal pstate : state := A;
  signal nstate : state;
begin
  PC:process(pstate,I) is begin
    if(I='1') then
      case pstate is
        when A => nstate <= Ax;
        when B => nstate <= Bx;
        when others => nstate <= Cx;
      end case;
    else
      case pstate is
        when Ax => nstate <= B;
        when Bx => nstate <= C;
        when others => nstate <= A;
      end case;
    end if;
  end process;
  PSQ: process(CLK) is begin
    if(rising_edge(CLK)) then
      pstate <= nstate;
    end if;
  end process;
  O <= "00" when (pstate=A)or(pstate=Ax) else
       "01" when (pstate=B)or(pstate=Bx) else "10";
end STM;
```

3. Complete the code that outputs the index of an `integer inp` in an unordered list of positive `integer`s within a maximum of `Nc` clock cycles where `Nc` is the size of the list. The output `EoS` is set to 1 at the same clock cycle and set to 0 when `Rst` input is 1 which also initiates the search. Search starts when `Rst` goes to 0. If the value sought is not found, a negative value will be output along with the `EoS` indicator.

```
entity Search is port (
  CLK : in  STD_LOGIC;
  Rst : in  STD_LOGIC;  -- restart search
  inp : in  integer;    -- integer to be searched
  inx : out integer;    -- index found
  EoS : out STD_LOGIC); -- End of Search indicator
end Search;
architecture Search of Search is
  constant Nc : integer := 100;
  type ListT is array (0 to Nc-1) of integer;
  signal List: ListT := (521,12642,7,…example values…);
  signal i: integer;
  signal EoSx : STD_LOGIC := '0';
begin
  Eos <= EoSx;
  process(CLK) is begin
    if(rising_edge(CLK)) then
      if(Rst='1') then
        EoSx <= '0'; -- restart search
        i <= 0;
      elsif(EoSx='0') then
        if(List(i)=inp) then
          inx <= i;
          EoSx <= '1';
        elsif(i=Nc-1) then
          inx <= -1; -- not found
          EoSx <= 1;
        else
          i <= i+1;
        end if;
      end if;
    end if;
  end process;
end Search;
```