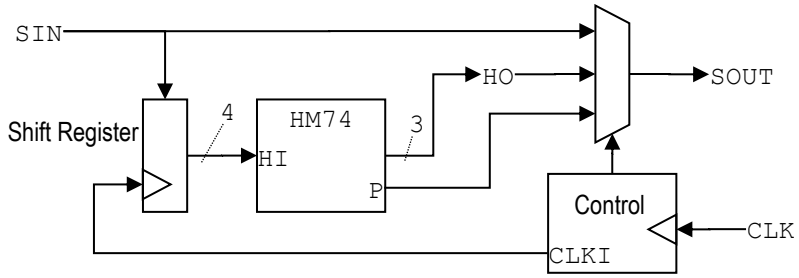
**No : Answers** **Name : Solutions**

**Eskişehir Osmangazi University - Faculty of Engineering and Architecture**

**Department of Electrical Engineering & Electronics**

"*Introduction To VHDL-FPGA*" Final Exam. 19.01.2021

The following is a block diagram of (7,4) error correcting coder circuit with serial-in and serial-out interface.




For each 4-bit input (serial-in) a 8-bit output (serial-out) is generated including additional parity bit. First 4 of these serialized bits are the original input bits, following 3 are codebits and 1 is the even-parity bit. Input bit-rate is half of the output bit-rate. Design the circuit in VHDL.

```
entity SCODER is Port (
  CLK, SIN : in STD_LOGIC;
  SOUT : out STD_LOGIC);
end SCODER;
architecture SCODER of SCODER is
  component HM74 is Port ( -- combinatorial encoder
    HI : in  STD_LOGIC_VECTOR(3 downto 0);
    HO : out STD_LOGIC_VECTOR(3 downto 0));
  end component;
-- your design below -
```

Since input bit-rate is half the output rate and it is not possible to generate 4 bit code (HM74 output) without receiving all 4 bits at the input, we have only two possibilities for implementation;

1. Delay the output until all conversion is done. (requires a 8-bit register before mux)
2. Store previous HM74 output and interlace it with the new input bits. (requires a 4-bit register after HM74)

The following is the first solution;

```
  signal SR,HO : STD_LOGIC_VECTOR(3 downto 0);
  signal HSOUT : STD_LOGIC_VECTOR(7 downto 0);
  signal SEL : integer range 0 to 7 :=0;
begin
  process(CLKI,SIN,SR) is begin -- input shift register
    if(RISING_EDGE(CLKI)) then
      SR <= SIN & SR(3 downto 1);
    end if;
  end process;
  HM74_inst: HM74 port map (HI => SR, HO => HO);
  control: process(CLK,CLKI,SEL) is begin
    if(RISING_EDGE(CLK)) then
      CLKI <= not CLKI; -- for input storage
      if(SEL=7) then SEL <= 0;
      else SEL <= SEL +1;
      if(SEL=0) then HSOUT <= SR & HO; end if;
      -- above is the reg. that stores everything after conversion
    end if;
  end process;
  SOUT <= HSOUT(SEL); -- output mux
end SCODER;
```

Duration for this question : thirty (30) minutes

The following is the second solution;

```
  signal SR,HO : STD_LOGIC_VECTOR(3 downto 0);
  signal HSOUT : STD_LOGIC_VECTOR(7 downto 0);
  signal HR    : STD_LOGIC_VECTOR(4 downto 0);
  signal SEL : integer range 0 to 7 :=0;
begin
  process(CLKI,SIN,SR) is begin -- input shift register
    if(RISING_EDGE(CLKI)) then
      SR <= SIN & SR(3 downto 1);
    end if;
  end process;
  HM74_inst: HM74 port map (HI => SR, HO => HO);
  control: process(CLK,CLKI,SEL) is begin
    if(RISING_EDGE(CLK)) then
      CLKI <= not CLKI; -- for input storage
      if(SEL=7) then SEL <= 0;
      else SEL <= SEL +1;
      if(SEL=0) then HR <= HO; end if;
      -- above is the reg. that stores HM74 output
    end if;
  end process;
  HSOUT <= SIN & HR(0) & SIN & HR(1) & SIN & HR(2) & SIN & HR(2);
  -- It looks like SIN is sent 4 times between HR bits. But its value is
  -- changed at every other CLK pulse, so they are different.
  -- Current input bits and previous code bits are interlaced.
  SOUT <= HSOUT(SEL); -- output mux
end SCODER;
```