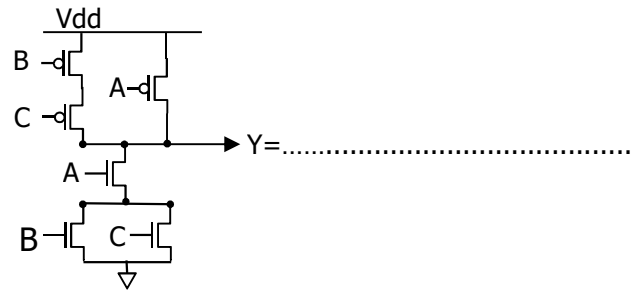
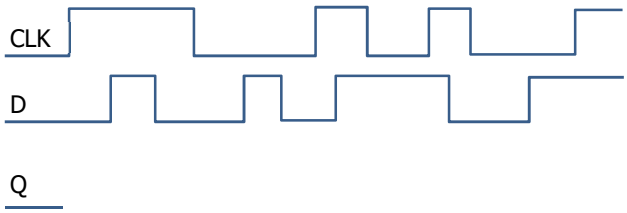
BỘ MÔN KTMT	MÔN : Thiết kế IC, IT4251 LẦN:...cuối kì.... LỚP : 97534 HỌ VÀ TÊN :..... STT:.....	ĐỀ
	Thời gian làm bài: ***90 phút*** *Ngày thi: 30/12/2017,* *Được phép sử dụng tài liệu* *- Khoanh tròn O lên đáp án trắc nghiệm, hoặc* *- Điền vào ô trống xám*	7295

PHẦN CÂU HỎI NHANH (4 điểm)

Câu 1. Cho thiết kế của 1 cổng logic gồm 6 transistor Biết rằng A, B, C là các tín hiệu vào tương ứng. Hãy viết biểu thức logic của đầu ra Y?



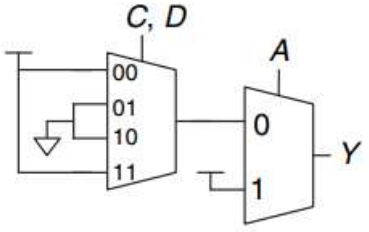
Câu 2. Một D Latch với dạng sóng tín hiệu vào như hình dưới. Hãy vẽ tín hiệu ra Q? Bỏ qua trễ



Câu 3. Cho bảng LUT – LookupTable 3 đầu vào. Hãy điền giá trị phù hợp vào các ngăn nhớ, để xây dựng hàm logic $Y = A + \bar{B}$. Chỉ rõ vai trò của A, B và I_2, I_1, I_0

	Địa chỉ	Ngăn nhớ	
	0		
	1		
I_2 ----→	2		
I_1 ----→	3		→ Y
I_0 ----→	4		
	5		
	6		
	7		

Câu 4. Hãy viết biểu thức logic của Y trong sơ đồ sau?



Y = ...

Câu 5. Hãy khoanh tròn vào đoạn mã viết đúng.

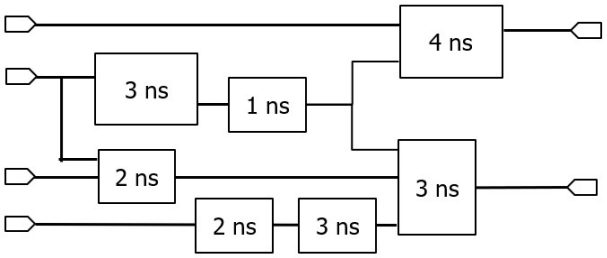
1)
```
process(clk) begin
    if rising_edge(clk) then
        q <= d;
end process;
```

2)
```
process(clk) begin
    if rising_edge(clk) then   q <= d;
    else   q <= q;
    end
end process;
```

3)
```
process(clk) begin
    if clk = '1' then q <= d;
    end if;
end process;
```

4)
```
process(clk, d) begin
    if clk = 0 then  q <= not d;
    end if
end process;
```

Câu 6. Sơ đồ sau thể hiện mạch tổ hợp với trễ truyền lan của từng thành phần. Hãy vẽ lát cắt thể hiện vị trí đặt thanh ghi pipeline, sao cho đạt được tốc độ tối ưu nhất? Được sử dụng 1 hoặc vô hạn thanh ghi pipeline.



Câu 7. Cho hàm logic $Y = A.B + \bar{A}.C$.
Phát biểu nào sau đây **sai**?

1) Xây dựng được Y từ duy nhất cổng NAND (không giới hạn số lượng)
2) Xây dựng được Y từ duy nhất phần từ D Latch
3) Xây dựng được Y từ 1 ROM 2^4 x 2-bit (4 đường địa chỉ, 2 đường dữ liệu
4) Xây dựng được Y từ 1 bộ chọn kênh 8x1

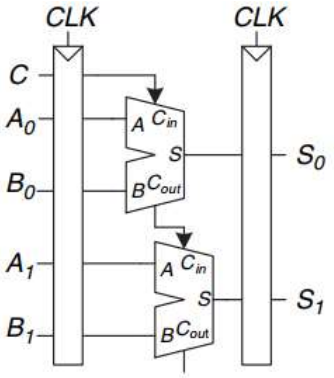
Câu 8.Với đoạn mã VHDL sau, Z bằng bao nhiêu?

```
Z <= A & B;
A <= "100";
B <= "101";
```

1) "100101"
2) không xác định
3) '1'
4) "100"

PHẦN CÂU HỎI TỰ LUẬN (6 điểm)

Câu 9. (2 điểm) Hình dưới đây là thiết kế của bộ cộng nhanh 2-bit trong bộ xử lý RePentium. Bộ cộng được thiết kế bởi 2 bộ toàn tổng với cờ Carry Out của bộ cộng trước nối với Carry In của bộ cộng sau. Bộ cộng cũng có thanh ghi đầu vào, thanh ghi đầu ra, và quá trình thực hiện phép cộng phải thực hiện xong trong một chu kì xung nhịp CLK.



Mỗi bộ toàn tổng có trễ truyền lan
- từ Cin **tới** S hoặc Cout là 22ps
- từ A, hoặc B **tới** S là 30ps
- từ A, hoặc B **tới** Cout là 27ps

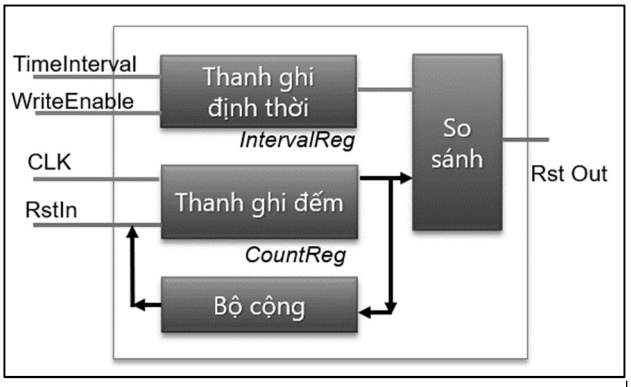
Mỗi bộ toàn tổng cũng có trễ lây nhiễm
- từ Cin **tới** S hoặc Cout là 12ps
- từ A, hoặc B **tới** S hoặc Count là 22ps

Mỗi flip-flop có tsetup = 30ps, thold = 10ps, tpcq = 35ps, tccq=21 ps

1) (0.5 đ) Trễ truyền lan của phần mạch tổ hợp gồm 2 bộ toàn tổng bằng bao nhiêu? Ghi rõ cách làm.
2) (0.5 đ) Trễ lây nhiễm của phần mạch tổ hợp nói trên bằng bao nhiêu? Ghi rõ cách làm.
3) (1.0 đ) Tần số xung nhịp CLK tối đa bằng bao nhiêu?

Câu 10.(2 điểm) Mạch watchdog là mạch phụ trợ, sẽ gửi tín hiệu reset (RstOut) tới mạch chính, nếu mạch chính không gửi reset tới mạch watchdog (Rst_In) sau khoảng thời gian nào đó. Sơ đồ nguyên lý và mã nguồn của watchdog như hình sau.

Hãy điền các lệnh còn thiếu vào ô trống màu xám




```
LIBRARY ieee;
    USE ieee.std_logic_1164.all;
    use ieee.std_logic_unsigned.all;

entity MyWatchDog is
    port (
      TimeInterval : in STD_LOGIC_VECTOR(4 downto 0);
      RstIn        : in STD_LOGIC;
      CLK          : in STD_LOGIC;
                                                    
                                                    
    );
end entity;

architecture EmGaiMua of MyWatchDog is
    signal IntervalReg : STD_LOGIC_VECTOR                 _
    signal    CountReg : STD_LOGIC_VECTOR                 _
begin
    IntervalProcess: process (WriteEnable)
    begin
        if rising_edge(WriteEnable) then
                                                          _
        end if;
    end process;

    CountProcess: process (CLK, RstIn)
    begin
        if (RstIn = '0') then
             CountReg <= (others=>'0');
        else
            if rising_edge(CLK) then
                                                          _
            end if;
        end if;
    end process;

    RstOut <= (CountReg(0) Xor IntervalReg(0)) or
              (CountReg(1) Xor IntervalReg(1)) or
              (CountReg(2) Xor IntervalReg(2)) or
                                                          _
                                                          _
end architecture;
```

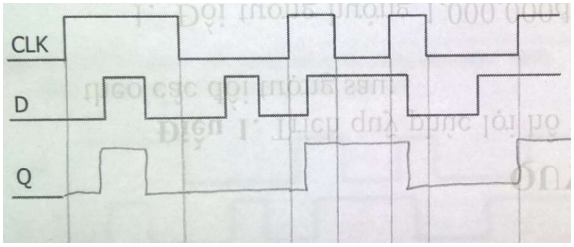
Câu 11.(2 điểm) Một bộ xử lý nhận mã lệnh từ bộ nhớ ROM thông qua bus địa chỉ lệnh 4-bit và bus lệnh 12-bit. Mỗi lệnh của bộ xử lý đều có độ dài bằng 12 bit. Sử dụng VHDL, hãy thiết kế bộ nhớ ROM cho bộ xử lý nói trên, với 3 lệnh máy 9BA, 28E, 1D bắt đầu từ địa chỉ số 2 với đầy đủ entity, architecture. với đầy đủ entity, architecture.

Lưu ý: trong code VHDL, phải viết tất cả nội dung lệnh máy ở dạng nhị phân.

ĐÁP ÁN

Câu 1: **Y =** $Y = A.\overline{C}.\overline{B} + \overline{A}$

Câu 2:



Câu 3:

Nếu A, B, C lần lượt là I0, I1, I2 thì đáp án là

	Địa chỉ	Ngăn nhớ	
	0	1	
	1	1	
I_2 ----→	2	0	
I_1 ----→	3	1	→ Y
I_0 ----→	4	1	
	5	1	
	6	0	
	7	1	

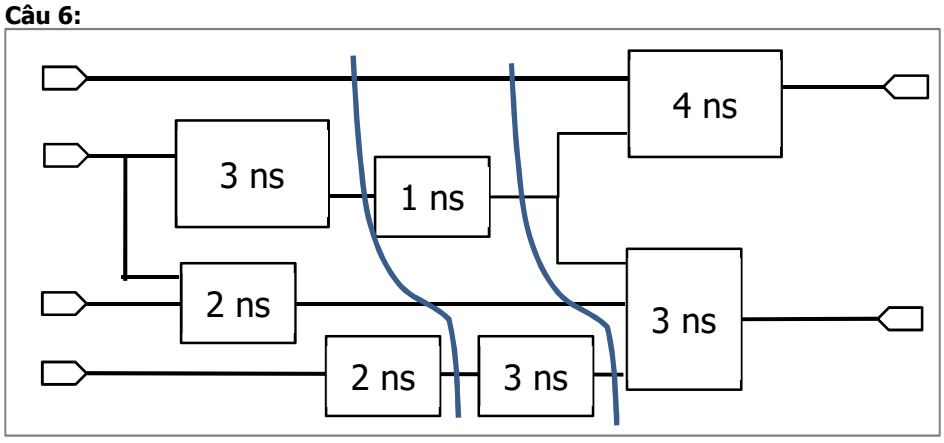
Nếu A, B, C lần lượt là I1, I0, I2 thì đáp án là

	Địa chỉ	Ngăn nhớ	
	0	1	
	1	0	
I_2 ----→	2	1	
I_1 ----→	3	1	→ Y
I_0 ----→	4	1	
	5	0	
	6	1	
	7	1	

Câu 4: $Y = A + \overline{A}.(\overline{C\ xor\ D})$

Câu 5: Đáp án 4

Câu 6:

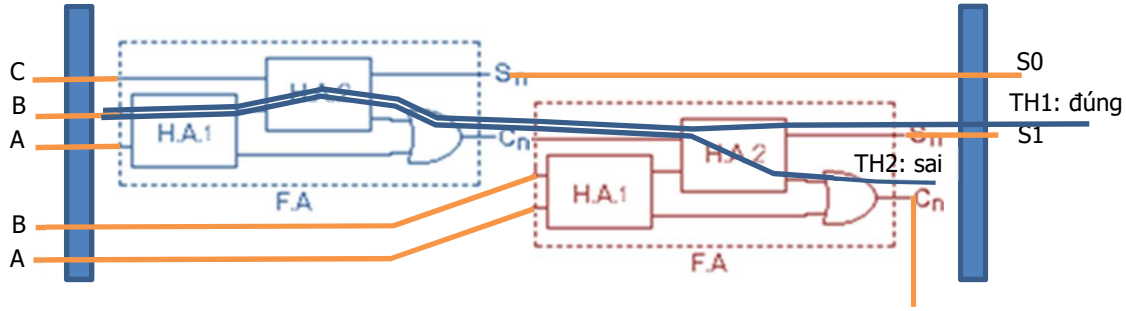


Câu 7: Đáp án 2 sai

Câu 8: Đáp án 1

Câu 9:

1) Tpd = Tpd(A-Cout) + Tpd(Cin-S) =27 + 22 = 49 ps
2) Tcd = Min(Tcd(Cin,S),Tcd(A,S),Tcd(A,Cout)) = Min(12,22,22)=12
3) Tc >= Tpcq + Tpd + Tsetup = 35 + 49 + 30 =114 ps → f = 1000/114 GHz = 8,7 GHz
   Thold < tccq + tcd <=> 10 < 21 + 12 = 33 okay



Câu 10

```
LIBRARY ieee;
    USE ieee.std_logic_1164.all;
    use ieee.std_logic_unsigned.all;

entity MyWatchDog is
    port (
      TimeInterval : in STD_LOGIC_VECTOR(4 downto 0);
      RstIn        : in STD_LOGIC;
      CLK          : in STD_LOGIC;
      WriteEnable  : in STD_LOGIC;
      RstOut       : out STD_LOGIC
    );
end entity;

architecture EmGaiMua of MyWatchDog is
    signal IntervalReg : STD_LOGIC_VECTOR (4 downto 0);
    signal    CountReg : STD_LOGIC_VECTOR (4 downto 0);
begin
    IntervalProcess: process (WriteEnable)
    begin
        if rising_edge(WriteEnable) then
            IntervalReg <=  TimeInterval;
        end if;
    end process;

    CountProcess: process (CLK, RstIn)
    begin
        if (RstIn = '0') then
             CountReg <= (others=>'0');
        else
            if rising_edge(CLK) then
                CountReg <= CountReg + 1;
            end if;
        end if;
    end process;

    RstOut <= (CountReg(0) Xor IntervalReg(0)) or
              (CountReg(1) Xor IntervalReg(1)) or
              (CountReg(2) Xor IntervalReg(2)) or
              (CountReg(3) Xor IntervalReg(3)) or
              (CountReg(4) Xor IntervalReg(4));
end architecture;
```

Câu 11:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity ROM is
        port(
              A: in STD_LOGIC_VECTOR(3 downto 0);    -- 4 bit dia chi
              D: out STD_LOGIC_VECTOR(11 downto 0);   -- 12 bit du lieu
            );
end ROM;

architecture behavior of ROM is
begin
       main:process (A)
       begin
              case A is
                     when "0010"    => D <= "100110111010";         --9BA
                     when "0011"    => D <= "001010001110";         --28E
                     when "0100"    => D <= "000000011101";         -- 1D
                     when others    => D <= (others => '0');
              end case;
       end process main;
end behavior;
```