TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN: ĐIỆN TỬ CÔNG NGHIỆP

ĐÁP ÁN ĐỀ THI CUỐI KỲ
HỌC KỲ I NĂM HỌC 2016-2017
Môn: Thiết kế vi mạch số dùng HDL
Mã môn học: DSIC330563
Đề số/Mã đề: 01. Đề thi có 2 trang.
Thời gian: 90 phút.

Câu 1: (2 điểm)

Sinh viên chọn 2 trong 5 câu sau đây:

a. Có bao nhiêu loại công nghệ thiết bị (Device Technologies) được sử dụng trong lĩnh vực thiết kế vi mạch? Mỗi loại hãy nêu ra tên, cấu trúc bên trọng và cho ví dụ về một IC mà anh/chị biết.

Hiện có 3 công nghệ được sử dụng để thiết kế vi mạch số:

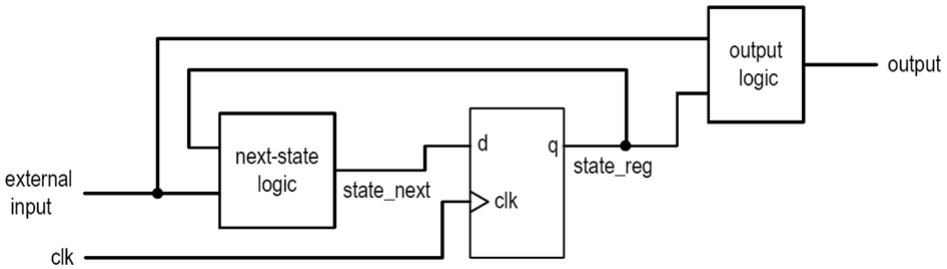
- ASIC Standard Cell:
  - Cấu trúc của Standard Cell ASIC là tập hộp những Logic Cell được cấu thành từ nhiều loại cổng Logic cho mỗi cell.
  - Những loại IC này thường được sản xuất trong nhà máy và được xuất xưởng với 1 chức năng cụ thể không thể thay đổi được trong quá trình sử dụng.
- ASIC Gate array:
  - Cấu trúc của Gate Array là tập hợp những Logic Cell được cấu thành từ một loại cổng Logic cho môi cell. Thường cổng logic được sử dụng là công NAND.
  - Những loại IC này thường được sản xuất trong nhà máy và được xuất xưởng với 1 chức năng cụ thể không thể thay đổi được trong quá trình sử dụng.
- Non-ASIC FPGA/CPLD
  - Cấu trúc của FPGA là tập hợp những Logic Cell được cấu thành chủ yếu bởi những lookup-table và những cổng logic khác. Các Logic Cell có thể lập trình kết nối, và những thành phần nhỏ trong mỗi cell cũng có thể được lập trình kết nối.
  - Những loại IC này thường được quy định chức năng bởi người sử dụng. Khi xuất xưởng, IC này chưa có chức năng cụ thể.

b. Định nghĩa Abstraction trong thiết kế hệ thống vi mạch số? Liệt kê các lớp trong Abstraction và cho biết loại tín hiệu quan tâm, phần tử cơ bản của mỗi lớp.

- Abstraction là sự khái quát hóa những đặc tính hoạt động của mỗi lớp hệ thống vi mạch số. Mỗi lớp hoạt động của hệ thống vi mạch số được đặc trưng bởi tín hiệu quan tâm xử lý, các đại lượng quan tâm xử lý, cấu trúc cơ bản nhất của mỗi lớp.
- Abstraction được cấu tạo phổ biến bao gồm những lớp sau:
  - Processor level:
    - Đối tượng quan tâm xử lý là những hành vi của hệ thống được thực hiện dưới dạng lưu đồ chương trình;
    - Các đại lượng quan tâm là những biến số và câu lệnh trong chương trình để thực hiện từng bước trong lưu đồ.
    - Thành phần cơ bản nhất là các lệnh trong lưu đồ chương trình.
  - Register Transfer Level:
    - Đối tượng quan tâm xử lý là tập hợp những bus dữ liệu tạo thành 1 con số cụ thể theo kiểu dữ liệu được quan tâm.
    - Các đại lượng quan tâm xử lý là những bus tín hiệu của từng khối trong hệ thống.
    - Thành phần cơ bản nhất của lớp này là các khối chức năng trong hệ thống. Mỗi khối có một chức năng cụ thể và kết hợp với nhau tạo thành một hệ thống hoàn chỉnh.
  - Logic Level:
    - Đối tượng quan tâm xử lý là từng bit dữ liệu để tạo thành bus dữ liệu hoàn chỉnh ở lớp Register Transfer Level.
    - Đại lượng quan tâm xử lý là giá trị logic 0 và 1 trong hàm logic.
    - Thành phần cơ bản nhất của lớp này là các cổng logic. Nhiều cổng Logic kết hợp lại thành khối logic bên trong RTL level.

- Physical Level:
  - Đối tượng quan tâm xử lý là các mức điện áp, tần số hoạt động và thời gian trễ của mạch.
  - Đại lượng quan tâm xử lý là các đại lượng vật lý cấp thấp như dòng điện, điện áp, tần số hoạt động, thời gian trễ.

c. Vẽ cấu trúc của mạch đồng bộ 3 thành phần và giải thích nguyên lý hoạt động.

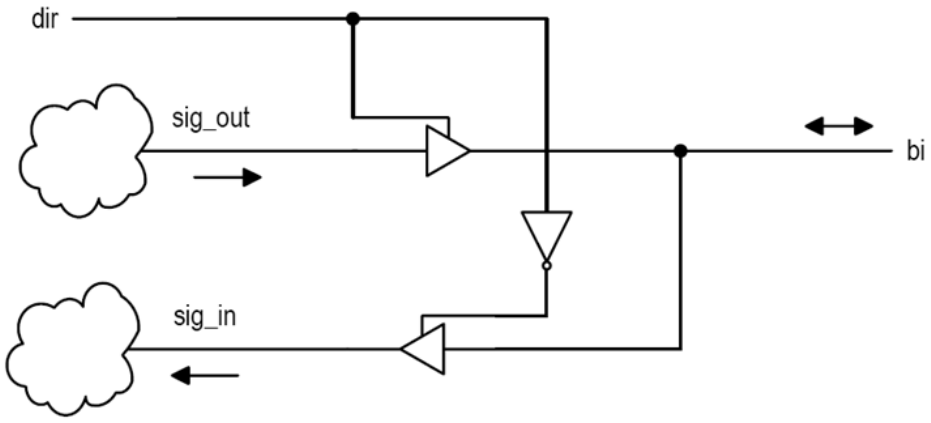


Giải thích chức năng từng khối:

- Khối Nex-State logic: Có chức năng tạo trạng thái mới dựa vào các tín hiệu ngõ vào và trạng thái hiện tại;
- Khối output logic: Có chức năng tạo ra tín hiệu ở ngõ ra dựa vào trạng thái hiện tại và các tín hiệu ngõ vào điều khiển;
- Mạch FlipFlop D: Có chức năng cập nhật trạng thái hiện tại bằng trạng thái mới;

Giải thích nguyên lý hoạt động: Ban đầu mạch đang ở trạng thái hiện tại, ngõ ra được gán một giá trị xác định bởi mạch tổ hợp ngõ ra dựa vào trạng thái hiện tại state_reg và ngõ vào điều khiển. Mạch tổ hợp ngõ vào tạo trạng thái mới state_next ở ngõ vào d của Flip Flop. Khi có cạnh lên clk tác động vào chân clk của Flip Flop, trạng thái hiện tại state_reg được cập nhật bằng trạng thái mới state_next. Quá trình tiếp tục khi có xung cạnh lên clk tiếp theo.

d. Vẽ cấu trúc của cổng 2 chiều dbùng 2 cổng đệm 3 trạng thái và viết chương trình VHDL mô tả cấu trúc này.

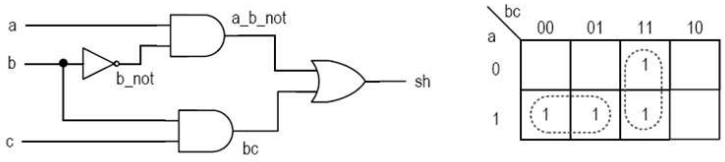



```
entity bi_demo is
   port(bi: inout std_logic;
 . . .

begin
   sig_out <= output_expression;
    . . .  <= expression_with_sig_in;
   . . .
   bi <= sig_out when dir='1' else 'Z';
   sig_in <= bi;
   . . .
```

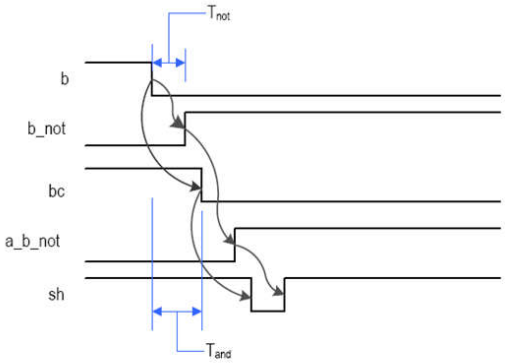
```
sig_in <= bi when dir='0' else 'Z';
```

e. Định nghĩa Hazard, Hazard tĩnh, Hazard động. Cho ví dụ một mạch điện tạo Hazard động và giải thích dạng sóng để chứng minh sự xuất hiện của Hazard.

Hazard là hiện tượng xảy ra trong khoảng thời gian trễ của hệ thống được định nghĩa là sự mất ổn định ở ngõ ra của mạch trước khi ổn định ở một trạng thái mà nó phải có.
Hazard tĩnh là sự mất ổn định xảy ra ở ngõ ra về mặt lý thuyết là ổn định.
Hazard động là sự bất ổn định xảy ra ở ngõ ra trong quá trình chuyển trạng thái. Về mặt lý thuyết, ngõ ra chuyển trạng thái từ 0 đến 1 hay từ 1 về 0 không có trạng thái trung gian.
Ví dụ một mạch điện tạo hiện tượng Hazard và giải thích dạng sóng:



(a) Karnaugh map and schematic



(b) Timing diagram

Câu 2: (2 điểm)

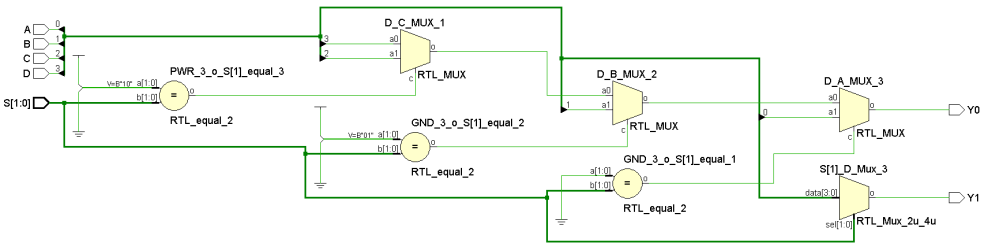
Sinh viên chọn 2 trong 5 câu sau đây:

a. Thiết kế mạch giải đa hợp từ 1 sang 4 có ngõ vào là I, ngõ ra là Y[3:0] sử dụng VHDL bằng 2 cấu trúc: lệnh gán có điều kiện, lệnh gán có lựa chọn. Vẽ sơ đồ khái niệm mỗi cấu trúc và giải thích cấu trúc nào phù hợp nhất.

Chương trình VHDL mô tả 2 cấu trúc:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity Mux4To1 is
    Port ( A, B, C, D : in  STD_LOGIC;
           Y0, Y1 : out  STD_LOGIC;
           S : in  STD_LOGIC_VECTOR (1 downto 0));
end Mux4To1;
architecture Behavioral of Mux4To1 is
begin
Y0 <= A WHEN S = "00" ELSE
    B WHEN S = "01" ELSE
    C WHEN S = "10" ELSE
    D;
WITH S SELECT
    Y1 <= A WHEN "00",
        B WHEN "01",
        C WHEN "10",
        D WHEN OTHERS;
end Behavioral;
```

Sơ đồ khái niệm dùng 2 cấu trúc:



Screen clipping taken: 28/12/2016 13:16

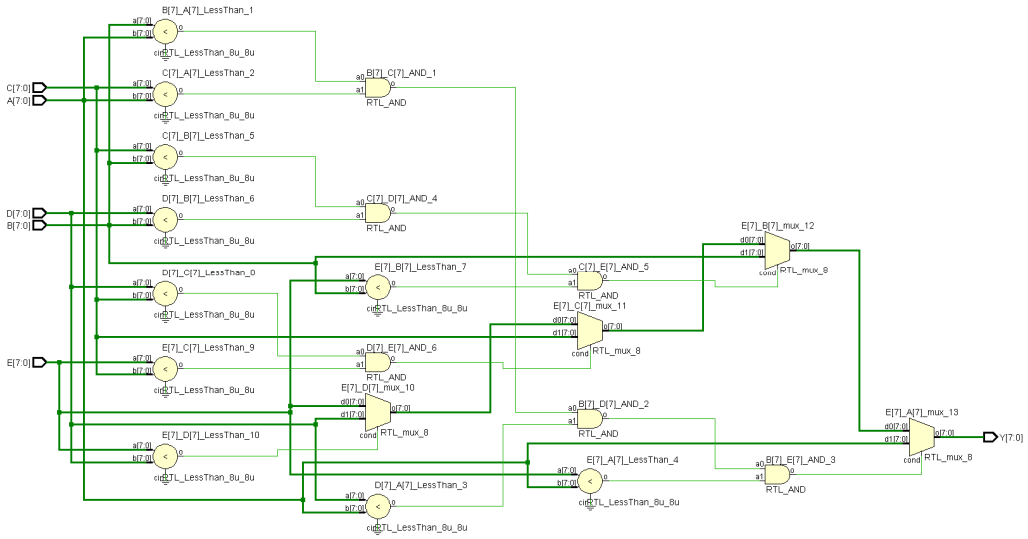
Trong 2 cấu trúc trên thì cấu trúc dùng lệnh gán có lựa chọn là phù hợp nhất vì sơ đồ khái niệm ngõ ra đúng là mạch mux từ 4 sang 1, hơn nữa các ngõ vào không có sự phân biệt về độ ưu tiên và thời gian trễ.

b. Thiết kế mạch tìm giá trị lớn nhất của 5 số. Ngõ vào là A, B, C, D, E có 8 bits, ngõ ra là MAX có 8 bits. Vẽ sơ đồ khái niệm của mạch.

Chương trình VHDL mô tả mạch tìm giá trị lớn nhất:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity MAX5INPUTS is
    Port (
    A, B, C, D, E : in  STD_LOGIC_VECTOR (7 downto 0);
    Y : out  STD_LOGIC_VECTOR (7 downto 0));
end MAX5INPUTS;

architecture Behavioral of MAX5INPUTS is

begin
    Y <= A WHEN (A > B) AND (A > C) AND (A > D) AND (A > E) ELSE
         B WHEN (B > C) AND (B > D) AND (B > E) ELSE
         C WHEN (C > D) AND (C > E) ELSE
         D WHEN (D > E) ELSE
         E;
end Behavioral;
```

Sơ đồ khái niệm của mạch tìm giá trị lớn nhất:



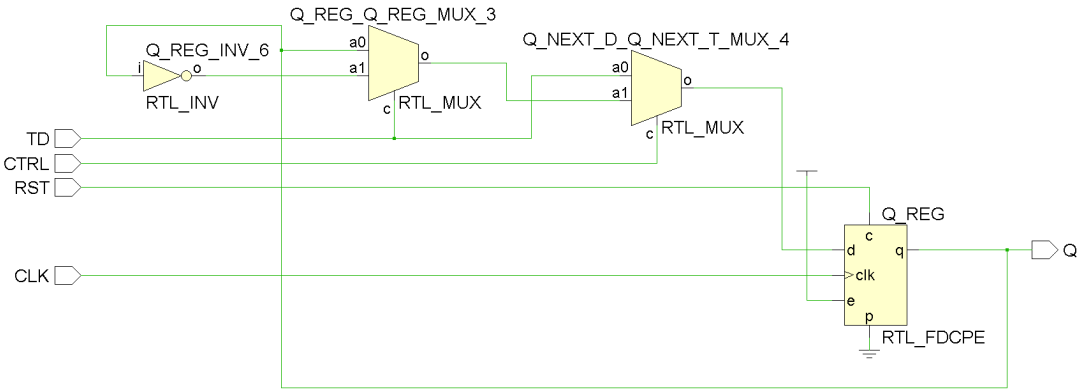
c. Thiết kế mạch tạo Flip Flop D hoặc T được điều khiển bằng chân chọn lựa. Ngõ vào là CK, Ctrl, D, ngõ ra là Q. Khi Ctrl = '0' mạch trở thành Flip-Flop D, khi Ctrl = '1' mạch trở thành Flip-Flop T.

Chương trình VHDL mô tả FlipFlop:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
entity FlipFLopTD is
    Port ( TD, CTRL : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           CLK : in  STD_LOGIC;
           RST : in  STD_LOGIC);
end FlipFLopTD;

architecture Behavioral of FlipFLopTD is
SIGNAL Q_REG, Q_NEXT_T, Q_NEXT_D, Q_NEXT : STD_LOGIC;
begin
-- D FLIP FLOP
PROCESS(CLK, Q_NEXT, RST)
BEGIN
    IF RST = '1' THEN
        Q_REG <= '0';
    ELSIF RISING_EDGE(CLK) THEN
        Q_REG <= Q_NEXT;
    END IF;
END PROCESS;
-- INPUT LOGIC
Q_NEXT_T <= NOT(Q_REG) WHEN TD = '1' ELSE
    Q_REG;
Q_NEXT_D <= TD;
Q_NEXT <= Q_NEXT_T WHEN CTRL = '1' ELSE
    Q_NEXT_D;
-- OUTPUT LOGIC
Q <= Q_REG;
end Behavioral;
```

Sơ đồ khái niệm của chương trình đã vẽ:



Screen clipping taken: 28/12/2016 13:41

d. Thiết kế mạch đếm mã Gray 8 bits sử dụng mô hình đồng bộ 3 thành phần. Ngõ vào là CK, RST, E, ngõ ra là Q[7:0].

Chương trình VHDL mô tả mạch đếm mã Gray:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
Use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity GRAYCOUNTER is
    Port ( CLK, RST, ENA : in  STD_LOGIC;
           Q : out  STD_LOGIC_VECTOR (7 downto 0));
end GRAYCOUNTER;
```

```
architecture Behavioral of GRAYCOUNTER is
SIGNAL Q_REG, Q_NEXT : STD_LOGIC_VECTOR(7 DOWNTO 0) := "00000000";
begin
-- D FLIP FLOP
PROCESS(CLK, RST, ENA)
BEGIN
    IF RST = '1' THEN
        Q_REG <= "00000000";
    ELSIF RISING_EDGE(CLK) THEN
        Q_REG <= Q_NEXT;
    END IF;
END PROCESS;
-- INPUT LOGIC
Q_NEXT <= Q_REG + 1 WHEN ENA = '1' ELSE
    Q_REG;
-- OUTPUT LOGIC, CONVERT TO THE GRAY CODE
Q <= Q_REG XOR '0'&Q_REG(7 DOWNTO 1);
end Behavioral;
```

e. Thiết kế mạch nhân 4 bits sử dụng VHDL. Ngõ vào là A[3:0], B[3:0], ngõ ra là Y[7:0].

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

entity MULTIPLYING_4BITS is
    Port ( A, B : in  STD_LOGIC_VECTOR (3 downto 0);
           Y : out  STD_LOGIC_VECTOR (7 downto 0));
end MULTIPLYING_4BITS;

architecture Behavioral of MULTIPLYING_4BITS is
    SIGNAL B0, B1, B2, B3 : STD_LOGIC_VECTOR(3 DOWNTO 0);
    SIGNAL P0, P1, P2, P3 : STD_LOGIC_VECTOR(7 DOWNTO 0);
begin
    B0 <= (OTHERS=>B(0));
    B1 <= (OTHERS=>B(1));
    B2 <= (OTHERS=>B(2));
    B3 <= (OTHERS=>B(3));

    P0 <= "0000"&(A AND B0);
    P1 <= "000"&(A AND B1)&'0';
    P2 <= "00"&(A AND B2)&"00";
    P3 <= '0'&(A AND B3)&"000";

    Y <= (P0 + P1) + (P2 + P3);
end Behavioral;
```

Câu 3: (3 điểm)

Thiết kế mạch ALU theo bảng sự thật sau:

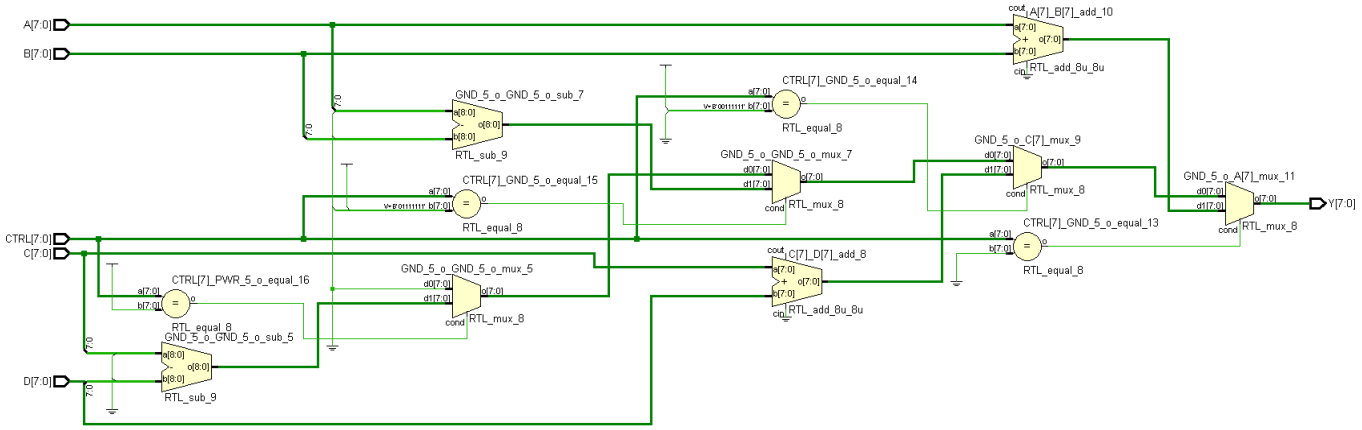
Ctrl[7:0]	Y[7:0]
0	A + B
63	C + D
127	A - B
255	C - D
Others	0

width	VHDL operator									
	nand	xor	$>_a$	$>_d$	=	$+1_a$	$+1_d$	$+_a$	$+_d$	mux
	area (gate count)									
8	8	22	25	68	26	27	33	51	118	21
16	16	44	52	102	51	55	73	101	265	42
32	32	85	105	211	102	113	153	203	437	85
64	64	171	212	398	204	227	313	405	755	171
	delay (ns)									
8	0.1	0.4	4.0	1.9	1.0	2.4	1.5	4.2	3.2	0.3
16	0.1	0.4	8.6	3.7	1.7	5.5	3.3	8.2	5.5	0.3
32	0.1	0.4	17.6	6.7	1.8	11.6	7.5	16.2	11.1	0.3
64	0.1	0.4	35.7	14.3	2.2	24.0	15.7	32.2	22.9	0.3

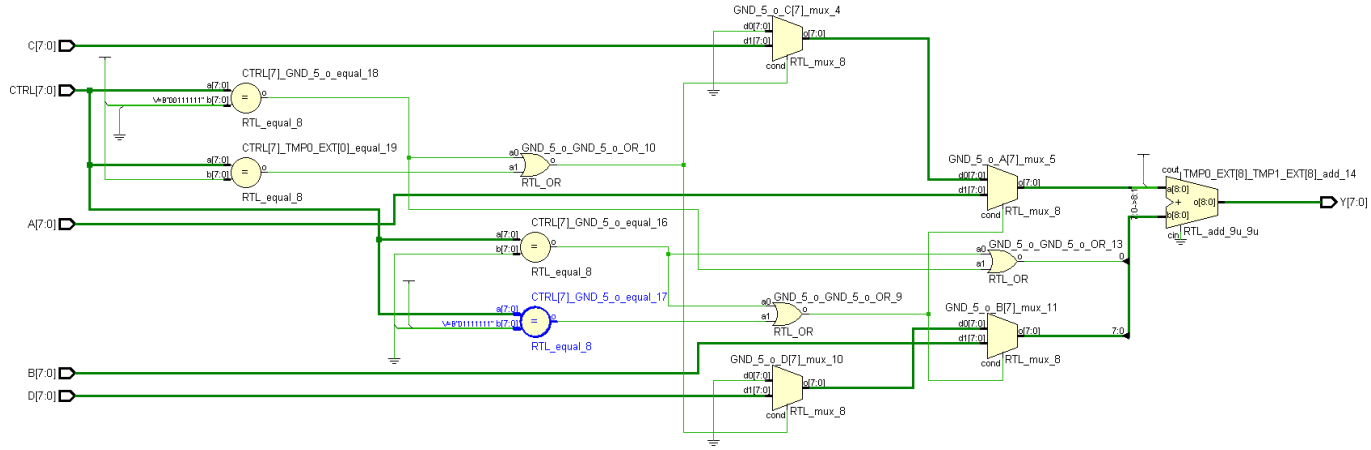
a. Mô tả mạch sử dụng VHDL sao cho dùng 2 mạch "+" và 2 mạch "-".

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ALU is
    Port ( A, B, C, D : in  STD_LOGIC_VECTOR (7 downto 0);
           Y : out  STD_LOGIC_VECTOR (7 downto 0);
           CTRL : in  STD_LOGIC_VECTOR (7 downto 0));
end ALU;

architecture Behavioral of ALU is

begin
    Y <= A + B WHEN CTRL = 0 ELSE
         C + D WHEN CTRL = 63 ELSE
         A - B WHEN CTRL = 127 ELSE
         C - D WHEN CTRL = 255 ELSE
         "00000000";
end Behavioral;
```

b. Vẽ sơ đồ khái niệm của mạch và tính toán tài nguyên và thời gian trễ.



c. Tối ưu sơ đồ khái niệm câu b sau cho chỉ sử dụng 1 mạch "+"; Tính toán tài nguyên và thời gian trễ của mạch.



Tài nguyên được sử dụng là:

- Cổng XOR: 3 cổng;
- Mạch so sánh "=" 8 bits: 3 mạch
- Mạch mux 2 sang 1 8 bits: 4 mạch
- Mạch "+" 9 bits: 1 mạch

Thời gian trễ của mạch:

- Đường nguy hiểm của hệ thống đi qua những linh kiện sau đây: Mạch so sánh "=" 8 bits, cổng OR, mạch mux 2 sang 1 8 bits, mạch cộng 9 bits.
- Thời gian trễ được tính bằng cách tính tổng các thời gian trễ của từng linh kiện được liệt kê.
  **Lưu ý: Tùy theo thiết kế của mạch mà tài nguyên và thời gian trễ sẽ khác nhau. Sinh viên tính theo những gì đã thiết kế. Câu này được tính đúng khi sinh viên tính theo phương pháp được hướng dẫn trong quá trình học.

d. Mô tả mạch vừa tối ưu ở câu c.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
entity ALU is
    Port ( A, B, C, D : in  STD_LOGIC_VECTOR (7 downto 0);
           Y : out  STD_LOGIC_VECTOR (7 downto 0);
           CTRL : in  STD_LOGIC_VECTOR (7 downto 0));
end ALU;
architecture Behavioral of ALU is
SIGNAL TMP0, TMP1 : STD_LOGIC_VECTOR(7 DOWNTO 0);
SIGNAL TMP0_EXT, TMP1_EXT, SUM_EXT : STD_LOGIC_VECTOR(8 DOWNTO 0);
SIGNAL C_IN : STD_LOGIC;
begin
    TMP0 <= A WHEN (CTRL = 0) OR (CTRL = 127) ELSE
        C WHEN (CTRL = 63) OR (CTRL = 255) ELSE
        "00000000";
    TMP1 <= B WHEN (CTRL = 0) OR (CTRL = 127) ELSE
        D WHEN (CTRL = 63) OR (CTRL = 255) ELSE
        "00000000";
    C_IN <= '1' WHEN (CTRL = 0) OR (CTRL = 63) ELSE
        '0';
    TMP0_EXT <= TMP0&'1';
    TMP1_EXT <= TMP1&C_IN;
    SUM_EXT <= TMP0_EXT + TMP1_EXT;
    Y <= SUM_EXT(8 DOWNTO 1);
end Behavioral;
```

Câu 4: (3 điểm)

Thiết kế mạch đếm đồng bộ dùng cấu trúc 3 thành phần theo yêu cầu sau:

Ctrl[1:0]	Chức năng
00	Mạch đếm lên BCD từ 0 đến 9
01	Mạch đếm xuống BCD từ 9 về 0
10	Mạch đếm lên từ 0 đến 15
11	Mạch đếm xuống từ 15 về 0

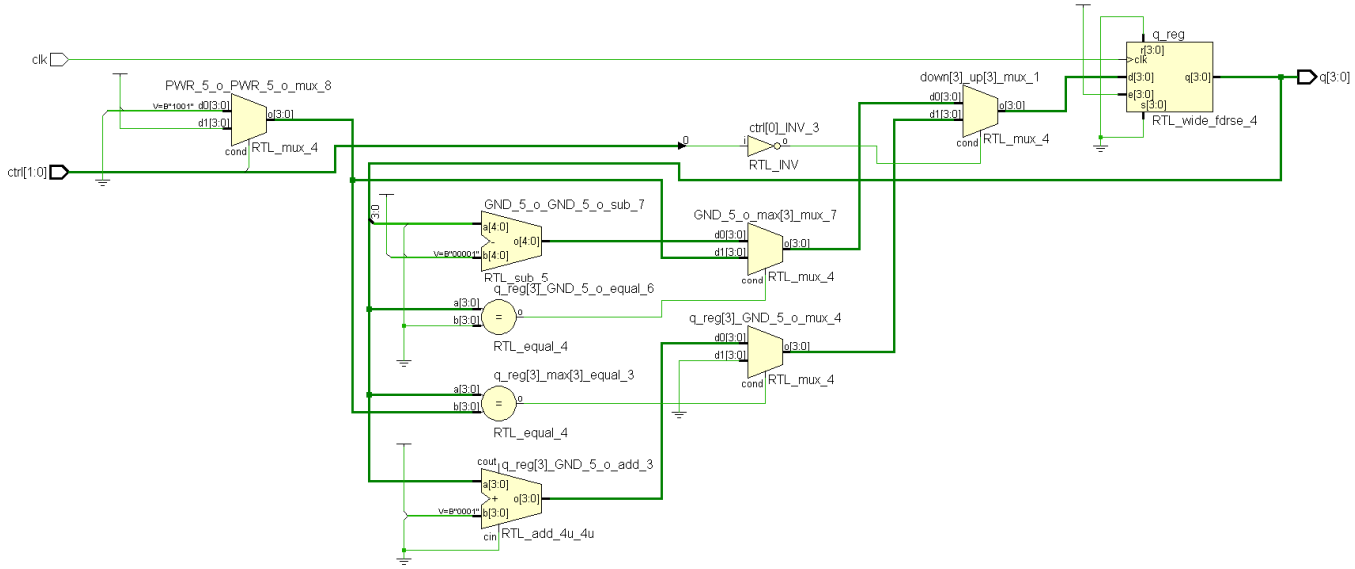
Mạch	Thời gian trễ
Mux 2-1 (4 bits)	2 ns
Mux 4-1 (4 bits)	2.5 ns
Adder (4 bits) "+"	4.5 ns
Subtractor (4 bits) "-"	5 ns
D Flip Flop	0.5 ns
Comparator (4 bits) "="	1 ns

a. Viết chương trình mô tả mạch đếm dùng VHDL bằng cấu trúc đồng bộ 3 thành phần.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;

entity Counter is
    Port ( ctrl : in  std_logic_vector (1 downto 0);
           clk : in std_logic;
```

```
                q : out  std_logic_vector (3 downto 0));
end Counter;

architecture Behavioral of Counter is
signal q_next, q_reg : std_logic_vector(3 downto 0) := "0000";
signal up, down : std_logic_vector(3 downto 0) := "0000";
signal max : std_logic_vector(3 downto 0);
begin
-- d flip flop
process(clk, q_next)
begin
    if rising_edge(clk) then
        q_reg <= q_next;
    end if;
end process;
-- input logic
q_next <= up when ctrl(0) = '0' else
    down;
up <= "0000" when q_reg = max else
    q_reg + 1;
down <= max when q_reg = "0000" else
    q_reg - 1;
max <= "1111" when ctrl(1) = '1' else
    "1001";
-- output logic
q <= q_reg;
end Behavioral;
```

b. Vẽ sơ đồ khái niệm của mạch.



c. Tính tần số hoạt động tối đa của mạch đếm này. Thời gian trễ của các linh kiện được cho trong bảng trên đây.

Thời gian trễ của mạch được tính toán theo sơ đồ khái niệm, vì thế tùy theo sơ đồ khái niệm đã vẽ mà có những giá trị thời gian trễ khác nhau của hệ thống. Câu này được tính là đúng khi sinh viên hoàn thiện đúng được những bước căn bản để tính tần số hoạt động của mạch đếm.

Những bước tính toàn thời gian trễ cho sơ đồ khái niệm của câu b.

- Xác định đường tín hiệu qua những linh kiện có thời gian trễ nhiều nhất: Đường tín hiệu có thời gian trễ dài nhất là đường đi qua những linh kiện như sau: Mạch cộng 4 bits, mạch mux từ 2 sang 1 4 bits, mạch mux từ 2 sang 1 4 bits, và flip flop D.
- Tổng thời gian trễ của những link kiện này là:

$$T_{Min} = T_{adder} + T_{mux} + T_{mux} + T_{flipflop} = 4.5 + 2 + 2 + 0.5 = 9ns$$

- Tần số hoạt động tối đa của mạch:

$$f_{Max} = \frac{1}{T_{Min}} = \frac{1}{9.10^{-12}} = 111.11GHz$$

Ghi chú: Cán bộ coi thi không được giải thích đề thi.

Chuẩn đầu ra của học phần (về kiến thức)	Nội dung kiểm tra
[CĐR 1.1]: Trình bày được các công nghệ thiết bị, quy trình thiết kế. Mô tả tổng quát, mô tả cấu trúc, mô tả hành vi, mô phỏng mạch. Trình bày tổng quan về ngôn ngữ mô tả phần cứng VHDL. Trình bày được cấu trúc ngôn ngữ cơ bản của VHDL.	Câu 1a, 1b, 1c, 1d, 1e Câu 2
[CĐR 2.1]: Hàm Big-O dùng để tính toán tài nguyên và hiệu suất của mạch theo ngõ vào. Phân tích, giải thích mạch điện của các toán tử sử dụng trong ngôn ngữ lập trình VHDL, tìm hàm Big-O. Phân tích thời gian trễ hệ thống, thời gian nguy hiểm, cách khắc phục.	Câu 1e Câu 3
[CĐR 2.2]: Phân tích chia sẻ toán tử, phân tích chia sẻ chức năng, phân tích bố trí linh kiện trong thiết kế để tìm cách tối ưu về tài nguyên, tối ưu về thời gian trễ. Phân tích và so sánh tài nguyên, thời gian trễ của mạch trước và sau khi tối ưu.	Câu 3
[CĐR 2.3]: Phương pháp thiết kế mạch tuần tự đồng bộ, thiết kế các mạch tuần tự cơ bản. Tính toán đáp ứng tần số cực đại của các mạch tuần tự.	Câu 4

Ngày 03 tháng 12 năm 2016

Thông qua Bộ môn

GVC. Ths. Nguyễn Đình Phú