GMU ECE 545

Fall 2007 Final

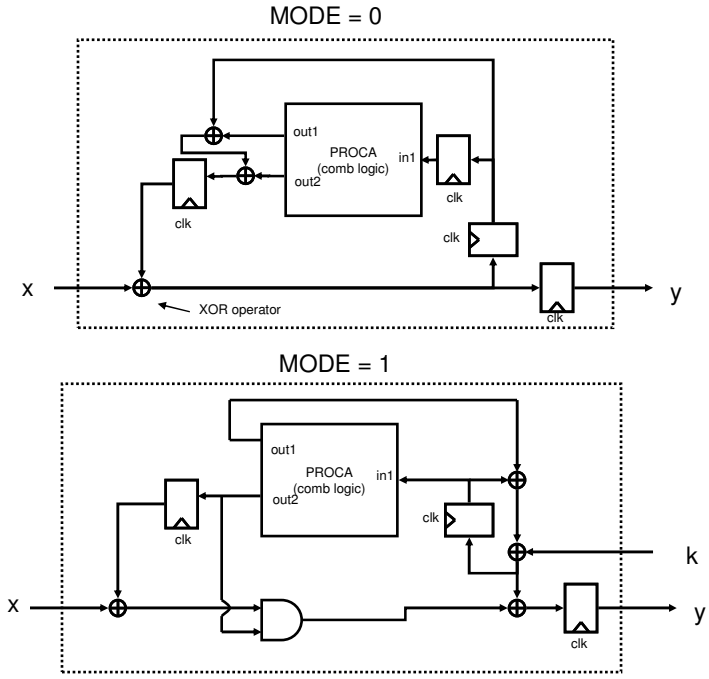
Name:_____

G Number:_____

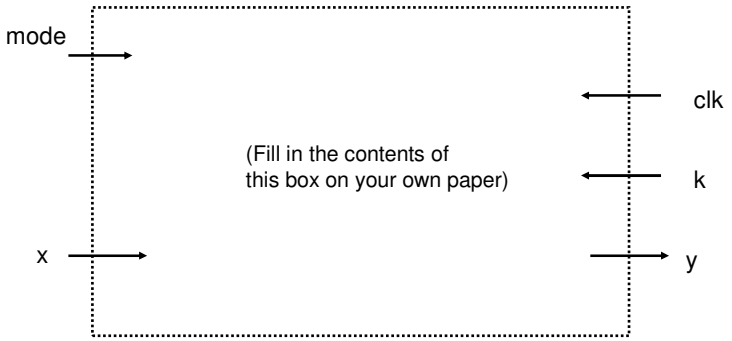
Problem 1: _____ / 5
Problem 2: _____ / 5
Problem 3: _____ / 5
Problem 4: _____ / 5
TOTAL: _____ / 20

Problem 1 (5 points)

The following circuit performs the logic below when configured in mode 0 and mode 1. Create a circuit which can perform both modes using a **mode** input pin. When mode = 0, the circuit is configured to perform mode 0. When mode = 1, the circuit is configured to perform mode 1. Assume x and y are a single bit only (not multi-bit). **Implement the circuit to have minimum area, i.e. no redundant hardware.**



(a) Draw the block diagram for the new circuit, using the following box as a template. (3 points)



(b) Write the VHDL code for the new circuit, **including library declaration, entity, and architecture**. Do not use process statements except to code the flip-flops. Assume the flip-flops have a clock called "clk" and no reset. (2 points)

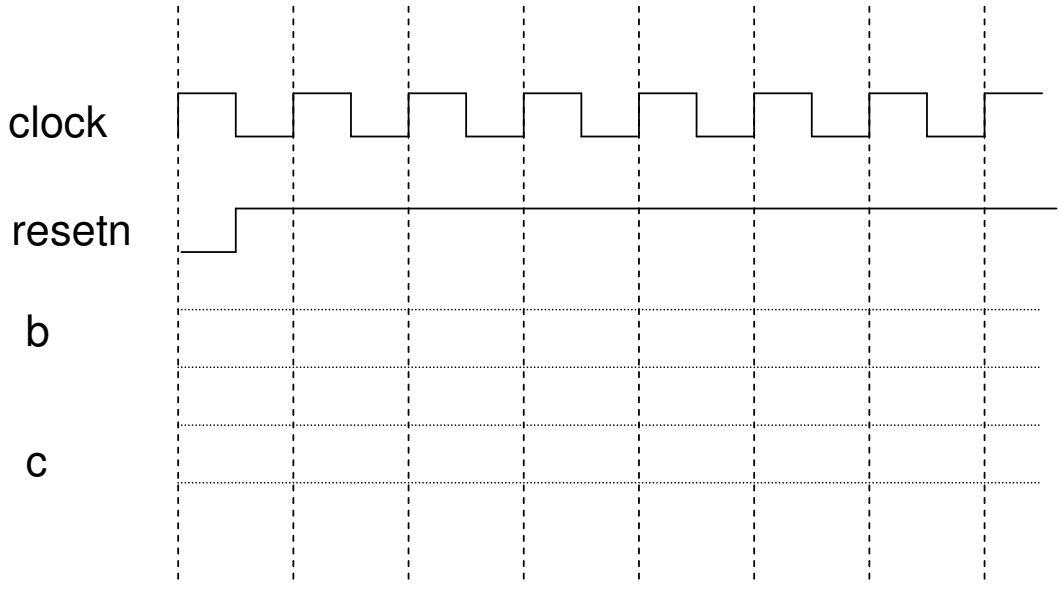
Assume the procedure PROCA, containing a combinational logic function, has been already defined in a package mypack in the work library. The procedure header in this package is:

procedure PROCA (in1: in std_logic; signal out1, out2: out std_logic);

Problem 2 (5 points)

(a) Consider the following VHDL code. Complete the waveform below for signals b and c. (3 points)

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity problem2 is
       port(
             clk, resetn: in std_logic;
             b,c : out std_logic_vector(3 downto 0));
end problem2;

architecture behavioral of problem2 is
      signal btmp, ctmp: std_logic_vector(3 downto 0);
begin
      process(clk,resetn)
            variable x,y: std_logic_vector(3 downto 0);
      begin
            if (resetn = '0') then
                  btmp <= "0000";
                  ctmp <= "0000";
            elsif (clk'event and clk='1') then
                  x := btmp;
                  y := ctmp;
                  x := x + 1;
                  btmp <= x + 2;
                  if (btmp(1)='1' and btmp(0)='0') then
                        btmp <= btmp + 1;
                  end if;
                  if (ctmp(1) = '1') then
                        y := y + 1;
                        ctmp <= ctmp + 3;
                  end if;
                  ctmp <= y + 1;
            end if;
      end process;
      b <= btmp;
      c <= ctmp;
end behavioral;
```



(b) The following VHDL code will give a synthesis warning when synthesized. Answer the two questions below the code.

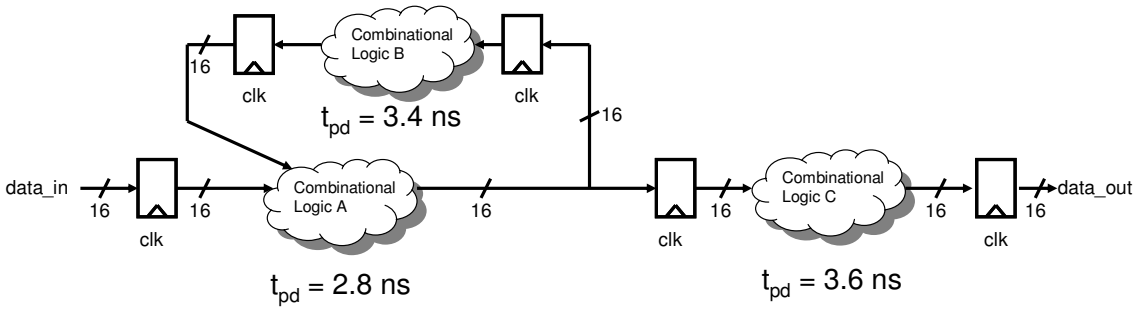
```
library ieee;
use ieee.std_logic_1164.all;

entity problem2b is
       port(
             a : in std_logic_vector(1 downto 0);
             b : out std_logic  );
end problem2b;

architecture dataflow of problem2b is
begin
      process(a)
      begin
            if (a = "00") then
                  b <= '1';
            elsif (a = "10") then
                  b <= '0';
            elsif (a = "11") then
                  b <= '1';
            end if;
      end process;

end dataflow;
```

Why does it give a synthesis warning? (1 point)

What code would you insert to remove this warning? (1 point)

Problem 3 (5 points)

Consider the circuit below.



Assume the setup time of each flip-flop (t_S) is 0.4 ns for all flip-flops. Assume the flip-flop propagation delay (t_{CLK2Q}) is 0.3 ns for all flip-flops. Assume the propagation delays for all combinational logic blocks are the values listed on the figure. Unless noted, assume there is no clock skew or jitter.

Note: If you do not have a calculator, leave the equations unsolved. However, be sure to put the correct units (ns, MHz, etc.)

Note: For full credit, show your equations as well as the final answer and make sure to include units.

(a) Draw the critical path of the circuit above. (0.5 points)

(b) What is the critical path delay of the circuit? (0.5 points)

(c) What is the maximum clock frequency of the circuit? (0.5 points)

(d) What is the maximum throughput of the circuit (i.e. throughput at maximum clock frequency)? (0.5 points)

(e) What is the latency from input to output in number of cycles? (0.5 points)

(f) What is the latency from input to output in nanoseconds, assuming maximum clock frequency? (0.5 points)

(g) Assume there is a clock jitter or skew of +/- 0.2 ns (i.e. clock skew can be positive or minus 0.2 ns between any two flip-flops). What is the maximum clock frequency of the circuit? (0.5 points)

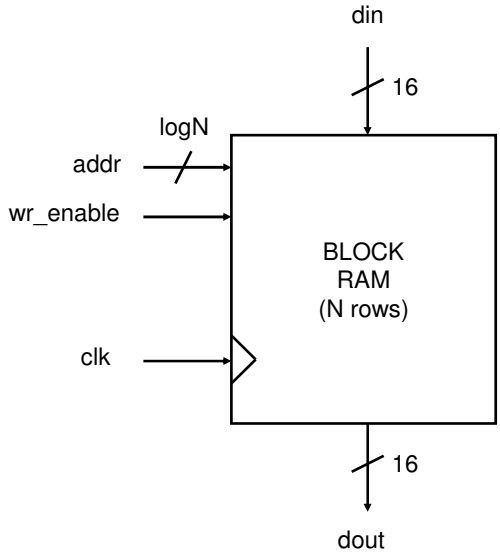
(h) Would pipelining help to reduce the critical path delay? (0.5 points)

(i) If pipelining would help question (h) and assuming a pipeline register can cut a combination logic block's delay exactly in half, where would you put the pipeline register? If pipelining would not help, put N/A. (0.5 points)

(j) If pipelining would help question (h) and assuming a pipeline register can cut a combinational logic block's delay exactly in half, what would be the new maximum clock frequency assuming no clock skew? If pipelining would not help, put N/A. (0.5 points)

Problem 4 (5 points)

(a) Code the following RAM in synthesizable VHDL code, **including library declaration, entity, and architecture**. Code such that the RAM will be inferred into a Xilinx Block RAM (i.e. synchronous read). Assume you are using XST, not Synplify Pro. Use generics for N and logN. The entity name should be blockram. All input and output ports should be std_logic or std_logic_vector. (2 points)



(b) Code the following block in synthesizable VHDL code, **including library declaration and architecture; you do not need to code the entity declaration**. Assume all signals are unsigned and resetn is an asynchronous active-low reset. (3 points)

For the adder block, the ovfl flag goes high when an overflow occurs in the addition. For the multiplier block, only the 16 MSBs of the multiplication product are kept.

Entity (you do not need to include this in your code):

```
entity problem4b is
   port (clk: in std_logic;
         resetn: in std_logic;
         dp_in: in std_logic_vector(15 downto 0);
         selA, selB, selC, wrena: in std_logic;
         addr: in std_logic_vector(4 downto 0);
         selD: in std_logic_vector(1 downto 0);
         dp_out: out std_logic_vector(15 downto 0);
         ovfl: out std_logic);
end problem4b;
```

Note: Assume blockram is a component declared in the package mypack in the work library.