Final Exam

(Time allowed: 120 minutes)
May 16, 2024

NOTE: Answer ALL **9** questions. Show all intermediate steps, except Question 1.

1. (10 points) Are the following statements necessarily true (Y) or not (N)?

(1) An edge-triggered D flip-flop changes states whenever the D input changes.
(2) A ring counter can use one flip-flop to represent two states in its sequence.
(3) For the Mealy machine, the outputs come directly from the combinational logic and not the memory.
(4) A synchronous counter is also known as an asynchronous counter.
(5) NAND gates can be used to produce the AND functions.
(6) Static RAM (SRAM) must be periodically refreshed to retain data.
(7) Registers are at the top of a memory hierarchy.
(8) A 2-input NAND gate and a 2-input negative-XOR gate have the same function.
(9) In the design of asynchronous counters with J-K flip-flops, all flip-flops should be connected to the same clock.
(10) In a computer, the overall processing speed is usually limited by the processor, not the memory.

Solution:

(1) N
(2) N
(3) Y
(4) N
(5) Y
(6) N
(7) Y
(8) N
(9) N
(10) N

CONTINUED

2. (10 points)

(1) (3 points)Convert the fractional binary number 10.0010101001_2 to octal *and* hexadecimal

(2) (3 points)Perform *binary* subtraction on the following signed numbers:

$$00110011 - 00100000$$

Show your result using the signed binary number.

(3) (4 points) Multiply 1011 by 1010 in binary form. Note that both numbers are in the 2's compliment form. Show your result using the signed binary number.

Solution:

(1)

$$10.1010101001_2 = 2.1244_8 = 2.2A4_{16}$$

(2) The 2's compliment of 00100000 is 11100000. Thus, we have

$$\begin{array}{r} 00110011 \\ +11100000 \\ =100010011 \end{array}$$

The answer is 00010011.

(3) The 2's compliment of 1011 and 1010 are 0101 and 0110, respectively.

$$\begin{array}{r} 0101 \\ \times 0110 \\ 0101 \\ 0101 \\ =011110 \end{array}$$

Therefore, the product is 011110.

CONTINUED

3. (10 points)

(1) (2 points) Develop a truth table for $\bar{A}BC + \bar{A}\bar{B}C + AB\bar{C}$

(2) (4 points) Convert $ABC + CD$ into the **standard** Product Of Sums (POS) form.

(3) (4 points) Convert $(A + B)(BC + D)$ into the **standard** Sum Of Products (SOP) form.

Solution:

(1)

A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

(2)

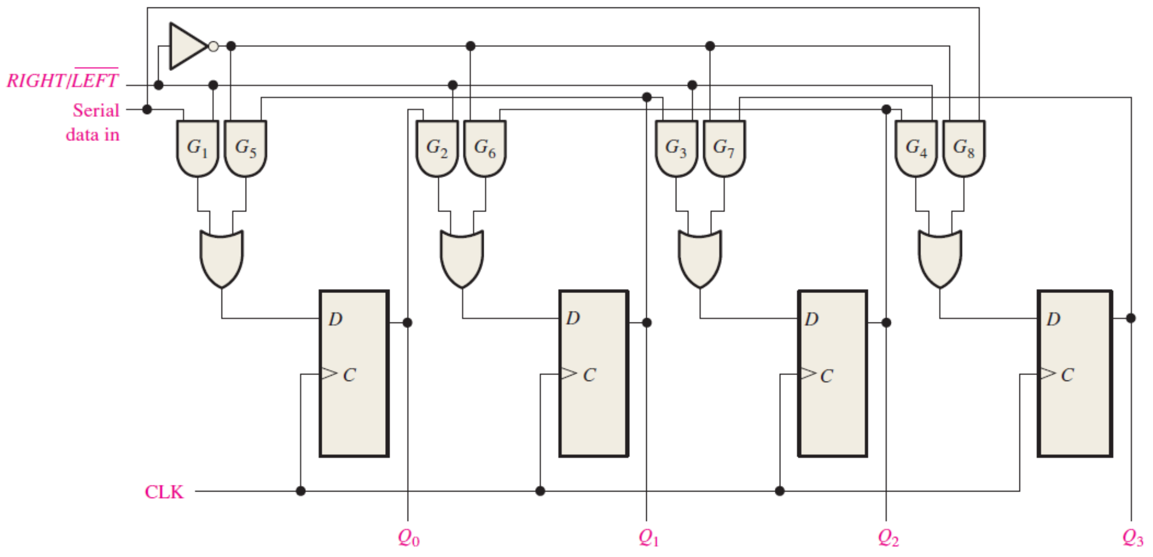
$$\begin{aligned}
&ABC + CD \\
=&(A + B + C + D)(A + B + C + \bar{D})(A + B + \bar{C} + D)(A + \bar{B} + C + D)(A + \bar{B} + C + \bar{D}) \\
&(A + \bar{B} + \bar{C} + D)(\bar{A} + \bar{B} + C + D)(\bar{A} + \bar{B} + C + \bar{D})(\bar{A} + B + C + D)(\bar{A} + B + C + \bar{D})(\bar{A} + B + \bar{C} + D)
\end{aligned}$$

(3)

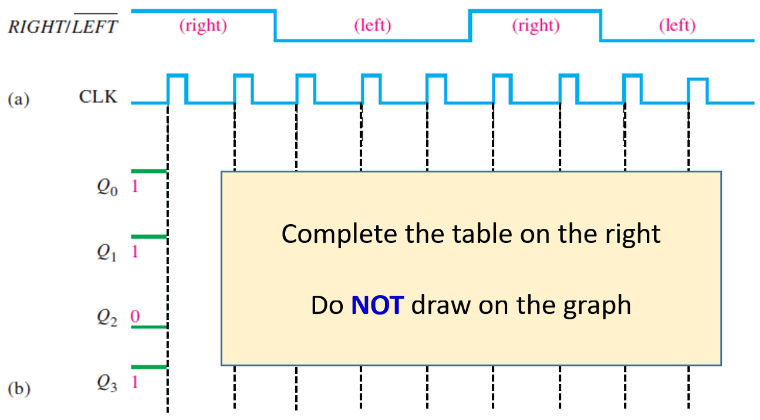
$$\begin{aligned}
&(A + B)(BC + D) \\
=&\bar{A}B\bar{C}D + \bar{A}BCD + \bar{A}BC\bar{D} \\
+&AB\bar{C}D + ABCD + ABC\bar{D} + A\bar{B}\bar{C}D + A\bar{B}CD
\end{aligned}$$

CONTINUED

4. (10 points) The following figure shows a 4-bit bidirectional shift register. A HIGH on the RIGHT/$\overline{\text{LEFT}}$ control input allows data bits inside the register to be shifted to the right, and a LOW enables data bits inside the register to be shifted to the left.



Determine the state of the shift register above after each clock pulse for the given RIGHT/LEFT control input waveform shown below. Assume that $Q_0 = 1$, $Q_1 = 1$, $Q_2 = 0$, and $Q_3 = 1$ and that the serial data-input line is LOW.



CLK	Q_0	Q_1	Q_2	Q_3
0	1	1	0	1
1				
2				
3				
4				
5				
6				
7				
8				

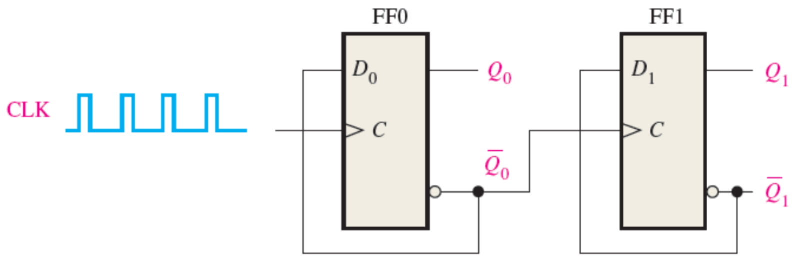
Solution:

CONTINUED

CLK	Q_0	Q_1	Q_2	Q_3
0	1	1	0	1
1	0	1	1	0
2	0	0	1	1
3	0	1	1	0
4	1	1	0	0
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	1	0	0

CONTINUED

5. (10 points) The following asynchronous counter of modulus 4 can be used as a frequency divider. More specifically, given a basic clock frequency of f_s Hz, the frequency of the Q_0 output waveform is $\frac{1}{2}f_s$ Hz while that of the Q_1 output waveform $\frac{1}{4}f_s$ Hz.



Now use two D flop flops and any necessary basic logic gates to design a *synchronous* counter of modulus 3 that generates an output waveform of $\frac{1}{3}f_s$ Hz.

(1) (3 points) Derive the next-state table;

(2) (4 points) Derive the expression for the input to each D flip flop;

(3) (3 points) Draw the circuit of the resulting counter of modulus 3.

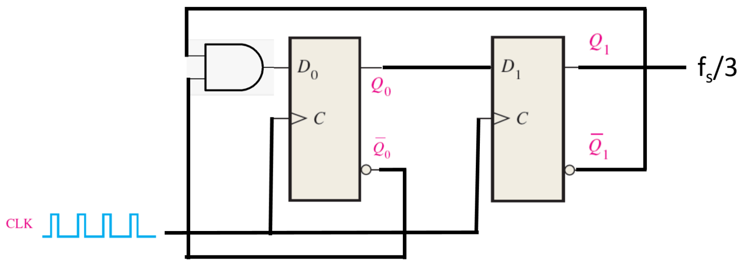
Solution:

(1) The next-state table is given as follows:

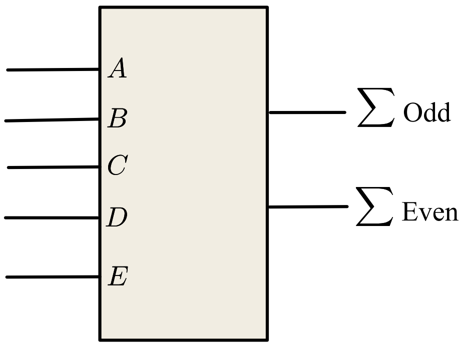
Present State		Next State			
Q_1	Q_0	Q_1	Q_0	D_1	D_0
0	0	0	1	0	1
0	1	1	0	1	0
1	0	0	0	0	0

(2) $D_1 = Q_0$ and $D_0 = \overline{Q}_1\overline{Q}_0$

(3) The resulting circuit is depicted in the following figure.

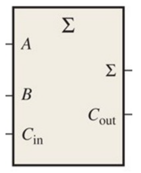


6. (12 points) A Parity Generator/Checker and its function table are shown below.

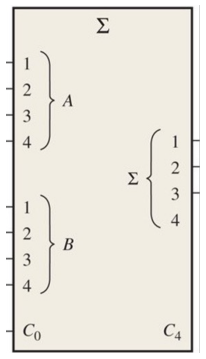


Number of Inputs A-E that Are High	**Outputs**	
	Σ Even	**Σ Odd**
0, 2, 4	H	L
1, 3, 5	L	H

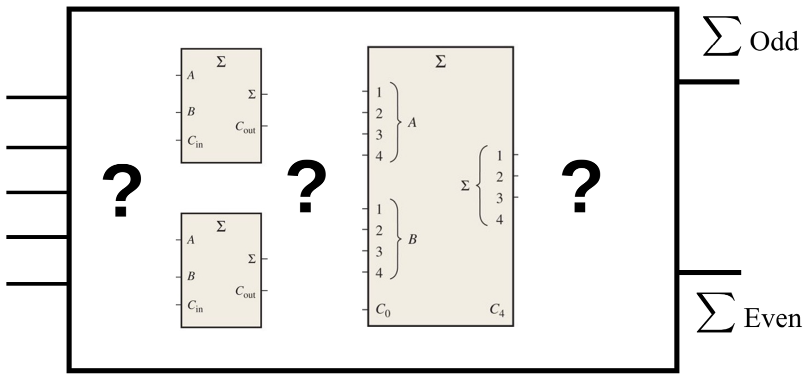
(1) (3 points) Use basic logic gates to design a 1-bit full adder as shown below where A and B are the input bits while C_{in} is the input carry. Finally, $\sum$ is the sum and C_{out} is the output carry.



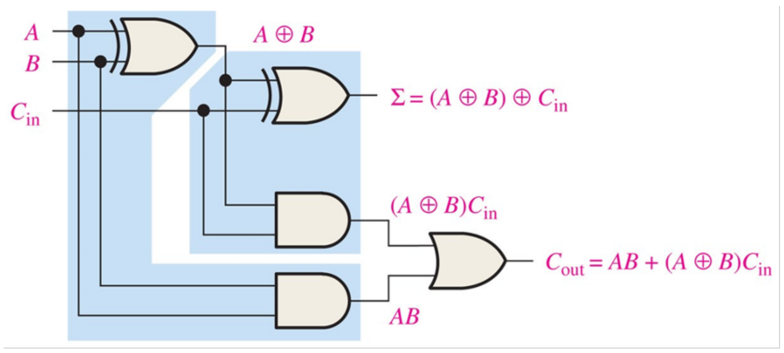
(2) (4 points) Use 4 1-bit full adders to implement a 4-bit adder as shown below. A and B are the 4-bit input number where $\sum$ is the 4-bit output. C_0 is the input carry and C_4 the is output carry.



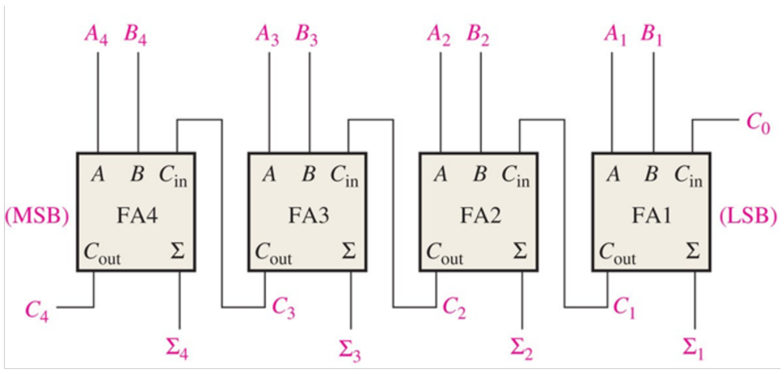
(3) (5 points) Finally, use two 1-bit full-adders and one 4-bit adder together with other basic logic gates to implement the Parity Generator/Checker.



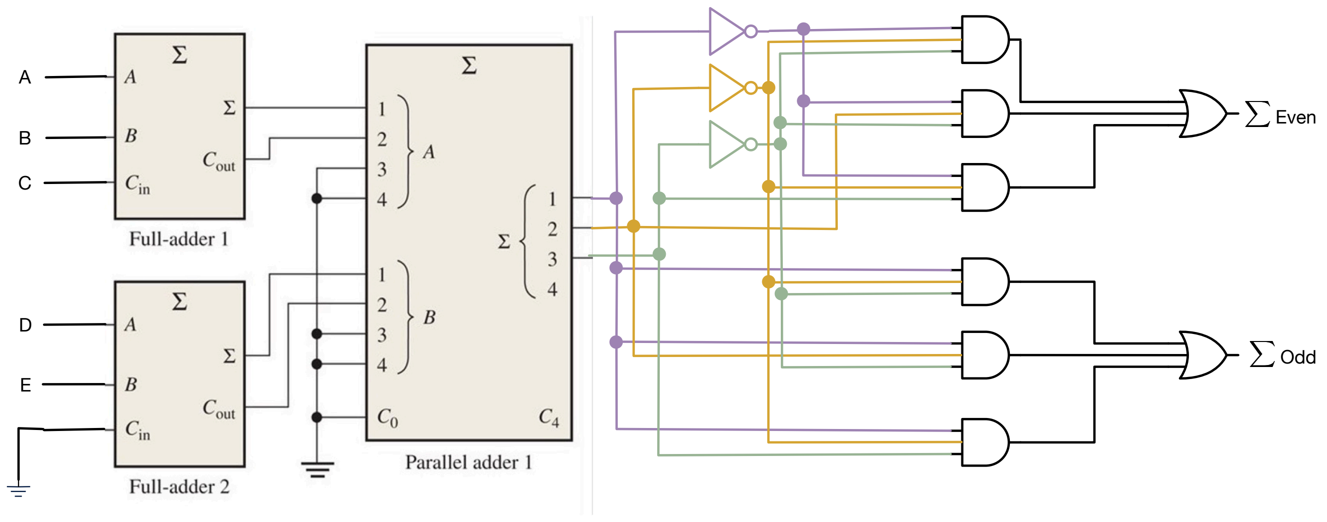
Solution:



A
B
C_{in}
$A \oplus B$
$\Sigma = (A \oplus B) \oplus C_{in}$
$(A \oplus B)C_{in}$
$C_{out} = AB + (A \oplus B)C_{in}$
AB



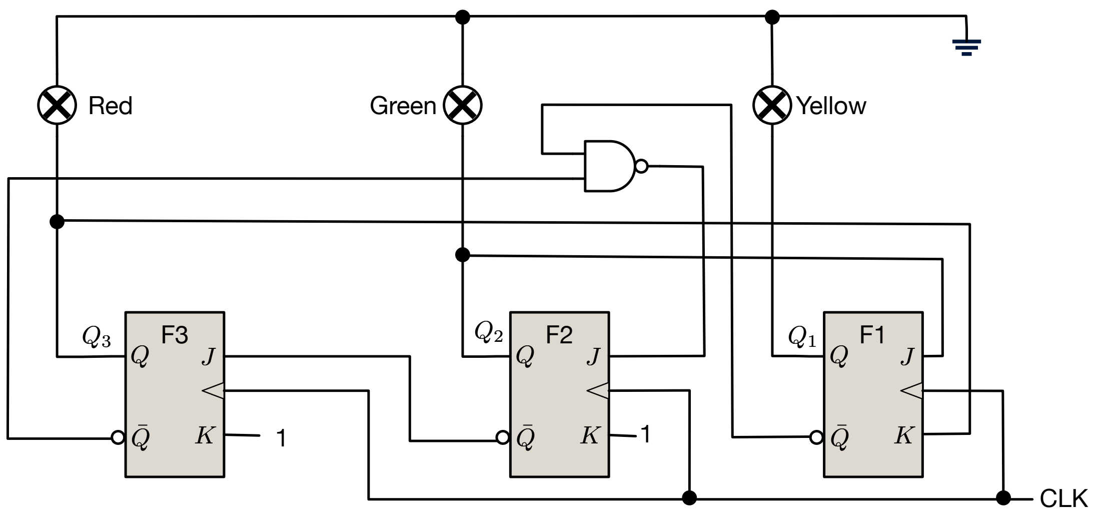
A_4 B_4
A_3 B_3
A_2 B_2
A_1 B_1
C_0
(MSB)
(LSB)
FA4
FA3
FA2
FA1
A B C_{in}
C_{out} Σ
C_4
C_3
C_2
C_1
Σ_4
Σ_3
Σ_2
Σ_1



A
B
C
D
E
Full-adder 1
Full-adder 2
Parallel adder 1
Σ
C_{out}
C_{in}
C_0
C_4
$\sum$ Even
$\sum$ Odd

CONTINUED

7. (15 points) A synchronous circuit with J-K Flip-Flops is shown below.



The truth table for a positive edge-triggered J-K flip-flop is given as follows.

Inputs			**Outputs**	
J	***K***	**CLK**	***Q***	$\bar{Q}$
0	0	↑	Q_0	$\bar{Q}_0$
0	1	↑	0	1
1	0	↑	1	0
1	1	↑	$\bar{Q}_0$	Q_0

↑ = clock transition LOW to HIGH
Q_0 = output level prior to clock transition

- The red light is on when $Q_3 = 1$, otherwise the red light is off;
- The green light is on when $Q_2 = 1$, otherwise the green light is off;
- The yellow light is on when $Q_1 = 1$, otherwise the yellow light is off.

(1) (4 points) Write down logic expressions for each Flip-Flop Input;

(2) (4 points) Write down the Next-State Table and draw the state diagram;

(3) (4 points) Write down the lighting sequence;

(4) (3 points) Assume the clock (CLK) frequency is 5Hz, write down each light on cumulative time during each state period.

Solution:

(1) $J_1 = Q_2$, $K_1 = Q_3$, $J_2 = Q_3 + Q_1$, $K_2 = 1$, $J_3 = \bar{Q}_2$, $K_3 = 1$.

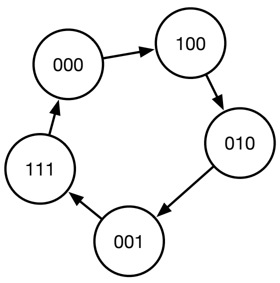
(2)

(3)

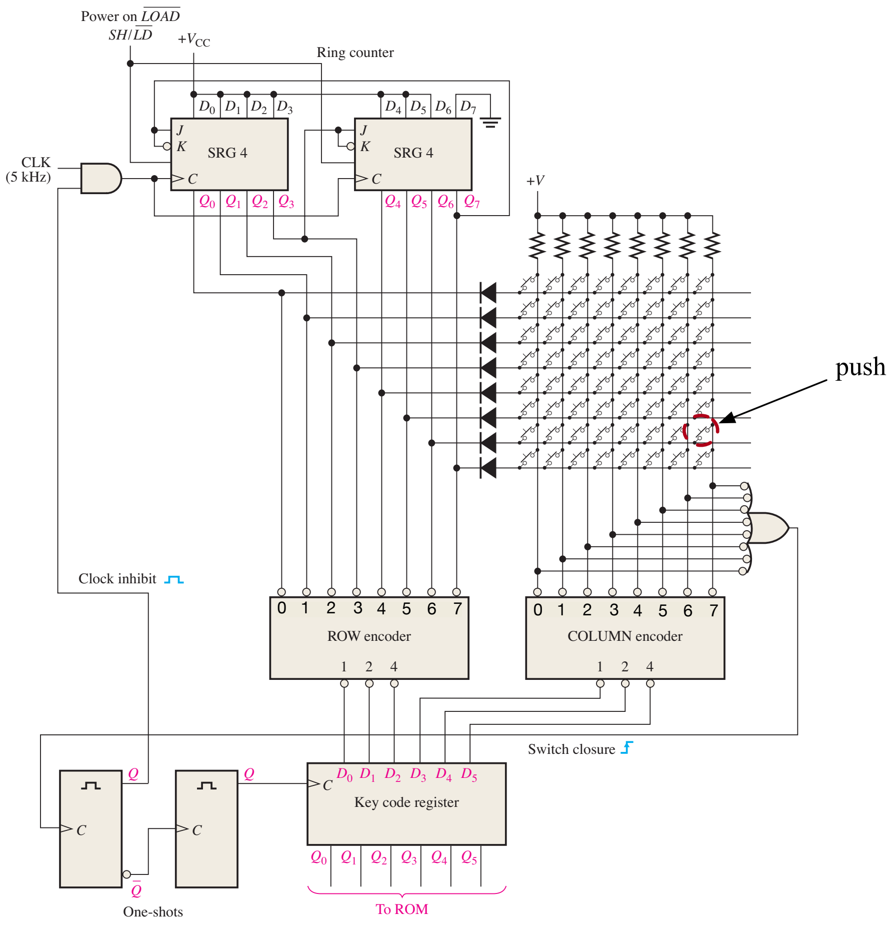
(4) Sequence: all off→ red on→ green on→ yellow on→ all on→ all off;

(5) red: 0.4s; green: 0.4s; yellow: 0.4s (period: 0.2s, all rights are on twice in each state period, there are 5 states.).

Present State			Next State		
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1
0	0	0	1	0	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	1	0	0	1
1	0	0	0	1	0
1	0	1	0	1	0
1	1	0	0	0	1
1	1	1	0	0	0

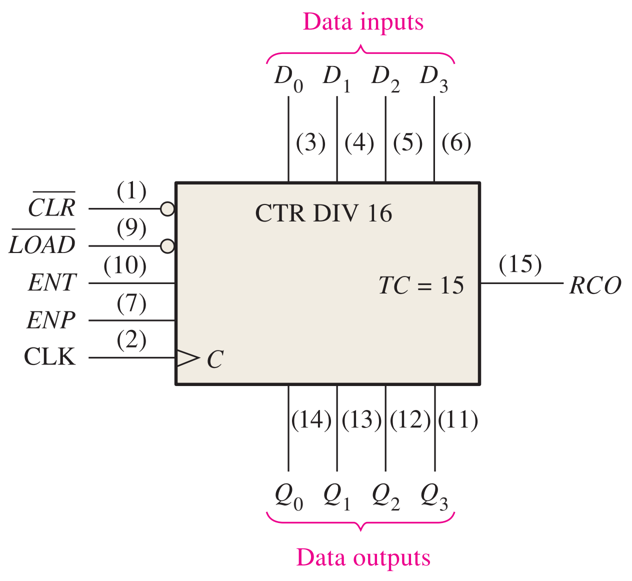


8. (19 points) A simplified keyboard encoding circuit is given below.



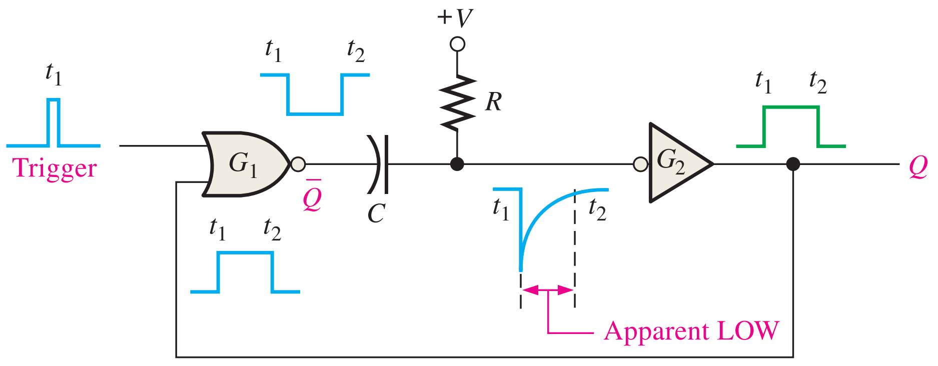
(1) (6 points) We only have a 100 MHz clock in hand. Design a circuit with several modulus-16 (binary) counters below to generate a 5 kHz clock (CLK). You can use other logic gates and set any value for each input;

CONTINUED



(2) (4 points) If we push the button labeled as "push" in the figure, what are the outputs of the ROW encoder and COLUMN encoder;

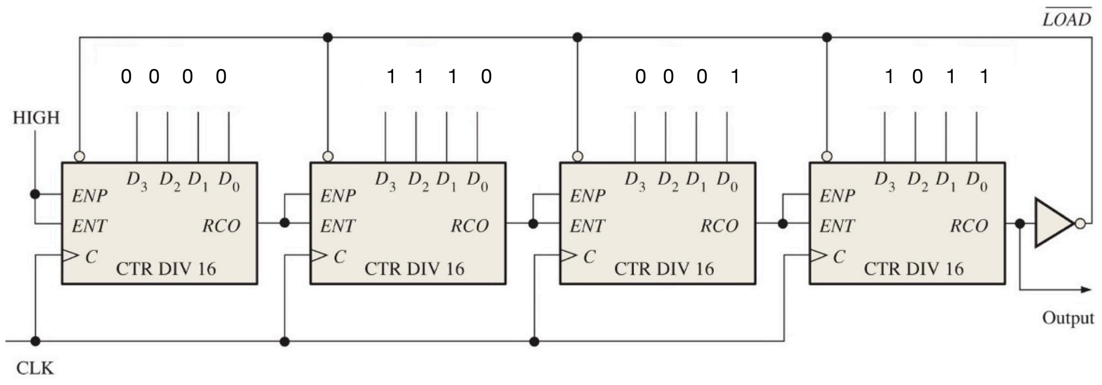
(3) (4 points) Assume the circuit below is used to implement the one-shot circuit. If we want to increase the pulse width, how to change the capacitor value C and resistor value R (increase or decrease)?



(4) (5 points) Use six JK Flip-Flops to implement the key code register.

Solution:

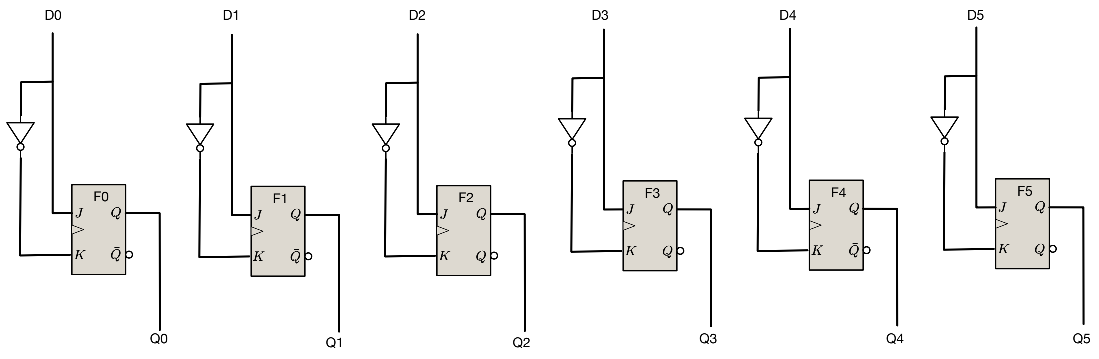
(1) We will need a modulus-(100000/5=20000) counter. 65536 - 45536 = 20000. 45536_{10} = 1011000111100000_2. A 5 kHz clock can be generated by:



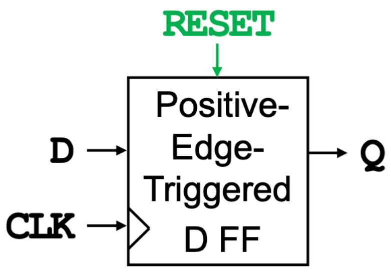
(2) ROW encoder: 001; COLUMN encoder: 000;

(3) Increase the capacitor or/and resistor values;

(4) The key code register is shown below.

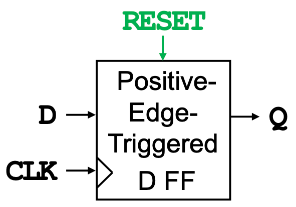


9. (4 points) Fill in the VHDL code below to implement a D flip-flop (DFF) with synchronous reset. (If you are not familiar with VHDL, you can use Verilog HDL to implement this DFF.)



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity DFF is
port(D,CLK,RESET: in std_logic;
                 Q: out std_logic);
end DFF;
architecture DFF_ARCH of DFF is
begin
  process(_______)
  begin
     if _____ and CLK'event then
        if (_______) then
          _________
        else
          Q <= D;
        end if;
     end if;
  end process;
end DFF_ARCH;
```

Solution:



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity DFF_SYNC is
port(D,CLK,RESET: in std_logic;
                Q: out std_logic);
end DFF_SYNC;
architecture DFF_SYNC_ARCH of DFF_SYNC is begin
begin
  process(CLK)
  begin
    if CLK = '1' and CLK'event then
       if (RESET = '1') then
          Q <= '0'; -- Reset Q anytime   ← Positive-edge- triggered
       else
          Q <= D; -- Q follows input D
       end if;
    end if;
  end process;
end DFF_SYNC_ARCH;
```

CONTINUED