***King Fahd University of Petroleum and Minerals***

***College of Computer Science and Engineering***

***Computer Engineering Department***

**COE 202: Digital Logic Design (3-0-3)**

**Term 122 (Spring 2013)**

**Major Exam II**

**Thursday April 18, 2013**

**Time: 150 minutes, Total Pages: 12**

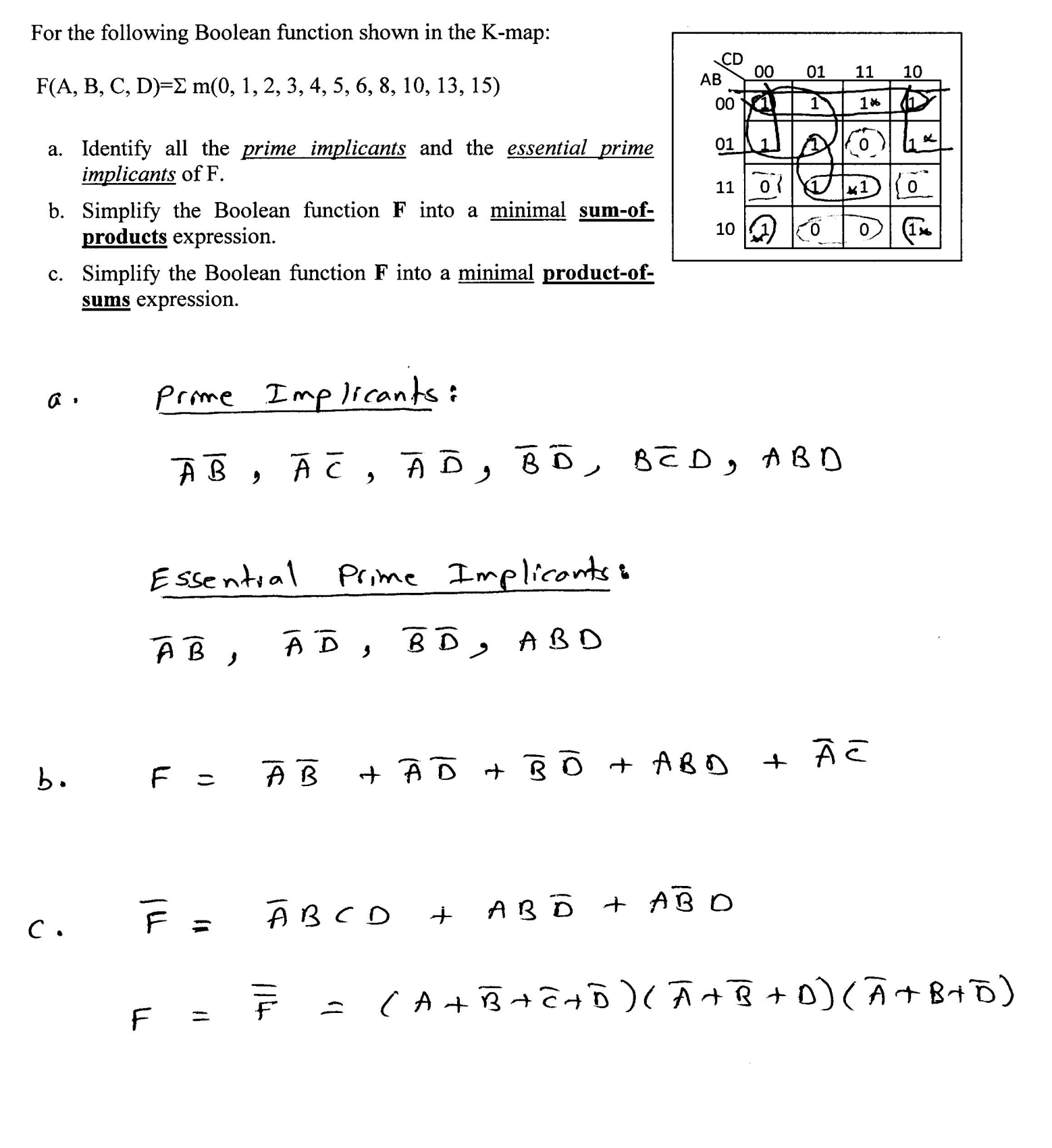
**Name:\_KEY\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_**

**Notes:**

* Do not open the exam book until instructed
* **Calculators are not allowed** (*basic, advanced, cell phones, etc*.)
* Answer all questions
* All steps must be shown
* Any assumptions made must be clearly stated

|  |  |  |
| --- | --- | --- |
| **Question** | **Maximum Points** | **Your Points** |
| **1** | **14** |  |
| **2** | **15** |  |
| **3** | **17** |  |
| **4** | **8** |  |
| **5** | **12** |  |
| **6** | **8** |  |
| **7** | **18** |  |
| **Total** | **92** |  |

**Question 1. (14 points)**



**Question 2. (15 Points)**



# It is required to design a Tripler circuit. The circuit receives an n-bit number X and computes the result Y=3\*X.

# If the input is an *n-*bit unsigned number, what is the size of the output “*y*” in bits?

# n+2



# The circuit can be constructed using *n identical copies* of the basic 1-bit cell shown to the right. The cell processes one input bit (X*i*) and produces one output bit (Y*i*) and two output carry bits (CO0 and CO1). To allow for cascading *n* such cells to implement an *n-*bit Tripler, the basic cell also accepts two input carry bits (CI0 and CI1). When the output carry equals 1 then CO1 CO0 = 01 while when it equals 2 then CO1 CO0 = 10.

# The Figure below shows how a 4-bit Tripler circuit is implemented using 4 copies of the basic 1-bit cell.

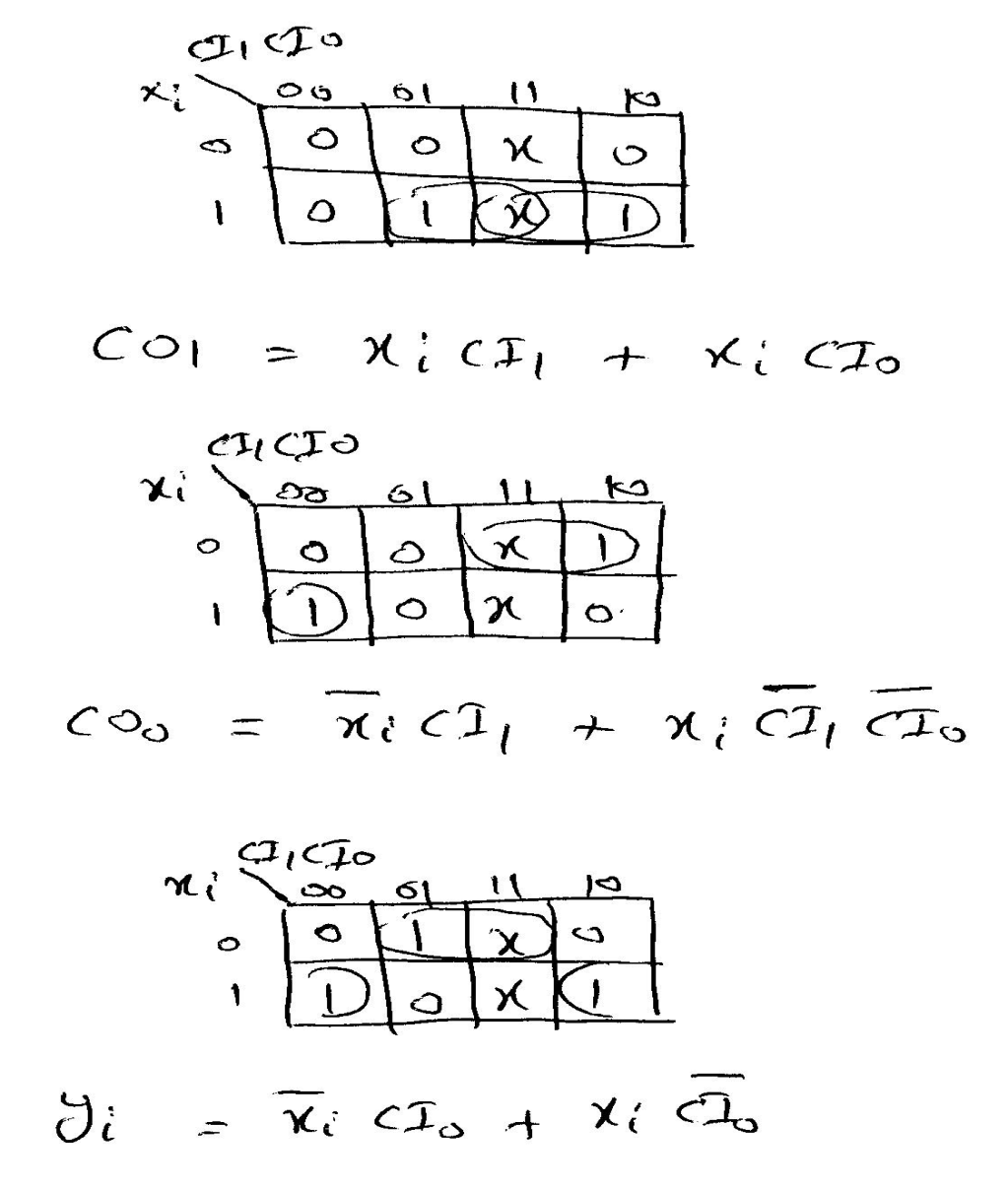


# Derive the truth table for the basic one-bit cell.

# (Hint: *As the initial input carries = 00, the maximum carry from one cell to the next is 2*)

# 

# Derive a minimized sum-of-product expressions for the outputs of the basic one-bit cell.

****

**Question 3. (17 Points)**

1. Fill in all blank cells in the two tables below. All binary representations use 7 bits

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Binary | Equivalent decimal value with the binary interpreted as: | | | |
| Unsigned number | Signed-magnitude number | Signed-1’s complement number | Signed-2’s complement number |
| 1011010 |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Decimal | Binary representation in: | | |
| Signed-magnitude notation | Signed-1’s complement notation | Signed-2’s complement notation |
| - 59 |  |  |  |

b. Using 2’s-complement signed arithmetic in **5 bits**, perform the following operations in binary. Show all your work. Verify that you get the expected decimal results.

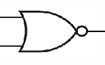
**Check for overflow and mark clearly any occurrences of it.**

|  |  |
| --- | --- |
| (i)  11010  + 11001 | (ii)  00101  - 10100 |
| (iii)  (+5)  + (-9) | (iv)  (-6)  - (+8) |

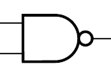
c. When doing signed 2’s complement arithmetic in **6 bit**s, the *smallest* binary number **that will cause overflow** when *subtracted* from 101000)2 is \_\_\_\_\_\_\_\_\_\_\_.

**Question 4. (8 Points)**

a. Show the logic diagrams that implement the given logic circuit using the minimum number of:

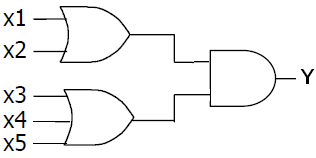


1. **2-input** NOR gates only. Use the following symbol for this NOR gate:



1. **2-input** NAND gates only. Use the following symbol for this NAND gate:

**Note**: Complements of the input variables are readily available.



1. In the logic circuit shown below, with **A** being any 4-bit input value,

* Output 1 = \_\_\_\_\_\_\_\_\_\_\_ (0 / 1 / Depends on A) , and
* Output 2 = \_\_\_\_\_\_\_\_\_\_\_ (0 / 1 / Depends on A).

**Note**: *The odd function gives a 1 output when the number of 1s in the input is odd*.

**Question 5. (12 Points)**



You are required to design a circuit that adds **four** *unsigned* 4-bit numbers. The following is the high-level diagram of the circuit.

1. What is the size of the output sum (S) in bits? [2 pts]
2. Using ONLY 4-bit adders, show how the above circuit can be constructed? [10 pts]

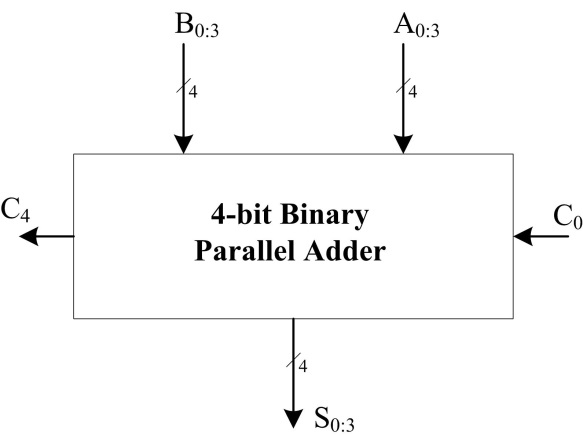
**Question 6. (8 Points)**

Assuming the availability of the true and complement of signals A, B and C, implement the following Boolean function.

1. Using a single minimum size multiplexer. [4 pts]
2. Using a single minimum size decoder and minimum other gates. [4 pts]

**Question 7. (18 Points)**

It is required to build a 4-bit binary parallel adder which adds two 4-bit numbers **A** (A*3* A*2* A*1* A*0*), and **B** (B*3* B*2* B*1* B*0*) together with an input carry bit **C*0*** (which may be 0 or 1).



The *full adder* circuit shown below will be used as a main building block of this adder.



1. Write the Boolean expressions of the P*i*, G*i*, S*i* and C*i+1* *full adder* signals. (2 Points)
2. If the 4-bit adder is implemented as a **Ripple Carry Adder** (RCA), draw the ***block diagram*** implementation of this adder. (1 Point)
3. Assuming gate delays of **3ns** forXOR gates and **1ns** for other gates; (3 Points)
4. What are the delays of the P0 and G0 signals?
5. What are the delays of the P3 and G3 signals?
6. What is the carry-propagation delay of a *single* full adder (i.e., delay from **C*i*** *to***C*i+1*)**?
7. What is the *worst case delay* of the 4-bit **RCA**? (4 Points)
8. If the 4-bit adder is implemented as a **Carry Lookahead Adder** (CLA), derive the Boolean expressions of the *four* Carry signals **C*1***, **C*2***, **C*3***, and **C*4***. (2 Points)
9. Draw a block diagram (*no detailed 2-level gate implementations*) of the **CLA** adder implementation (2 Points)
10. What is the *worst case delay* of the CLA? (4 Points)