Objective: Understand binary addition and subtraction
Language: C or C++
Points: 20
Release date: 24th August 2020
Deadline: 7 September 2020

Following functions need to be implemented: Signature of the function needs to be kept the same, else you won't get the marks.

(a) Void BitAddition(char A, char B, char CIN, char &Sum, char &COUT)
// This function takes three bits represented as char as input and gives Sum and COUT as output
(b) Void NBitAddition(char* A, char *B, char CIN, char* Sum, char &COUT)
//This function assumes A, B and Sum as N bit char array (not strings). Sum of A and B is calculated using binary addition methods. BitAddition can be used for doing binary addition
(c) Void Inverse(char* A, char* B)
//A is input array and B is output array. Every character of B is the logical inverse of A
(d) Void TwoSCompliment(char* A, char *B)
//A is an input array, and B is output 2's complement. 2's complement can be calculated by adding 1 to the inverse. Thus, using Inverse() function and NBitAddition()
(e) Bool isOverflow(char* A, char* B, char* Sum, char COUT, int operation)
//returns true if overflow condition is true, else false. A and B are input arrays and Sum and COUT are output that is received from either NBitAddition or NBitSubtrcation. Operation is 0 if ADD and 1 if Subtract.
(f) Void NBitSubtraction(char* A, char *B, char CIN, char* Sum, char &COUT)
//Do the subtraction. It can be done by finding 2'c complement of B and then using NBitAddition
//Arguments: A and B bit arrays of size N. B: number to be subtracted. Sum: Output of subtraction actually reflect A - B. COUT: Carry generated by NBitAddition
CIN: Can be viewed as borrow. If CIN is 1, Sum = A - B -1. If CIN is 0, Sum = A - B

All the functions assume the availability of ~~#DEFINE~~ global variable N

----------TA announcement----------

1: Link for the contest: https://www.hackerrank.com/cs203-lab1
2: You have to use institute email id for hacker rank signup/login
3: Sample test cases can be seen in the contest. A sample test case can be as follows
Sample Input 1:
5
1 1 1
01001 10100 1

10010
00111
10011 10111 0
00111 10011 1
Sample Output 1:
1 1
11110 0
01101
11001
0
10011 1

Sample Input 2:
8
1 1 0
10011110 11010011 0
11010011
10011110
10011110 00101101 1
10011110 00101101 0

Sample Output 2:
0 1
01110001 1
00101100
01100010
1
01110001 1

Sample Input 3:
8
1 1 0
00110101 01011011 0
11111111
01010101
00110101 01011011 0
00110101 01011011 0

Sample Output 3:
0 1
10010000 0
00000000
10101011

1
11011010 0

4: Kindly strictly follow the syntax defined above. Even if you get the correct output by not following the syntax, you will be not be awarded any marks. *For example: if some argument is char then use char, do not replace it by int.*
5: Kindly do not copy codes from your friends or the internet, it may lead you to failing of the course.
6: Multiple submission will create multiple copies of your assignment. It may create confusion. Please use run option before submitting and try to keep submission count as low as possible, i.e. one or two

----------Frequently asked questions----------
Question: where on HackerRank(what link can we follow) to complete this assignment?
Answer: https://www.hackerrank.com/cs203-lab1

Question: Need a sample test case.
Answer: Refer the same document

Question: For isOverflow, are A and B in two's complement?
Answer: There will be a case, for example -A-B. In this case, we can treat A, B are in 2's complement

Question: In NBitAddition and NBitSubtraction, can it be assumed that overflow will not occur and the answer will also be N bit? In case the answer is greater than N bit, should the extra bit(s) be considered?
Answer: As of now N bit addition and N bit subtraction having Cout as output parameter so if an overflow occurs and no carry, your output will be N bit. If Carry occurs in your N bit addition or subtraction, you have to print extra bit as COUT.

Questions what does CIN mean in functions like NbitAddition?
Answer: CIN in NbitAddition makes your n bit addition generic. You can use two such function to find out 2NbitAddition or any larger size addition. Here it was specified for sake of completeness

Suppose you are using two NbitAddition units in your design, it may be possible that one unit generates a carry which can go as an input to the second NbitAdder. Here CIN represents such a case, your design should work in both cases first when your unit takes a CIN in such case you consider it as '1' or second case where there is no carry from the preceding unit in such case you take CIN as '0'.

Question: How CIN and COUT parameters will help in NbitAddition and NbitSubtraction?

Answer: As Neeraj sir mentioned in the above question, I will give an example and intuition behind taking the help of carry in both input and output streams.
Where carry bit is used as a mechanism to chain multiple 1-bit additions together. We can take input of a carry bit into a 1-bit addition that has been generated from the output of a previous 1-bit addition. You will understand the clear intuition of these chaining of 1-bit additions in future modules.
Ex:

CIN:	0	1	1	0	0
A:	1	0	0	1	1
B:	0	0	1	1	0
COUT:	0	0	1	1	0
Sum:	1	1	0	0	1

Where black color represents our input (A, B, CIN) and blue represents output (Sum, COUT). Check with orange color, both CIN and COUT were the same, we were using COUT of Nth bit addition to (N-1)th bit addition as a CIN and so on. As of now we can call this as chaining of 1-bit additions.

For NbitSubtraction: CIN and COUT behaviour will be the same because A-B will be calculated as A+(2's complement(B)) during this addition of A, 2's complement(B) we will use CIN and retrieve COUT as above mentioned.

Question: One statement mentions that we have to consider CIN in Nbitsubtraction as the borrow bit, while the other statements mentions that CIN, COUT in Nbitsubtraction will behave the same as in NBitAddition (ie, CIN would behave like a carry bit, because essentially we are adding A with 2's complement of B).

Sir these statements are contradictory and consequently, creating confusion. It would be extremely helpful, if you can clarify about the situation, when CIN=1 in Nbitsubtraction. Giving an test case related to the situation would help even better.

Answer:Both statements were correct.
1st is saying about concept/intuition behind A-B, where 2nd Image saying and giving a hint for easy implementation. Check the below explanation.

1st :

it quotes CIN in Nbitaddition is like A+B+CIN for cascade addition(parallel 1-bit addition). You can check an example in the document for cascade addition example. It will be applicable for Nbitsubtraction as same as addition like A-B-CIN for maintaining parallel in subtraction as well. Where these things can be clarified in future classes about full adder, full subtractor, and full subtractor will be implemented using full adder by using xor gates.

A-B= A+2'scomplement(B)-CIN

2nd :
when CIN=1
A-B= A-B-1=A+2scomplement(B)-1=A+1's complement(B).
when CIN=0
A-B=A+2'scomlement(B)+0= A+2'scomplement(B).

where from 2nd image case we can conclude that if CIN=1 we have to do Nbitaddition with (A, inverse(B)).
if CIN =0 we have to do Nbitaddition(A,twos(B)).

I hope this helps in your lab point of view.

Question: How to submit a program on hackerrank?
Answer: A sample test can be attempted on the following URL:
https://www.hackerrank.com/ones-and-twos-complement

.

----------Applicable from lab 2 onward----------
Logistics: All these functions should be written in file lab1_<entry_number>.c or lab1_<entry_number>.cpp
Lab1_enry_number> file should not have “main” function. “Main” function should be there in another file.
Labs will be evaluated automatically, so it is important to adhere to the function signature.