Booth Algorithm

Explanation of input format for code :

1) Input the number of register bits (Note : **Number of register bits should be SLIGHTLY" greater than the number of bits required to represent either of the number that are to be multiplied** ; in short - No of register bits is large enough to hold both number
2) Input whether the first number(M) you put in , is positive or not , by pressing "Y"(for yes) and "N"(for no) .
3) **Input your first number with a sign**. (Say if you wish to enter a positive number , say "7" , so **the input format is "7" , instead of "+7"**)
4) Input whether the second number(Q) you put in , is positive or not , by pressing "Y"(for yes) and "N"(for no) .
5) **Input your second number with a sign**. (Say if you wish to enter a negative number , say "-7" , so **the input format is "-7" , instead of "7"**)
6) Sit back and enjoy the code processings and your result

Note/Explanation/Clarification : If you wish to enter a positive number , just do it without the "+" sign , but for a negative number , you need to put "-" too.
For example, you wish to input "negative 5" , so it would be inputted as "-5" and not as "5" .

How to run:

Just run the code(python) file and enter input as described above

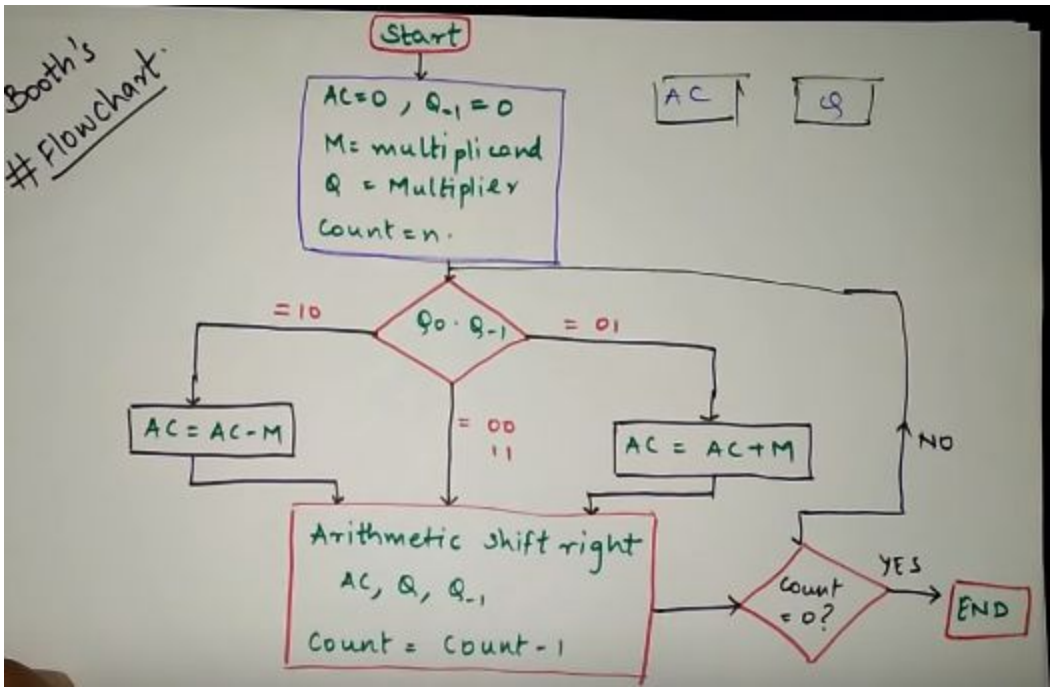
Assumptions and Constraints

1. Numbers are in decimal (base 10) and are integers
2) Numbers are inputted with their appropriate signs .
3) No of register bits is large enough to hold both number
4) Please read the "Explanation of input format for code "(above) carefully

Output :

After the code is fully executed ,**the result is displayed in both Binary and Decimal.**

Explanation of code :



This is the algorithm which is followed in the code.
Our first number is M (Multiplicand)
Our second number is Q (Multiplier)
We initialize “Q-1”(pronounced : Q minus 1 ; not to be confused with subtracting ”1” from Q) with zero(“0”)

We initialize “AC” with “000…….upto the number of register bits” . That means if the number of registers bit is equal to 4 , so AC = 0000.

Now we use a for loop to do the below mentioned process repeatedly for n times , where n = number of register bits.

We check the last bit of “Q” and “(Q-1)” .

If they are equal to “00” or “11” , then we do Arithmetic Right Shift and move towards the next iteration.

Else If they are equal to “01” , then we do a binary sum of “AC” and “+M” and store the result in “AC”(ie AC = AC + M) and move towards the next iteration.

Else if they are equal to “10” then we do binary sum of “AC” and “-M” and store the result in “AC”(ie AC = AC + (-M)) and move towards the next iteration.

After the for loop is fully executed ,**the result is displayed in both Binary and Decimal.**

.Credits : https://www.youtube.com/watch?v=DHhcnjEKEFo

Code made by -
Harshal Dev
2019306
CSD