Batch: A2 **Roll No.:** 16010122041

Experiment / assignment / tutorial No: 2

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

TITLE: To study and implement Booth's Multiplication Algorithm.

AIM: Booth's Algorithm for Multiplication

Expected OUTCOME of Experiment: (Mention CO/CO's attained here)

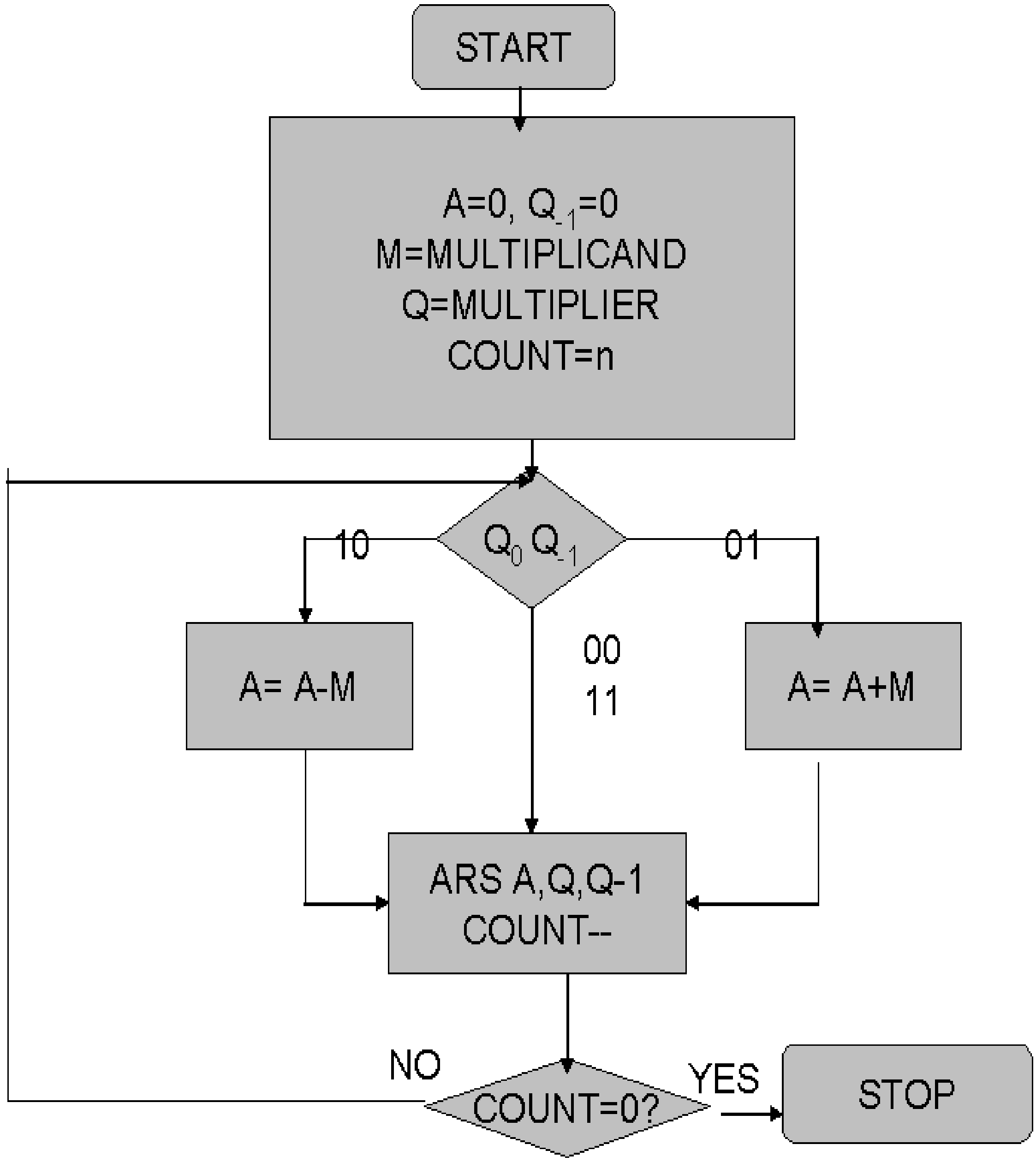
Books/ Journals/ Websites referred:

1. Carl Hamacher, Zvonko Vranesic and Safwat Zaky, "Computer Organization", Fifth Edition, TataMcGraw-Hill.
2. William Stallings, "Computer Organization and Architecture: Designing for Performance", Eighth Edition, Pearson.
3. Dr. M. Usha, T. S. Srikanth, "Computer System Architecture and Organization", First Edition, Wiley-India.

Pre Lab/ Prior Concepts:

It is a powerful algorithm for signed number multiplication which generates a 2n bit product and treats both positive and negative numbers uniformly. Also the efficiency of the algorithm is good due to the fact that, block of 1's and 0's are skipped over and subtraction/addition is only done if pair contains 10 or 01

Flowchart:


START
A=0, Q-1=0
M=MULTIPLICAND
Q=MULTIPLIER
COUNT=n
Q0 Q-1
10
01
00
11
A= A-M
A= A+M
ARS A,Q,Q-1
COUNT--
NO
COUNT=0?
YES
STOP

Design Steps:

1. Start
2. Get the multiplicand (M) and Multiplier (Q) from the user
3. Initialize A= Q_{-1} =0
4. Convert M and Q into binar
5. Compare Q_0 and Q_{-1} and perform the respective operation.

Q_0 Q_{-1}	**Operation**
00/11	Arithmetic right shift
01	A+M and Arithmetic right shift
10	A-M and Arithmetic right shift

6. Repeat steps 5 till all bits are compared

7. Convert the result to decimal form and display

8. End

Example: (Handwritten solved problem needs to be uploaded)

M = 7 : 0111   -M : 1001
Q = -3 : 1101

A	Q	Q_{-1}	Opperation
0000	1101	0	Initial Value.
1001	1101	0	A ← A − M
1100	1110	1	Arithmetic right shift
0011	1110	1	A ← A + M
0001	1111	0	Arithmetic right shift
1010	1111	0	A ← A − M
1101	0111	1	Arithmetic right shift
1110	1011	1	Arithmetic right shift.

Final answer: 1110 1011
↓ 2's comp.
0001 0101 = −21

M = 5 : 0101   -M : 1011
Q = 5 : 0101

A	Q	Q_{-1}	Opperation.
0000	0101	0	Initial Value.
1011	0101	0	A ← A − M
1101	1010	1	Arithemetic shift right
0010	1010	1	A ← A + M
0001	0101	0	Arithmetic right shift
1100	0101	0	A ← A − M
1110	0010	1	Arithmetic right shift
0011	0010	1	A ← A + M
0001	1001	0	Arithmetic right shift

Final ans: 0001 1001 = <u>25</u>

Code:

```
#include <bits/stdc++.h>
using namespace std;
int findbit(int m,int q){
    m=max(abs(m),abs(q));
    for(q=0;pow(2,q)<m;q++);
    return (max((q+1),4));
}
int* binary(int a,int num){
    int* ptr=(int*)malloc(num*sizeof(int));
    int acopy=abs(a),check=1;
    for (int i = 0;i<num; i++){
        ptr[i] = acopy % 2;
        acopy = acopy/2;
    }
    if (a < 0){
        for (int i = 0; i <num; i++){
            if (ptr[i] == 1 && check==1)
                check=0;
            else if(ptr[i] == 1 && check==0)
                ptr[i]=0;
            else if(ptr[i] == 0 && check==0)
                ptr[i]=1;
        }
    }
    return ptr;
}
void printbinary(int* ans,string s,int num){
    for(int i=2*num;i>num;i--)
        cout<<ans[i]<<" ";
    cout<<"\t";
    for(int i=num;i>0;i--)
        cout<<ans[i]<<" ";
    cout<<"\t"<<ans[0]<<"\t"<<s<<endl;
}
void binaryadd(int* ans,int* n,int num){
    int carry=0;
```

```
    for(int i=num+1;i<=2*num;i++){
        if(ans[i]+n[i-num-1]+carry==1){
            ans[i]=1;
            carry=0;
        }
        else if(ans[i]+n[i-num-1]+carry==2){
            ans[i]=0;
            carry=1;
        }
        else if(ans[i]+n[i-num-1]+carry==3){
            ans[i]=1;
            carry=1;
        }
    }
}
int main()
{
    int m,q;
    cout<<"Enter M and Q: ";
    cin>>m>>q;
    int num=findbit(m,q);
    int ans[2*num+1]={0};
    int *arr=binary(q,num);
    for(int i=num;i>0;i--)
        ans[i]=arr[i-1];
    cout<<endl<<"A\t\tQ\t\tQ-1\t\tOperation"<<endl<<endl;
    printbinary(ans,"Initial Value",num);
    for(int i=0;i<num;i++){
        if(ans[1]==0 && ans[0]==1){
            binaryadd(ans,binary(m,num),num);
            printbinary(ans,"A <- A + M",num);
        }
        else if(ans[1]==1 && ans[0]==0){
            binaryadd(ans,binary(-m,num),num);
            printbinary(ans,"A <- A - M",num);
        }
        for(int i=0;i<2*num;i++)
```

```
            ans[i]=ans[i+1]; // Right Shifting
        printbinary(ans,"Arithemetic Right Shift",num);
    }
    printbinary(ans,"Final Answer",num);
}
```

Output:

```
Enter M and Q: 5 5

A               Q               Q-1             Operation

0 0 0 0         0 1 0 1         0       Initial Value
1 0 1 1         0 1 0 1         0       A <- A - M
1 1 0 1         1 0 1 0         1       Arithemetic Right Shift
0 0 1 0         1 0 1 0         1       A <- A + M
0 0 0 1         0 1 0 1         0       Arithemetic Right Shift
1 1 0 0         0 1 0 1         0       A <- A - M
1 1 1 0         0 0 1 0         1       Arithemetic Right Shift
0 0 1 1         0 0 1 0         1       A <- A + M
0 0 0 1         1 0 0 1         0       Arithemetic Right Shift
0 0 0 1         1 0 0 1         0       Final Answer
```

```
Enter M and Q: 11 -10

A               Q               Q-1             Operation

0 0 0 0 0       1 0 1 1 0       0       Initial Value
0 0 0 0 0       0 1 0 1 1       0       Arithemetic Right Shift
1 0 1 0 1       0 1 0 1 1       0       A <- A - M
1 1 0 1 0       1 0 1 0 1       1       Arithemetic Right Shift
1 1 1 0 1       0 1 0 1 0       1       Arithemetic Right Shift
0 1 0 0 0       0 1 0 1 0       1       A <- A + M
0 0 1 0 0       0 0 1 0 1       0       Arithemetic Right Shift
1 1 0 0 1       0 0 1 0 1       0       A <- A - M
1 1 1 0 0       1 0 0 1 0       1       Arithemetic Right Shift
1 1 1 0 0       1 0 0 1 0       1       Final Answer
```

Conclusion:

Learnt and implemented booths algorithm along with the understanding of computer bits and operations like arithmetic shift right.

Post Lab Descriptive Questions

1. Explain advantages and disadvantages of Booth's algorithm.

Advantages of booth's multiplication:

- Easy calculation of multiplication problem.
- Consecutive additions will be replaced.
- Less complex and ease scaling.

Disadvantages of booth's multiplication:

- This algorithm will not work for isolated 1's.
- It is time consuming.
- If digital gates are more, chip area would be large.

2. Is Booth's recoding better than Booth's algorithm? Justify

Advantage of Booth's recoding is that it reduces the number of 1's and increases the number of 0's in a binary number. Having more number of 0's is advantageous for easier calculation.

For Example: (01111)2 is equivalent to (+1 0 0 0 -1) in Booth Recoding. Hence it is more efficient and less time consuming in comparison to Booth's algorithm.

Date: ______________ **Signature of faculty in-charge**