



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Experiment No. 7
Aim : Implement Booth's algorithm using c-programming
Name: Mitanksh Gosalia
Roll Number:13
Date of Performance:
Date of Submission:



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Aim: To implement Booth's algorithm using c-programming.

Objective -

1. To understand the working of Booths algorithm.
2. To understand how to implement Booth's algorithm using c-programming.

Theory:

Booth's algorithm is a multiplication algorithm that multiplies two signed binary numbers in 2's complement notation. Booth used desk calculators that were faster at shifting than adding and created the algorithm to increase their speed.

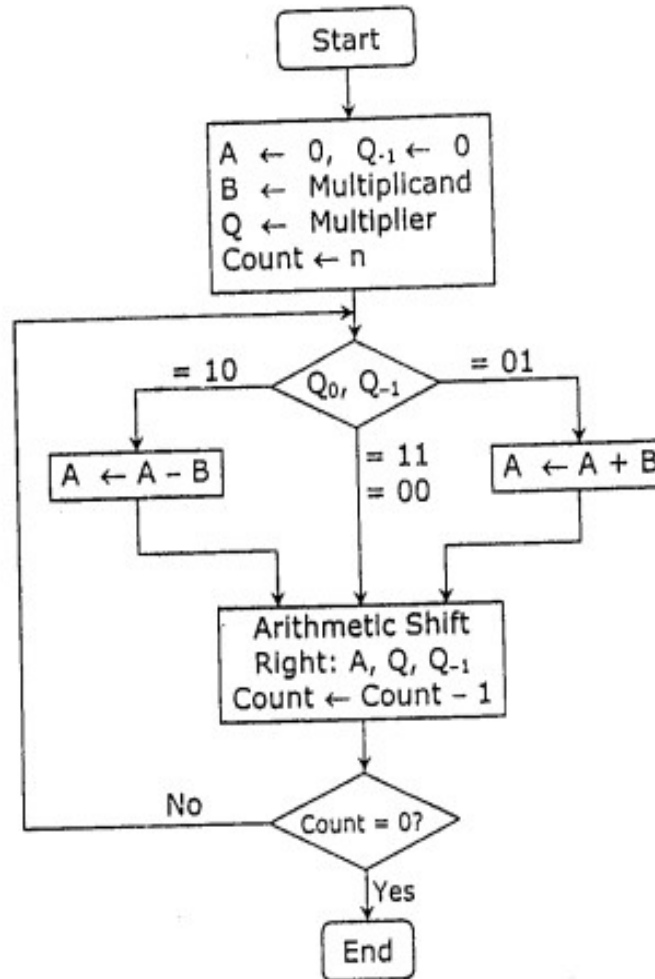
The algorithm works as per the following conditions :

1. If Q_n and Q_{-1} are same i.e. 00 or 11 perform arithmetic shift by 1 bit.
2. If $Q_n Q_{-1} = 10$ do $A = A - B$ and perform arithmetic shift by 1 bit.
3. If $Q_n Q_{-1} = 01$ do $A = A + B$ and perform arithmetic shift by 1 bit.



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science



Multiplicand (B) ← 0 1 0 1 (5), Multiplier (Q) ← 0 1 0 0 (4)				
Steps	A	Q	Q-1	Operation
	0 0 0 0	0 1 0 0	0	Initial
Step 1 :	0 0 0 0	0 0 1 0	0	Shift right
Step 2 :	0 0 0 0	0 0 0 1	0	Shift right
Step 3 :	1 0 1 1	0 0 0 1	0	A ← A - B
	1 1 0 1	1 0 0 0	1	Shift right
Step 4 :	0 0 1 0	1 0 0 0	1	A ← A + B
	0 0 0 1	0 1 0 0	0	Shift right
Result	0 0 0 1 0 1 0 0 = +20			

Program:



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
1. #include <stdio.h>
2. #include <math.h>
3.
4. int a = 0, b = 0, c = 0, a1 = 0, b1 = 0, com[5] = { 1, 0, 0, 0, 0};
5. int anum[5] = {0}, anumcp[5] = {0}, bnum[5] = {0};
6. int acomp[5] = {0}, bcomp[5] = {0}, pro[5] = {0}, res[5] = {0};
7.
8. void binary(){
9.     a1 = fabs(a);
10.    b1 = fabs(b);
11.    int r, r2, i, temp;
12.    for (i = 0; i < 5; i++){
13.        r = a1 % 2;
14.        a1 = a1 / 2;
15.        r2 = b1 % 2;
16.        b1 = b1 / 2;
17.        anum[i] = r;
18.        anumcp[i] = r;
19.        bnum[i] = r2;
20.        if(r2 == 0){
21.            bcomp[i] = 1;
22.        }
23.        if(r == 0){
24.            acomp[i] = 1;
25.        }
26.    }
27.
28.    c = 0;
29.    for (i = 0; i < 5; i++){
30.        res[i] = com[i] + bcomp[i] + c;
31.        if(res[i] >= 2){
32.            c = 1;
33.        }
34.        else
35.            c = 0;
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
36.     res[i] = res[i] % 2;
37. }
38. for (i = 4; i >= 0; i--){
39.     bcomp[i] = res[i];
40. }
41.
42. if (a < 0){
43.     c = 0;
44.     for (i = 4; i >= 0; i--){
45.         res[i] = 0;
46.     }
47.     for (i = 0; i < 5; i++){
48.         res[i] = com[i] + acomp[i] + c;
49.         if (res[i] >= 2){
50.             c = 1;
51.         }
52.         else
53.             c = 0;
54.         res[i] = res[i]%2;
55.     }
56.     for (i = 4; i >= 0; i--){
57.         anum[i] = res[i];
58.         anumcp[i] = res[i];
59.     }
60.
61. }
62. if(b < 0){
63.     for (i = 0; i < 5; i++){
64.         temp = bnum[i];
65.         bnum[i] = bcomp[i];
66.         bcomp[i] = temp;
67.     }
68. }
69.}
70. void add(int num[]){
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
71.  int i;
72.  c = 0;
73.  for ( i = 0; i < 5; i++){
74.      res[i] = pro[i] + num[i] + c;
75.      if (res[i] >= 2){
76.          c = 1;
77.      }
78.      else{
79.          c = 0;
80.      }
81.      res[i] = res[i]%2;
82.  }
83.  for (i = 4; i >= 0; i--){
84.      pro[i] = res[i];
85.      printf("%d",pro[i]);
86.  }
87.  printf(".");
88.  for (i = 4; i >= 0; i--){
89.      printf("%d", anumcp[i]);
90.  }
91. }
92. void arshift(){
93.     int temp = pro[4], temp2 = pro[0], i;
94.     for (i = 1; i < 5 ; i++){
95.         pro[i-1] = pro[i];
96.     }
97.     pro[4] = temp;
98.     for (i = 1; i < 5 ; i++){
99.         anumcp[i-1] = anumcp[i];
100.    }
101.    anumcp[4] = temp2;
102.    printf("\nAR-SHIFT: ");
103.    for (i = 4; i >= 0; i--){
104.        printf("%d",pro[i]);
105.    }
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
106.     printf(".");
107.     for(i = 4; i >= 0; i--){
108.         printf("%d", anumcp[i]);
109.     }
110. }
111.
112. void main(){
113.     int i, q = 0;
114.     printf("\t\tBOOTH'S MULTIPLICATION ALGORITHM");
115.     printf("\nEnter two numbers to multiply: ");
116.     printf("\nBoth must be less than 16");
117.     //simulating for two numbers each below 16
118.     do{
119.         printf("\nEnter A: ");
120.         scanf("%d",&a);
121.         printf("Enter B: ");
122.         scanf("%d", &b);
123.     }while(a >=16 || b >=16);
124.
125.     printf("\nExpected product = %d", a * b);
126.     binary();
127.     printf("\n\nBinary Equivalents are: ");
128.     printf("\nA = ");
129.     for (i = 4; i >= 0; i--){
130.         printf("%d", anum[i]);
131.     }
132.     printf("\nB = ");
133.     for (i = 4; i >= 0; i--){
134.         printf("%d", bnum[i]);
135.     }
136.     printf("\nB'+ 1 = ");
137.     for (i = 4; i >= 0; i--){
138.         printf("%d", bcomp[i]);
139.     }
140.     printf("\n\n");
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
141.     for (i = 0; i < 5; i++){
142.         if (anum[i] == q){
143.             printf("\n-->");
144.             arshift();
145.             q = anum[i];
146.         }
147.         else if (anum[i] == 1 && q == 0){
148.             printf("\n-->");
149.             printf("\nSUB B: ");
150.             add(bcomp);
151.             arshift();
152.             q = anum[i];
153.         }
154.         else{
155.             printf("\n-->");
156.             printf("\nADD B: ");
157.             add(bnum);
158.             arshift();
159.             q = anum[i];
160.         }
161.     }
162.
163.     printf("\nProduct is = ");
164.     for (i = 4; i >= 0; i--){
165.         printf("%d", pro[i]);
166.     }
167.     for (i = 4; i >= 0; i--){
168.         printf("%d", anumcp[i]);
169.     }
170. }
```

Output:



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

BOOTH'S MULTIPLICATION ALGORITHM

Enter two numbers to multiply:

Both must be less than 16

Enter A: 1

Enter B: 11

Expected product = 11

Binary Equivalents are:

A = 0001

B = 0101

B'+ 1 = 1010

-->

SUB B: 1010:0001

AR-SHIFT: 1101:1000

-->

ADD B: 0010:1000

AR-SHIFT: 0010:1100

-->

AR-SHIFT: 0001:0110

-->

AR-SHIFT: 0000:1011

-->

AR-SHIFT: 0000:0101

Product is = 00000101

Conclusion -

In Conclusion, the **Booth algorithm** is a helpful technique for binary multiplication of signed integers in the **2's complement** representation. Compared to conventional multiplication techniques, the process just requires shifting and either adding or removing the multiplicand based on the **multiplier bit** value. We have supplied a bitwise operation-based implementation of **Booth's** method in C along with a practical application.