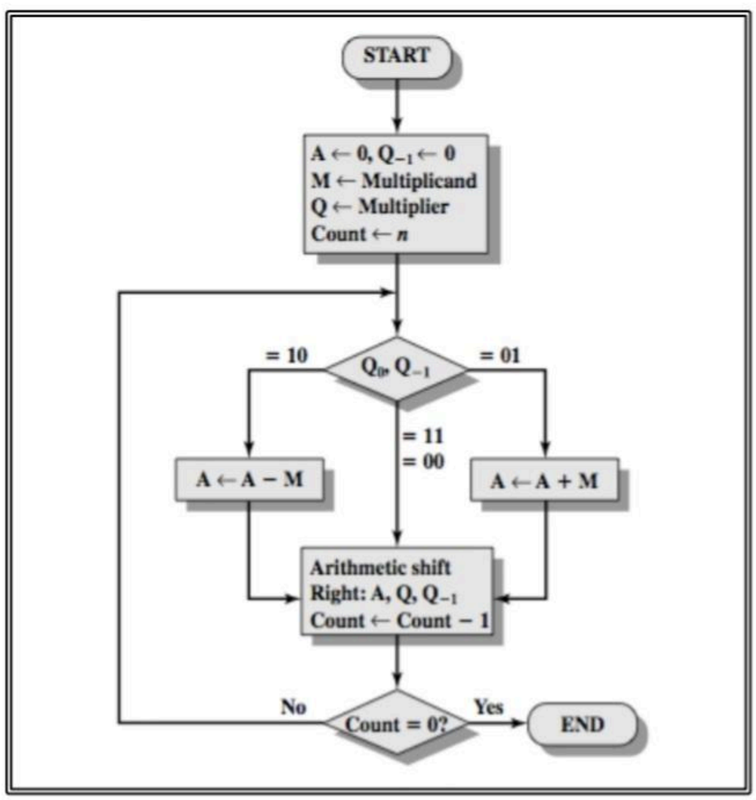
Bhartiya Vidya Bhavan's

Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

NAME:	Sujal Sandeep Dingankar
UID:	2024301005
SUBJECT:	CAO
EXPERIMENT NO :	Experiment 7
DATE OF PERFORMANCE	20/10/2024
AIM:	**Program for Booth's Algorithm**
Flowchart :-	Booth's Flowchart..
	FLOWCHART FOR BOOTH'S ALGORITHM 

Theory :-	The Booth algorithm is an efficient method for multiplying binary numbers, particularly designed to handle signed numbers in two's complement representation. Here's a brief overview: **Key Concepts:** 1. **Binary Representation**: Numbers are represented in binary, with signed numbers using two's complement. 2. **Multiplicand and Multiplier**: The algorithm processes a multiplicand (the number being multiplied) and a multiplier (the number by which we multiply). **Steps:** 1. **Initialization**: Set up the necessary registers: one for the multiplicand, one for the multiplier, and an additional register to store the result. Initialize a bit to keep track of the previous state of the least significant bit (LSB) of the multiplier. 2. **Iteration**: Loop through the bits of the multiplier: ○ If the current LSB is `1` and the previous bit was `0`, add the multiplicand to the result. ○ If the current LSB is `0` and the previous bit was `1`, subtract the multiplicand from the result. ○ Shift the multiplier and the result right by one bit. Update the previous bit. 3. **Repeat**: Continue the process for a predetermined number of bits (usually the length of the multiplier).

PROGRAM:

```
#include<stdio.h>

#include<stdlib.h>

#include<string.h>



//utility functions

void add(int* result, int* multiplicand, int size){

int carry=0;

for(int i=size-1; i>=0; i--){

if(result[i]+multiplicand[i]+carry<2){

result[i]=result[i]+multiplicand[i]+carry;

carry=0;

}

else if(result[i]+multiplicand[i]+carry==2){
```

```
result[i]=0;
carry=1;
}
else{
result[i]=1;
carry=1;
}
}
}

void arithmeticRightShift(int* result, int size){
for(int i=size-1; i>0; i--){
result[i]=result[i-1];
}
}


//main function
void main(){
//taking the user inputs
char M[' '];
char Q[' '];
printf("\nEnter the multiplicand      > ");
```

```
scanf("%s",M);
printf("Enter the multiplier       > ");
scanf("%s",Q);


//finding length of the input to determine the array size
int Mlen=strlen(M), Qlen=strlen(Q);
int maxlen=Mlen>Qlen?Mlen:Qlen;


//creating all the required arrays
int* result=(int*)calloc(2*maxlen+1,sizeof(int));
int* addM=(int*)calloc(maxlen,sizeof(int));
int* subM=(int*)calloc(maxlen,sizeof(int));
int* tem=(int*)calloc(maxlen,sizeof(int));


//array initialization
int temp=0;
for(int i=2*maxlen-Qlen; i<2*maxlen; i++){
if(Q[temp]=='1'){
result[i]=1;
}
temp++;
}
```

	temp=0;
	for(int i=maxlen-Mlen; i<maxlen; i++){
	if(M[temp]=='1'){
	addM[i]=1;
	}
	temp++;
	}
	for(int i=0; i<maxlen;
	i++){ if(addM[i]==1){
	subM[i]=0;
	}
	else{
	subM[i]=1;
	}
	}
	tem[maxlen-1]=1;
	add(subM,tem,maxlen);
	
	//implementing the algorithm
	int count=maxlen;
	while(count--){
	if(result[2*maxlen]==1&&result[2*maxlen-1]==0){

```
add(result,addM,maxlen);

}

else if(result[2*maxlen]==0&&result[2*maxlen-1]==1){

add(result,subM,maxlen);

}

arithmeticRightShift(result,2*maxlen+1);

}



//printing the result

printf("Result       > ");

for(int i=0; i<2*maxlen; i++){

printf("%d",result[i]);

}

printf("\n\n");

}
```

RESULT:

```
Enter the multiplicand -----> 0111
Enter the multiplier -----> 0011
Result -----> 00010101

PS C:\SPIT\VSCODE\.vscode\CAO>
```

CONCLUSION:	By performing the experiment, I understood the concept and implementation of Booth's Algorithm.