Experiment No. 8
Aim :Implement Restoring algorithm using c-programming
Name:Mitanksh Gosalia
Roll Number:13
Date of Performance:
Date of Submission:

Aim: To implement Restoring division algorithm using c-programming.

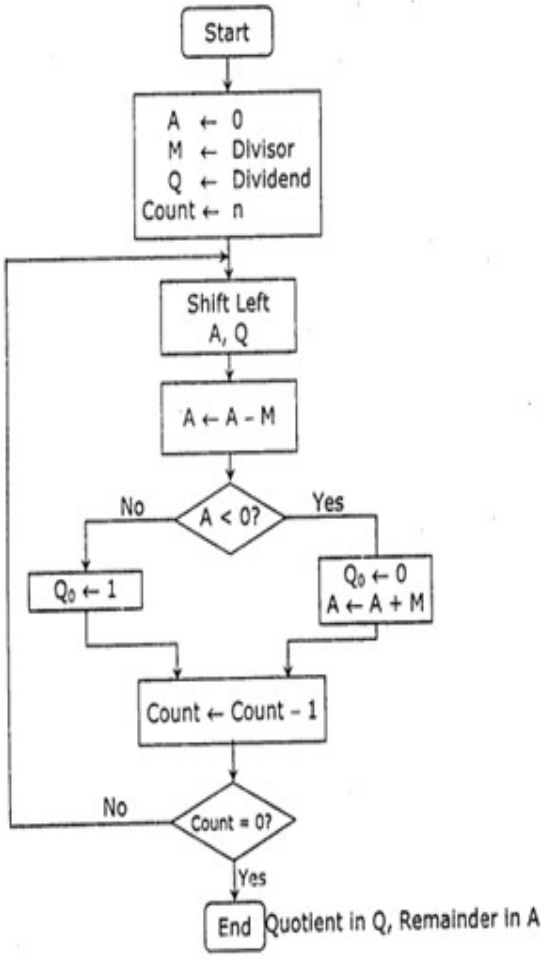
Objective -

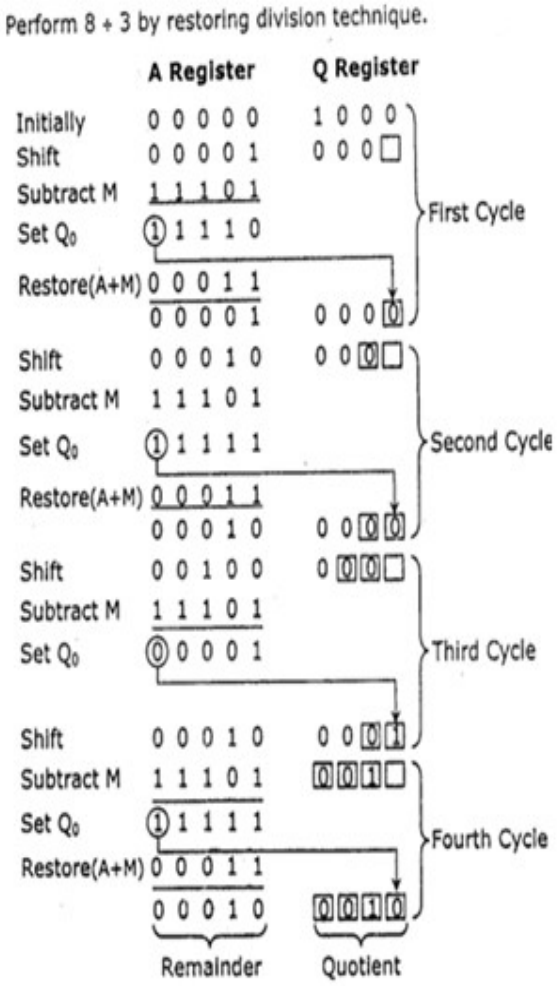
1. To understand the working of Restoring division algorithm.
2. To understand how to implement Restoring division algorithm using c-programming.

Theory:

1) The divisor is placed in M register, the dividend placed in Q register.
2) At every step, the A and Q registers together are shifted to the left by 1-bit
3) M is subtracted from A to determine whether A divides the partial remainder. If it does, then Q0 set to 1-bit. Otherwise, Q0 gets a 0 bit and M must be added back to A to restore the previous value.
4) The count is then decremented and the process continues for n steps. At the end, the quotient is in the Q register and the remainder is in the A register.

Flowchart





Program-

```
#include<stdlib.h>

#include<stdio.h>

int acum[100]={0}           ;

void add(int acum[],int b[],int n);

int q[100],b[100];

int main()

{

int x,y;
```

```
printf("Enter the Number :");

scanf("%d%d",&x,&y);

int i=0;

while(x>0||y>0)

{

if(x>0)

{

q[i]=x%2;

x=x/2;

}

else

{

q[i]=0;

}

if(y>0)

{

b[i]=y%2;

y=y/2;

}

else
```

```
{
b[i]=0;
}
i++;
}

int n=i;
int bc[50];
printf("\n");
for(i=0;i<n;i++)
{
if(b[i]==0)
{
bc[i]=1;
}
else
{
bc[i]=0;
}
}
```

```
bc[n]=1;

for(i=0;i<=n;i++)

{

if(bc[i]==0)

{

bc[i]=1;

i=n+2;

}

else

{

bc[i]=0;

}

}

int l;

 b[n]=0;

int k=n;

int n1=n+n-1;

int j,mi=n-1;

for(i=n;i!=0;i--)

{
```

```
for(j=n;j>0;j--)

{

acum[j]=acum[j-1];



}

acum[0]=q[n-1];

for(j=n-1;j>0;j--)

{

q[j]=q[j-1];

}



add(acum,bc,n+1);

if(acum[n]==1)

{

q[0]=0;

add(acum,b,n+1);

}

else

{

q[0]=1;
```

```
}

}

printf("\nQuoient   : ");


for(  l=n-1;l>=0;l--)

{

printf("%d",q[l]);



}

printf("\nRemainder : ");

for( l=n;l>=0;l--)

{

printf("%d",acum[l]);

}

return 0;

}

void add(int acum[],int bo[],int n)

{

int i=0,temp=0,sum=0;

for(i=0;i<n;i++)
```

```
{
sum=0;
sum=acum[i]+bo[i]+temp;
if(sum==0)
{
acum[i]=0;
temp=0;
}
else if (sum==2)
{
acum[i]=0;
temp=1;
}
else if(sum==1)
{
acum[i]=1;
temp=0;
}
else if(sum==3)
{
```

```
acum[i]=1;

temp=1;

}

}

}
```

Output -

```
Input:
15 7

Output:
Enter the Number :

Quoient: 0010
Remainder: 00001
```

Conclusion –

The experiment demonstrated the design and implementation of a restoring division algorithm, which is a type of logic circuit that can divide two n-bit binary numbers by using repeated shifting, addition, and subtraction operations.The experiment used integrated circuits (ICs) to implement the registers and the arithmetic logic unit (ALU) on a breadboard, and verified their functionality using a multimeter or an oscilloscope.The experiment recorded the input and output values of each register and the ALU for different combinations of binary inputs and constructed the corresponding truth tables.The experiment observed the advantages and disadvantages of the restoring division algorithm compared to other division algorithms, such as simplicity, accuracy, but slow speed and long carry propagation delay.The experiment learned how to use the restoring division algorithm to design various arithmetic circuits, such as counters, registers, shift registers, etc. that can perform operations such as division, modulo, etc. on binary numbers.