



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Experiment No. 9
Implement Non-Restoring algorithm using c-programming
Name: Nitish Jha
Roll Number: 18
Date of Performance:
Date of Submission:

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

Objective -

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

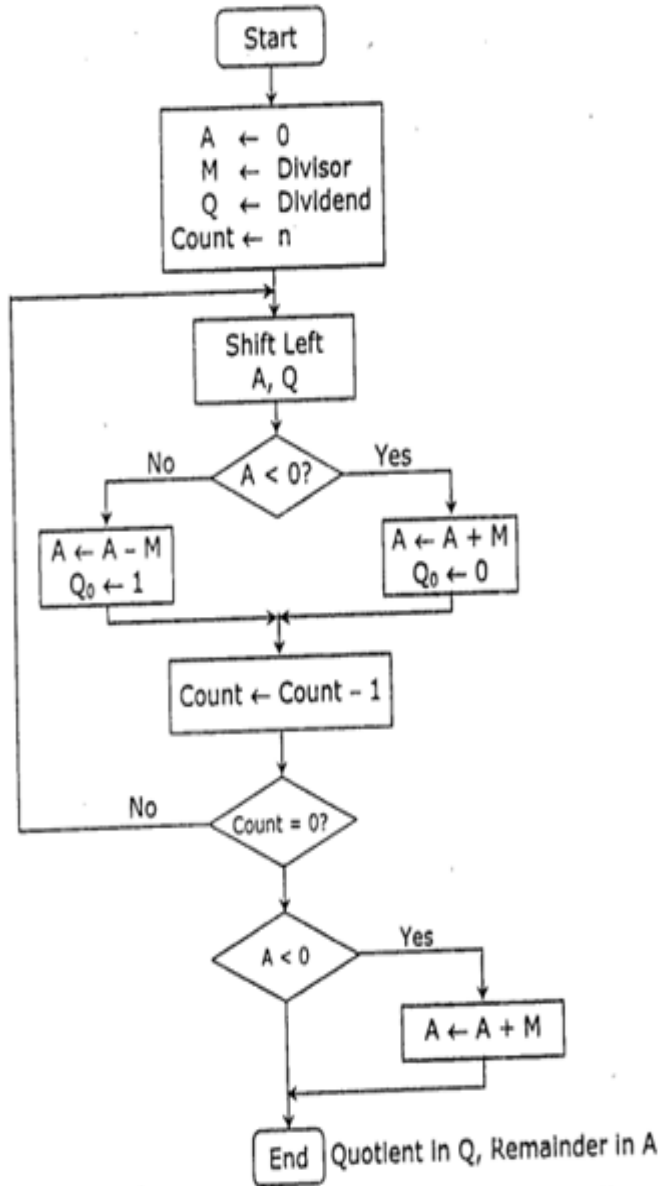
Theory:

In each cycle content of the register, A is first shifted and then the divisor is added or subtracted with the content of register A depending upon the sign of A. In this, there is no need of restoring, but if the remainder is negative then there is a need of restoring the remainder. This is the faster algorithm of division.



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science



Perform $8 + 3$ by non-restoring division technique.

	A Register	Q Register	
Initially	0 0 0 0	1 0 0 0	
Shift	0 0 0 0	1 0 0 0	
Subtract	1 1 1 0		
Set Q ₀	1 1 1 0	0 0 0 0	First Cycle
Shift	1 1 1 0	0 0 0 0	
Add	0 0 0 1		
Set Q ₀	1 1 1 1	0 0 0 0	Second Cycle
Shift	1 1 1 1	0 0 0 0	
Add	0 0 0 1		
Set Q ₀	0 0 0 1	0 0 0 1	Third Cycle
Shift	0 0 0 1	0 0 0 1	
Subtract	1 1 1 0		
Set Q ₀	1 1 1 1	0 0 0 1	Fourth Cycle
Add	1 1 1 1		
	0 0 0 1		
	0 0 0 1		
			Quotient
			Remainder



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Program -

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
int dec_bin(int, int []);
```

```
int twos(int [], int []);
```

```
int left(int [], int []);
```

```
int add(int [], int []);
```

```
int main()
```

```
{
```

```
    int a, b, m[4]={0,0,0,0}, q[4]={0,0,0,0}, acc[4]={0,0,0,0}, m2[4], i, n=4;
```

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

```
    scanf("%d", &a);
```

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

```
    scanf("%d", &b);
```

```
    dec_bin(a, q);
```

```
    dec_bin(b, m);
```

```
    twos(m, m2);
```

```
    printf("\nA\tQ\tComments\n");
```

```
    for(i=3; i>=0; i--)
```

```
    {
```

```
        printf("%d", acc[i]);
```

```
    }
```

```
    printf("\t");
```

```
    for(i=3; i>=0; i--)
```

```
    {
```

```
        printf("%d", q[i]);
```

```
    }
```

```
    printf("\tStart\n");
```

```
    while(n>0)
```

```
    {
```

```
        left(acc, q);
```

```
        for(i=3; i>=0; i--)
```

```
        {
```

```
            printf("%d", acc[i]);
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
}
printf("\t");
for(i=3; i>=1; i--)
{
    printf("%d", q[i]);
}
printf(" _\tLeft Shift A,Q\n");
add(acc, m2);
for(i=3; i>=0; i--)
{
    printf("%d", acc[i]);
}
printf("\t");
for(i=3; i>=1; i--)
{
    printf("%d", q[i]);
}
printf(" _\tA=A-M\n");
if(acc[3]==0)
{
    q[0]=1;
    for(i=3; i>=0; i--)
    {
        printf("%d", acc[i]);
    }
    printf("\t");
    for(i=3; i>=0; i--)
    {
        printf("%d", q[i]);
    }
    printf("\tQ0=1\n");
}
else
{
    q[0]=0;
    add(acc, m);
    for(i=3; i>=0; i--)
    {
        printf("%d", acc[i]);
    }
}
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
    }
    printf("\t");
    for(i=3; i>=0; i--)
    {
        printf("%d", q[i]);
    }
    printf("\tQo=0; A=A+M\n");
}
n--;
}
printf("\nQuotient = ");
for(i=3; i>=0; i--)
{
    printf("%d", q[i]);
}
printf("\tRemainder = ");
for(i=3; i>=0; i--)
{
    printf("%d", acc[i]);
}
printf("\n");
return 0;
}
```

```
int dec_bin(int d, int m[])
{
    int b=0, i=0;
    for(i=0; i<4; i++)
    {
        m[i]=d%2;
        d=d/2;
    }
    return 0;
}
```

```
int twos(int m[], int m2[])
{
    int i, m1[4];
    for(i=0; i<4; i++)
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
{
    if(m[i]==0)
    {
        m1[i]=1;
    }
    else
    {
        m1[i]=0;
    }
}
for(i=0; i<4; i++)
{
    m2[i]=m1[i];
}
if(m2[0]==0)
{
    m2[0]=1;
}
else
{
    m2[0]=0;
    if(m2[1]==0)
    {
        m2[1]=1;
    }
    else
    {
        m2[1]=0;
        if(m2[2]==0)
        {
            m2[2]=1;
        }
        else
        {
            m2[2]=0;
            if(m2[3]==0)
            {
                m2[3]=1;
            }
        }
    }
}
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
        else
        {
            m2[3]=0;
        }
    }
}
return 0;
}
```

```
int left(int acc[], int q[])
{
    int i;
    for(i=3; i>0; i--)
    {
        acc[i]=acc[i-1];
    }
    acc[0]=q[3];
    for(i=3; i>0; i--)
    {
        q[i]=q[i-1];
    }
}
```

```
int add(int acc[], int m[])
{
    int i, carry=0;
    for(i=0; i<4; i++)
    {
        if(acc[i]+m[i]+carry==0)
        {
            acc[i]=0;
            carry=0;
        }
        else if(acc[i]+m[i]+carry==1)
        {
            acc[i]=1;
            carry=0;
        }
    }
}
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
else if(acc[i]+m[i]+carry==2)
{
    acc[i]=0;
    carry=1;
}
else if(acc[i]+m[i]+carry==3)
{
    acc[i]=1;
    carry=1;
}
}
return 0;
}
```

Output:

Enter the Dividend: 10

Enter the Divisor: 2

A	Q	Comments
0000	1010	Start
0001	010_	Left Shift A,Q
1111	010_	A=A-M
0001	0100	Q ₀ =0; A=A+M
0010	100_	Left Shift A,Q
0000	100_	A=A-M
0000	1001	Q ₀ =1
0001	001_	Left Shift A,Q
1111	001_	A=A-M
0001	0010	Q ₀ =0; A=A+M
0010	010_	Left Shift A,Q
0000	010_	A=A-M
0000	0101	Q ₀ =1

Quotient = 0101 Remainder = 0000



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Conclusion -

The Non-Restoring Division Algorithm is another approach to binary division, similar to the Restoring Division Algorithm, but with a different method for handling negative remainders. It is a widely used technique in digital systems and microprocessors for efficient binary division. This C program illustrates the theoretical principles of the Non-Restoring Division Algorithm by performing binary division on the provided dividend and divisor. It follows the key steps of the algorithm, including left-shifting, subtraction, quotient bit determination, and correction. The program outputs the quotient and remainder, which represent the result of the division operation.