TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

A LAB REPORT
ON

Division of two unsigned integers by restoring Methods

Lab No: 2
Experiments Date:
Submission Date:

Submitted By:
Name: Nabin Khanal
Group: (076 BCT 036)
Roll No: Group B

Submitted To:
Department of
Electronics and
computer
Engineering

TITLE: Division of two unsigned integer binary numbers

OBJECTVE:

To implement restoring division algorithm in digital computer.

Theory:

Division of two fixed-point binary numbers in signed magnitude representation is done with paper and pencil by a process of successive compare, shift and subtract operations. Binary division is simpler than decimal division because the quotient digits are either 0 or 1. and there is no need to estimate how many times the dividend or partial remainder fits into the divisor.

When the division is implemented in digital computer, instead of shifing the divisor to the right, the dividend or partial remainder is shifted to the left, thus leaving the two numbers in the required relative position. Subtraction may be achieved by adding A to the 2's ~~component~~ complement at B. The information about relative magnitudes is then available from end carry.

- Division can be implemented with restoring algorithm as follows:

Q holds the dividend, M holds the divisor and A register is 0.

On completion, Q will get the Quotient and A will get the remainder.

The number of steps is equal to the number of bits in the dividend.

1) At each step left shift the AQ registers collectively. by 1 position.

(2) Subtract the divisor from AC perform A-M)

3) If the result is positive then the step is successful. In this case the quotient bit will be 1 and restoration is not required.

4) If the result is negative the step is said to be unsuccessful, quotient bit will be 0.
Here restoration is performed by adding back the divisor.

Repeat steps 1 to 4 for all bits of dividend.

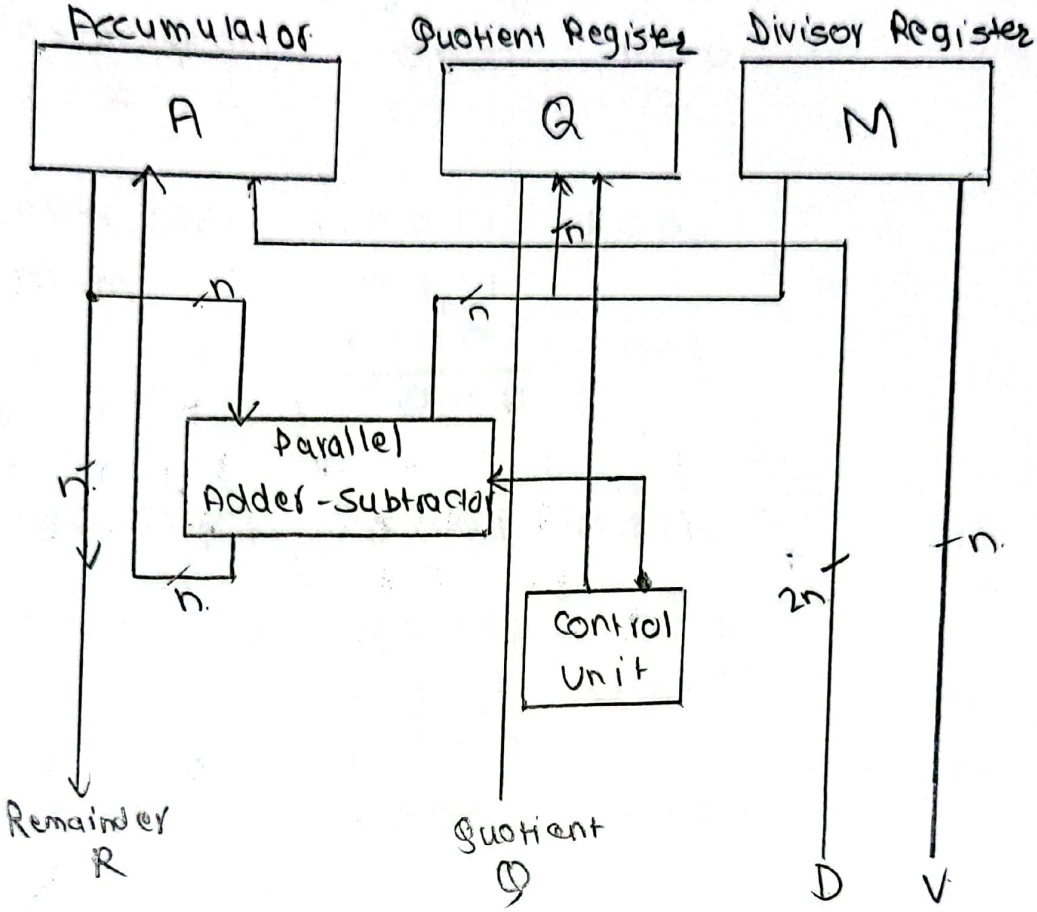

Fig: Hardware implementation of restoring division algorithm

Example. $7 \div 3$: -3 : 1101

	Accumulator	Dividend	Divisor.
Initial values	0000	0111	0011
: left shift	0000	111-	
A-M	1101		
	1101		
Restoration	0000	1110.	

(2) Left Shift

A-M	0001	110_
	0001	
	+1101	1100
	1110.	
Restoration	0001	1100.

(3) left shift

	0011	100_
A-M	0011	
	+1101	1001
	0000	

(4) left shift

	0001	001_
A-M	0001	
	+1101	0010.
	1110	
Restoration	0001	0010.

∴ Quotient = 0010 = 2
Remainder = 0001 = 1

⇒ 7 ÷ 3 ⇒ Quotient 2
Remainder 1

Source code:

```
from difference import subtract

def shift(A, Q):
    return A[1:]+Q[0], Q[1:]+'_'


def divide(dividend, divisor, n):
    A="".zfill(n)
    Q= dividend
    M= divisor

    for in in range(n):
        A,Q= shift(A,Q)
        sub = subtract(A,M,n)
        if (sub[0]=='1'):
            Q = Q[:len(Q)-1]+'0'
        else:
            A= sub
            Q = Q[:len(Q)-1]+'1'
    return Q,A

def main():
    n= int(input('Enter the numberof bits'))

    n1 = input('Enter first number:')
    n2 = input('Enter second number:')

    n1 = n1.zfill(n)
    n2 = n2.zfill(n)
    print(divide(n1,n2,n))

main()
```

Output:

```
Enter the no of bits: 4
Enter the first number: 0111
Enter the second number: 0011

('0010', '0001')
```

Discussion

Restoring division algorithm is implemented in this lab using python programming language. The division function restoring() is provided with dividend, divisor and number of bits. It takes the help of subtract function from previous lab and performs division and returns quotient and remainder.

Conclusion

To conclude, restoring algorithm can be implemented easily but it is slow due to continuous restores. So non-restoring algorithm makes its entry in division of two numbers.