Batch: B2 Roll No.: 16010124107
Experiment / assignment / tutorial No. 03
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

TITLE : To study and implement Restoring method of division

AIM : The basis of algorithm is based on paper and pencil approach and the operation involves repetitive shifting with addition and subtraction. So the main aim is to depict the usual process in the form of an algorithm.

Expected OUTCOME of Experiment: (Mention CO /CO's attained here)

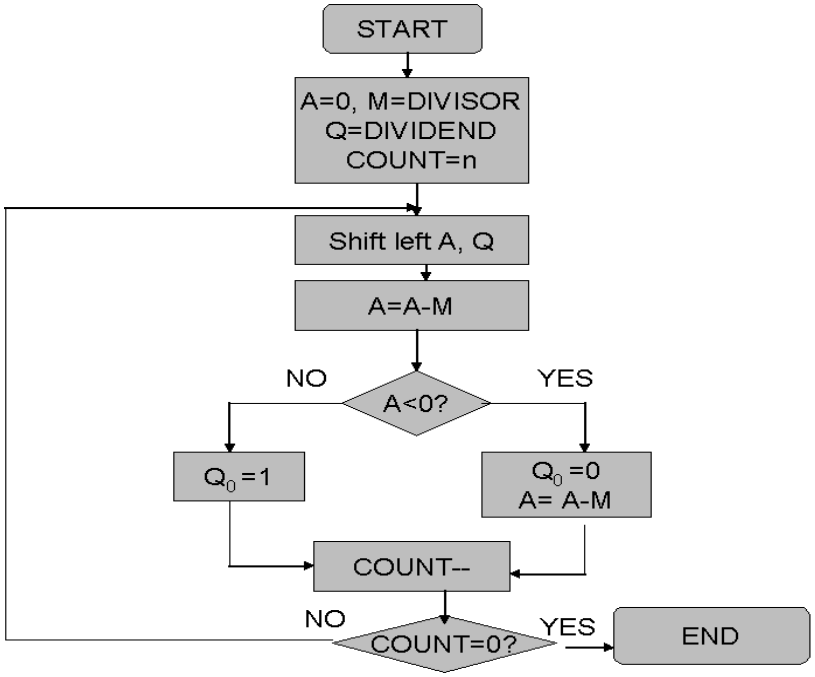
Books/ Journals/ Websites referred:

1. Carl Hamacher, Zvonko Vranesic and Safwat Zaky, "Computer Organization", Fifth Edition, TataMcGraw-Hill.
2. William Stallings, "Computer Organization and Architecture: Designing for Performance", Eighth Edition, Pearson.
3. Dr. M. Usha, T. S. Srikanth, "Computer System Architecture and Organization", First Edition, Wiley-India.

Pre Lab/ Prior Concepts:

The Restoring algorithm works with any combination of positive and negative numbers

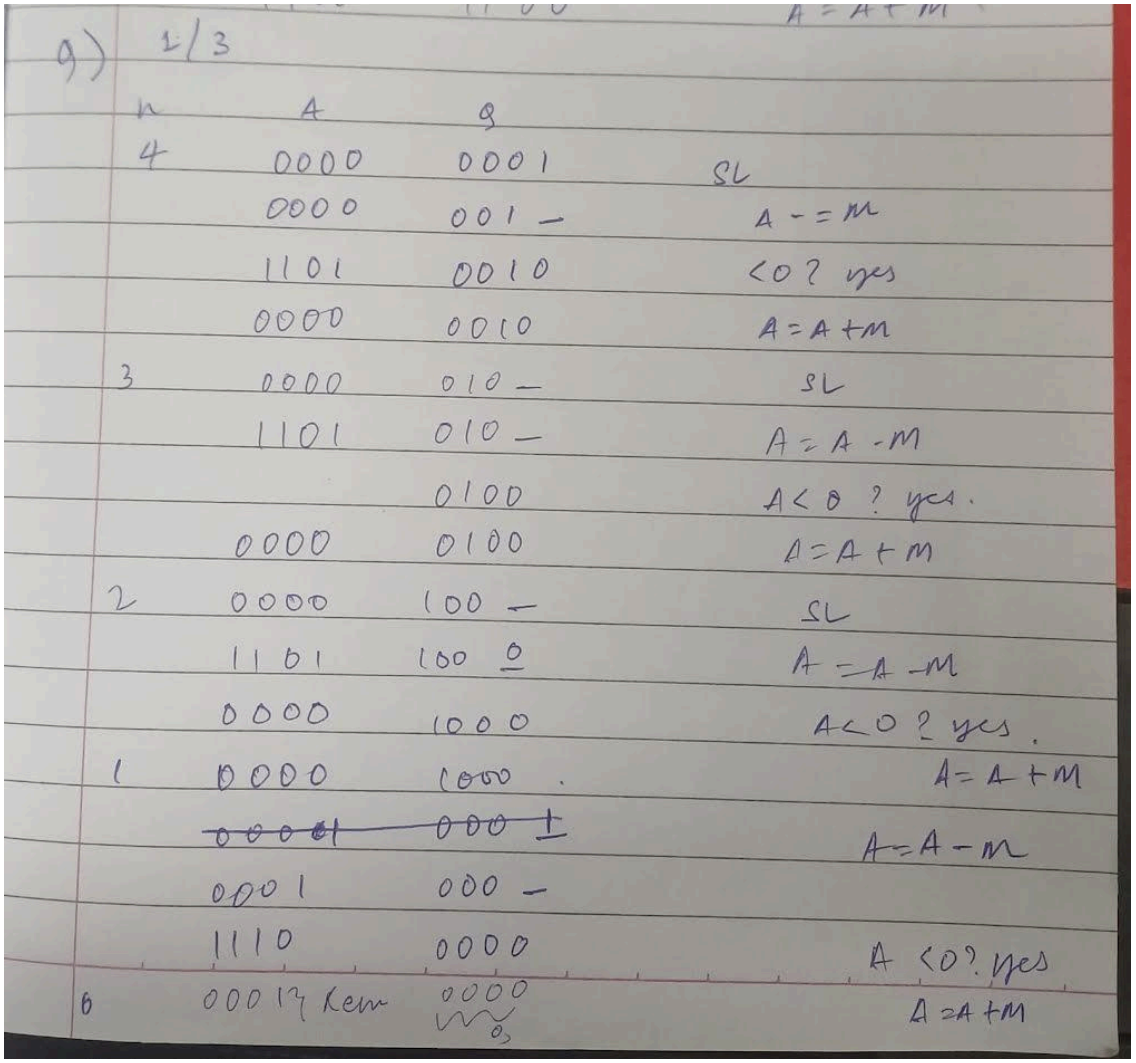
Flowchart for Restoring of Division:



Design Steps:

1. Start
2. Initialize A=0, M=Divisor, Q=Dividend and count=n (no of bits)
3. Left shift A, Q
4. If MSB of A and M are same
5. Then A=A-M
6. Else A=A+M
7. If MSB of previous A and present A are same
8. Q_0=0 & store present A
9. Else Q_0=0 & restore previous A
10. Decrement count.
11. If count=0 go to 11
12. Else go to 3
13. STOP

Example: - (Handwritten solved problem needs to be uploaded):-

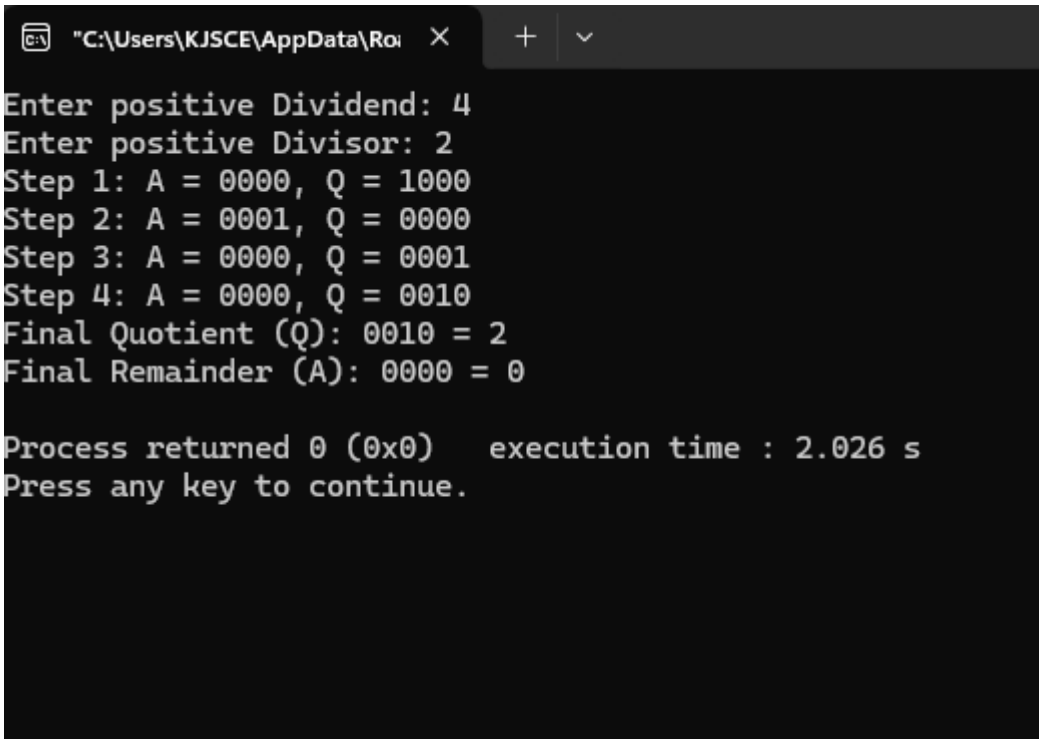


CODE:-

```
#include <iostream>

#include <bitset>

using namespace std;


const int n = 8;
```

```
void leftShift(bitset<n>& A, bitset<n>& Q) {
    bool msbQ = Q[n - 1];
    A <<= 1;
    A[0] = msbQ;
    Q <<= 1;
}

void printState(int step, bitset<n> A, bitset<n> Q) {
    cout << "Step " << step << ": A = " << A << ", Q = " << Q << endl;
}

int main() {
    int dividend, divisor;
    cout << "Enter Dividend: ";
    cin >> dividend;
    cout << "Enter Divisor: ";
    cin >> divisor;

    bitset<n> A(0);
    bitset<n> Q(dividend);
    bitset<n> M(divisor);

    int count = n;
```

```
    for (int step = 1; count > 0; --count, ++step) {

        leftShift(A, Q);

        int tempA = (int)(A.to_ulong()) - (int)(M.to_ulong());

        if (tempA < 0) {
            Q[0] = 0;
            tempA += M.to_ulong();
        } else {
            Q[0] = 1;
        }

        A = bitset<n>(tempA);
        printState(step, A, Q);
    }

    cout << "Final Quotient (Q): " << Q << " = " << Q.to_ulong() << endl;
    cout << "Final Remainder (A): " << A << " = " << A.to_ulong() << endl;

    return 0;
}
```



Conclusion:-

Restoring division is a method for performing binary division, commonly used in computer architecture. It involves repeatedly subtracting the divisor from the dividend and restoring the remainder if the result is negative. This process continues until the quotient and remainder are obtained.

Post Lab Descriptive Questions

1. **What are the advantages of restoring division over non restoring division?**

- The restoring division algorithm is conceptually simpler
- Restoring division ensures that the remainder is always non-negative
- Restoring division avoids the issue of sign correction at the end of the division process

Date: 01/08/2025