Batch: B-2 Roll No.: 16010122151
Experiment / assignment / tutorial No. 4
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

TITLE : To study and implement Non Restoring method of division

AIM : The basis of algorithm is based on paper and pencil approach and the operation involve 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)

CO1:Describe and define the structure of a computer with buses structure and detail

working of the arithmetic logic unit and its sub modules

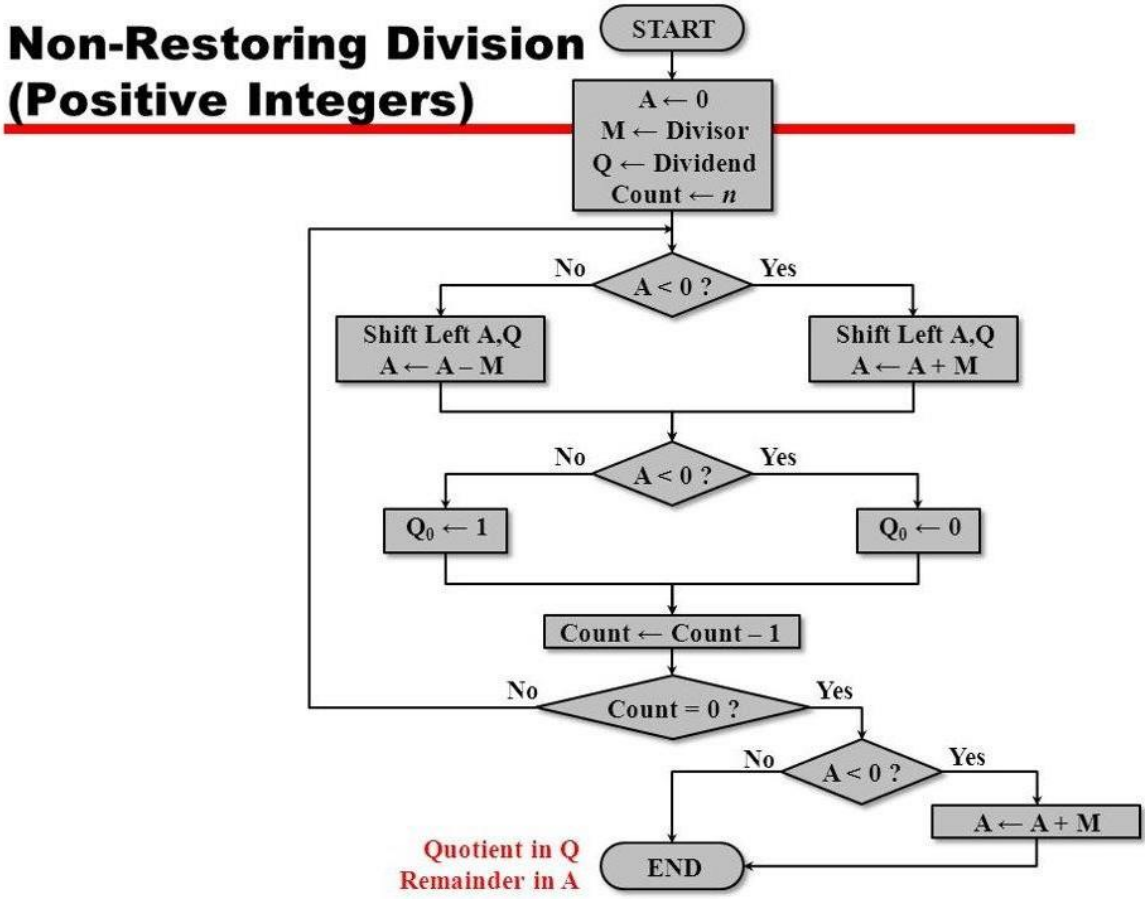
Books/ Journals/ Websites referred:

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 Non Restoring algorithm works with any combination of positive and negative numbers.

Flowchart for Non Restoring of Division:



Code:

```
#include <iostream>
#include <string>
using namespace std;
 // Function to add two binary numbers
string add(string A, string M)
{
   int carry = 0;
   string Sum = "";
      for (int i = A.length() - 1; i >= 0; i--) {
            int temp = (A[i] - '0') + (M[i] - '0') + carry;
       // If the binary number exceeds 1
      if (temp > 1) {
         Sum += to_string(temp % 2);
         carry = 1;
      }
      else {
         Sum += to_string(temp);
         carry = 0;
      }
   }
   // MSB to LSB
   return string(Sum.rbegin(), Sum.rend());
}
```

```
string compliment(string m)
{
    string M = ""; // Iterating through the number
    for (int i = 0; i < m.length(); i++) {
        // Computing the compliment
        M += to_string((m[i] - '0' + 1) % 2);
    }
     // Adding 1 to the computed
    // value
    M = add(M, "0001");
    return M;
}
 void nonRestoringDivision(string Q, string M, string A)
{
    // Computing the length of the
    // number
    int count = M.length();
    string comp_M = compliment(M);
    string flag = "successful";
     cout << "Initial Values: A: " << A << " Q: " << Q
         << " M: " << M << endl;
     while (count) {
        // Printing the values at every step
        cout << "\nstep: " << M.length() - count + 1;
         cout << " Left Shift and ";
```

```
A = A.substr(1) + Q[0];
 if (flag == "successful") {
    A = add(A, comp_M);
    cout << "subtract: ";
}
else {
    A = add(A, M);
    cout << "Addition: ";
}
 cout << "A: " << A << " Q: " << Q.substr(1) << "_";
 if (A[0] == '1') {
          Q = Q.substr(1) + "0";
    cout << " -Unsuccessful";
    flag = "unsuccessful";
    cout << " A: " << A << " Q: " << Q
        << " -Addition in next Step" << endl;
}
else {
     Q = Q.substr(1) + "1";
    cout << " Successful";
    flag = "successful";
    cout << " A: " << A << " Q: " << Q
        << " -Subtraction in next step" << endl;
}
```

```
        count--;
    }
    cout << "\nQuotient(Q): " << Q << " Remainder(A): " << A
         << endl;
}
 // Driver code
int main()
{
    string dividend = "0111";
    string divisor = "0101";
    string accumulator = string(dividend.size(), '0');
    nonRestoringDivision(dividend, divisor, accumulator);
    return 0;
}
```

OUTPUT:

```
Initial Values: A: 0000 Q: 0111 M: 0101
Step: 1 Left Shift and subtract: A: 1011 Q: 111_ -Unsuccessful A: 1011 Q: 1110 -Addition in next Step
Step: 2 Left Shift and Addition: A: 1100 Q: 110_ -Unsuccessful A: 1100 Q: 1100 -Addition in next Step
Step: 3 Left Shift and Addition: A: 1110 Q: 100_ -Unsuccessful A: 1110 Q: 1000 -Addition in next Step
Step: 4 Left Shift and Addition: A: 0010 Q: 000_ Successful A: 0010 Q: 0001 -Subtraction in next step
Quotient(Q): 0001 Remainder(A): 0010
```

Conclusion:

We have learned and written the code for Non restoring Division.

Post Lab Descriptive Questions

1. **What are the advantages of non restoring division over restoring division?**
   The advantage of using non-restoring arithmetic over the standard restoring division is **that a test subtraction is not required**; the sign bit determines whether an addition or subtraction is used.

Date:___24/08/23_____ **Signature of faculty in-charge**