**Batch: C3 Roll No.: 16010123217**

**Experiment / assignment / tutorial No. 2**

|  |
| --- |
| **TITLE:** To study and implement Booth’s Multiplication Algorithm. |

**AIM:** Booth’s Algorithm for Multiplication

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

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

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.

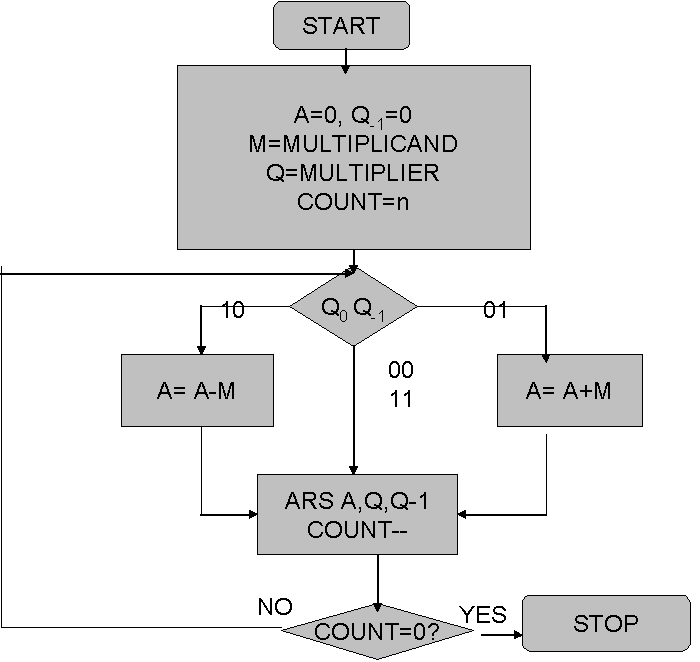
3. Dr. M. Usha, T. S. Srikanth, “Computer System Architecture and Organization”, First Edition, Wiley-India.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:**

It is a powerful algorithm for signed number multiplication which generates a 2n bit product and treats both positive and negative numbers uniformly. Also the efficiency of the algorithm is good due to the fact that, block of 1’s and 0’s are skipped over and subtraction/addition is only done if pair contains 10 or 01

**Flowchart:**

****

**Design Steps**:

1. Start
2. Get the multiplicand (M) and Multiplier (Q) from the user
3. Initialize A= Q-1 =0
4. Convert M and Q into binary
5. Compare Q0 andQ-1 and perform the respective operation.

|  |  |
| --- | --- |
| **Q0 Q-1** | **Operation** |
| 00/11 | Arithmetic right shift |
| 01 | A+M and Arithmetic right shift |
| 10 | A-M and Arithmetic right shift |

6. Repeat steps 5 till all bits are compared

7. Convert the result to decimal form and display

8. End

Code:  
#include <iostream>

#include <bitset>

#include <string>

using namespace std;

string int2binary(int a) {

bitset<6> bs(a);

return bs.to\_string();

}

int binary2int(string a, string q) {

// Combine the binary string and the multiplier string

string combined = a + q;

// Convert the binary string to an integer

int result = 0;

for (char bit : combined) {

result = (result << 1) | (bit - '0');

}

return result;

}

string addbinary(const string& bin1, const string& bin2) {

int carry = 0;

string result;

for (int i = 5; i >= 0; --i) {

int bit1 = bin1[i] - '0';

int bit2 = bin2[i] - '0';

int sum = bit1 + bit2 + carry;

result = char((sum % 2) + '0') + result;

carry = sum / 2;

}

return result;

}

string twosComplement(const string& binary) {

string onesComplement;

for (char bit : binary) {

onesComplement += (bit == '0') ? '1' : '0';

}

return addbinary(onesComplement, "000001");

}

void rightShift(string& a, string& q, int& q1) {

q1 = q.back() - '0';

q = a.back() + q.substr(0, 5);

a = a[0] + a.substr(0, 5);

}

int main() {

int m, q, n = 6, q1 = 0;

string a = "000000";

cout << "Om Thanage\t16010123217\n";

cout << "Welcome to Booth's Algorithm (upto 6 bits)\n";

cout << "Enter Multiplicand M: ";

cin >> m;

cout << "Enter Multiplier Q: ";

cin >> q;

string bm = int2binary(m);

string bq = int2binary(q);

string bm\_twosComplement = twosComplement(bm);

cout << "Binary of M: " << bm << endl;

cout << "Binary of Q: " << bq << endl;

cout << "2's Complement of M: " << bm\_twosComplement << endl;

cout << "\nInitial values: \n";

cout << "A: " << a << ", Q: " << bq << ", Q-1: " << q1 << ", M: "<< bm <<"\n\n";

while (n--) {

if (bq.back() == '0' && q1 == 1) {

a = addbinary(a, bm);

cout << "Step " << 6 - n << ": A = A + M = " << a << "\n";

} else if (bq.back() == '1' && q1 == 0) {

a = addbinary(a, bm\_twosComplement);

cout << "Step " << 6 - n << ": A = A - M = " << a << "\n";

}

rightShift(a, bq, q1);

cout << "Step " << 6 - n << ": Right Shift -> A: " << a << ", Q: " << bq << ", Q-1: " << q1 << ", M: "<< bm <<"\n\n";

}

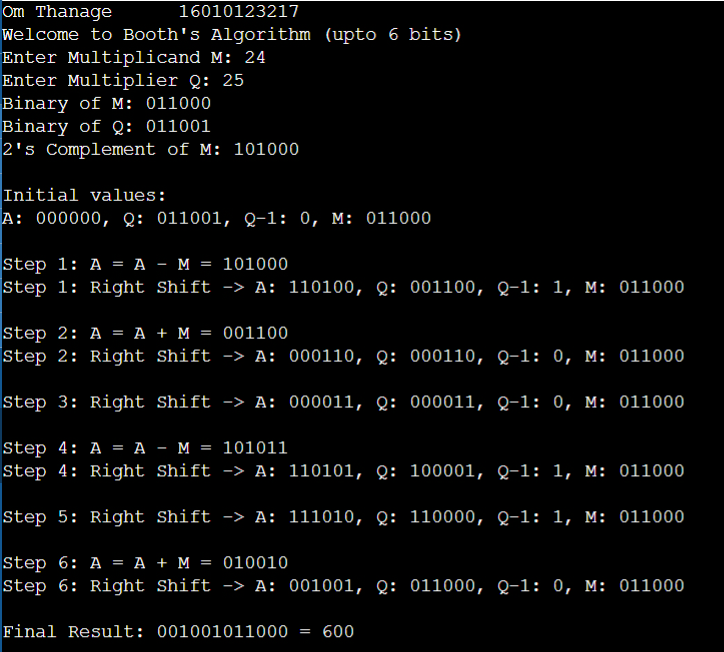
int finalResult = binary2int(a, bq);

cout << "Final Result: " << a << bq << " = " << finalResult << endl;

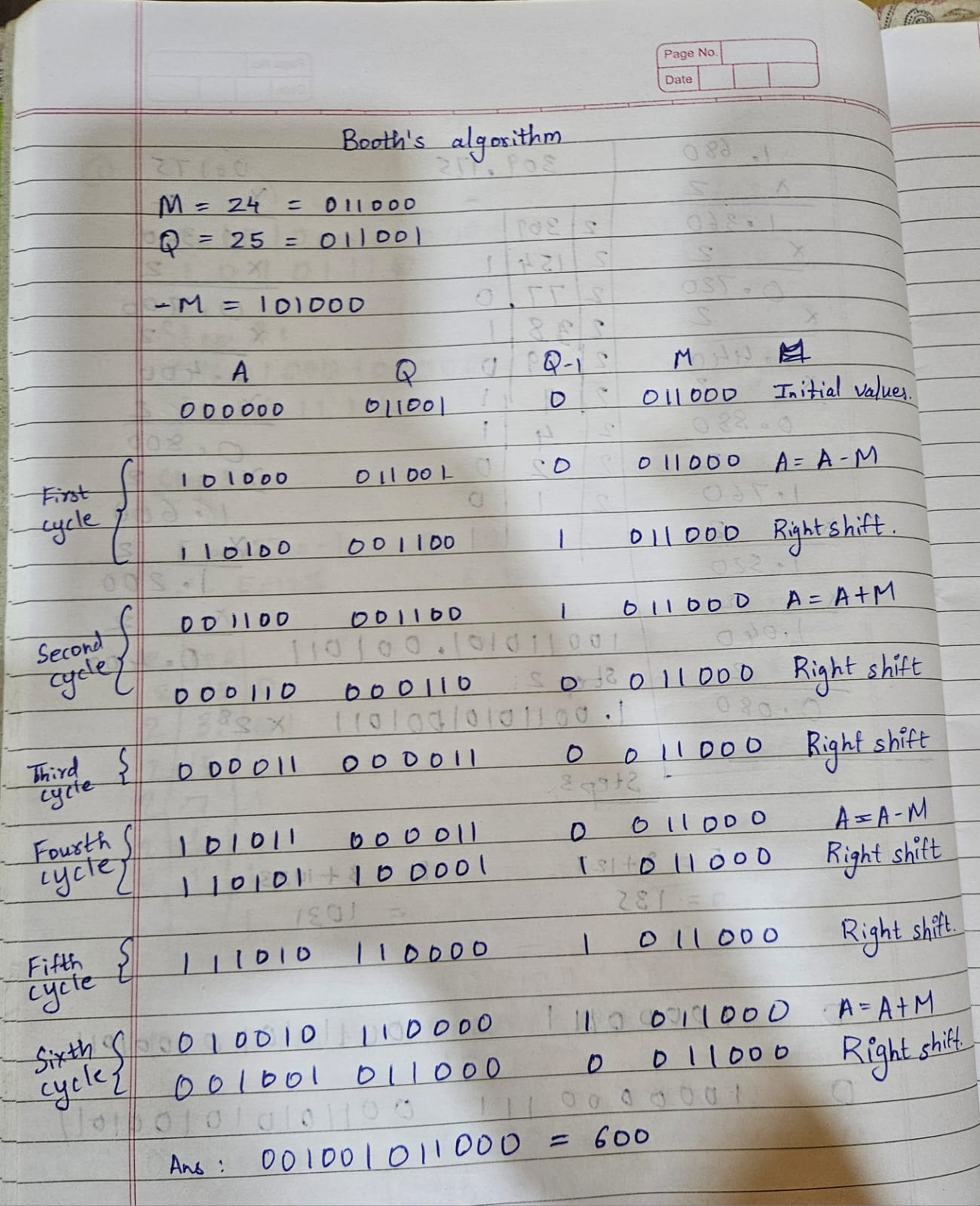
return 0;

}

Output:



Example: (Handwritten solved problem needs to be uploaded)



**Conclusion:**

**In this experiment we learnt how to multiply two positive numbers using Booth's algorithm. It is also faster because of its conditional adding and subtracting for pairs 10 and 01 thus reducing the steps.**

**Post Lab Descriptive Questions**

1. **Explain advantages and disadvantages of Booth’s algorithm.**

## Ans.

## Advantages:

**Faster than traditional multiplication**

**Efficient for signed numbers**

**Lower hardware requirement**

**Widely used in hardware**

## Disadvantages:

**Complex to understand**

**Limited applicability**

**Higher latency**

**Higher power consumption**

1. **Is Booth’s recoding better than Booth’s algorithm? Justify**

Ans. Booth's Recoding can be considered better than Booth's Algorithm in specific contexts where optimization of partial products is crucial. Booth's Recoding reduces the number of non-zero digits in the multiplier, leading to fewer operations and potentially faster multiplication. However, Booth's Algorithm is more general and directly handles multiplication efficiently by reducing the number of operations when dealing with sequences of 1s. The choice between them depends on the specific needs of the system—if minimizing partial products is the priority, Booth's Recoding might be preferred.

**Date: 9/08/24**