LAB1. Verify the 1's and 2's complement operations of computer system by using c or c++ codes.

#include <iostream>

#include <string.h>

using namespace std;

int main()

{

string bin, temp, ones, twos;

int len;

cout << "\t\t\*\*\*\*\*1's and 2's COMPLEMENT\*\*\*\*\*\*\n\n";

cout << "Enter a valid binary number: ";

cin >> bin;

len = bin.length();

int count = 0;

while (count != len)

{

if (bin[count] != '1' && bin[count] != '0')

{

cout << "Invalid binary number." << endl;

main();

}

count += 1;

}

// 1's complement

for (int i = 0; i < len; i++)

{

if (bin[i] == '1')

{

bin[i] = '0';

}

else

{

bin[i] = '1';

}

}

ones = bin;

temp = ones;

bin = temp;

// 2's complement

int carry = 1;

for (int i = len - 1; i >= 0; i--)

{

if (bin[i] == '1' && carry == 1)

{

bin[i] = '0';

}

else if (bin[i] == '0' && carry == 1)

{

bin[i] = '1';

carry = 0;

}

}

twos = bin;

cout << "The 1s complement of the entered number is: " << ones << endl;

cout << "The 2s complement of the entered number is: " << twos << endl;

return 0;

}

**Output:**

Enter a valid binary number: 10001

The 1s complement of the entered number is: 01110

The 2s complement of the entered number is: 01111

LAB2. Verify the Multiplication of Signed Magnitude data of computer system by using c or c++ codes.

#include <iostream>

#include <math.h>

int q = 0, b = 0, c = 0, e = 0, q1 = 0, b1 = 0;

int qnum[4] = {0}, bnum[4] = {0};

int acc[4] = {0}, res[4] = {0};

int sc = 0, bs = 0, qs = 0, asize = 0;

void binary()

{

b1 = fabs(b);

q1 = fabs(q);

int r1, r2, i;

for (i = 0; i < 4; i++)

{

r1 = b1 % 2;

b1 = b1 / 2;

r2 = q1 % 2;

q1 = q1 / 2;

bnum[i] = r1;

qnum[i] = r2;

}

sc = sizeof(qnum) / sizeof(int);

asize = sc;

}

void add(int num[])

{

int i;

c = 0;

for (i = 0; i < 4; i++)

{

res[i] = acc[i] + num[i] + c;

if (res[i] >= 2)

{

c = 1;

}

else

{

c = 0;

}

e = c;

res[i] = res[i] % 2;

}

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

{

acc[i] = res[i];

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

}

printf(" : ");

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

{

printf("%d", qnum[i]);

}

}

void rshift()

{

int temp2 = acc[0], i;

for (i = 1; i < 4; i++)

{

acc[i - 1] = acc[i];

}

acc[3] = e;

e = 0;

for (i = 1; i < 4; i++)

{

qnum[i - 1] = qnum[i];

}

qnum[3] = temp2;

printf("\n R-SHIFT: ");

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

{

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

}

printf(":");

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

{

printf("%d", qnum[i]);

}

}

int main()

{

int i;

int p = 0, n = 1;

printf("\t\t\*\*\*\*\*\*SIGNED MAGNITUDE MULTIPLICATION ALGORITHM\*\*\*\*\*\*\n");

printf("\nEnter two numbers to multiply: ");

printf("\nBoth must be less than 16");

do

{

printf("\nEnter b: ");

scanf("%d", &b);

printf("Enter Q: ");

scanf("%d", &q);

} while (b >= 16 || q >= 16);

printf("\n Expected product = %d", b \* q);

binary();

printf("\nS.C. = %d", sc);

printf("\n\n Signed Binary Equivalents are: ");

printf("\n b = ");

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

{

printf("%d", bnum[i]);

}

printf("\n q = ");

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

{

printf("%d", qnum[i]);

}

printf("\n\n");

while (sc != 0)

{

printf("\nS.C. = %d", sc);

if (qnum[0] == 0)

{

printf("\n-->");

rshift();

}

else

{

printf("\n-->");

printf("\n ADD B: ");

add(bnum);

rshift();

}

sc--;

}

printf("\nproduct is = ");

if ((b < 0 && q > 0) || (b > 0 && q < 0))

printf("%d", n);

else

printf("%d", p);

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

{

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

}

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

{

printf("%d", qnum[i]);

}

return 0;

}

**Output:**

Enter two numbers to multiply:

Both must be less than 16

Enter b: 14

Enter Q: 7

Expected product = 98

S.C. = 4

Signed Binary Equivalents are:

b = 1110

q = 0111

S.C. = 4

-->

ADD B: 1110 : 0111

R-SHIFT: 0111:0011

S.C. = 3

-->

ADD B: 0101 : 0011

R-SHIFT: 1010:1001

S.C. = 2

-->

ADD B: 1000 : 1001

R-SHIFT: 1100:0100

S.C. = 1

-->

R-SHIFT: 0110:0010

product is = 001100010

LAB3. Verify Booth multiplication algorithm of computer system by using c or c++ codes.

#include <iostream>

using namespace std;

int a = 0, b = 0, c = 0, com[5] = {1, 0, 0, 0, 0};

int anum[5] = {0}, bnum[5] = {0}, anumcp[5] = {0};

int acomp[5] = {0}, bcomp[5] = {0}, pro[5] = {0}, res[5] = {0};

void binary()

{

a = abs(a);

b = abs(b);

int r, r2, i;

for (i = 0; i < 5; i++)

{

r = a % 2;

a /= 2;

r2 = b % 2;

b /= 2;

anum[i] = r;

anumcp[i] = r;

bnum[i] = r2;

if (r2 == 0)

bcomp[i] = 1;

if (r == 0)

acomp[i] = 1;

}

c = 0;

for (i = 0; i < 5; i++)

{

res[i] = com[i] + bcomp[i] + c;

if (res[i] >= 2)

c = 1;

else

c = 0;

res[i] = res[i] % 2;

}

for (i = 4; i >= 0; i--)

bcomp[i] = res[i];

}

void add(int num[])

{

int i, c = 0;

for (i = 0; i < 5; i++)

{

res[i] = pro[i] + num[i] + c;

if (res[i] >= 2)

c = 1;

else

c = 0;

res[i] = res[i] % 2;

}

for (i = 4; i >= 0; i--)

{

pro[i] = res[i];

cout << pro[i];

}

cout << " : ";

for (i = 4; i >= 0; i--)

cout << anumcp[i];

}

void arshift()

{

int temp = pro[4], temp2 = pro[0], i;

for (i = 1; i < 5; i++)

pro[i - 1] = pro[i];

pro[4] = temp;

for (i = 1; i < 5; i++)

anumcp[i - 1] = anumcp[i];

anumcp[4] = temp2;

cout << "\nAR-SHIFT: ";

for (i = 4; i >= 0; i--)

cout << pro[i];

cout << " : ";

for (i = 4; i >= 0; i--)

{

cout << anumcp[i];

}

}

int main()

{

int i, q = 0;

cout << "\t\t\*\*\*\*\*\*BOOTH'S MULTIPLICATION ALGORITHM\*\*\*\*\*\*\n";

cout << "\nEnter two numbers to multiply: ";

cout << "\nBoth must be less than 16";

do

{

cout << "\nEnter A: ";

cin >> a;

cout << "Enter B: ";

cin >> b;

} while (a >= 16 || b >= 16);

cout << "\nExpected product = " << a \* b;

binary();

cout << "\n\nBinary Equivalents are: ";

cout << "\nA = ";

for (i = 4; i >= 0; i--)

cout << anum[i];

cout << "\nB = ";

for (i = 4; i >= 0; i--)

cout << bnum[i];

cout << "\nB'+ 1 = ";

for (i = 4; i >= 0; i--)

cout << bcomp[i];

cout << "\n\n";

for (i = 0; i < 5; i++)

{

if (anum[i] == q)

{

cout << "\n-->";

arshift();

q = anum[i];

}

else if (anum[i] == 1 && q == 0)

{

cout << "\n-->";

cout << "\nSUB B: ";

add(bcomp);

arshift();

q = anum[i];

}

else

{

cout << "\n-->";

cout << "\nADD B: ";

add(bnum);

arshift();

q = anum[i];

}

}

cout << "\nProduct: ";

for (i = 4; i >= 0; i--)

cout << pro[i];

for (i = 4; i >= 0; i--)

cout << anumcp[i];

return 0;

}

**Output:**

\*\*\*\*\*\*BOOTH'S MULTIPLICATION ALGORITHM\*\*\*\*\*\*

Enter two numbers to multiply:

Both must be less than 16

Enter A: 15

Enter B: 4

Expected product = 60

Binary Equivalents are:

A = 01111

B = 00100

B'+ 1 = 11100

-->

SUB B: 11100 : 01111

AR-SHIFT: 11110 : 00111

-->

AR-SHIFT: 11111 : 00011

-->

AR-SHIFT: 11111 : 10001

-->

AR-SHIFT: 11111 : 11000

-->

ADD B: 00011 : 11000

AR-SHIFT: 00001 : 11100

Product: 0000111100