**AIM**-

Perform Division of following numbers using Restoring Division Algorithm

Dividend=1011 Divisor=0011

**THEORY-**

Normal division :

Divisor 00001101 Quotient

1011 10010011 Dividend

* 1011

001110

* 1011 Partial Remainders

001111

* 1011

0100 Remainder

First bits of dividend are examined from left to right, until the set of bits examined represents a number greater than or equal to the divisor. This is refer to as the divisor being able to divide the number.

Until this event occurs, zero’s are placed in the quotient from left to right.

When the event occurs , a one is placed in the quotient and the divisor is subtracted from the partial dividend. The result is referred as partial remainder . The process continues until all the bits of dividend are exhausted.

* **Flowchart for unsigned binary division**

START

No

No

Yes

Yes

END

Count = 0

Count 🡨 count - 1

Q0🡨 1 , A 🡨A + M

Q0🡨 1

A < 0

A 🡨 A - M

Shift left A, Q

A 🡨 0 , M 🡨 divisor ,

Q 🡨 dividend , count 🡨 n

**DESCRIPTION-**

Figure shows a machine algorithm that corresponds to a long division process.

The divisor is placed in the M register , the dividend in the Q register.

At each step , the A and Q registers together are shifted to left by one-bit.

M is subtracted from A to determine whether A divides the partial remainder. If it does , then Q0 gets a 1 bit. Otherwise , Q0 gets a 0 bit and M must be added back to A to restore the previous value.

The count is then decremented and the process continues for n-steps. At the end, the quotient is in the Q register and the remainder is in the A register.

* **Algorithm for division of signed integers**

**Step I )**Load the divisor into the M register and the dividend into A,Q registers. The dividend must be expressed as a 2n-bit 2’s compliment number.

**Step II )** Shift A,Q left one-bit position.

**Step III )**If M and A have the same signs, perform A🡨A – M , otherwise A 🡨 A + M.

**Step IV )** The preceding operation is successful if the sign of A is the same before and after the operation.

1. If the operation is successful or A = 0 , then set Q0🡨 1.
2. If the operation is unsuccessful and A ≠ 0 then set Q0🡨 0 and restore the previous value of A.

**Step V )**Repeat the steps 2 through 4 as many times there are bit positions in Q.

**Step IV )**The remainder is in A . If signs of the divisor and dividend were the same , then the quotient is in Q , otherwise the correct quotient is the 2’s complement of Q.

***D = Q \* V + R***

Here , D = Dividend , Q = Quotient , V = Divisor and R = Remainder

Examples :-

Perform the following division using Restoring Division Algorithm.

**Dividend = 0111 (+7)** **Divisor = 0011 (+3)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| (Cycle) | A | Q | M |  |
|  | 0000 | 0111 | 0011 | Initial values |
| C1 | 0000  1011  0000 | 1110  1110  1110 | 0011  0011  0011 | Shift left  A 🡨 A – M  Restore |
| C2 | 0001  1110  0001 | 1100  1100  1100 | 0011  0011  0011 | Shift left  A 🡨 A – M  Restore |
| C3 | 0011  0000  0000 | 1000  1000  1001 | 0011  0011  0011 | Shift left  A 🡨 A – M  Set Q0 = 1 |
| C4 | 0001  1110  0001 | 0010  0010  0010 | 0011  0011  0011 | Shift left  A 🡨 A – M  Restore |

(Remainder) (Quotient)

**Quotient = 0010 (+2)**

**Remainder = 0001 (+1)**

**CONCLUSION-**

Thus we studied how to perform division using restoring division

algorithm.