Restoring Division Algorithm For Unsigned Integer

- A division algorithm provides a ***quotient*** and a ***remainder*** when we divide two number.

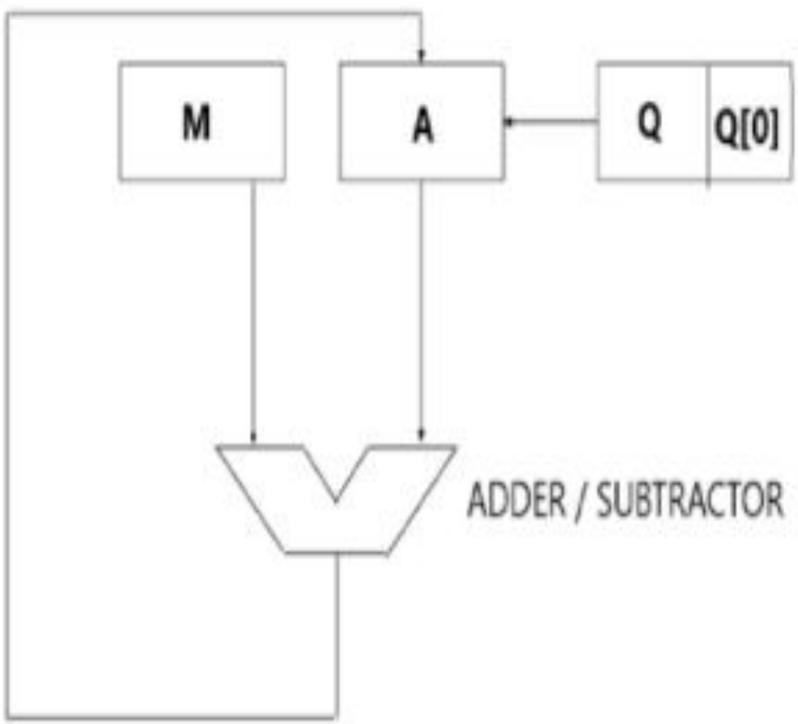
- They are generally of two type **slow algorithm and fast algorithm**.

- Slow division algorithm are
  - Restoring,
  - non-restoring,
  - non-performing restoring,
  - SRT algorithm

- Under fast comes
  - Newton–Raphson and
  - Goldschmidt

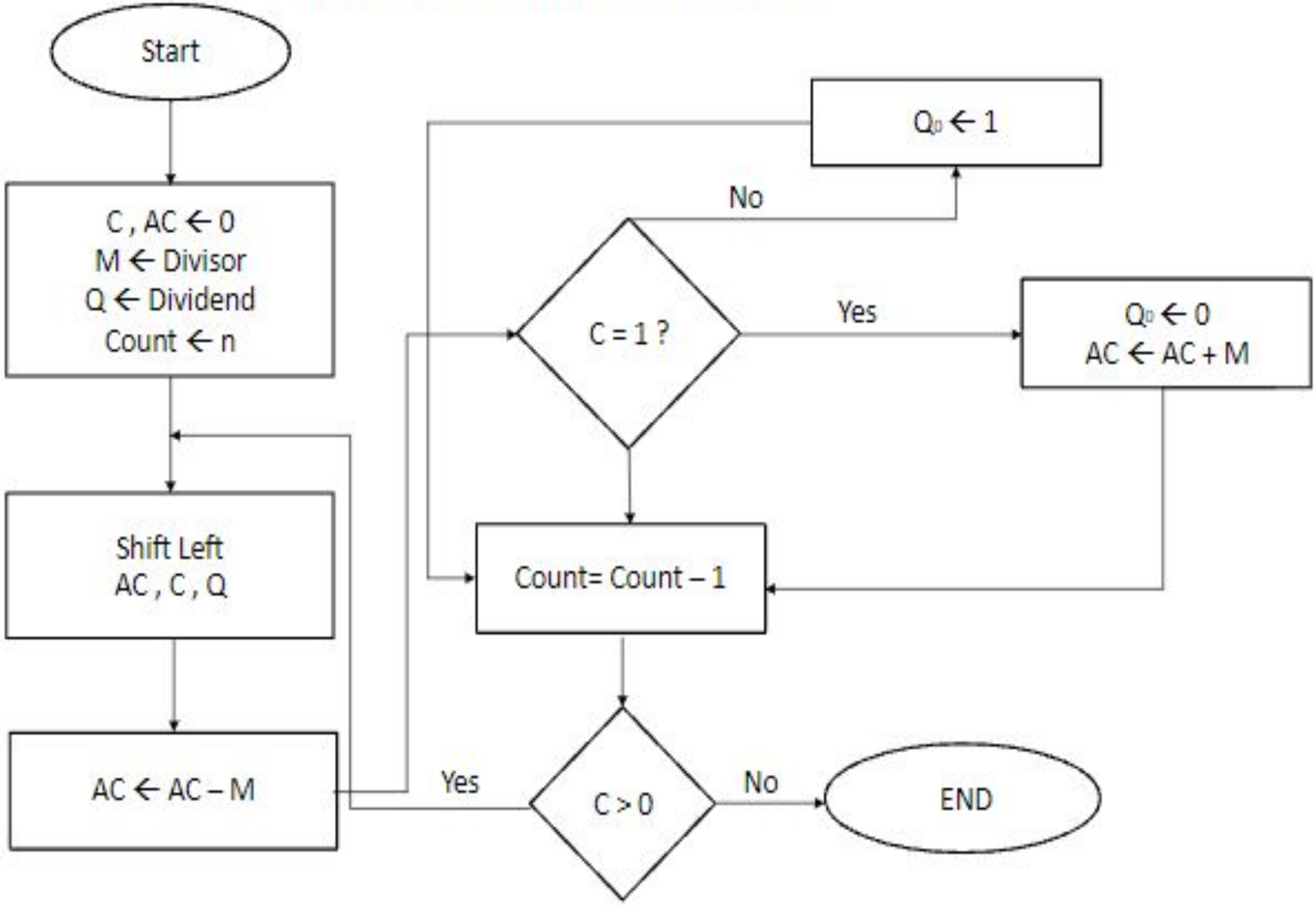
Restoring Division Algorithm For Unsigned Integer

Restoring term is due to fact that value of **register A** is restored after each iteration.



Here, **register Q** contain quotient
and
register A contain remainder.
Here,
n-bit dividend is loaded in **Q** and
divisor is loaded in **M.**
Value of Register is initially kept **0**
and this is the register whose value
is restored during iteration due to which it is
named **Restoring** .

Restoring Division Flowchart



Example:

Perform Non-Restoring Division for Unsigned Integer.

Dividend =11 Divisor =3 -M =11101

- **Step-1:** First the registers are initialized with corresponding values (Q = Dividend, M = Divisor, A = 0, n = number of bits in dividend)
- **Step-2:** Then the content of register A and Q is shifted right as if they are a single unit
- **Step-3:** Then content of register M is subtracted from A and result is stored in A
- **Step-4:** Then the most significant bit of the A is checked if it is 0 the least significant bit of Q is set to 1 otherwise if it is 1 the least significant bit of Q is set to 0 and value of register A is restored i.e the value of A before the subtraction with M
- **Step-5:** The value of counter n is decremented
- **Step-6:** If the value of n becomes zero we get of the loop otherwise we repeat from step 2
- **Step-7:** Finally, the register Q contain the quotient and A contain remainder

N	M	A	Q	OPERATION
4	00011	00000	1011	initialize
	00011	00001	011_	shift left AQ
	00011	11110	011_	A=A-M
	00011	00001	0110	Q[0]=0 And restore A

N	M	A	Q	OPERATION
3	00011	00010	110_	shift left AQ
	00011	11111	110_	A=A-M
	00011	00010	1100	Q[0]=0
	00011	00010	110_	shift left AQ

N	M	A	Q	OPERATION
2	00011	00101	100_	shift left AQ
	00011	00010	100_	A=A-M
	00011	00010	1001	Q[0]=1
	00011	00101	100_	shift left AQ

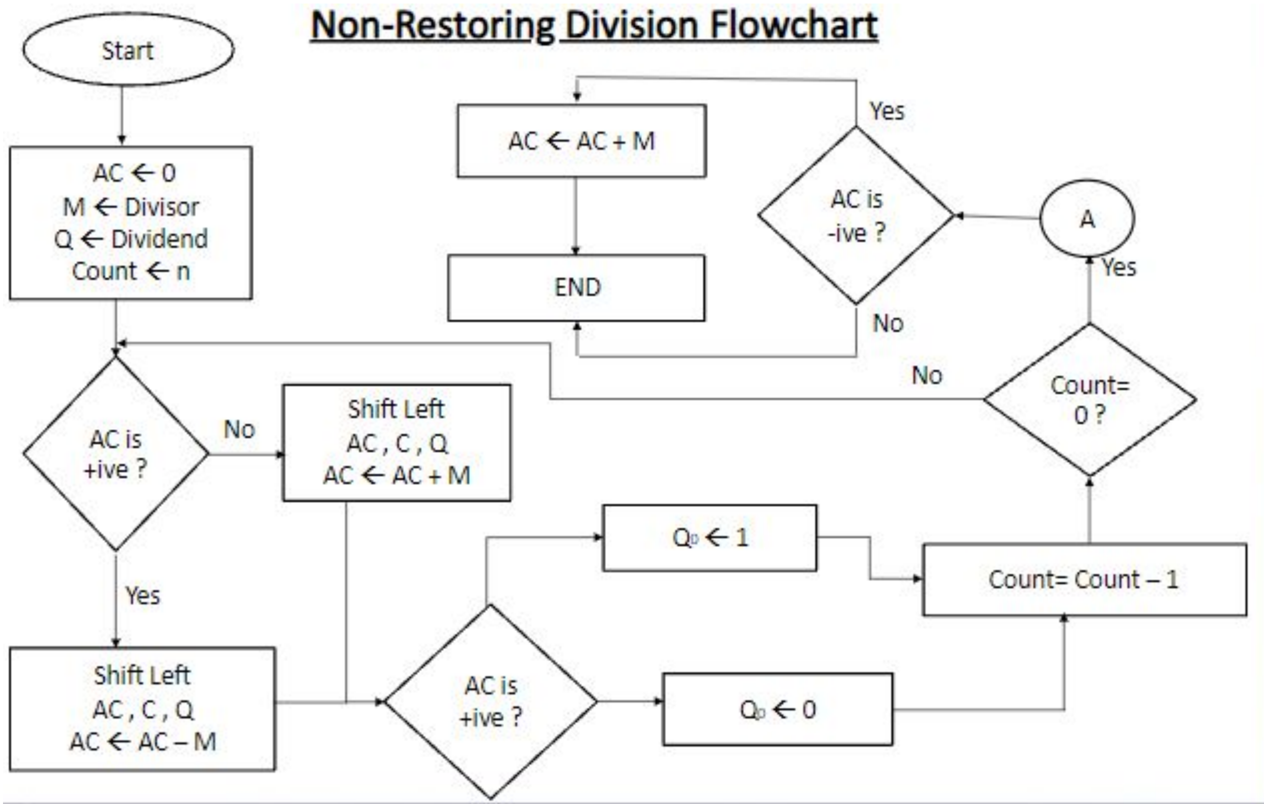
N	M	A	Q	OPERATION
1	00011	00101	001_	shift left AQ
	00011	00010	001_	A=A-M
	00011	00010	0011	Q[0]=1
	00011	00101	001_	shift left AQ

Non-Restoring Division For Unsigned Integer

- Non-Restoring division is less complex than the restoring one because simpler operation are involved i.e. addition and subtraction, also now restoring step is performed.
- In this method, rely on the sign bit of the register which initially contain zero named as **A**.

- 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. The disadvantage, though, is that an extra bit must be maintained in the partial remainder to keep track of the sign.

Non-Restoring Division Flowchart



- **Step-1:** First the registers are initialized with corresponding values (Q = Dividend, M = Divisor, A = 0, n = number of bits in dividend)
- **Step-2:** Check the sign bit of register A
- **Step-3:** If it is 1 shift left content of AQ and perform A = A+M, otherwise shift left AQ and perform A = A-M (means add 2's complement of M to A and store it to A)
- **Step-4:** Again the sign bit of register A
- **Step-5:** If sign bit is 1 Q[0] become 0 otherwise Q[0] become 1 (Q[0] means least significant bit of register Q)
- **Step-6:** Decrements value of N by 1
- **Step-7:** If N is not equal to zero go to **Step 2** otherwise go to next step
- **Step-8:** If sign bit of A is 1 then perform A = A+M
- **Step-9:** Register Q contain quotient and A contain remainder

Example:

Perform Non-Restoring Division for Unsigned Integer

Dividend =11 Divisor =3 -M =11101

- **Step-1:** First the registers are initialized with corresponding values (Q = Dividend, M = Divisor, A = 0, n = number of bits in dividend)
- **Step-2:** Check the sign bit of register A
- **Step-3:** If it is 1 shift left content of AQ and perform A = A+M, otherwise shift left AQ and perform A = A-M (means add 2's complement of M to A and store it to A)
- **Step-4:** Again the sign bit of register A
- **Step-5:** If sign bit is 1 Q[0] become 0 otherwise Q[0] become 1 (Q[0] means least significant bit of register Q)
- **Step-6:** Decrements value of N by 1

Example:

Perform Non-Restoring Division for Unsigned Integer

Dividend =11 Divisor =3 -M =11101

- **Step-7:** If N is not equal to zero go to **Step 2** otherwise go to next step
- **Step-8:** If sign bit of A is 1 then perform A = A+M
- **Step-9:** Register Q contain quotient and A contain remainder

N	M	A	Q	ACTION
4	00011	00000	1011	Start
		00001	011_	Left shift AQ
		11110	011_	A=A-M

N	M	A	Q	ACTION
3		11110	0110	Q[0]=0
		11100	110_	Left shift AQ
		11111	110_	A=A+M

N	M	A	Q	ACTION
2		11111	1100	Q[0]=0
		11111	100_	Left Shift AQ
		00010	100_	A=A+M

N	M	A	Q	ACTION
1		00010	1001	Q[0]=1
		00101	001_	Left Shift AQ
		00010	001_	A=A-M

N	M	A	Q	ACTION
0		00010	0011	Q[0]=1

From calculations we get

Quotient = 3 (Q)

Remainder = 2 (A)