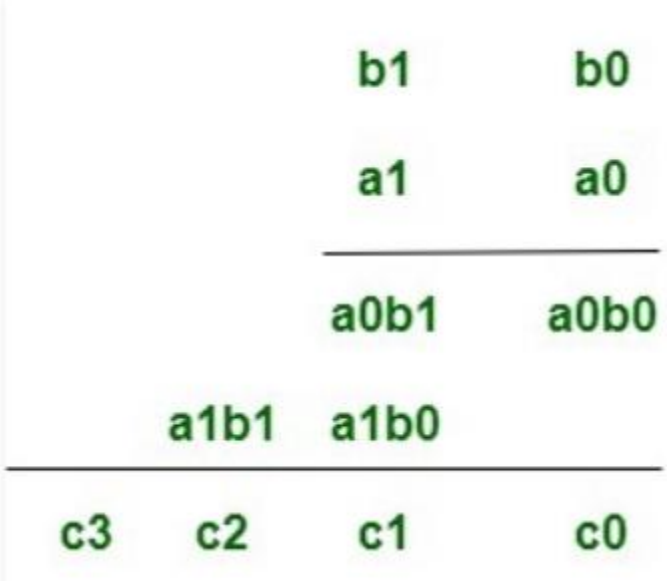
Module 3
Part 1

Arithmetic Algorithms

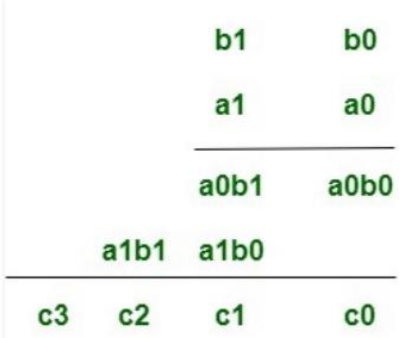
Array Multiplier

- Binary multiplication can be implemented in a combinational two-dimensional logic array called **array multiplier.**
- The main component in each in each cell is a full adder, FA.
- The AND gate in each cell determines whether a multiplicand bit mj , is added to the incoming partial product bit based on the value of the multiplier bit, qi .
- Each row i, where 0<= i <=3, adds the multiplicand (appropriately shifted) to the incoming parcel product, PPi , to generate the outgoing partial product, PP(i+1), if qi .=1
- If qi .=0, PPi is passed vertically downward unchanged. PP0 is all 0's and PP4 is the desired product. The multiplication is shifted left one position per row by the diagonal signal path.

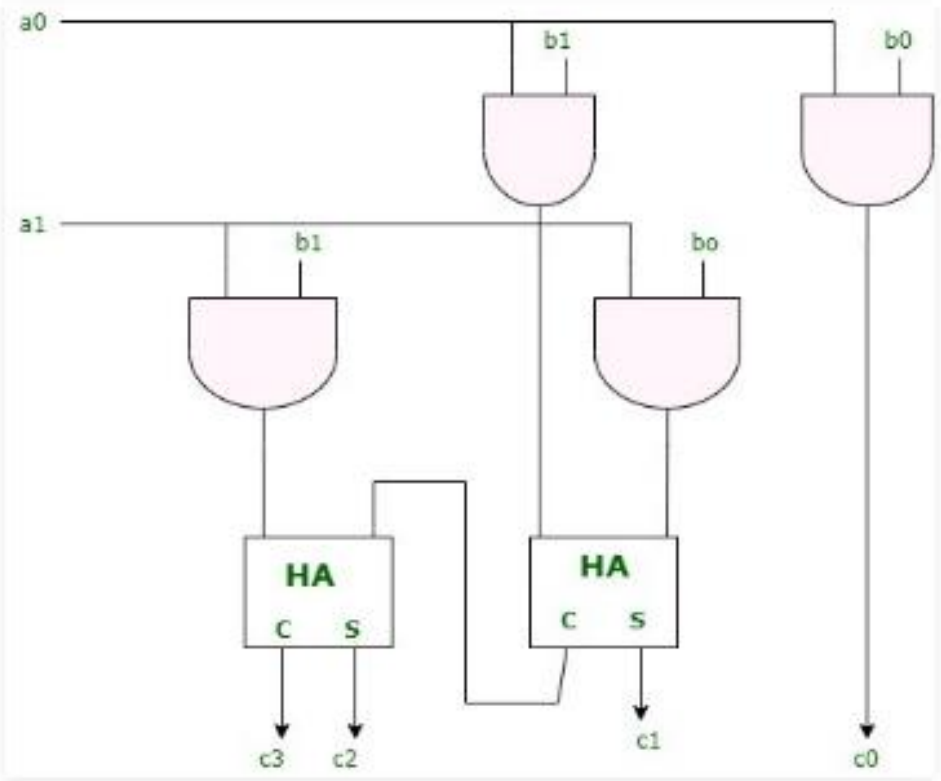
- For implementation of array multiplier with a combinational circuit, consider the multiplication of two 2-bit numbers as shown in figure.
- The multiplicand bits are b1 and b0, the multiplier bits are a1 and a0, and the product is
- **C3C2C1C0**



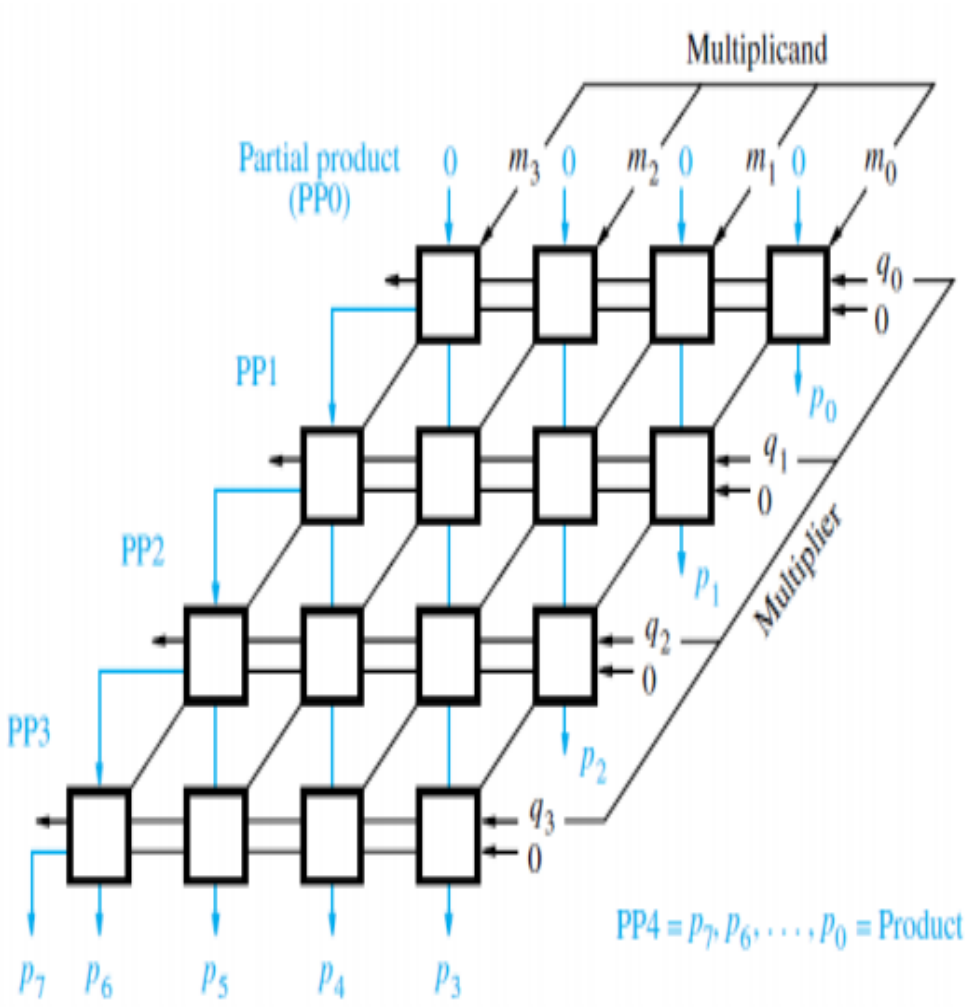
- Assuming A = a1a0 and B= b1b0, the various bits of the final product term P can be written as:-



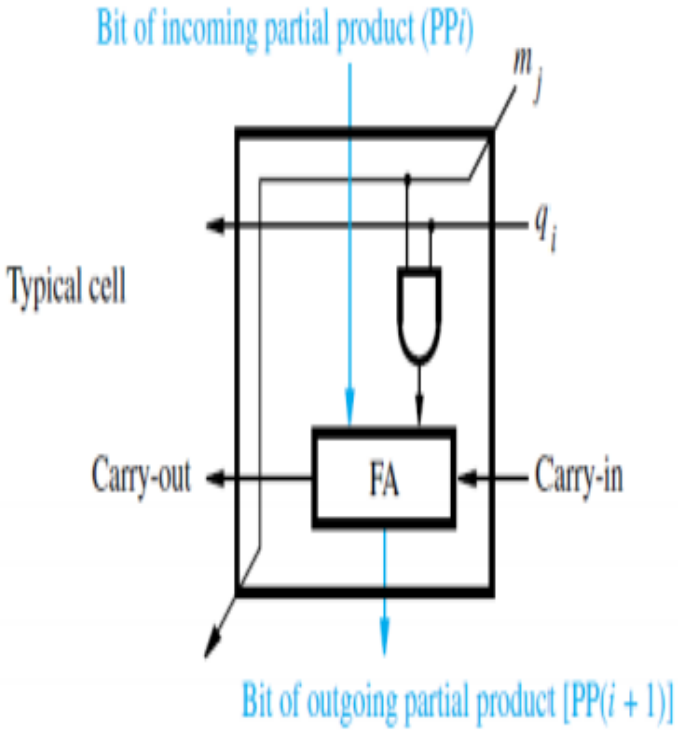
1. P(0)= a0b0
2. P(1)=a1b0 + b1a0
3. P(2) = a1b1 + c1 where c1 is the carry generated during the addition for the P(1) term.
4. P(3) = c2 where c2 is the carry generated during the addition for the P(2) term.



For j multiplier bits and k multiplicand we need j*k AND gates and (j-1) k-bit adders to produce a product of j+k bits.



(a)Array multiplication of positive binary operands



(b) Multiplier cell

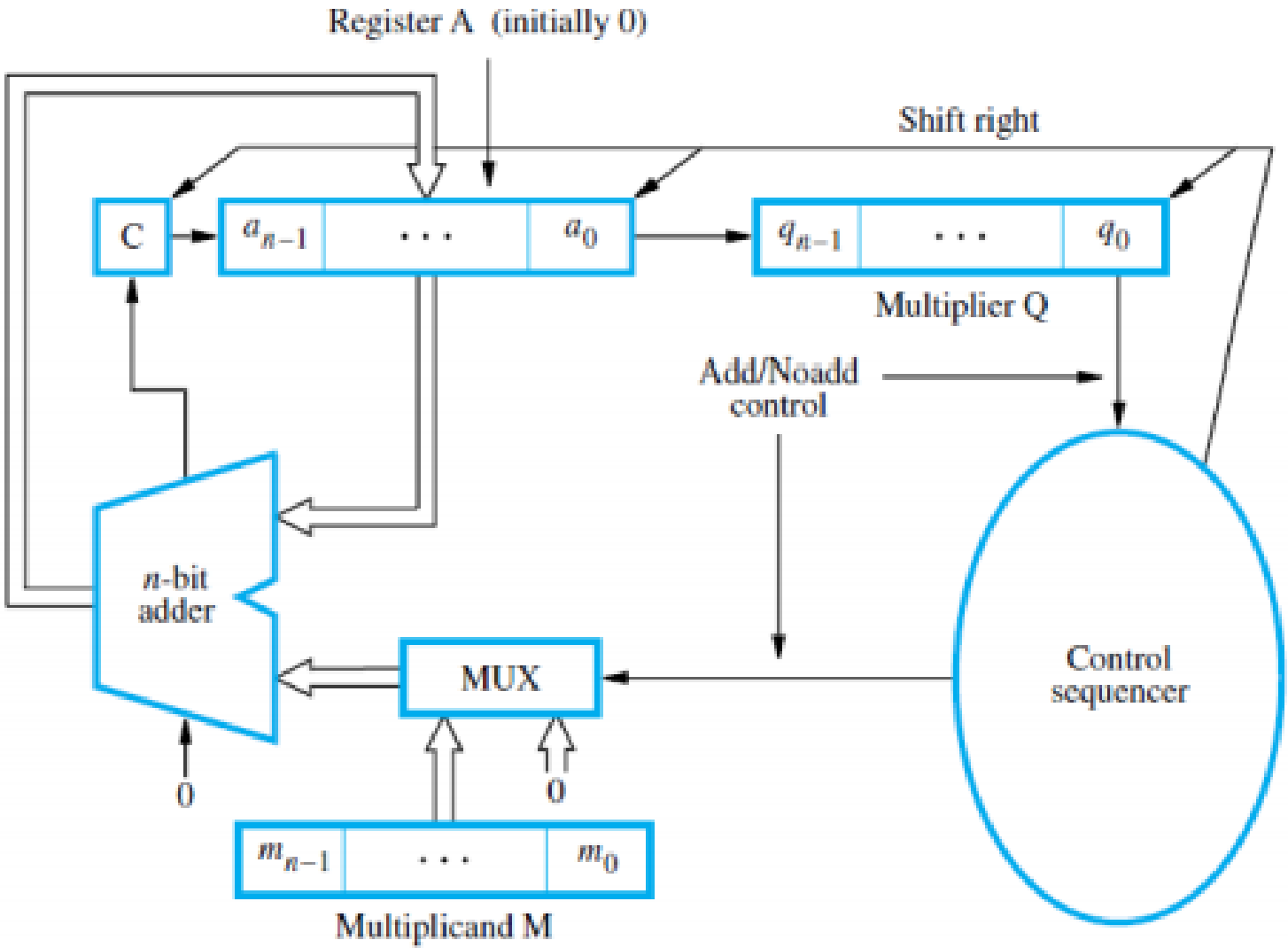
- **Disadvantages:**

(1) An n bit by n bit array multiplier requires n2 AND gates and n(n-2) full adders and n half adders.(Half aders are used if there are 2 inputs and full adder used if there are 3 inputs).

(2) The longest part of input to output through n adders in top row, n -1 adders in the bottom row and n-3 adders in middle row. The longest in a circuit is called critical path.

Sequential Circuit Multiplier

- Multiplication is performed as a series of (n) conditional addition and shift operation such that if the given bit of the multiplier is 0 then only a shift operation is performed, while if the given bit of the multiplier is 1 then addition of the partial products and a shift operation are performed.
- The combinational array multiplier uses a large number of logic gates for multiplying numbers.
- Multiplication of two n-bit numbers can also be performed in a sequential circuit that uses a single n bit adder.



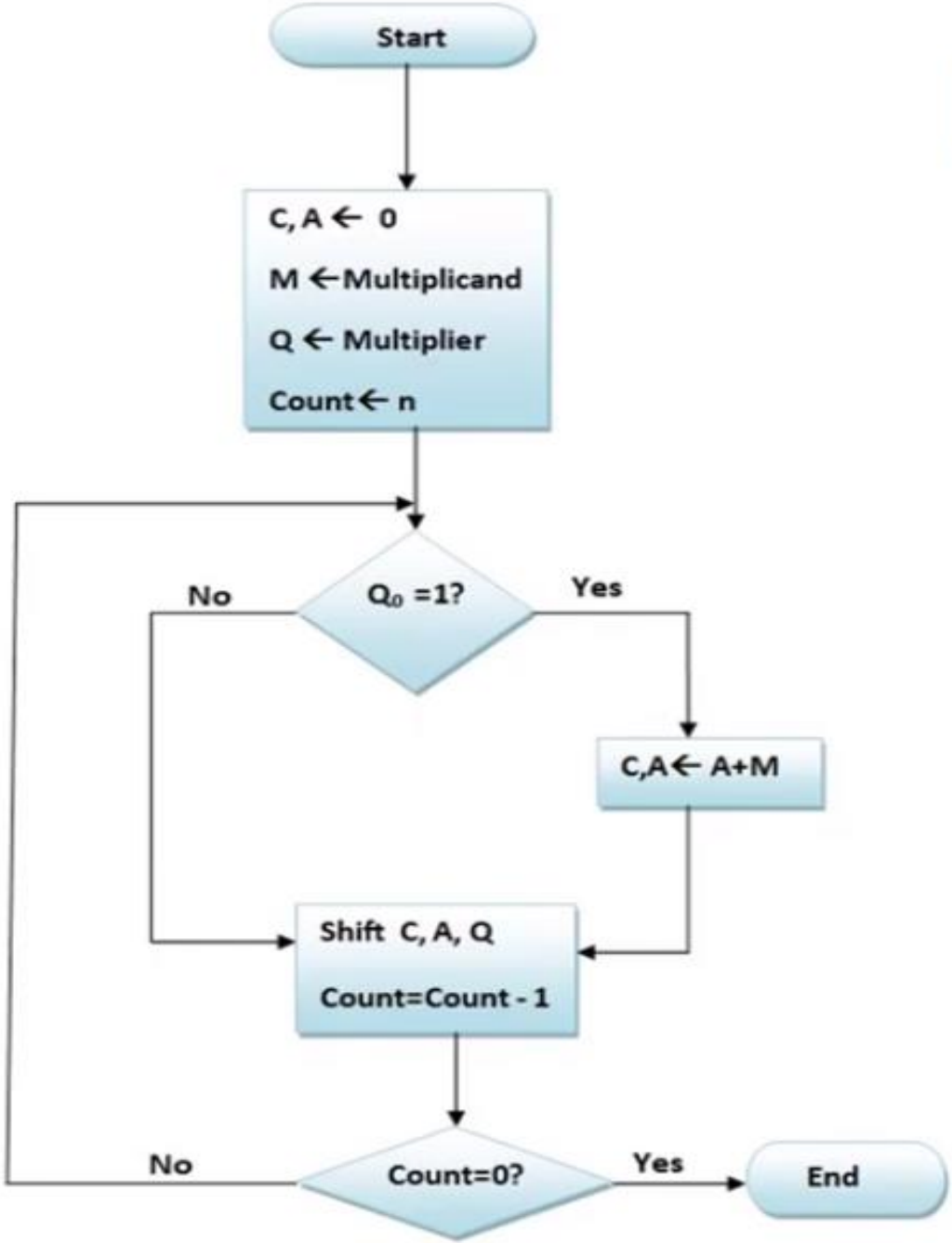
(a) Register configuration

Multiplication

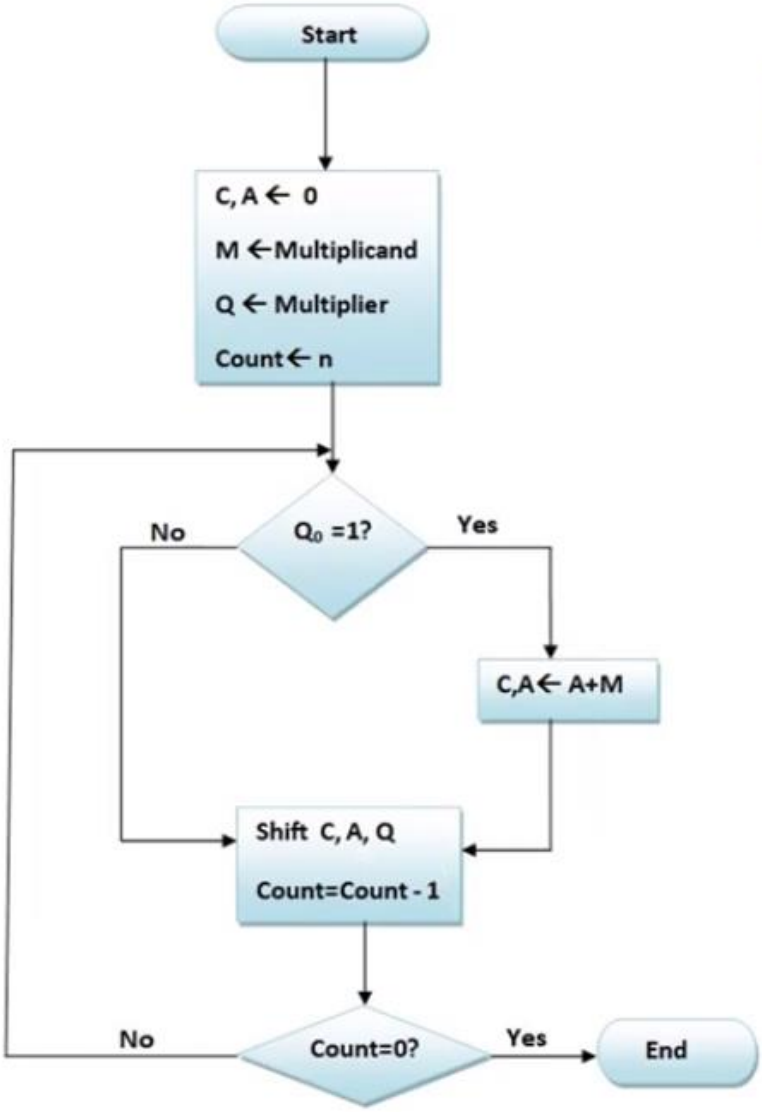
- Find partial product for each digit of multiplier
- Add partial products

```
    1 0 1 1  X  [multiplicand]
    1 1 0 1     [Multiplier]
    1 0 1 1
  0 0 0 0
1 0 1 1
1 0 1 1

-----------
1 0 0 0 1 1 1 1
```

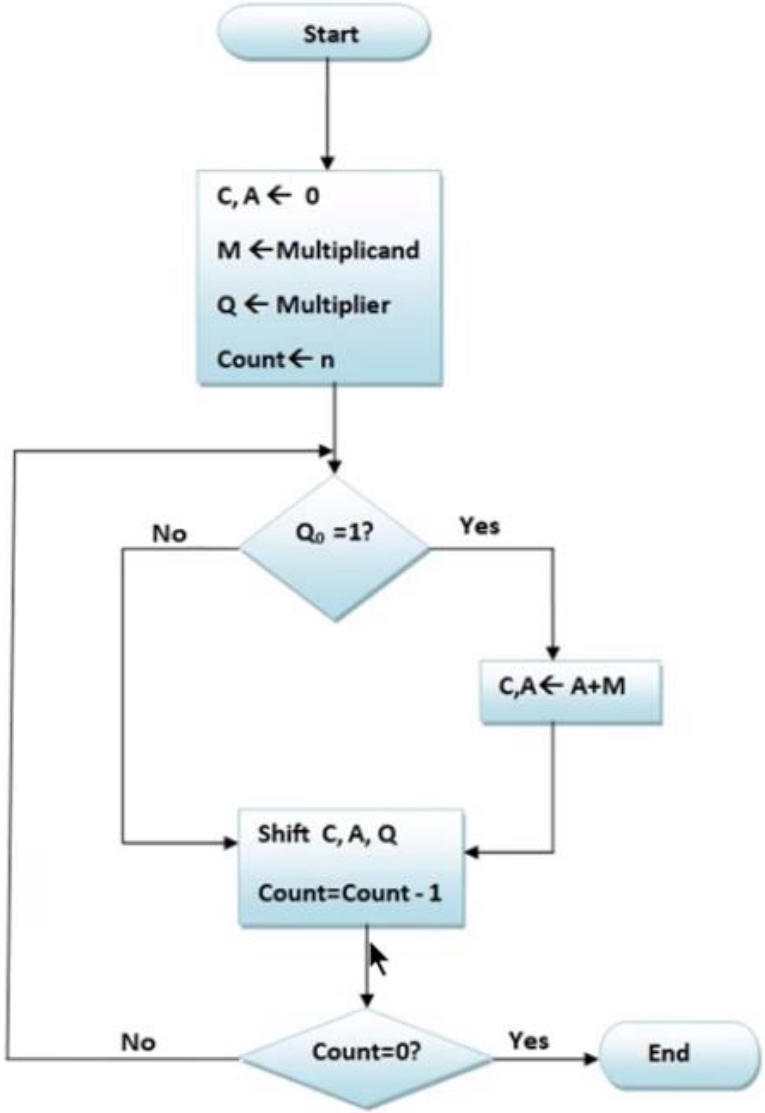

Start
C, A ← 0
M ← Multiplicand
Q ← Multiplier
Count ← n
Q_0 =1?
No
Yes
C,A ← A+M
Shift C, A, Q
Count=Count - 1
Count=0?
No
Yes
End

Algorithm for Multiplication



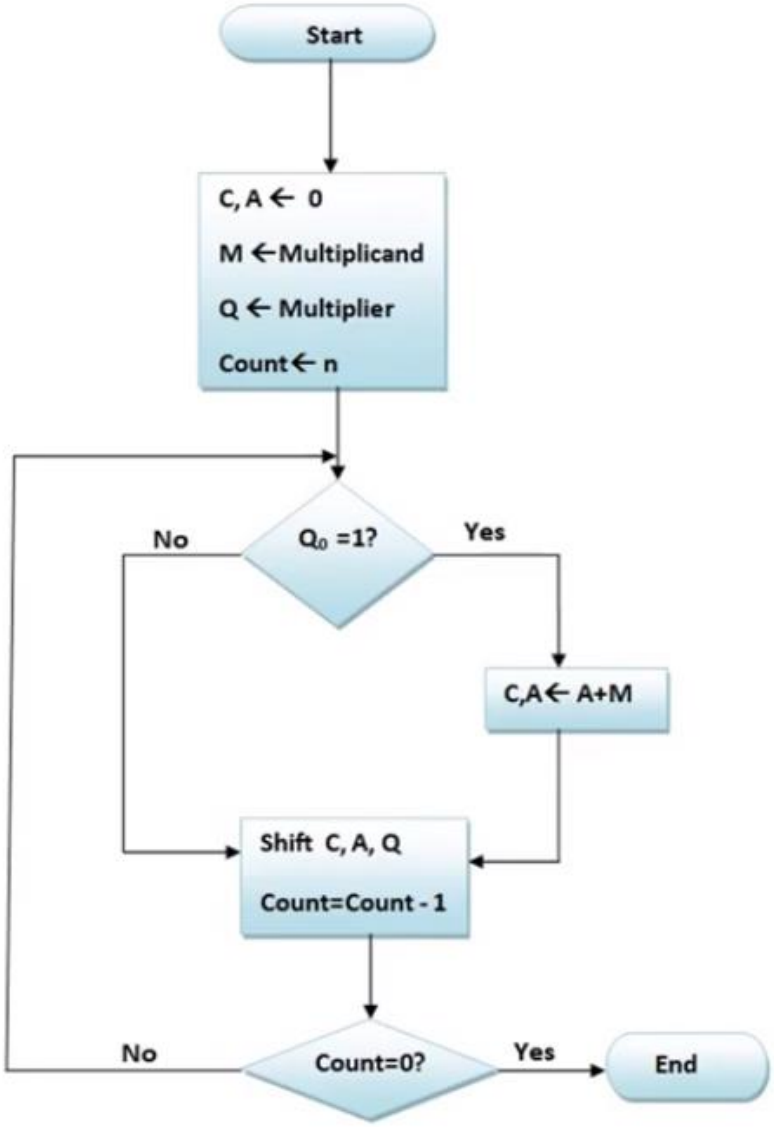
C	A	Q	M	Steps
0	0000	1101	1011	Initialize
0	1011	1101	1011	Add
0	0101	1110	1011	Right shift
0	0010	1111	1011	Right shift
0	1101	1111	1011	Add
0	0110	1111	1011	Right shift
1	0001	1111	1011	Add
0	1000	1111	1011	Right shift

Algorithm for Multiplication



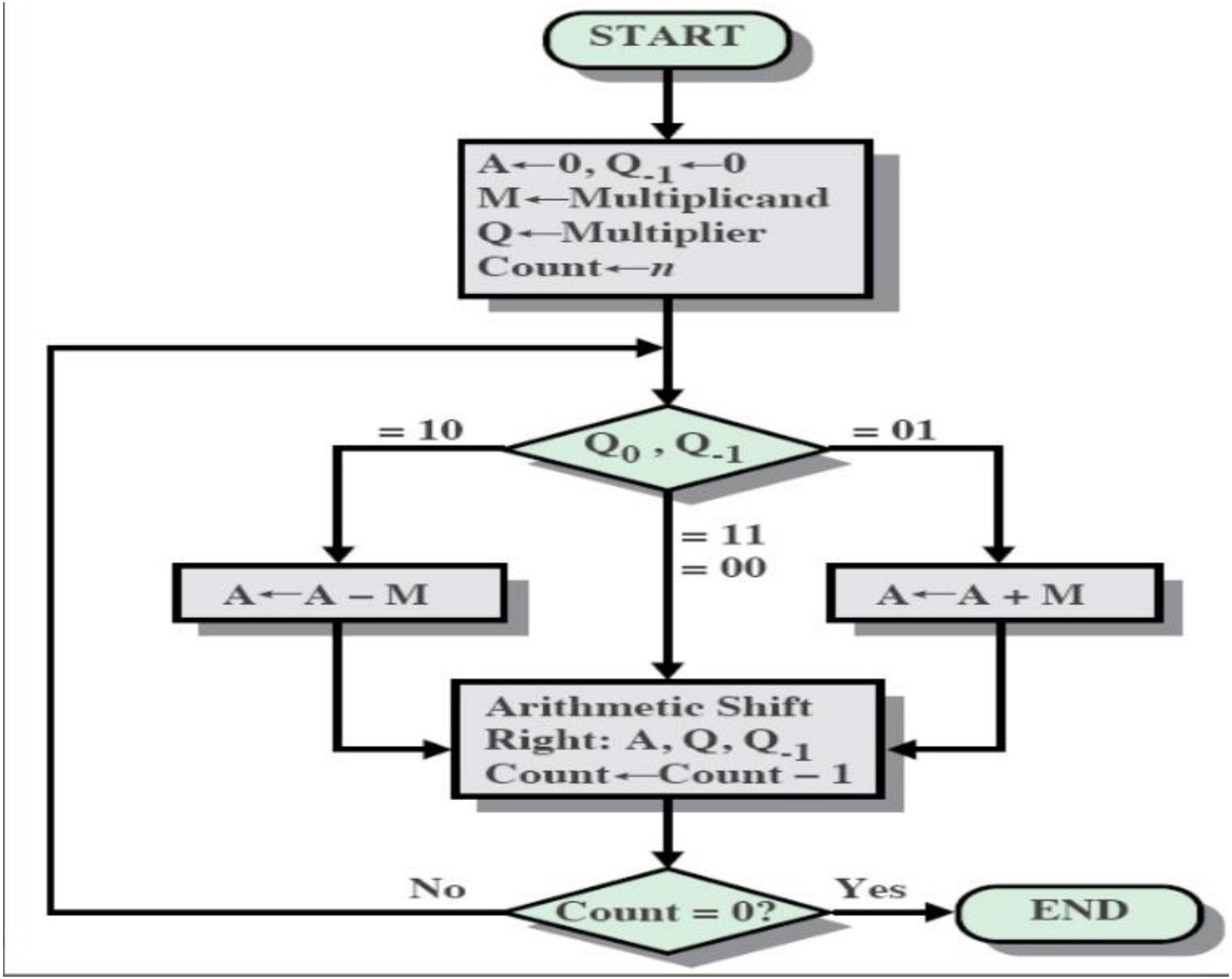
C	A	Q	M	Steps
0	0000	1101	1011	Initialize
0	1011	1101	1011	Add
0	0101	1110	1011	Right shift
0	0010	1111	1011	Right shift
0	1101	1111	1011	Add
0	0110	1111	1011	Right shift
1	0001	1111	1011	Add
0	1000	1111	1011	Right shift

Algorithm for Multiplication



C	A	Q	M	Steps
0	0000	1101	1011	Initialize
0 0	1011 0101	1101 1110	1011 1011	Add Right shift
0	0010	1111	1011	Right shift
0 0	1101 0110	1111 1111	1011 1011	Add Right shift
1 0	0001 1000	1111 1111	1011 1011	Add Right shift

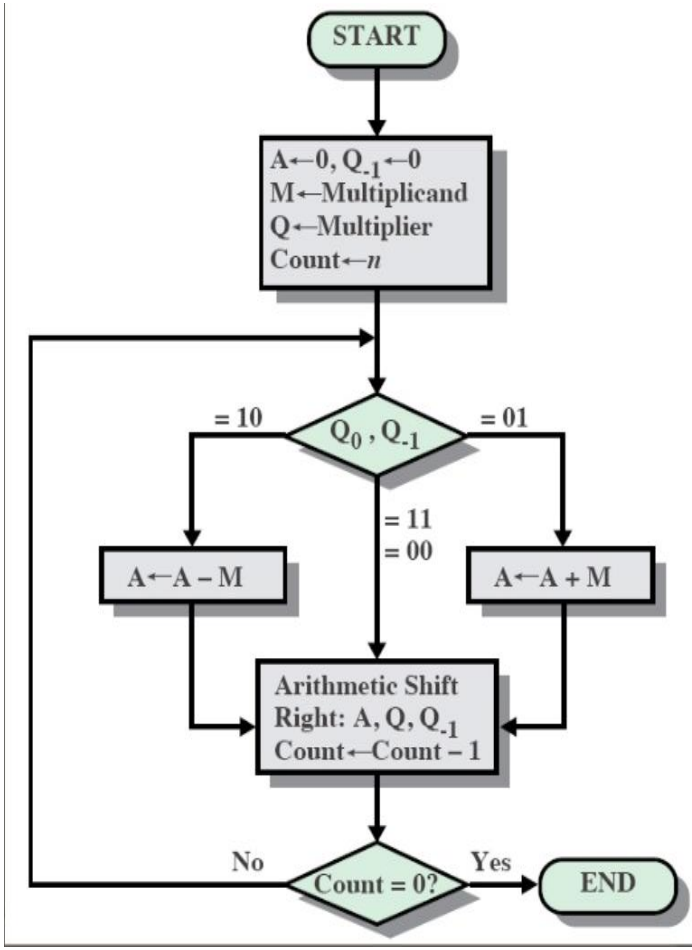
BOOTH'S ALGORITHM


START
A←0, Q-1←0
M←Multiplicand
Q←Multiplier
Count←n
Q0 , Q-1
= 10
= 01
= 11
= 00
A←A – M
A←A + M
Arithmetic Shift
Right: A, Q, Q-1
Count←Count – 1
No
Yes
Count = 0?
END

- Possible arithmetic actions:
  - `00` → no arithmetic operation
  - `01` → add multiplicand to left half of product
  - `10` → subtract multiplicand from left half of product
  - `11` → no arithmetic operation

STEPS	A	Q	Q-1	OPERATION	
0	0000	0011	0		
1	1001 1100	0011 1001	0 1	10->A=A-M A=A+M’+1 ASHR	M=0111 M’=1000+1=1001
2	1110	0100	1	11-> SHIFT	
3	1110+ 0111 -------- 0101 0010	0100 0100 1010	1 1 0	01->A=A+M ASHR	
4	0010 0001	1010 0101	0 0	00->SHIFT	

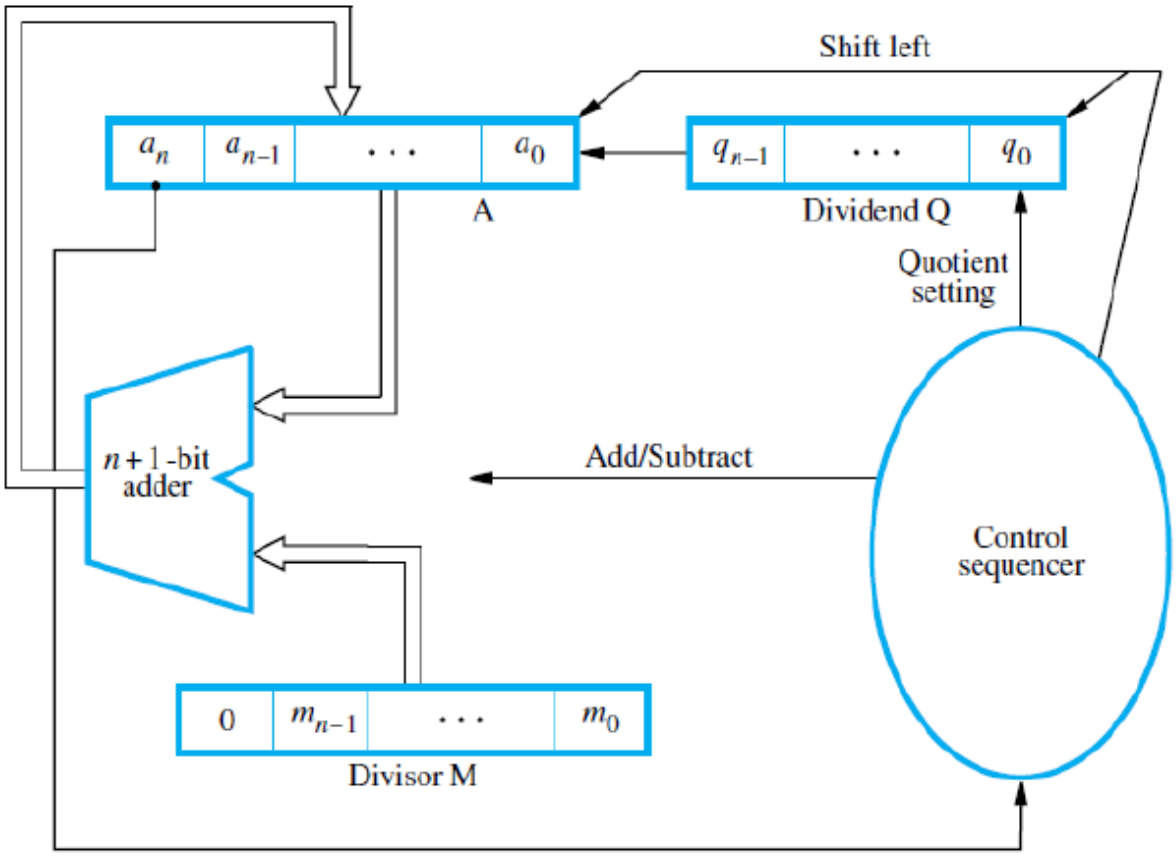
MULTIPLICAND -7 : 0111(M)
MULTIPLIER – 3 : 0011(Q)



Booth : (7) x (-3)

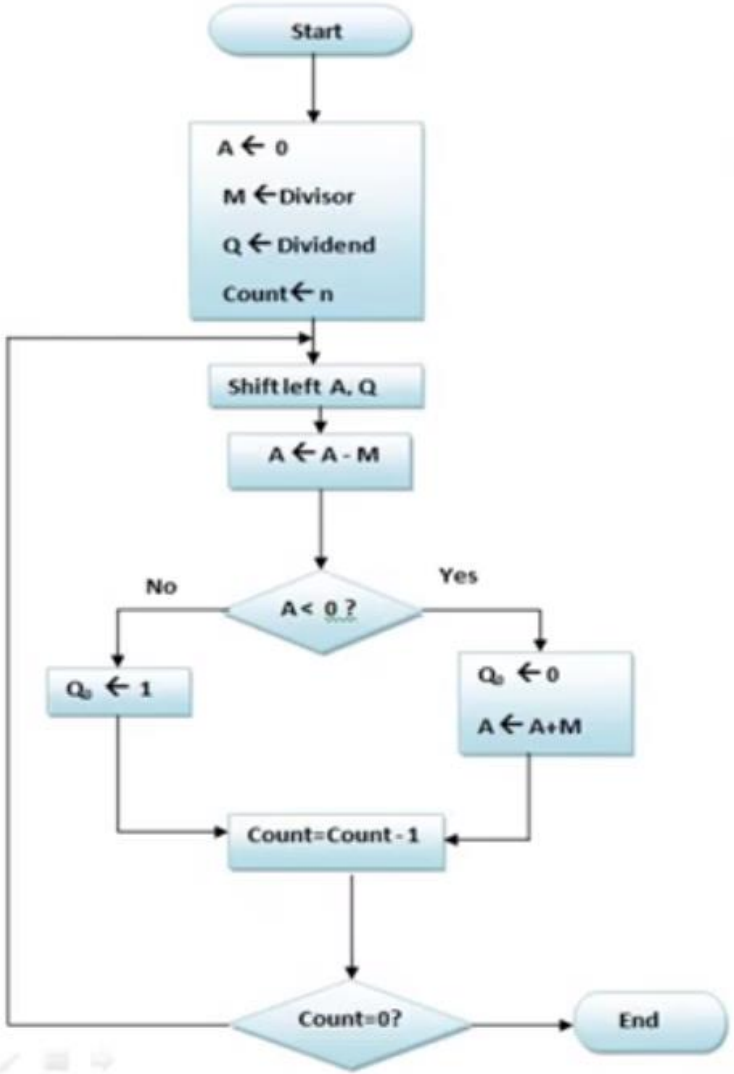
A	Q	Q-1 (-3)	M 7	
0000	1101	0	0111	
1001	1101	0	0111	A <- (A - M) 1st cycle
1100	1110	1	0111	Shift
0011	1110	1	0111	A <- (A + M) 2nd cycle
0001	1111	0	0111	Shift
1010	1111	0	0111	A <- (A - M) 3rd cycle
1101	0111	1	0111	Shift
1110	1011	1	0111	Shift

BINARY DIVISION

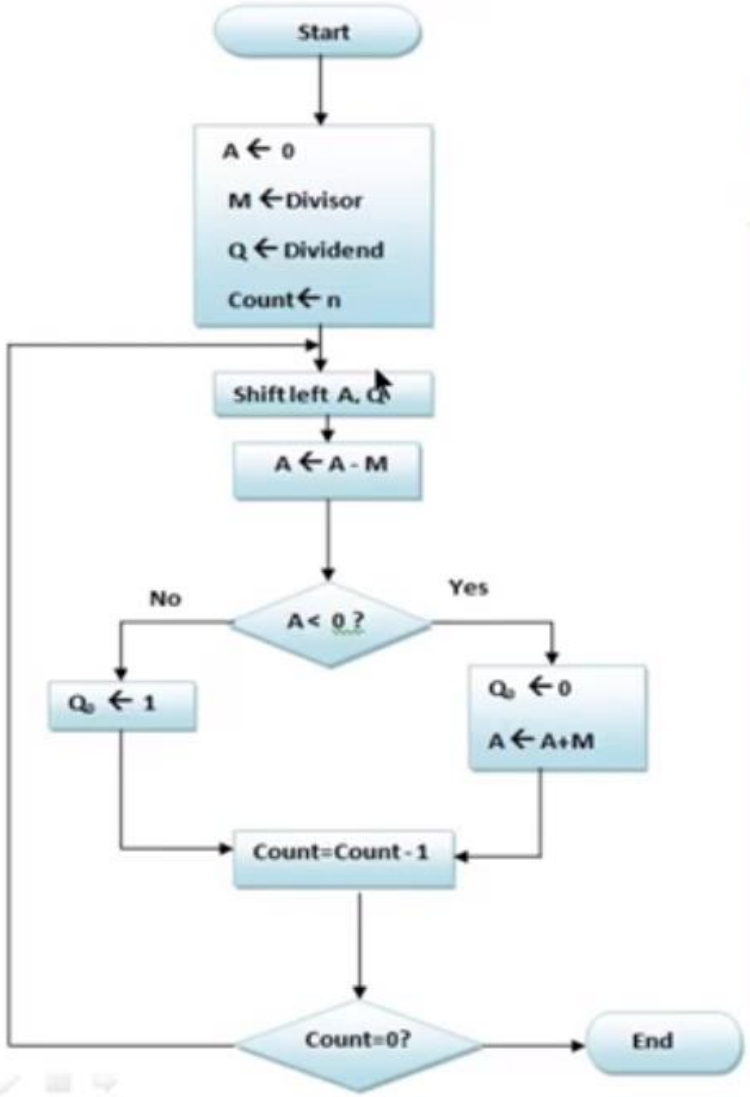


Circuit arrangement for binary division.

BINARY DIVISION(RESTORING METHOD)

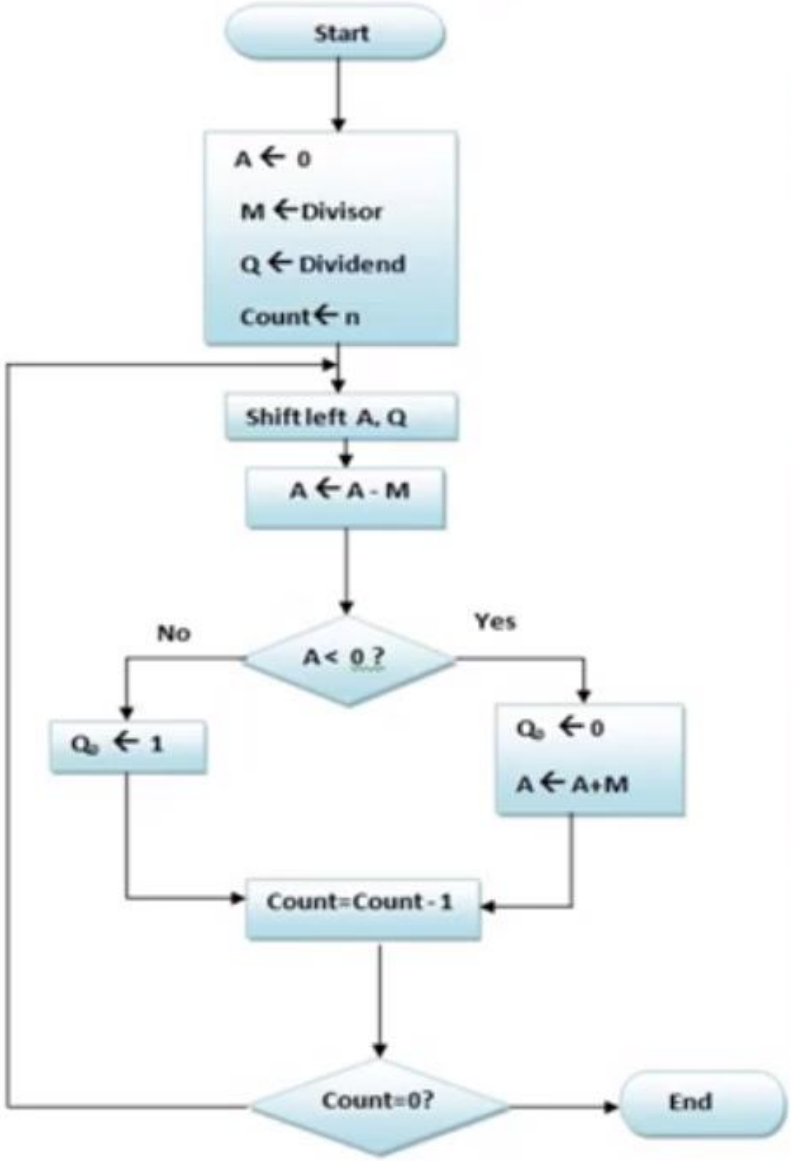


DIVIDEND : 7 0111
DIVISOR : 3 0011



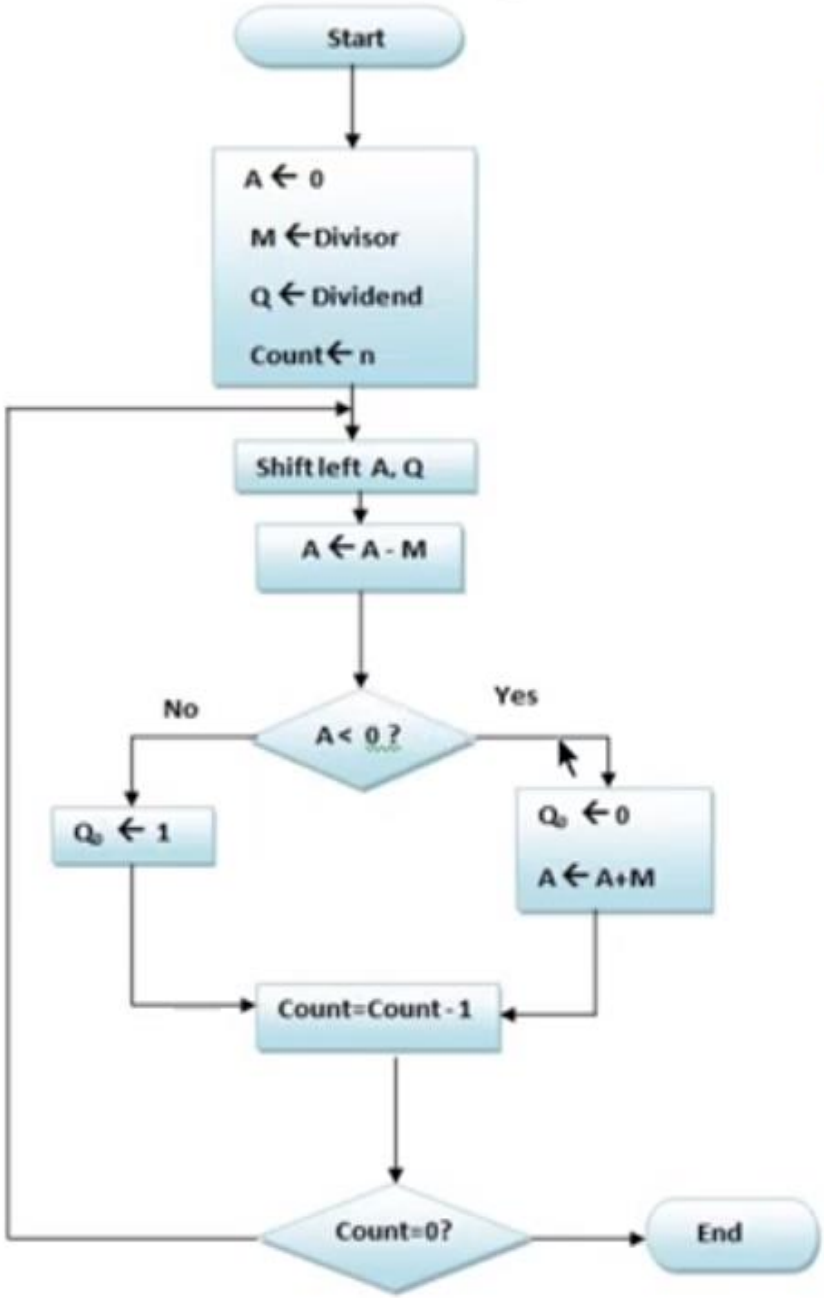
Count	A	Q	M	Operation
4	0000	0111	0011	Intialization

Algorithm for Division



Count	A	Q	M	Operation
4	0000	0111	0011	Intialization
	0000	1110	0011	Left Shift
	1101	1110	0011	A ← A- M
3	0000	1110	0011	Restore (A←A +M) and Q_0 ← 0

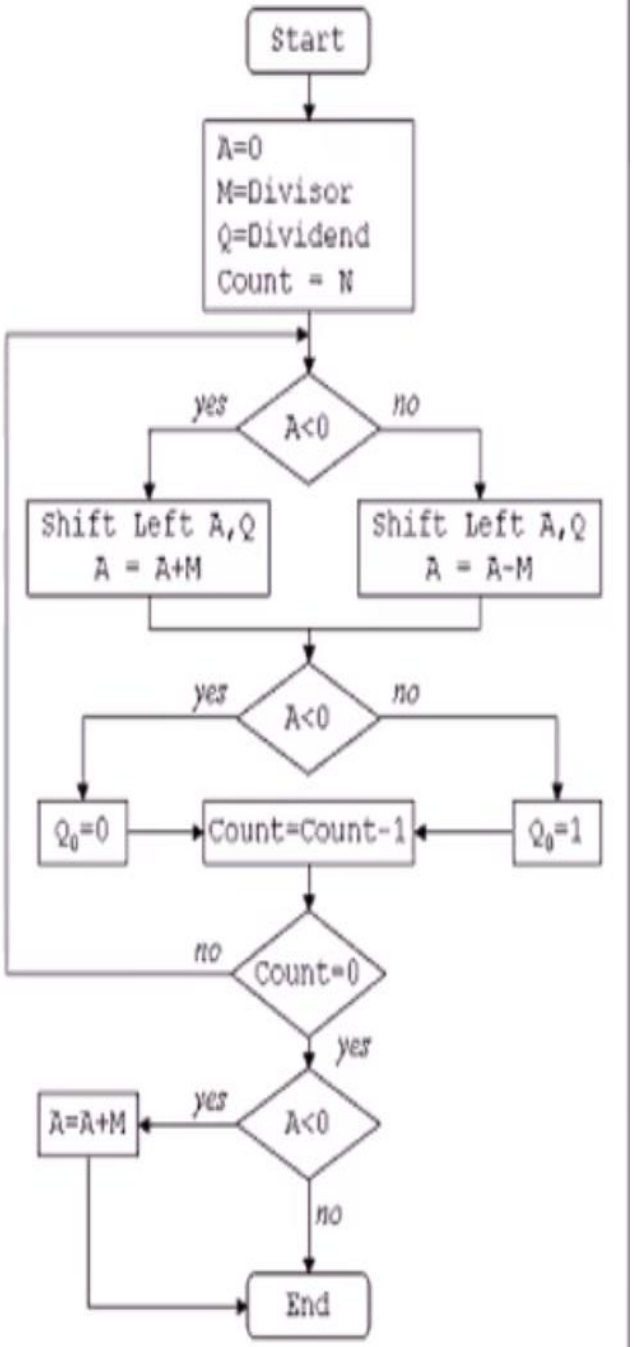
Algorithm for Division



Count	A	Q	M	Operation
4	0000	0111	0011	Intialization
3	0000 1101 0000	1110 1110 1110	0011 0011 0011	Left Shift A ← A- M Restore (A←A +M) and $Q_0 \leftarrow 0$
2	0001 1110 0001	1100 1100 1100	0011 0011 0011	Left shift A ← A-M Restore (A←A +M) and $Q_0 \leftarrow 0$
1	0011 0000 0000	1000 1000 1001	0011 0011 0011	Left Shift A ← A-M Set $Q_0 \leftarrow 1$
0	0001 1110 0001 Remainder	0010 0010 0010 Quotient	0011 0011 0011	Left Shift A ← A- M Restore (A←A +M) and $Q_0 \leftarrow 0$

Non-Restoring Division Algorithm

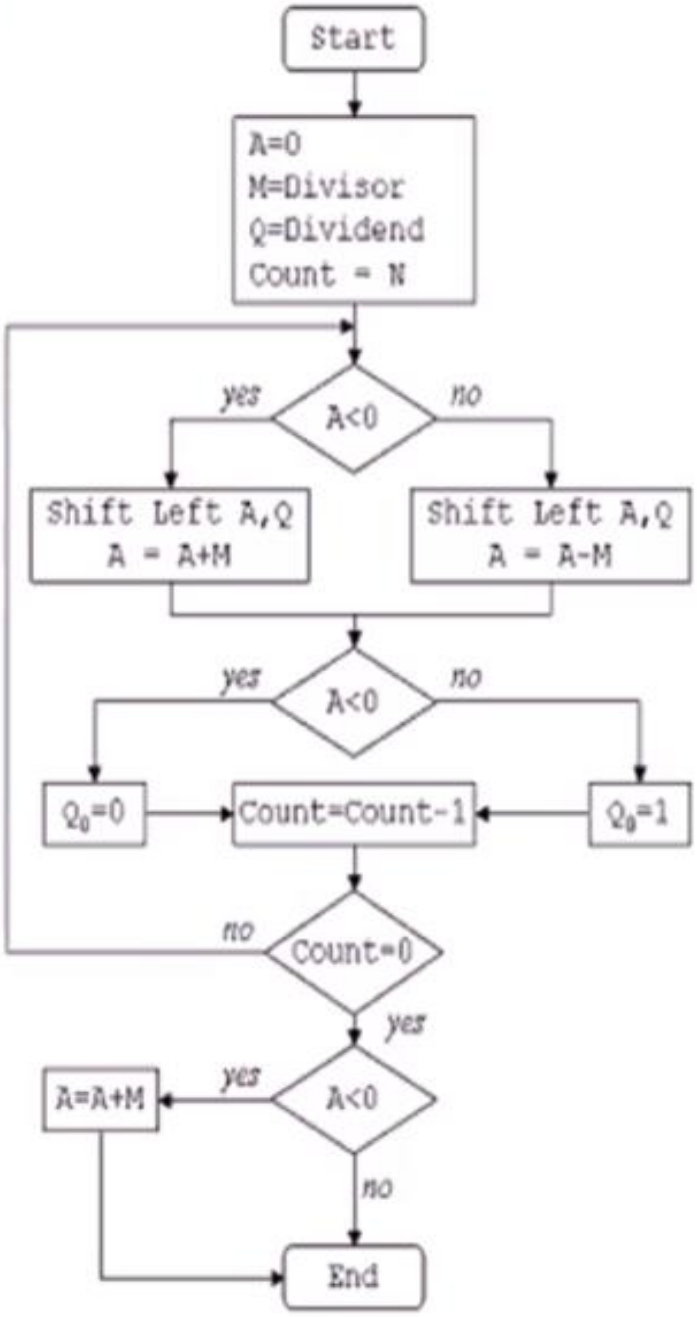
- Input:
  - M – positive divisor (n-bit)
  - Q – positive dividend (n-bit)
- Output:
  - Q – Quotient
  - A – Remainder
- Begin
  - $A \leftarrow 0$
  - Do n times
    - If the sign of A is 0
      - Shift A and Q left one bit position and $A \leftarrow A - M$
    - else
      - Shift A and Q left one bit position and $A \leftarrow A + M$
    - If Sign of A is 0
      - $q_0 \leftarrow 1$
    - Else
      - $q_0 \leftarrow 0$
  - If sign of A is 1
    - $A \leftarrow A + M$
- End



EG: DIVIDEND -Q : 7
DIVISOR –M : 3

$M = 0011$, $\overline{M} + 1 = 1101$

Comment	A	Q	SC
	0000	0111	4
Shl	0000	1110	
A = A - M	1101		
	1101		3
Shl	1011	1100	
A = A + M	0011		
	1110		2
Shl	1101	1000	
A = A + M	0011		
	0000	1001	1



Comment	A	Q	SC
Shl	0001	0010	
A = A - M	1101		
	1110	0010	0
A = A + M	0011		
	0001