# Computer Organization
(Adder/Subtractor, multiplication/div)

KR Chowdhary
Professor & Head
*Email: kr.chowdhary@gmail.com*
*webpage: krchowdhary.com*

Department of Computer Science and Engineering
MBM Engineering College, Jodhpur

November 14, 2013

# Complement of Numbers

```
  2345
 -0789
  ----
  1556
  -----
```

```
Instead we use:              2345
9's complement of 0789 =+9210 (this is:(r-1)th complement,
                       -------  r for radix)
                      1 1555
                          +1 (the overflow 1
                      -------  is added here)
                        1556
```

```
This can be done in r's complement as:
     2345
     9211 (this is 10's complement of 0789)
   -------
   1 1556, extra 1 is dropped.
```

In binary, r's (radix, i.e. 10) complement is 2's complement and 1's complement is (r-1)'s complement.

# Hardware for adder/ subtractor

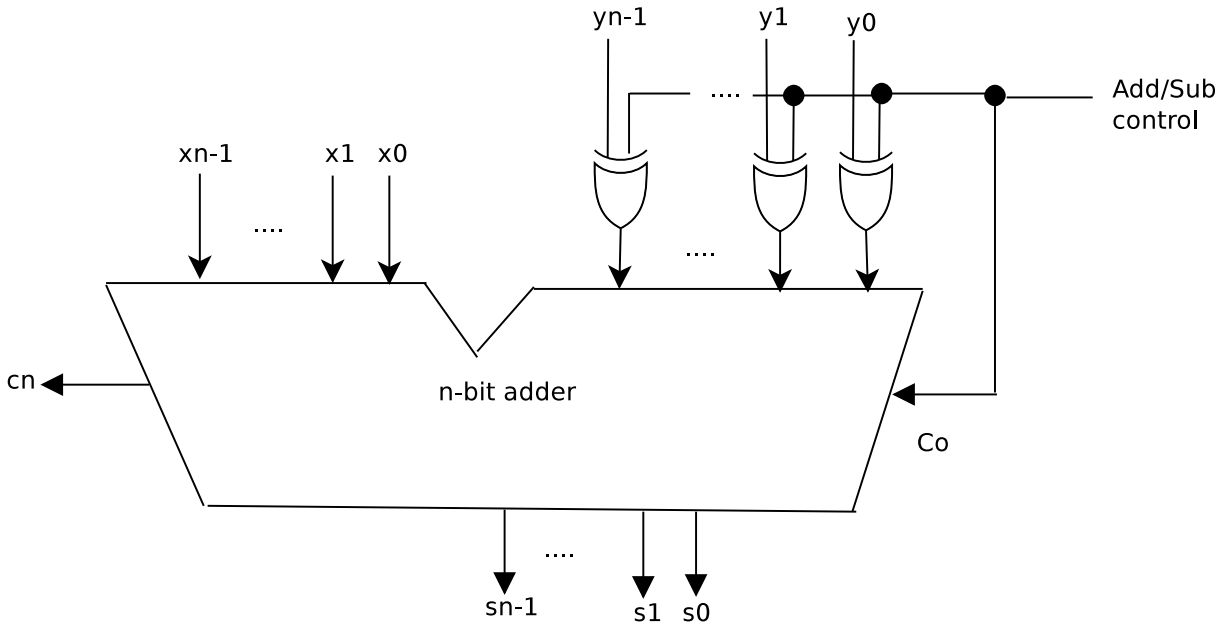


Figure 1: Adder/Subtractor

- Add/subtract line as 0/1 sends $Y$ bits as they were or after complementing as 2s complement.
- $c_0$ true is for 2's complement.
- For addition, $S \leftarrow X + Y$, for subtraction, $S \leftarrow X - Y$.
- Signed numbers' over flow (carry) $C = x_{n-1}y_{n-1}\bar{s}_{n-1} + \bar{x}_{n-1}\bar{y}_{n-1}s_{n-1}$
- $n$ number of FAs for ripple carry adder, or high-speed adder circuit.

# Integer Multiplication

```
                1 0 1 1  (multiplicand (11))
              x 1 1 0 1   (multiplier (13)
 ---------------------------
                1 0 1 1
              0 0 0 0
            1 0 1 1                 partial products
          1 0 1 1
 ----------------------------
        1 0 0 0 1 1 1 1  (product  = 143)
----------------------------
```

After every partial product is added into progressive sum, the result is shifted to right. So that next addition will take place one digit left of previous.

# Multiplication of positive numbers

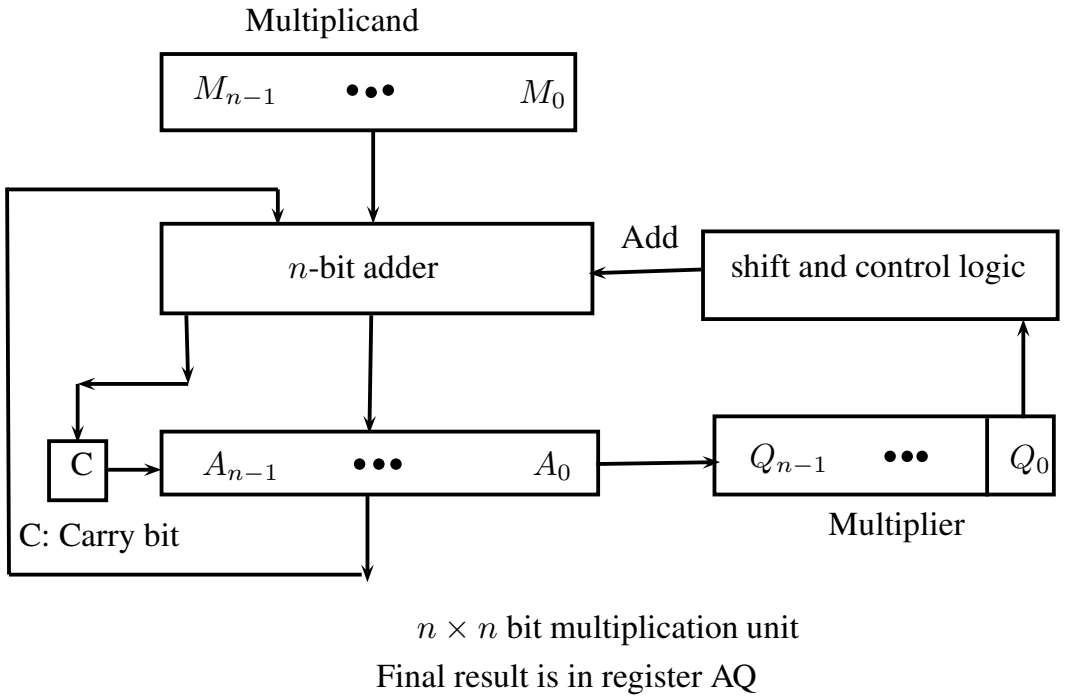


Figure 2: $n \times n$ bit multiplication

- Thus multiplication is relatively complex operation.
- For two operands $|M| = n$ for multiplicand and $|Q| = n$ for multiplier, result size $= 2n$.

# Multiplication flow-chart

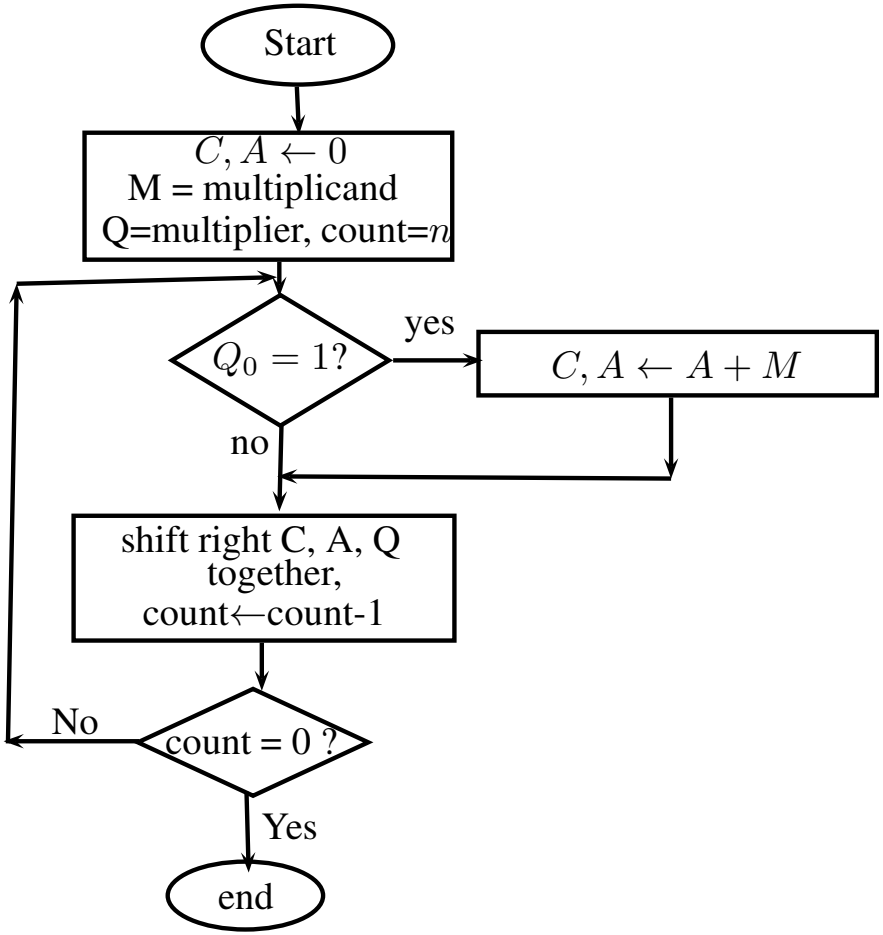


Figure 3: nxn bit multiplication.

# Integer Division

```
            0 0 0 0 1 1 0 1 =Quotient
          -------------------------
   1011 )  1 0 0 1 0 0 1 1 = Dividend
Divisor       1 0 1 1
              -----------
              0 0 1 1 1 0
                  1 0 1 1
              -----------
                  0 0 1 1 1 1
                      1 0 1 1
              ---------------
                      0 1 0 0 =remainder
```

When divisor is subtracted from resultant dividend, result is shifted to right.

Initial: Register $Q$ holds Dividend, $M$ holds Divisor

Final: Register $A$ holds remainder, $Q$ holds Quotient.

# Division flow-chart

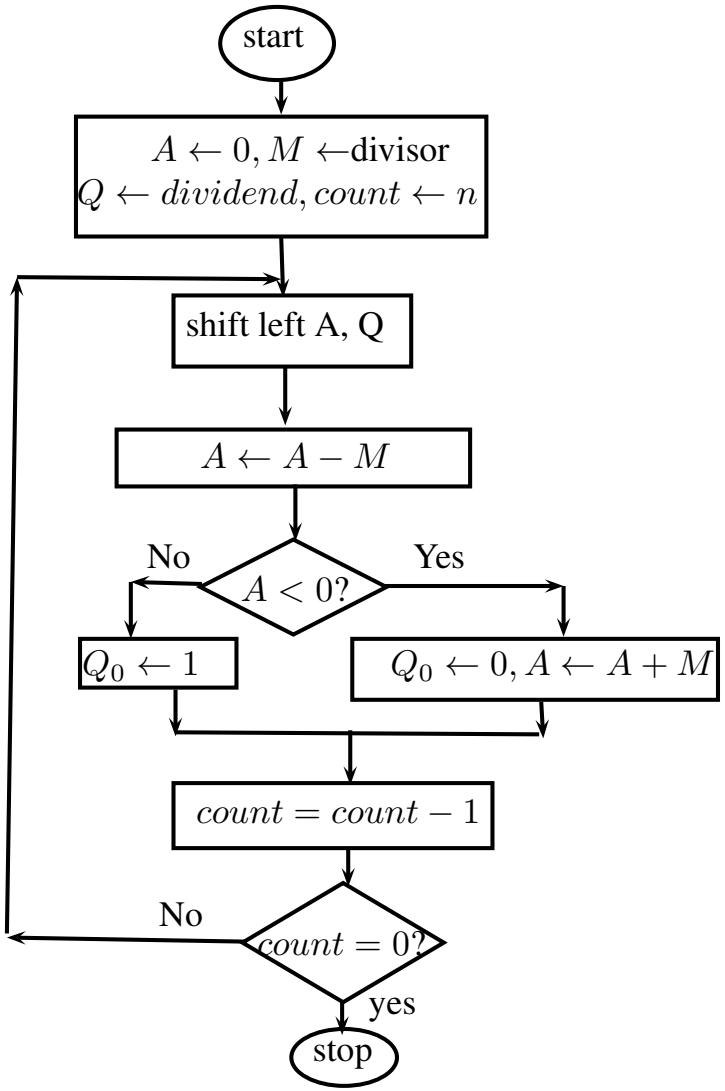


Figure 4: n by n-bit Division

# Division

For two operands $|M| = n$ for divisor and $|Q| = n$ for dividend, result size $= n$.

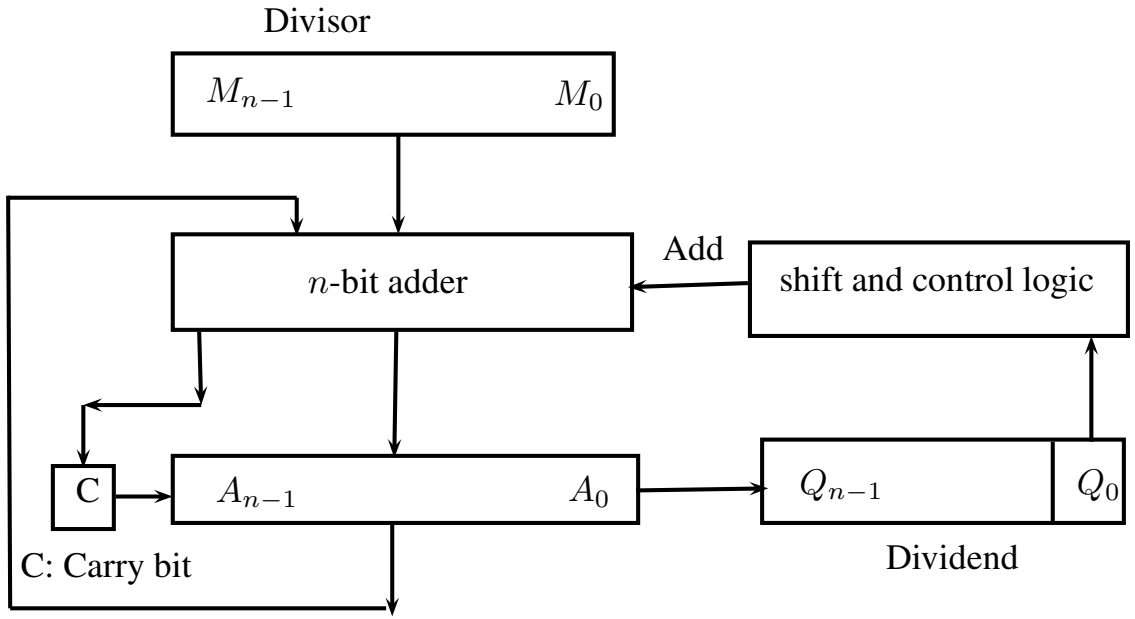


Figure 5: $n \div n$ bit division

- When divisor is subtracted from resultant dividend, result is shifted to right.
- Initial: Register $Q$ holds Dividend, $M$ holds Divisor
- Final: Register $A$ holds remainder, $Q$ holds Quotient.

# Points to ponder

1. Give a logical justification for $C = x_{n-1}y_{n-1}\bar{s}_{n-1} + \bar{x}_{n-1}\bar{y}_{n-1}s_{n-1}$ in slide no. 3.
2. What modifications are required in slides 5,6 to perform signed multiplication?
3. What modifications are required in slides 8,9 to perform signed division?
4. Why it is not possible to provide exact expression for number of bit-level operations, as a function of input size $n$, for the division algorithm discussed in this slide?