

Division Algorithm for Signed Magnitude Data

Division Algorithm for Signed Magnitude Data

- ☐ The **Division** of two fixed-point binary numbers in the signed-magnitude representation is done by the **cycle of successive compare, shift, and subtract operations**
- ☐ The **binary division** is **easier** than the **decimal division** because the **quotient digit** is **either 0 or 1**
- ☐ There is no need to estimate how many times the dividend or partial remainders adjust to the divisor

Division Algorithm for Signed Magnitude Data

- ✓ ☐ The **Division** of two fixed-point binary numbers in the signed-magnitude representation is done by the **cycle of successive compare, shift, and subtract operations**
- ✓ ☐ The **binary division** is **easier** than the **decimal division** because the **quotient digit** is **either 0 or 1**
- ✓ ☐ There is no need to estimate how many times the dividend or partial remainders adjust to the divisor

$$\begin{array}{r}
 01101 \rightarrow Q \\
 10001 \overline{) 01110} \\
 \underline{-0000} \\
 11100 \\
 \underline{-10001} \\
 10110 \\
 \underline{-10001} \\
 10100 \\
 \underline{-10001} \\
 110 \leftarrow \underline{\underline{R.}}
 \end{array}$$

Dividend (A): 01110

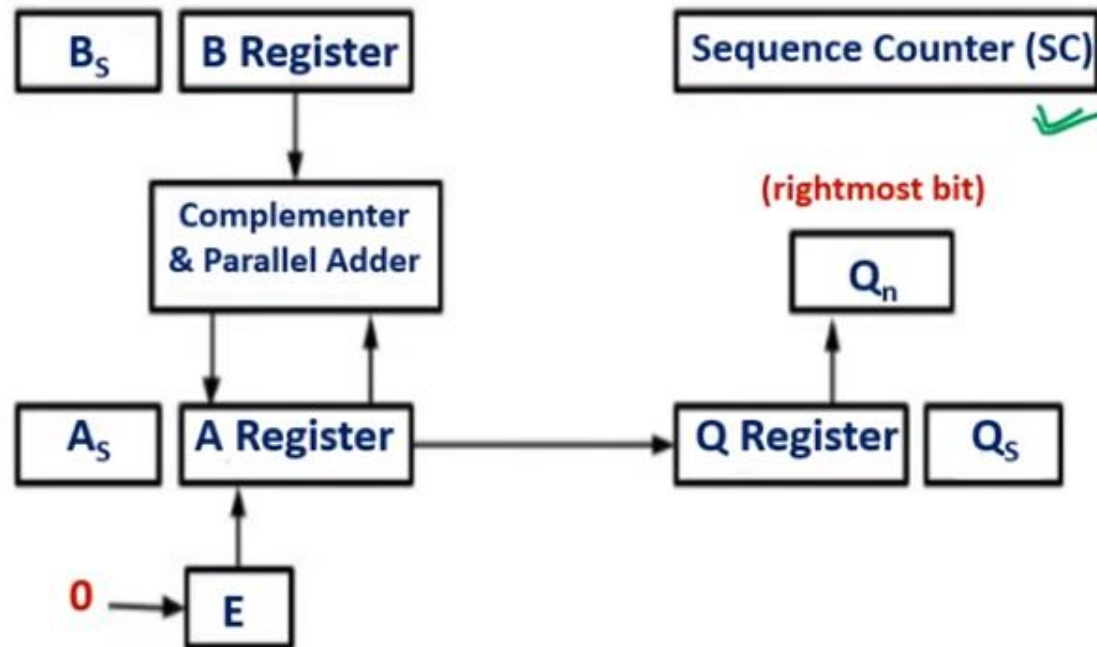
Divisor (B): 10001

✓ Quotient:?

✓ Remainder:?

$$A < B, \quad R = 01$$

Division Algorithm for Signed Magnitude Data



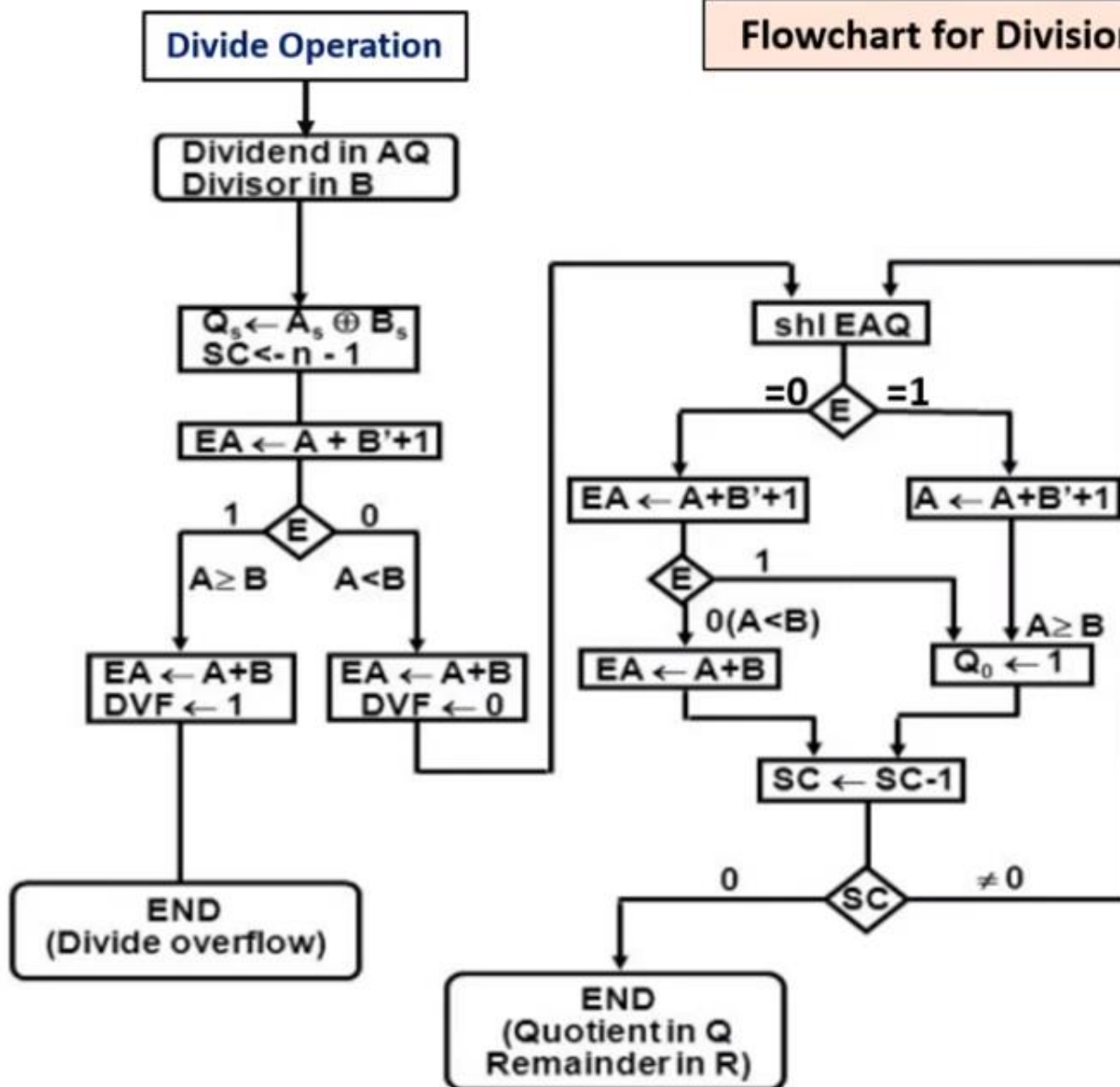
Hardware Implementation for Divide Operation

- The hardware implementation in the division operation is **identical** to that required for multiplication
- Here, **Registers B** is used to **store divisor** and the **double-length dividend** is stored in **registers A and Q**
- The information for the relative magnitude is given in **E**
- Sequence Counter register (**SC**) is used to store the number of bits in the **dividend**

E A Q
←

2

Division Algorithm for Signed Magnitude Data



- ❑ The **Division** of two fixed-point binary numbers in the signed-magnitude representation is done by the **cycle of successive compare, shift, and subtract operations**
- ❑ The **binary division** is **easier** than the **decimal division** because the **quotient digit** is **either 0 or 1**

Division Algorithm for Signed Magnitude Data

OPERATION	E	A	Q	SC
Initial	0	01110	00000	5
shl EAQ	0	11100	00000	
A + B' + 1	1	01011	00001	4

Question: Compute 01110 / 10001

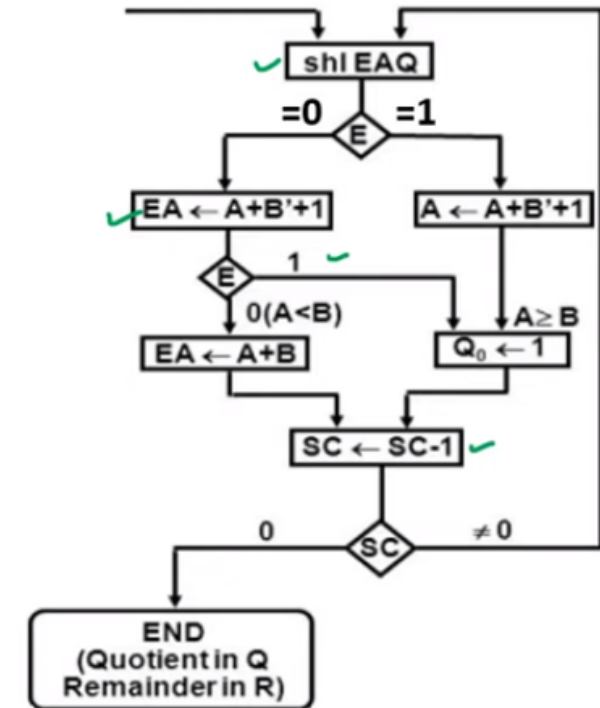
✓ Dividend (A) = 01110

✓ Divisor (B) = 10001

Initially A = 0000, Q = 00000, SC = 5

B' = 01110

✓ B' + 1 = 01111



Division Algorithm for Signed Magnitude Data

OPERATION	E	A	Q	SC
Initial	0	01110	00000	5
shl EAQ	0	11100	00000	
$A + B' + 1$	✓1	01011	00001	4
shl EAQ	0	10110	00010	
$A + \bar{B} + 1$	1	00101	00011	3

Question: Compute 01110 / 10001

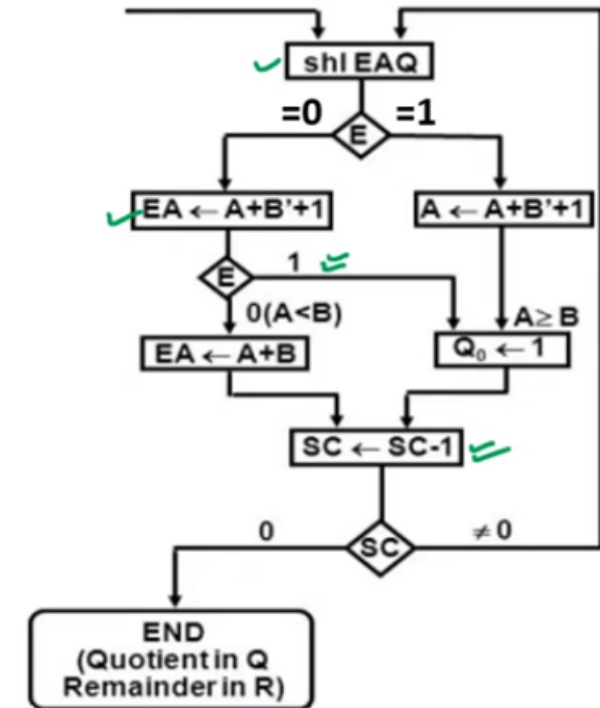
✓ Dividend (A) = 01110

✓ Divisor (B) = 10001

Initially A = 0000, Q = 00000, SC = 5

B' = 01110

✓ B' + 1 = 01111



Division Algorithm for Signed Magnitude Data

OPERATION	E	A	Q	SC
Initial	0	01110	00000	5
shl EAQ	0	11100	00000	
$A + B' + 1$	✓ 1	01011	00001	4
shl	0	10110	00010	
$A + \bar{B} + 1$	1	00101	00011	3
shl	0	01010	00110	
$A + \bar{B} + 1$		11001		
$A + B$	1	01010	00110	2

Question: Compute 01110 / 10001

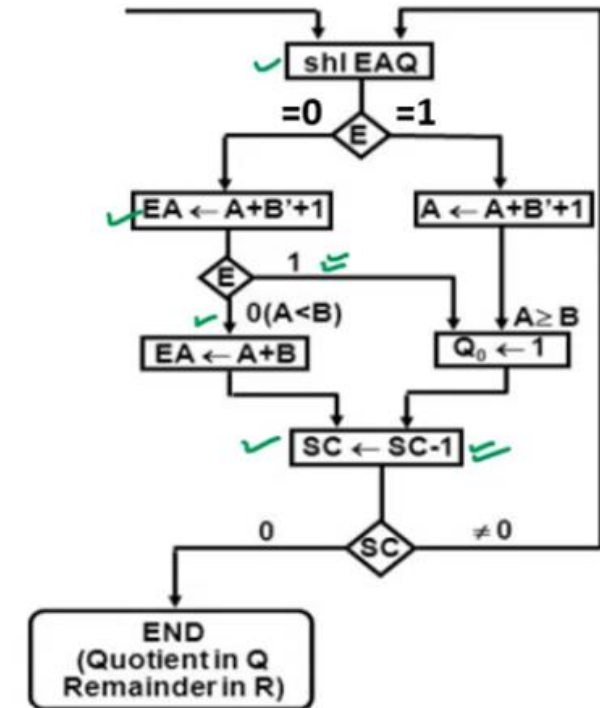
✓ Dividend (A) = 01110

✓ Divisor (B) = 10001

Initially A = 0000, Q = 00000, SC = 5

B' = 01110

✓ B' + 1 = 01111



Division Algorithm for Signed Magnitude Data

OPERATION	E	A	Q	SC
Initial	0	01110	00000	5
shl EAQ	0	11100	00000	
$A + B' + 1$	✓ 1	$\begin{array}{r} 11100 \\ + 01111 \\ \hline 01011 \end{array}$	$\begin{array}{r} 00000 \\ \hline 00001 \end{array}$	4
shl	0	10110	00010	
$A + \bar{B} + 1$	1	$\begin{array}{r} 10110 \\ + 01111 \\ \hline 00101 \end{array}$	$\begin{array}{r} 00010 \\ \hline 00011 \end{array}$	3
shl	0	01010	00110	
$A + \bar{B} + 1$		$\begin{array}{r} 01010 \\ + 01111 \\ \hline 11001 \end{array}$		
$A + B$	1	$\begin{array}{r} 11001 \\ + 10001 \\ \hline 01010 \end{array}$	$\begin{array}{r} 00110 \\ \hline 00110 \end{array}$	2
shl	0	10100	01100	
$A + \bar{B} + 1$	1	$\begin{array}{r} 10100 \\ + 01111 \\ \hline 00011 \end{array}$	$\begin{array}{r} 01100 \\ \hline 01101 \end{array}$	1

Question: Compute 01110 / 10001

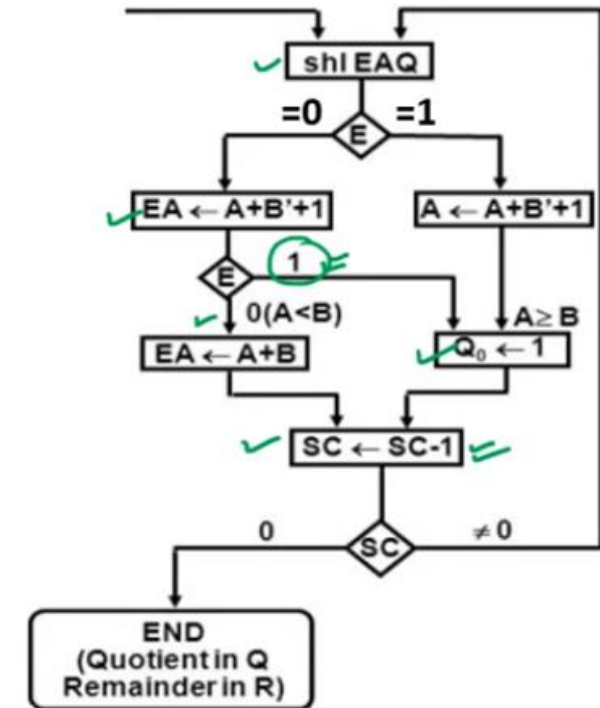
✓ Dividend (A) = 01110

✓ Divisor (B) = 10001

Initially A = 0000, Q = 00000, SC = 5

B' = 01110

✓ B' + 1 = 01111



Division Algorithm for Signed Magnitude Data

OPERATION	E	A	Q	SC
Initial	0	01110	00000	5
shl EAQ	0	11100	00000	
A + B' + 1	✓ 1	01011	00001	4
shl	0	10110	00010	
A + B' + 1	1	00101	00011	3
shl	0	01010	00110	
A + B' + 1		11001		
A + B	1	01010	00110	2
shl	0	10100	01100	
A + B' + 1	1	00011	01101	1
shl	0	00110	11010	
A + B' + 1		10101		
	1	00110	11010	0

Handwritten notes in the table:

- Initial: A = 01110, Q = 00000, SC = 5
- shl EAQ: A = 11100, Q = 00000
- A + B' + 1: A = 01011, Q = 00001, SC = 4
- shl: A = 10110, Q = 00010
- A + B' + 1: A = 00101, Q = 00011, SC = 3
- shl: A = 01010, Q = 00110
- A + B' + 1: A = 11001
- A + B: A = 01010, Q = 00110, SC = 2
- shl: A = 10100, Q = 01100
- A + B' + 1: A = 00011, Q = 01101, SC = 1
- shl: A = 00110, Q = 11010
- A + B' + 1: A = 10101
- Final: A = 00110, Q = 11010, SC = 0

Question: Compute 01110 / 10001

✓ Dividend (A) = 01110

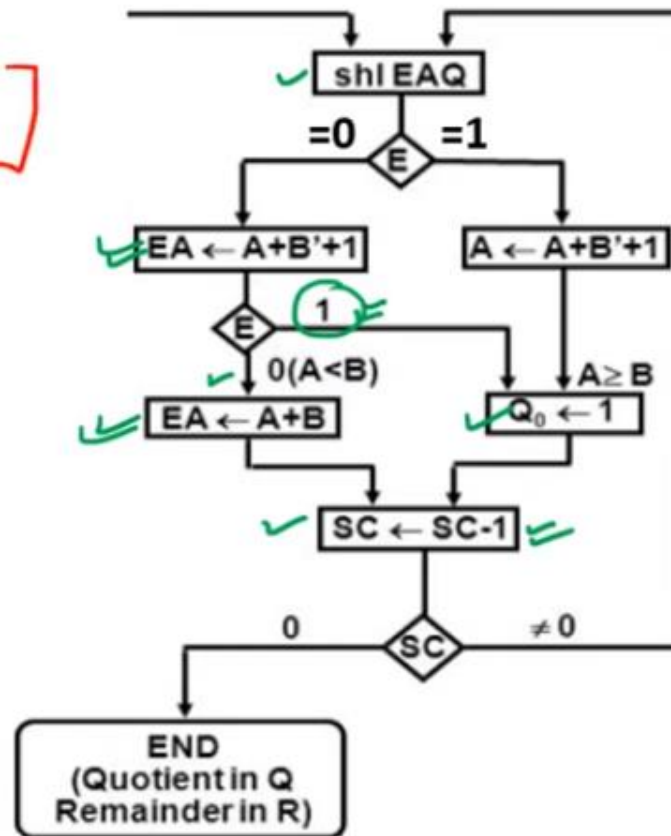
✓ Divisor (B) = 10001

Initially A = 0000, Q = 00000, SC = 5

B' = 01110

✓ B' + 1 = 01111

R = 00110
Q = 11010



Practice Problems

1. Compute $21 / 5$ by Restoring algorithm for signed magnitude data.
2. Solve $11010 / 1011$ by Restoring algorithm for signed magnitude data.