Booth’s Multiplication Algorithm

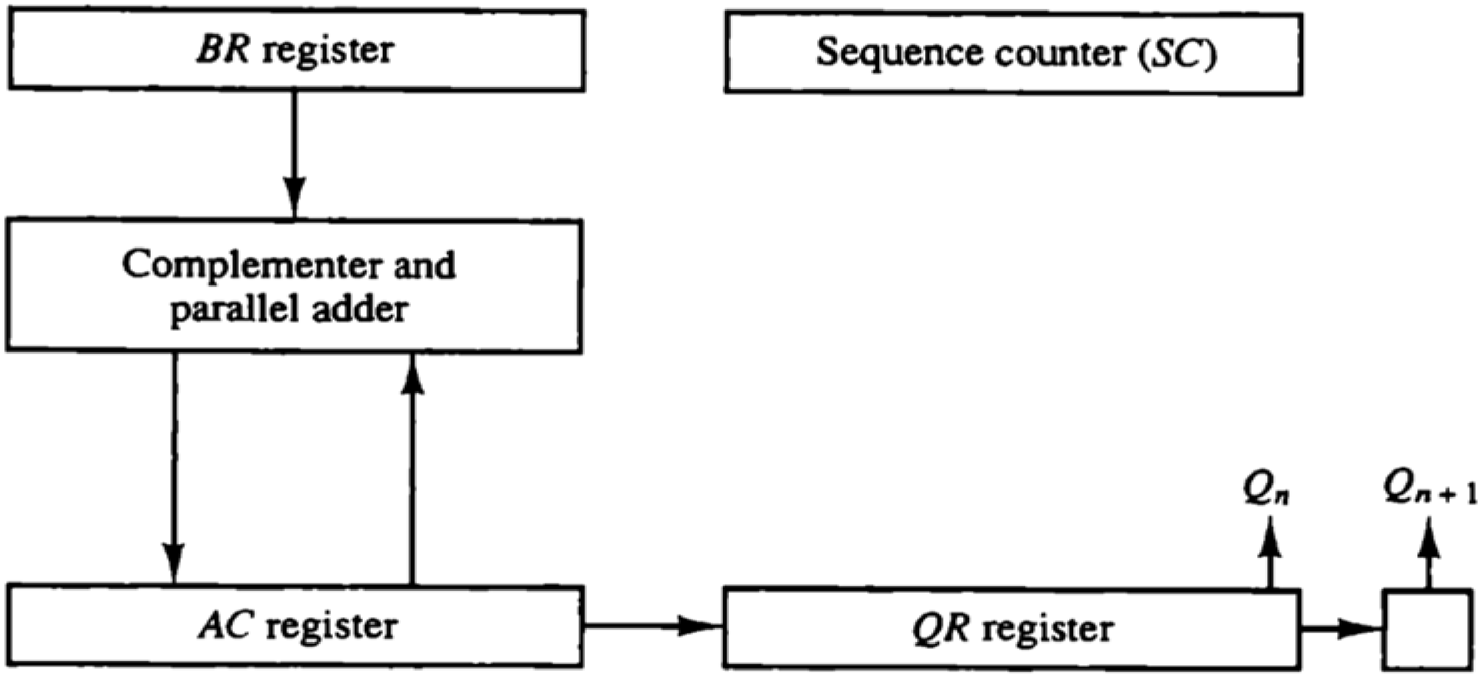
Booth’s Multiplication Algorithm

- Booth’s Algorithm gives a procedure for multiplying binary integers in signed-2’s complement representation.
- It operates on the fact that string 0’s in the multiplier requires no addition or subtraction but just shifting and string of 1’s in the multiplier require addition or subtraction followed by shifting.

Hardware for booth algorithm

- The algorithm requires the register configuration as shown in fig.

Hardware for Booth algorithm.



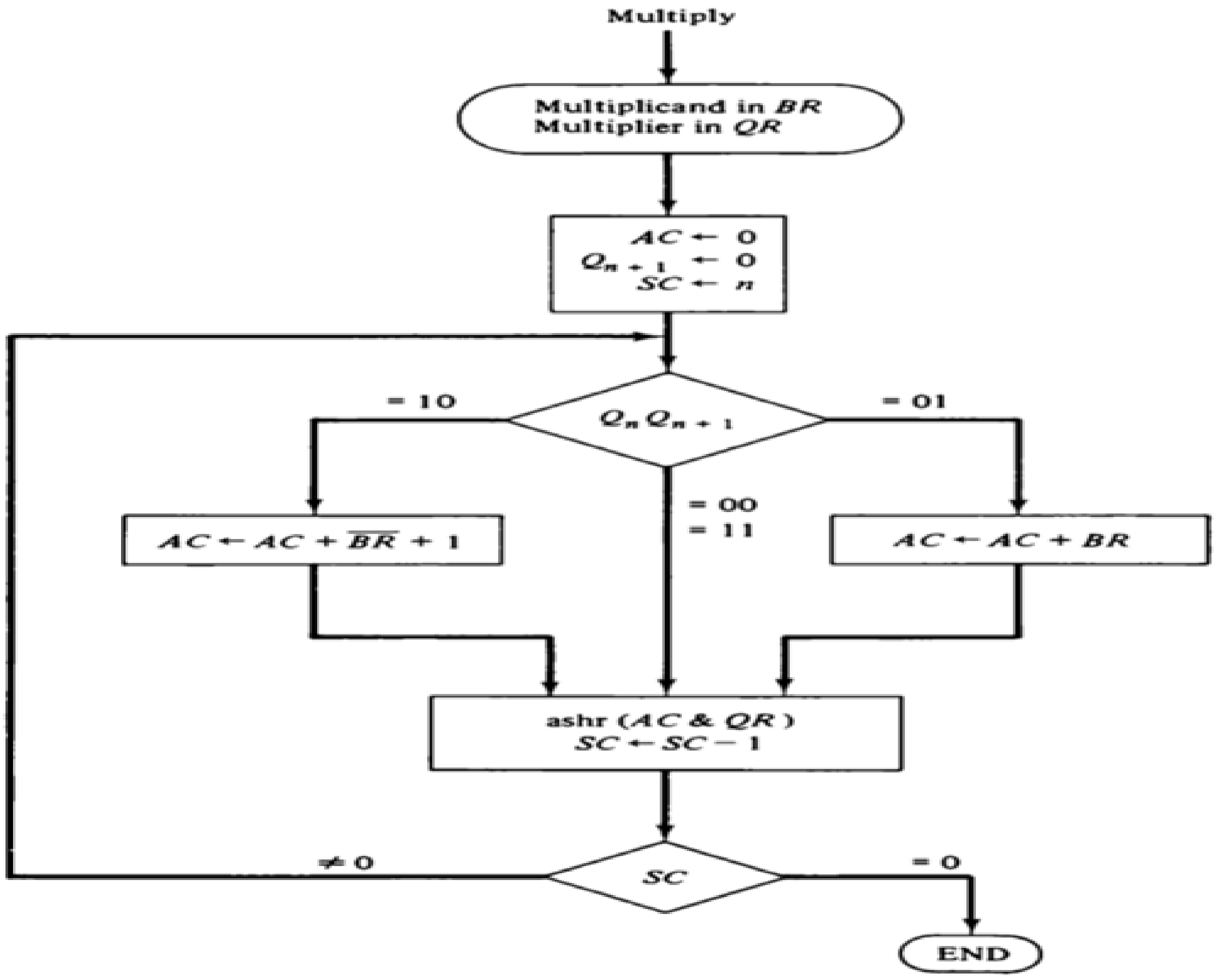
Hardware for booth algorithm

The following rules are required for Booths Algorithm

1. The multiplicand is subtracted from partial product upon encountering first LSB 1 in a multiplier.
2. The multiplicand is added to the partial product upon encountering the first 0(provided that there was a previous 1) in the multiplier.
3. The partial product does not change when the multiplier bit is identical to the previous multiplier bit.

Here an extra flip-flop Q_{n+1} is appended to QR to facilitate a double bit inspection of the multiplier.

Booth algorithm for multiplication of signed-2's complement



Booth algorithm for multiplication of signed-2's complement

- Initially AC and Qn+1 bit is cleared to zero and SC is set to number n equal to number of bits in multiplier.
- The two bits of the multiplier in Qn and Qn+1 are inspected.
- If two bits are equal to 01, the multiplicand is added to partial product in AC.
- If two bits are equal to 10, the multiplicand is subtracted from the partial product in AC.

Booth algorithm for multiplication of signed-2's complement

- When two bits are equal, partial product does not change.
- The next step is to do arithmetic shift right the partial product in AC and Multiplier in QR which leaves the sign bit in AC unchanged.
- The SC is decremented by one and the loop computation is repeated for n times.
- Finally the result is available in AC and QR with 2's complement representation for negative numbers.

Example

-9 X -13 = + 117

Q_n Q_{n+1}	$BR = 10111$ $\overline{BR} + 1 = 01001$	AC	QR	Q_{n+1}	SC
	Initial	00000	10011	0	101
1 0	Subtract BR	01001			
		01001			
	ashr	00100	11001	1	100
1 1	ashr	00010	01100	1	011
0 1	Add BR	10111			
		11001			
	ashr	11100	10110	0	010
0 0	ashr	11110	01011	0	001
1 0	Subtract BR	01001			
		00111			
	ashr	00011	10101	1	000