Multiplication Algorithms

Name :Sumit kumar Roll No:202151165

1. Booth’s Algorithm

Booth algorithm gives a procedure for **multiplying binary integers** in signed 2’s complement representation **in efficient way**, i.e., less number of additions/subtractions required. It operates on the fact that strings of 0’s in the multiplier require no addition but just shifting and a string of 1’s in the multiplier from bit weight 2^k to weight 2^m can be treated as 2^(k+1 ) to 2^m.

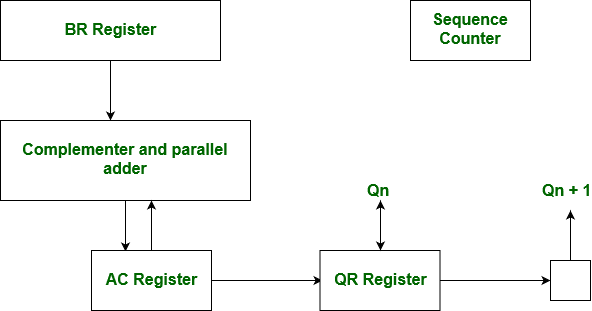
As in all multiplication schemes, booth algorithm requires examination **of the multiplier bits** and shifting of the partial product. Prior to the shifting, the multiplicand may be added to the partial product, subtracted from the partial product, or left unchanged according to following rules:

The multiplicand is subtracted from the partial product upon encountering the first least significant 1 in a string of 1’s in the multiplier

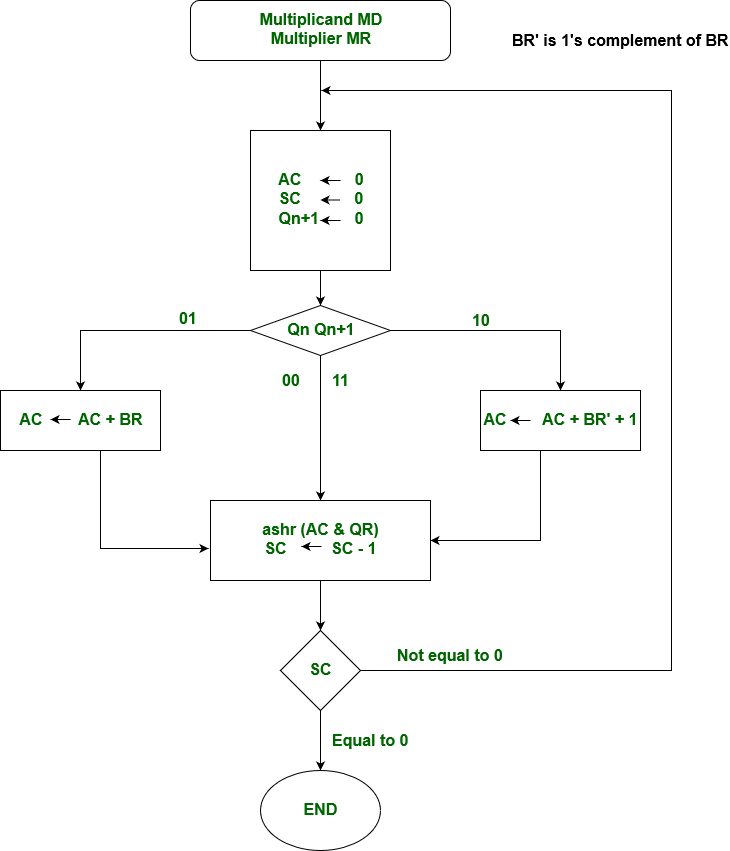
The multiplicand is added to the partial product upon encountering the first 0 (provided that there was a previous ‘1’) in a string of 0’s in the multiplier.

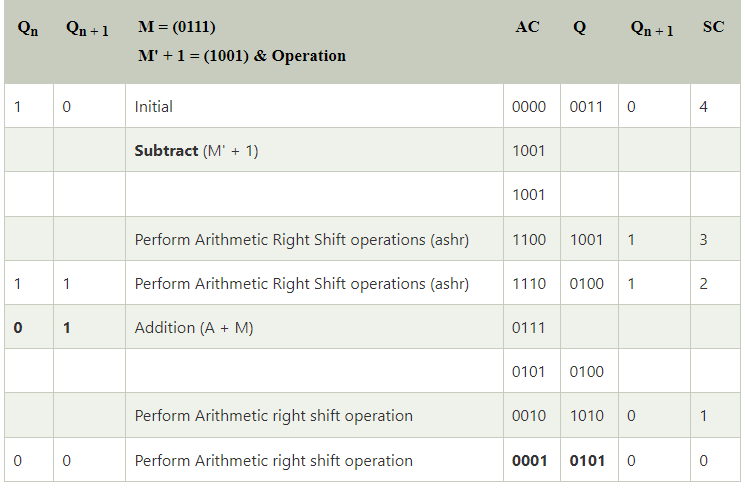
The partial product does not change when the multiplier bit is identical to the previous multiplier bit.

**Hardware Implementation of Booths Algorithm –** The hardware implementation of the booth algorithm requires the register configuration shown in the figure below.



1. **We name the register as A, B and Q, AC, BR and QR respectively. Qn designates the least significant bit of multiplier in the register QR. An extra flip-flop Qn+1is appended to QR to facilitate a double inspection of the multiplier.The flowchart for the booth algorithm is shown below.** Set the Multiplicand and Multiplier binary bits as M and Q, respectively.
2. Initially, we set the AC and Qn + 1 registers value to 0.
3. SC represents the number of Multiplier bits (Q), and it is a sequence counter that is continuously decremented till equal to the number of bits (n) or reached to 0.
4. A Qn represents the last bit of the Q, and the Qn+1 shows the incremented bit of Qn by 1.
5. On each cycle of the booth algorithm, Qn and Qn + 1 bits will be checked on the following parameters as follows:
   1. When two bits Qn and Qn + 1 are 00 or 11, we simply perform the arithmetic shift right operation (ashr) to the partial product AC. And the bits of Qn and Qn + 1 is incremented by 1 bit.
   2. If the bits of Qn and Qn + 1 is shows to 01, the multiplicand bits (M) will be added to the AC (Accumulator register). After that, we perform the right shift operation to the AC and QR bits by 1.
   3. If the bits of Qn and Qn + 1 is shows to 10, the multiplicand bits (M) will be subtracted from the AC (Accumulator register). After that, we perform the right shift operation to the AC and QR bits by 1.
6. The operation continuously works till we reached n - 1 bit in the booth algorithm.
7. Results of the Multiplication binary bits will be stored in the AC and QR registers.





The numerical example of the Booth's Multiplication Algorithm is 7 x 3 = 21 and the binary representation of 21 is 10101. Here, we get the resultant in binary 00010101. Now we convert it into decimal, as (000010101)10 = 2\*4 + 2\*3 + 2\*2 + 2\*1 + 2\*0 => 21.