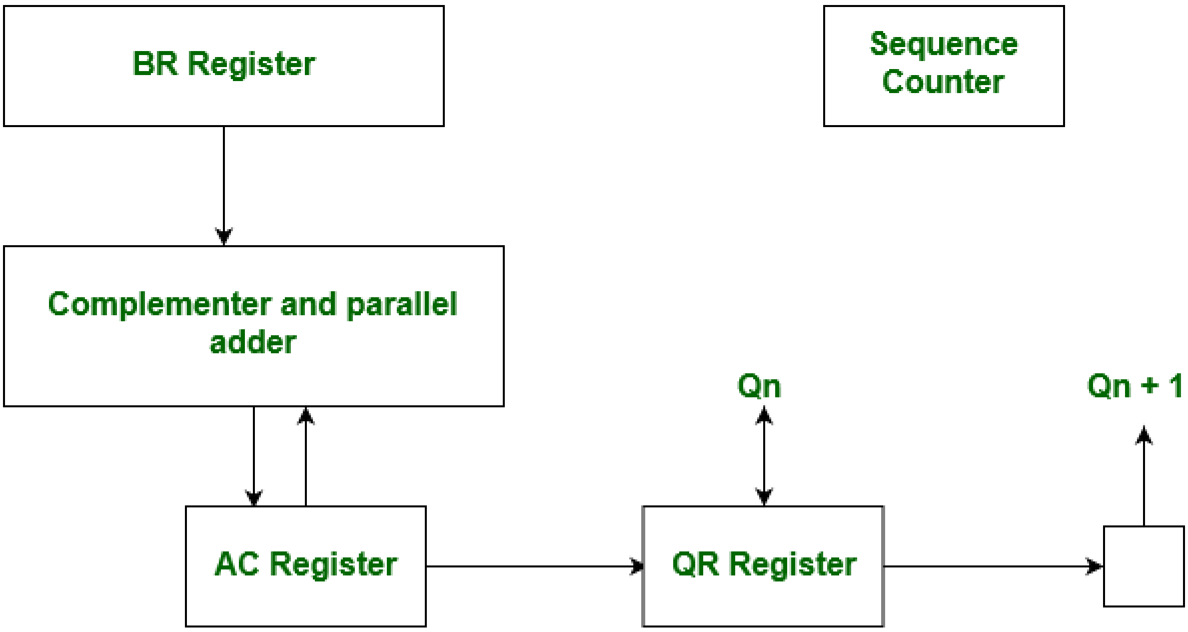
Booth's Algorithm

Booth's algorithm is a method for multiplying signed binary numbers in two's complement representation. It improves efficiency by minimizing the number of required arithmetic operations.

- The method works by examining pairs of adjacent bits in the multiplier and deciding whether to add, subtract, or do nothing with the multiplicand, followed by an arithmetic shift.
- This approach simplifies handling of consecutive sequences of 1s or 0s, making multiplication faster and more effective than the standard binary method.

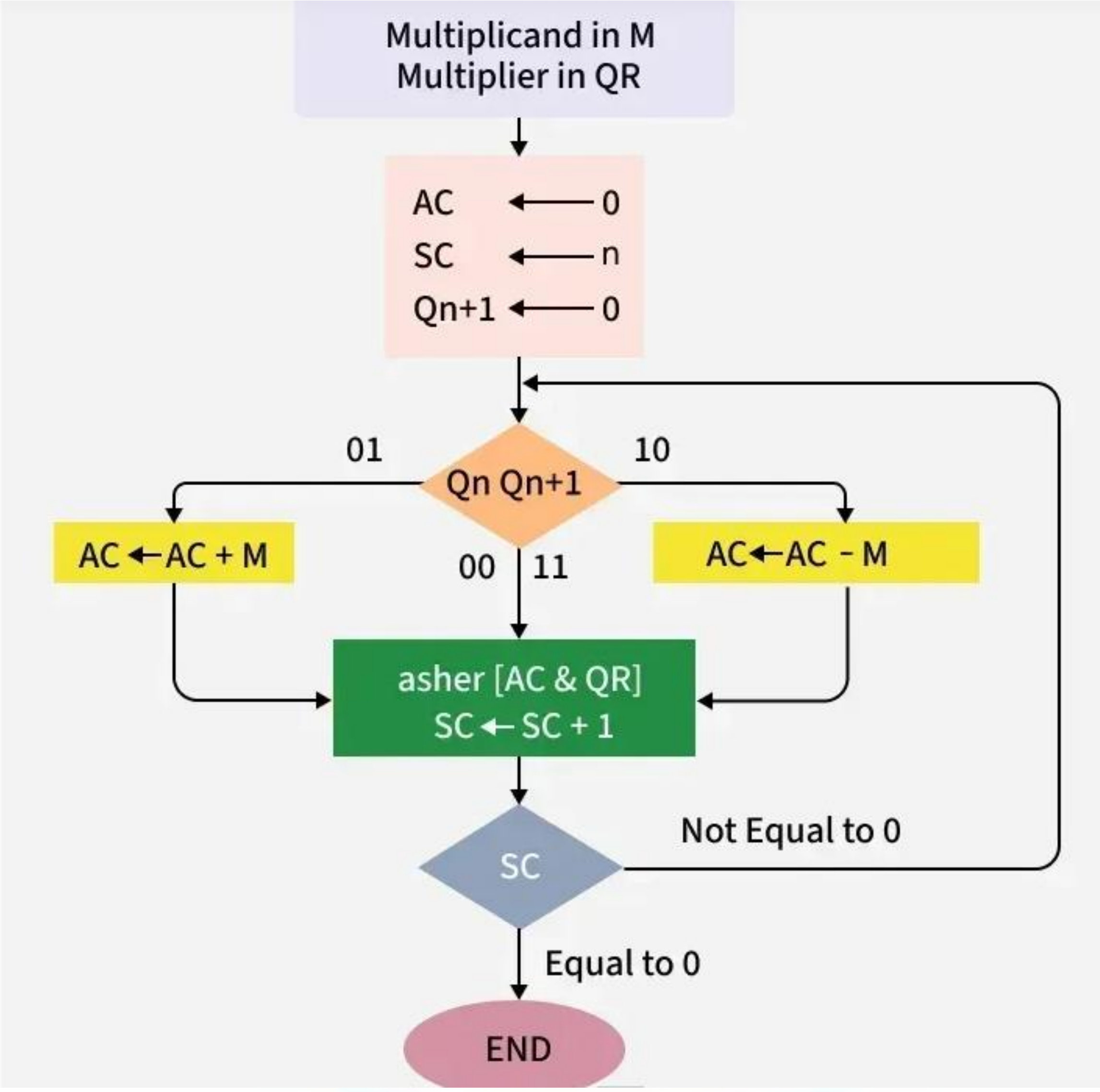
Hardware Implementation of Booth's Algorithm

The hardware implementation of the booth algorithm requires the register configuration shown in the figure below:



Booth's Algorithm Flowchart

We name the registers as A, B and Q, AC, BR and QR, respectively. Qn designates the least significant bit of the 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:



Booth's Algorithm Flowchart

Booth's Algorithm Steps

- Initialize **AC = 0**, **Qn+1 = 0**, and set **SC = n** (number of multiplier bits).
- Check the two least significant bits (**Qn** and **Qn+1**).
- If bits = **10** → Subtract multiplicand (M) from AC (first 1 in a sequence).
- If bits = **01** → Add multiplicand (M) to AC (first 0 after ones).
- If bits = **00** or **11** → No change in AC (partial product unchanged).
- Perform an **Arithmetic Shift Right (ashr)** on [AC, QR, Qn+1] (sign bit in AC remains).
- Decrement sequence counter (SC).

- Repeat steps until SC = 0 (n iterations complete).
- Handle negative numbers: if multiplicand/multiplier is negative, take its **2's complement** to simplify operations.
- Final product obtained in [AC, QR] after all steps.

Application of Booth's Algorithm

- **Processors and ALUs:** Booth's Algorithm enables efficient signed multiplication inside microchips and processors by reducing the number of additions/subtractions, leading to faster arithmetic logic unit (ALU) operations essential for computing, graphics, and cryptography.
- **Digital Signal Processing (DSP):** It accelerates multiplication tasks in DSP applications such as filtering and convolution for real-time audio, video, and signal processing.
- **Hardware Accelerators:** Specialized hardware for image processing, neural networks, and AI use Booth's Algorithm to speed up multiplication operations.
- **Cryptography:** Cryptographic operations involving large-number exponentiation benefit from faster multiplication with Booth's Algorithm, improving encryption and signature processes.
- **High-Performance Computing (HPC):** Large-scale scientific and mathematical computations employ Booth's Algorithm for optimized multiplication, enhancing overall system performance.
- **Embedded Systems:** Resource-limited embedded devices improve multiplication efficiency and power consumption by using Booth's Algorithm.
- **Network Packet Processing:** Booth's Algorithm helps efficient multiplication in network devices for operations on packet headers and payloads.