```
                          Arithmetic Logic Unit (ALU 9+8 bit)
 ____________________________________________________________________________________________
    bit 1                  bit
    sign extension         123456789

         operand 1     AL1 000000000  00000000   ALBA   Factor for addition
         operand 1     AL2 000000000  00000000   ALBS   Factor for subtraction
         carryover     ALC 000000000
         result        ALR 000000000  00000000 0 ALx  extra eit
                           1+8 bit    ARL 8 bit
                 result:   High Byte  Low Byte
 ____________________________________________________________________________________________
```

The sequence control is done according to the right bit of ARL (result low byte) and the extra bit ALx.

>> The result line (result) ALR ARL ALx is shifted 1 bit to the right, the auxiliary bit is dropped out.
On the left, the bit identical to the one shifted away is added as a sign extension.

ARL	ALx	short form	Meaning
...0	0	>>	(shift only) shift >>
...0	1	add >>	factor is added, then >>
...1	0	sub >>	-factor is added, then >>
...1	1	>>	(shift only) shift >>

ALR is copied to AL1 as a subtotal.

The multiplication algorithm (according to Booth, for signed factors)

```
  set ALBA factor 1        //  factor for addition
  set ALBS -factor 1       // -faktor (to calculate in advance, two's complement for subtraction)
  set ALR  000000000       // 9 bit: initial value for continuous summation
  set ARL  factor 2        // set 8 bits into the result low byte
  set ALx  0               // 1 extra bit (initial value 0)
 //-------------------------------------------------------
  loop 8                   // loop through 8 times
    b1 = right(ARL)        // right bit of ARL
    b2 = get(ALx)          // extra bit
    if b1=b2               // 0 0 and 1 1
      shift >>             // shift right only
    else
      if b1=0 ∧ b2=1
        set AL2 ALBA       // set factor 1 for addition of ALBA in AL2
      endif
      if b1=1 ∧ b2=0
        set AL2 ALBS       // set -factor 1 for addition of ALBS in AL2
      endif
      set AL1 ALR          // transfer current total to AL1
        // Perform addition in the ALU, the sign bit of AL2 is duplicated in advance.
        //
        add   //  <-- addition algorithm add_ALU_9 (AL1+AL2) in 9 bit arithmetic unit with sign extension
        //
        shift >>           //  shift right
    endif
  end_loop                 // end of loop
```

If the number range -128..127 is not exceeded, the 8 bit product is in the low byte ARL.