Unit V:
Computer Arithmetic Techniques

The Arithmetic and Logic Unit, Multiplication of positive numbers, Signed operand multiplication, Booths algorithm, Integer division, Floating point representation – IEEE standard.

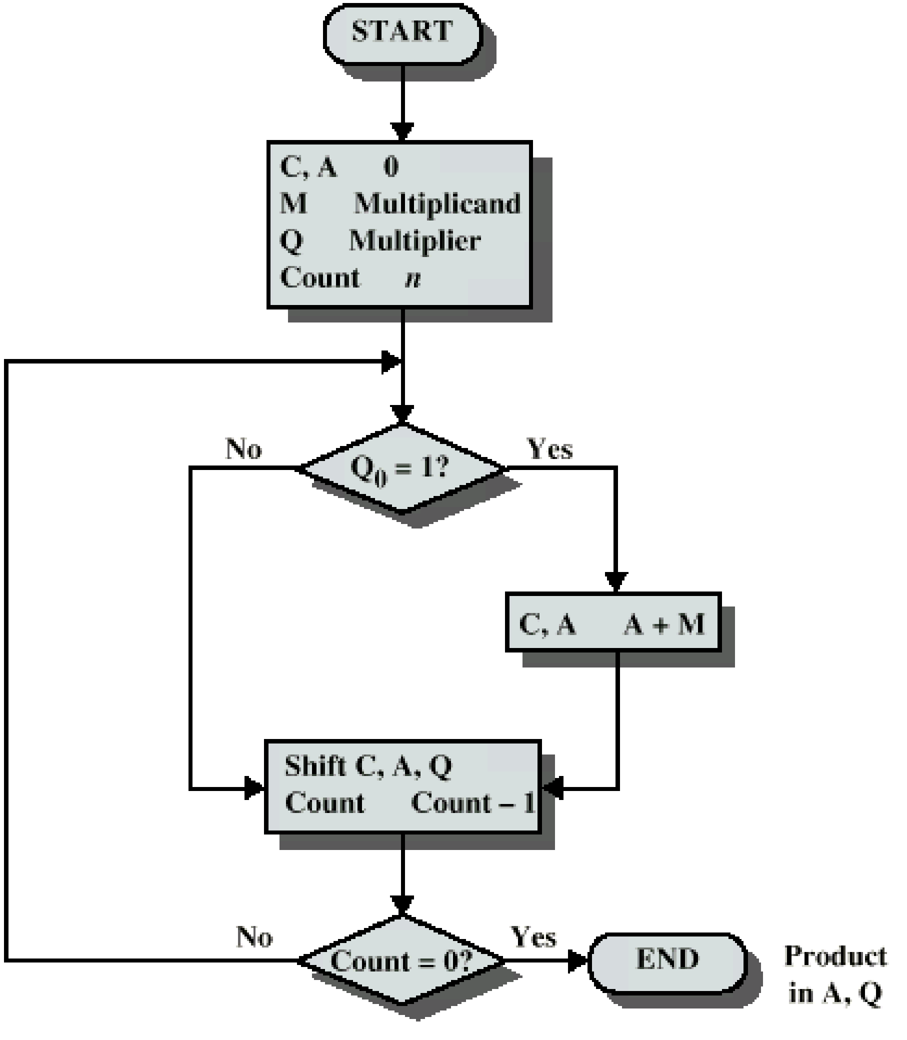
Multiplication

- Complex
- Work out partial product for each digit
- Take care with place value (column)
- Add partial products

Multiplication Example (unsigned) (long hand)

- 1011 Multiplicand (11 dec)
- x 1101 Multiplier (13 dec)
- 1011 Partial products
- 00000 Note: if multiplier bit is 1 copy
- 1011*00* multiplicand (place value)
- 1011*000* otherwise zero
- 10001111 Product (143 dec)
- Note: need double length result

Flowchart for Unsigned Binary Multiplication



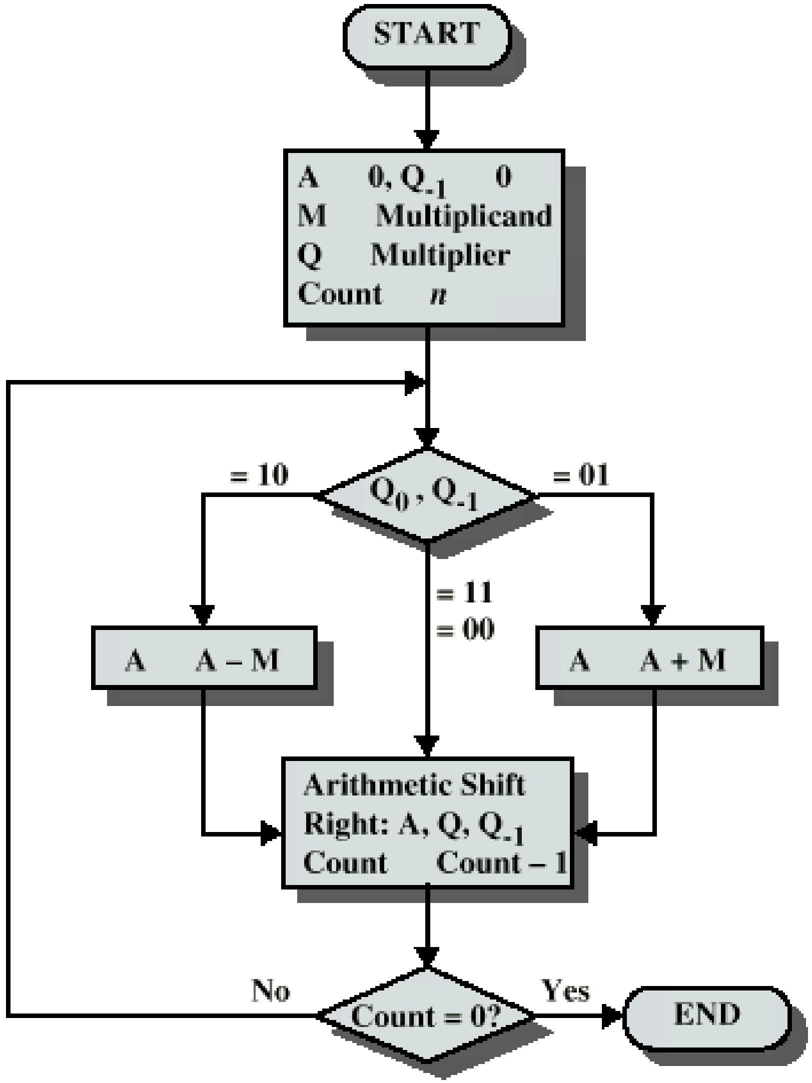
Execution of Example

C	A	Q	M		
0	0000	1101	1011	Initial Values	
0	1011	1101	1011	Add	First Cycle
0	0101	1110	1011	Shift	
0	0010	1111	1011	Shift	Second Cycle
0	1101	1111	1011	Add	Third Cycle
0	0110	1111	1011	Shift	
1	0001	1111	1011	Add	Fourth Cycle
0	1000	1111	1011	Shift	

Multiplying Negative Numbers

- This does not work!
- Solution 1
  - Convert to positive if required
  - Multiply as above
  - If signs were different, negate answer
- Solution 2
  - Booth's algorithm

Booth's Algorithm



Example of Booth's Algorithm

- 3*7
- First setup the columns and initial values.

A	Q	Q_{-1}	M	Comment
00000	00011	0	00111	Init values

- This case 3 is in Q and M is 7.
  - But could put 7 in Q and M as 3

Example of Booth's Algorithm

- First cycle: Now look at Q_0 and Q_{-1}

A	Q	Q_{-1}	M	Comment
00000	00011	0	00111	Init values

- With a 10, we Sub (A=A-M), then shift (always to the right)

A	Q	Q_{-1}	M	Comment	
00000	00011	0	00111	Init values	
11001	00011	0	00111	Sub (A=A-M)	First
11100	10001	1	00111	Shift	Cycle

- Second cycle: looking at Q_0 and Q_{-1}
  - With a 11, we only shift.

Example of Booth's Algorithm

- 2nd cycle Result

A	Q	Q_{-1}	M	Comment	
00000	00011	0	00111	Init values	
11001	00011	0	00111	Sub (A=A-M)	First
11100	10001	1	00111	Shift	Cycle
11110	01000	1	00111	Shift	Second Cycle

- Third cycle, Q_0 and Q_{-1} have 01
  - —So we will Add (A=A+M), then shift

Example of Booth's Algorithm

- 3[nd] cycle Result

A	Q	Q_{-1}	M	Comment	
00000	00011	0	00111	Init values	
11001	00011	0	00111	Sub (A=A-M)	First
11100	10001	1	00111	Shift	Cycle
11110	01000	1	00111	Shift	Second Cycle
00101	01000	1	00111	Add (A=A+M)	Third
00010	10100	0	00111	Shift	Cycle

- 4[th] cycle, Q_0 and Q_{-1} have 00
  - So we only shift

Example of Booth's Algorithm

- 4^{nd} cycle Result

A	Q	Q_{-1}	M	Comment	
00000	00011	0	00111	Init values	
11001	00011	0	00111	Sub (A=A-M)	First Cycle
11100	10001	1	00111	Shift	
11110	01000	1	00111	Shift	Second Cycle
00101	01000	1	00111	Add (A=A+M)	Third Cycle
00010	10100	0	00111	Shift	
00001	01010	0	00111	Shift	Fourth Cycle

- 5^{th} cycle, Q_0 and Q_{-1} have 00
  - So we only shift

Example of Booth's Algorithm

5th cycle Result

A	Q	Q-1	M	Comment	
00000	00011	0	00111	Init values	
11001	00011	0	00111	Sub (A=A-M)	First Cycle
11100	10001	1	00111	Shift	
11110	01000	1	00111	Shift	Second Cycle
00101	01000	1	00111	Add (A=A+M)	Third Cycle
00010	10100	0	00111	Shift	
00001	01010	0	00111	Shift	Fourth Cycle
00000	10101	0	00111	Shift	Fifth Cycle

Since we are working in 5 bits, we only repeat 5 times

Example of Booth's Algorithm

A	Q	Q_{-1}	M	Comment	
00000	00011	0	00111	Init values	
11001	00011	0	00111	Sub (A=A-M)	First
11100	10001	1	00111	Shift	Cycle
11110	01000	1	00111	Shift	Second Cycle
00101	01000	1	00111	Add (A=A+M)	Third
00010	10100	0	00111	Shift	Cycle
00001	01010	0	00111	Shift	Fourth Cycle
00000	10101	0	00111	Shift	Fifth Cycle

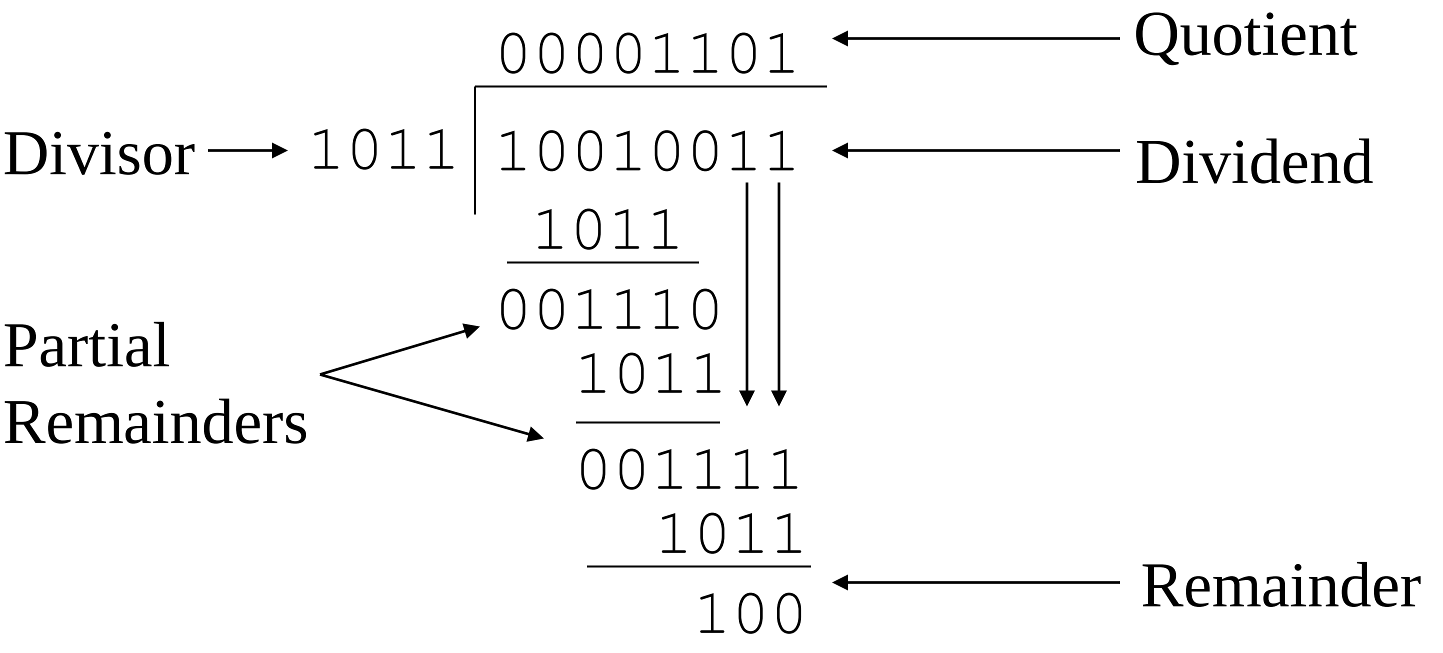
Result is A and Q so 0000010101 which is 21.

Note: The sign bit is the last bit in A.

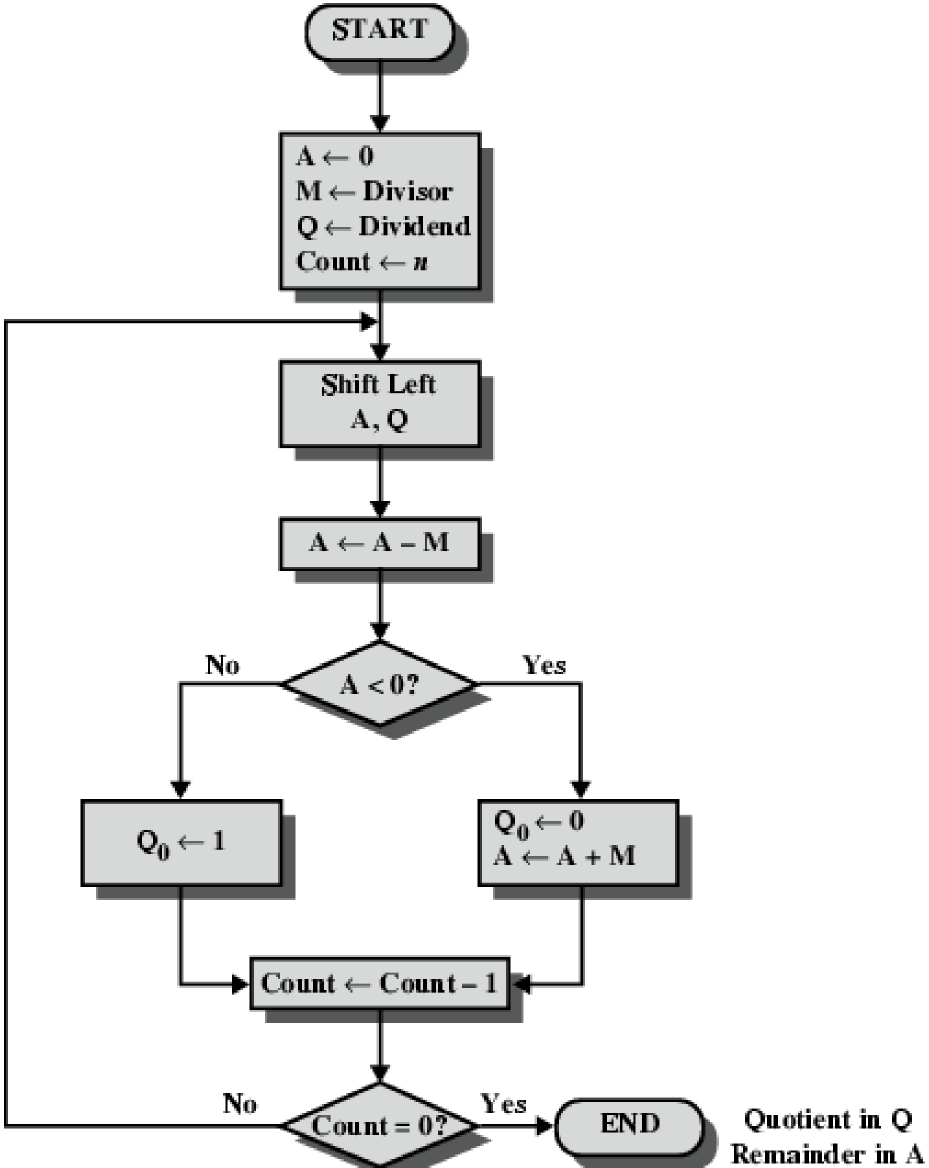
Division

- More complex than multiplication
- Negative numbers are really bad!
- Based on long division

Division of Unsigned Binary Integers



Flowchart for Unsigned Binary Division



Signed Division

1. Load divisor into M and the dividend into A, Q registers. Dividend must be 2n-bit twos complement number
   - 0111 (7) becomes 00000111
   - 1001 (-7) becomes 11111001
2. Shift A, Q left 1 bit position
3. If M and A have the same signs, $A \leftarrow A - M$ else $A \leftarrow A + M$

4. Step 3 is successful if sign of A is the same as before step 3 and at the end of step 3
   A. if successful or (A=0 AND Q=0) then set Q0←1
   B. if unsuccessful and ($A \neq 0$ OR $Q \neq 0$) then restore the previous value of A
5. Repeat steps 2 through 4 as many times as there are bit positions in Q.
6. The reminder is in A. If the signs are of Divisor and dividend were the same, the quotient is in Q, otherwise the correct quotient is the twos complement of Q.

Examples of division (signed)

A	Q	M=0011		A	Q	M=0011
0000	0111	Initial Value		1111	1001	Initial Value
0000	1110	shift		1111	0010	shift
1101		subtract		0010		Add
0000	1110	restore		1111	0010	Restore
0001	1100	shift		1110	0100	Shift
1110		subtract		0001		Add
0001	1100	restore		1110	0100	Restore
0011	1000	shift		1100	1000	Shift
0000		subtract		1111		Add
0000	1001	set Q0 = 1		1111	1001	Q0 =1
0001	0010	Shift		1111	0010	Shift
1110		subtract		0010		Add
0001	0010	restore		1111	0010	Restore

(a) 7/3 (b) -7/3

Q&A