ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΥΠΟΛΟΓΙΣΤΩΝ

–

ΑΡΙΘΜΗΤΙΚΕΣ ΠΡΑΞΕΙΣ

ΣΗΜΜΥ, 5^{Ο} ΕΞΑΜΗΝΟ

http://www.cslab.ece.ntua.gr/courses/comparch

ΑΡΙΘΜΟΙ

- Decimal
  - Εύκολο για τον άνθρωπο
  - Ιδιαίτερα για την εκτέλεση αριθμητικών πράξεων
- Hex
  - Χρήσιμο για την απεικόνιση δυαδικών αριθμών (4 bits / symbol)
  - Αρκετά δύσκολο για αριθμητικές πράξεις
- Binary
  - Χρησιμοποιείται εσωτερικά στους υπολογιστές για την απεικόνιση των αριθμών
  - Αριθμητικές πράξεις +,-,*,/ ????

ΑΡΙΘΜΗΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

- Βάση (radix) r ή β

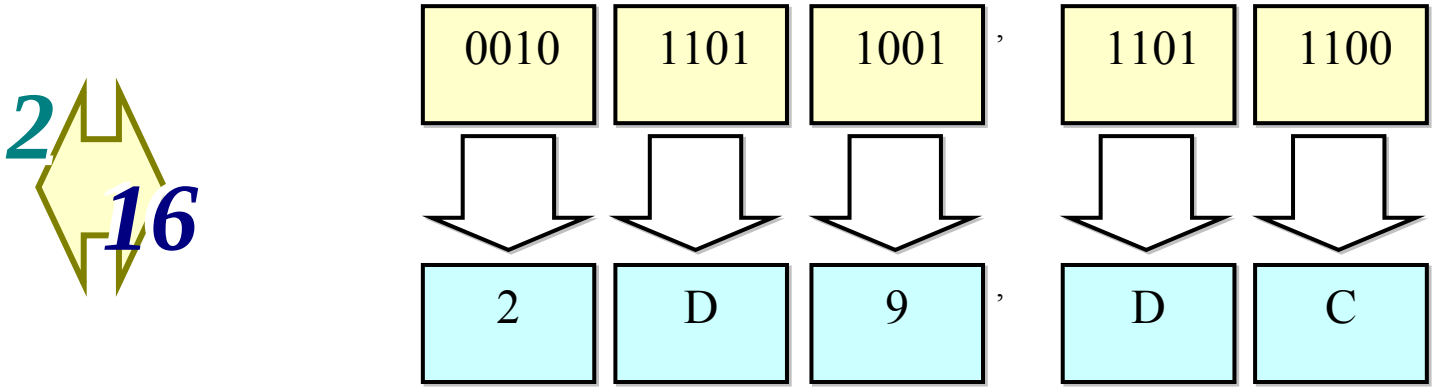
$$N=\sum_{i=-n}^{m-1}\alpha_i\cdot\beta^i=\underbrace{\alpha_{m-1}\cdot\beta^{m-1}+\alpha_{m-2}\cdot\beta^{m-2}+\ldots+\alpha_1\cdot\beta+\alpha_0}_{\text{ακέραιο μέρος του αριθμού}}+\underbrace{\alpha_{-1}\cdot\frac{1}{\beta}+\alpha_{-2}\cdot\frac{1}{\beta^2}+\ldots+\alpha_{-n}\cdot\frac{1}{\beta^n}}_{\text{κλασματικό μέρος του αριθμού}}$$

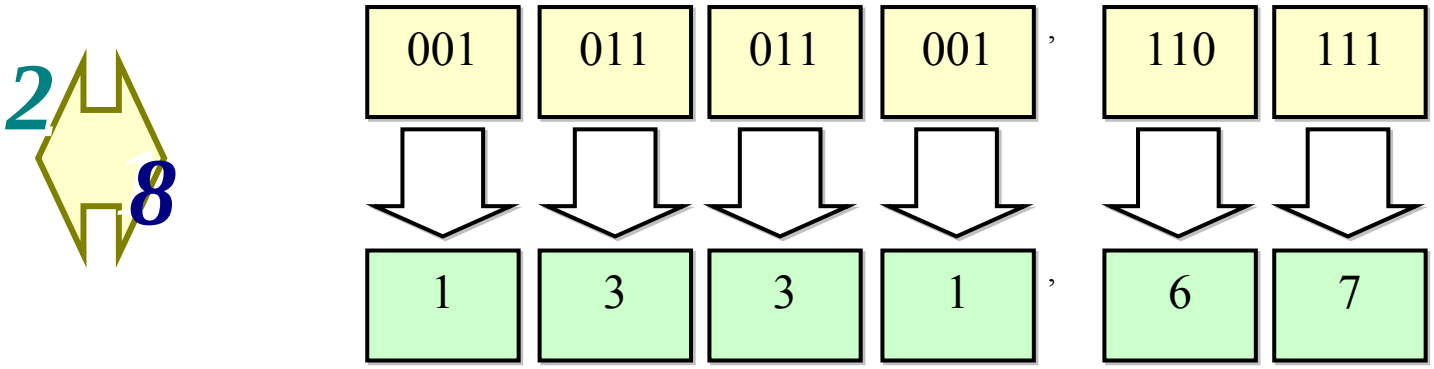
- Παράσταση αριθμού: $\alpha_{m-1}\alpha_{m-2}\ldots\alpha_0$, $\alpha_{-1}\alpha_{-2}\ldots\alpha_{-n}$
- Ψηφία του συστήματος: 0,…, β-1

π.χ. $1821{,}352_{(10)} = 1\times10^3 + 8\times10^2 + 2\times10^1 + 1\times10^0 + 3\times10^{-1} + 5\times10^{-2} + 2\times10^{-3}$

$1256{,}124_{(8)} = 1\times8^3 + 2\times8^2 + 5\times8^1 + 6\times8^0 + 1\times8^{-1} + 2\times8^{-2} + 4\times8^{-3}$

ΜΕΤΑΤΡΟΠΕΣ ΜΕΤΑΞΥ ΣΥΣΤΗΜΑΤΩΝ





COMPUTER ARITHMETIC

- Θετικοί ακέραιοι
  - Διευθύνσεις, pointers κτλ.
- Προσημασμένοι ακέραιοι
  - Αριθμητικές πράξεις ακεραίων αριθμών
- Πραγματικοί αριθμοί
  - Αριθμητικές πράξεις πραγματικών αριθμών
- Περιορισμένος αριθμός bits
  - Υπερχείλιση (Overflow)
  - 1101 + 0101 = 1 0010

ΑΝΑΠΑΡΑΣΤΑΣΗ ΑΡΙΘΜΩΝ

- n bits
  - Μπορούν να ανπαρασταθούν 2^n διαφορετικά στοιχεία
- MSB (Most Significant Bit), LSB (Less Significant Bit)
- 3 διαφορετικές αναπαραστάσεις:
  - Πρόσημο – μέτρο, Συμπλήρωμα ως προς 1, Συμπλήρωμα ως προς 2

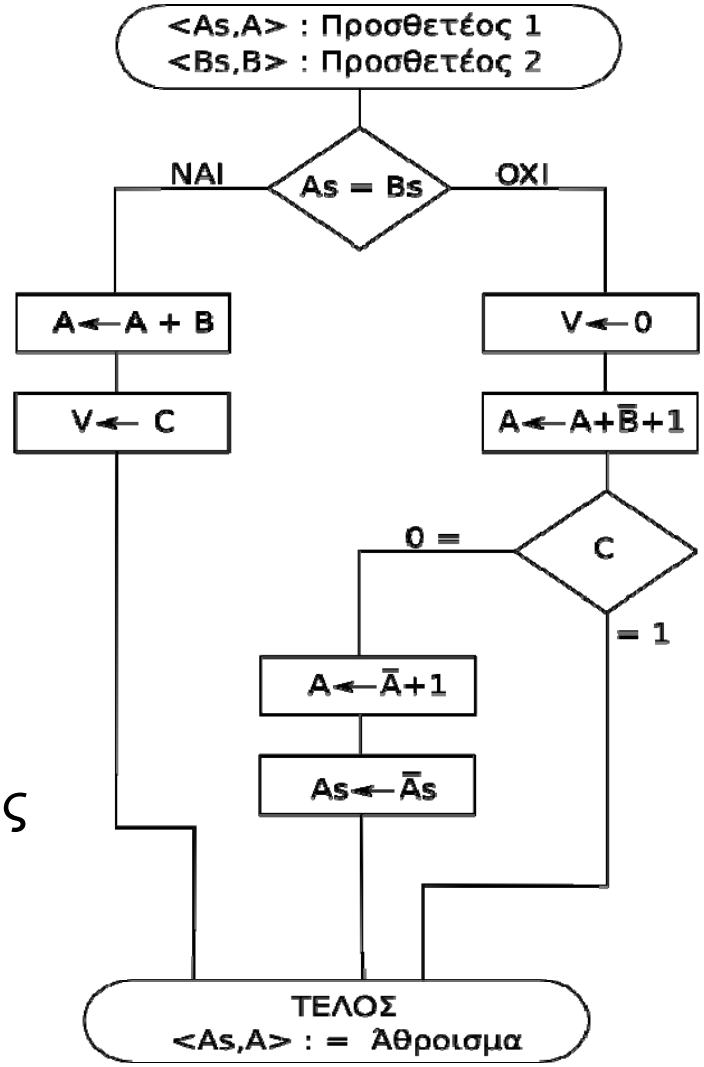
Sign Magnitude	One's Complement	Two's Complement
000 = +0	000 = +0	000 = +0
001 = +1	001 = +1	001 = +1
010 = +2	010 = +2	010 = +2
011 = +3	011 = +3	011 = +3
100 = -0	100 = -3	100 = -4
101 = -1	101 = -2	101 = -3
110 = -2	110 = -1	110 = -2
111 = -3	111 = -0	111 = -1

ΠΡΟΣΗΜΟ - ΜΕΤΡΟ

- Το MSB αφιερώνεται για να περιγράψει το πρόσημο (0 → +, 1 → -)
- Τα υπόλοιπα bits καθορίζουν το μέτρο του αριθμού
- Π.χ. Σε 12 bits
  - το $+1764_{(10)}$ παρίσταται ως: 011011100100
  - το $-1764_{(10)}$ παρίσταται ως: 111011100100
- Παρίστανται οι αριθμοί από $-2^{n-1}+1$ εώς $2^{n-1}-1$
- Το 0 έχει 2 παραστάσεις (000…0 και 100…0)

ΠΡΟΣΗΜΟ - ΜΕΤΡΟ

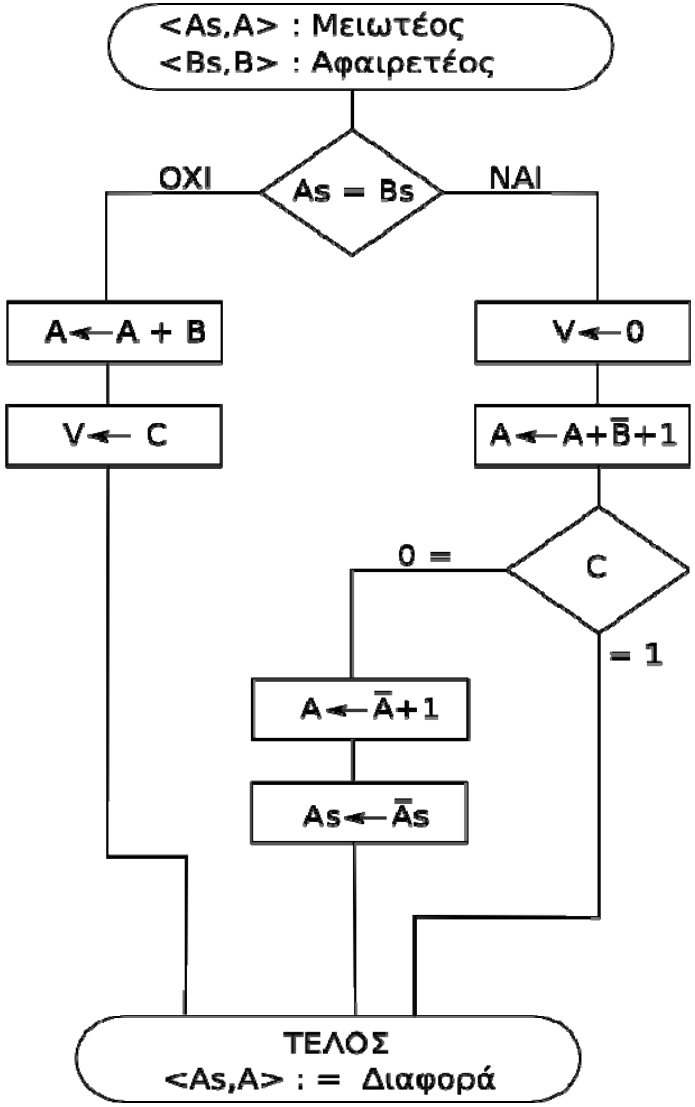
- Πρόσθεση
- A, B ομόσημοι
  - Υπάρχει περίπτωση υπερχείλισης
  - Εντοπίζεται στο κρατούμενο C_n
- A, B ετερόσημοι
  - Δεν υπάρχει υπερχείλιση
  - Το κρατούμενο C_n προσδιορίζει το πρόσημο του τελικού αποτελέσματος



ΠΡΟΣΗΜΟ - ΜΕΤΡΟ

- Αφαίρεση

- Παρόμοια με την πρόσθεση
  - Αφαίρεση ομόσημων = πρόσθεση ετερόσημων
  - Αφαίρεση ετερόσημων = πρόσθεση ομόσημων



ΠΡΟΣΗΜΟ - ΜΕΤΡΟ

- Πλεονεκτήματα:
  - Εύκολα αντιληπτή από τον άνθρωπο
  - Γρήγορη εύρεση του αντιθέτου
- Μειονεκτήματα:
  - Μη αποδοτική στις στοιχειώδεις πράξεις, π.χ. για υλοποίηση πρόσθεσης:
    - Έλεγχος προσήμου
    - Σύγκριση μέτρου
    - Διεξαγωγή πράξης
  - Διπλή αναπαράσταση για το 0
  - Δύσκολη επέκταση προσήμου

ΣΥΜΠΛΗΡΩΜΑ ΩΣ ΠΡΟΣ 1

- Το MSB προσδιορίζει το πρόσημο (0→+, 1→-)
- Οι αρνητικοί ακέραιοι προκύπτουν από τον αντίστοιχο θετικό με αντιστροφή και των n bits
- Π.χ. Σε 12 bits
  - το $+1764_{(10)}$ παρίσταται ως: 011011100100
  - το $-1764_{(10)}$ παρίσταται ως: 100100011011
- Παρίστανται οι αριθμοί από $\mathbf{-2^{n-1}+1}$ εώς $\mathbf{2^{n-1}-1}$
- Το 0 έχει 2 παραστάσεις (000…0 και 111…1)

- Μειονεκτήματα
  - Διπλή αναπαράσταση για το 0
  - Δύσκολο για τον άνθρωπο

ΣΥΜΠΛΗΡΩΜΑ ΩΣ ΠΡΟΣ 2

- Το MSB προσδιορίζει το πρόσημο (0→+, 1→-)
- Οι αρνητικοί ακέραιοι προκύπτουν από τον αντίστοιχο θετικό με αντιστροφή και των n bits και **πρόσθεση της μονάδας**
- Η ίδια διαδικασία εφαρμόζεται για την εύρεση του αντίθετου είτε ο αριθμός είναι θετικός είτε είναι αρνητικός
- Π.χ. Σε 12 bits
  - το $+1764_{(10)}$ παρίσταται ως: 011011100100
  - το $-1764_{(10)}$ παρίσταται ως: 100100011011

```
                  100100011011
                + 000000000001
                  ------------
                  100100011100
```

- Αν ẑ το συμπλήρωμα ως προς 2 του z, στα n bits ισχύει: $\mathbf{z + \hat{z} = 2^n}$
- Παρίστανται οι αριθμοί από $\mathbf{-2^{n-1}}$ εώς $\mathbf{2^{n-1}-1}$
- Το 0 έχει 1 παράσταση (000…0)

ΣΥΜΠΛΗΡΩΜΑ ΩΣ ΠΡΟΣ 2

- Πρόσθεση
- Αφαίρεση
  - Πρόσθεση του αντίθετου αριθμού

- Πρόσθεση bit-bit από δεξιά προς αριστερά με μεταφορά κρατουμένου

001100	001100	110100	110100	101110
+010001	+101111	+010001	+101111	+101110
---	---	---	---	---
011101	111011	1000101	1100011	1011100

ΣΥΜΠΛΗΡΩΜΑ ΩΣ ΠΡΟΣ 2

- Η διαδικασία της πρόσθεσης/αφαίρεσης είναι αρκετά απλή
- Πιο γρήγορη από την αντίστοιχη διαδικασία για αριθμούς σε μορφή “πρόσημο-μέτρο”
  - Δεν απαιτείται έλεγχος προσήμου
- Πιθανότητα υπερχείλισης και άρα λανθασμένου αποτελέσματος
- Μειονεκτήματα:
  - Δύσκολο για τον άνθρωπο
  - Δύσκολη εύρεση του αντιθέτου

ΣΥΜΠΛΗΡΩΜΑ ΩΣ ΠΡΟΣ 2

- Καμία πιθανότητα υπερχείλισης
  - Πρόσθεση ετερόσημων αριθμών
  - Αφαίρεση ομόσημων αριθμών

- Υπερχείλιση όταν :
  - Πρόσθεση δύο θετικών αριθμών δίνει αρνητικό αριθμό
  - Πρόσθεση δύο αρνητικών αριθμών δίνει θετικό αριθμό
  - Αφαίρεση αρνητικού από θετικό δίνει αρνητικό
  - Αφαίρεση θετικού από αρνητικό δίνει θετικό

ΣΥΜΠΛΗΡΩΜΑ ΩΣ ΠΡΟΣ 2

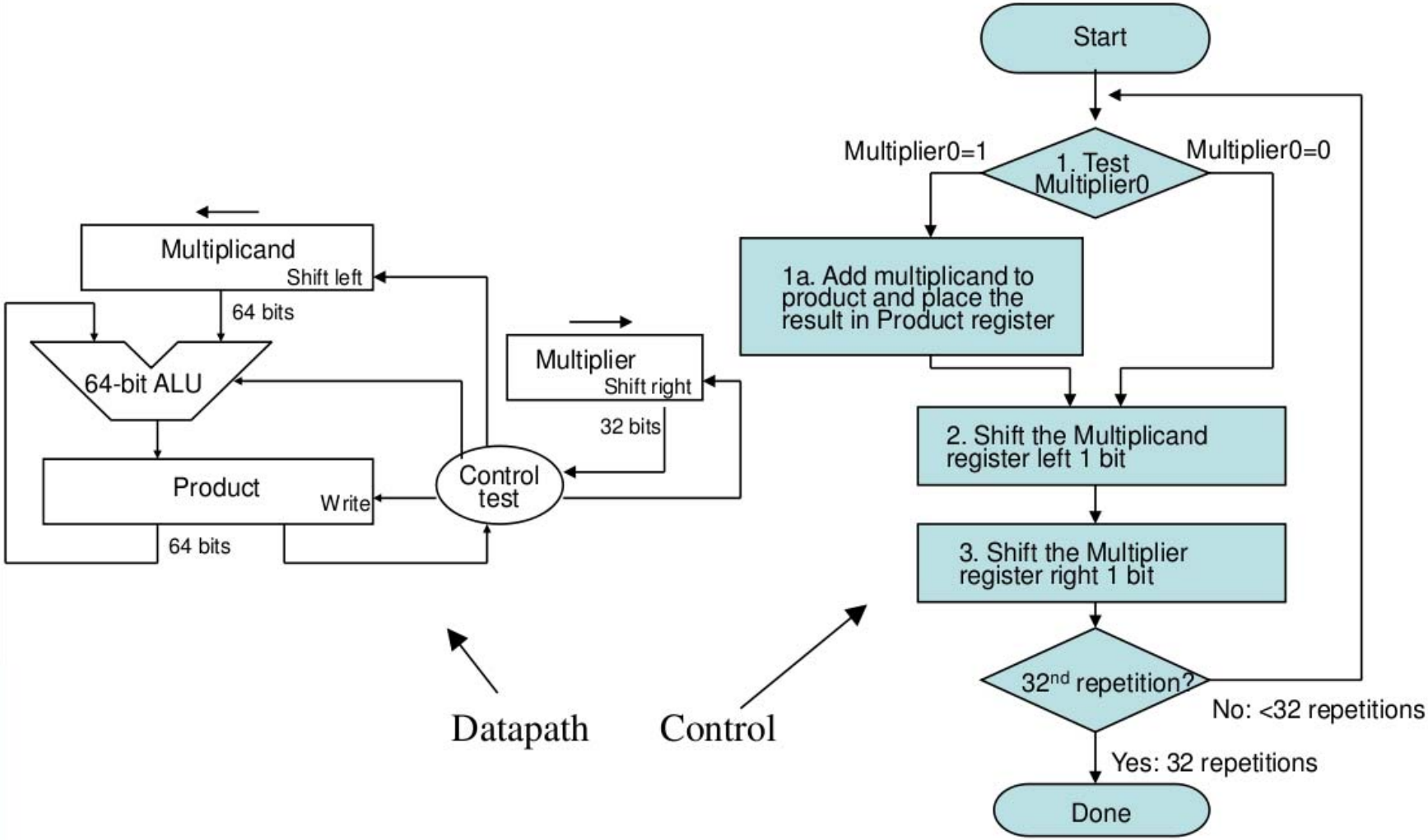
ΠΡΑΞΗ	A	B	ΑΠΟΤΕΛΕΣΜΑ
A + B	>0	>0	<0
A + B	<0	<0	>0
A – B	>0	<0	<0
A – B	<0	>0	>0

- Όταν το σύστημα εντοπίσει μια υπερχείλιση ειδοποιεί το πρόγραμμα που την προκάλεσε (interrupt / exception)
- Άλλες γλώσσες προγραμματισμού τις αγνοούν (π.χ C) ενώ άλλες απαιτούν να ειδοποιείται το πρόγραμμα και ο χρήστης (πχ. Fortran, Ada)

ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ

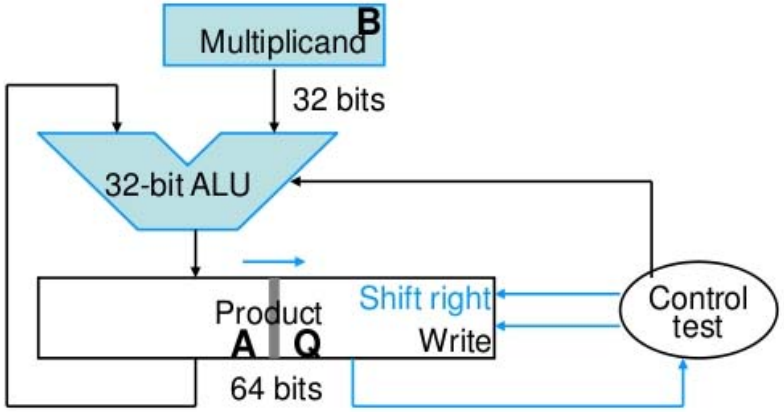
- Γενικά πιο πολύπλοκος από την πρόσθεση
- Υλοποιείται με διαδοχικές ολισθήσεις και προσθέσεις
- Απαιτεί περισσότερο χρόνο καθώς και περισσότερο υλικό
  - Αν ο Α έχει n ψηφία και ο Β m, τότε το γινόμενο θα έχει n+m
- Τί γίνεται όταν πολλαπλασιάζουμε ετερόσημους αριθμούς;

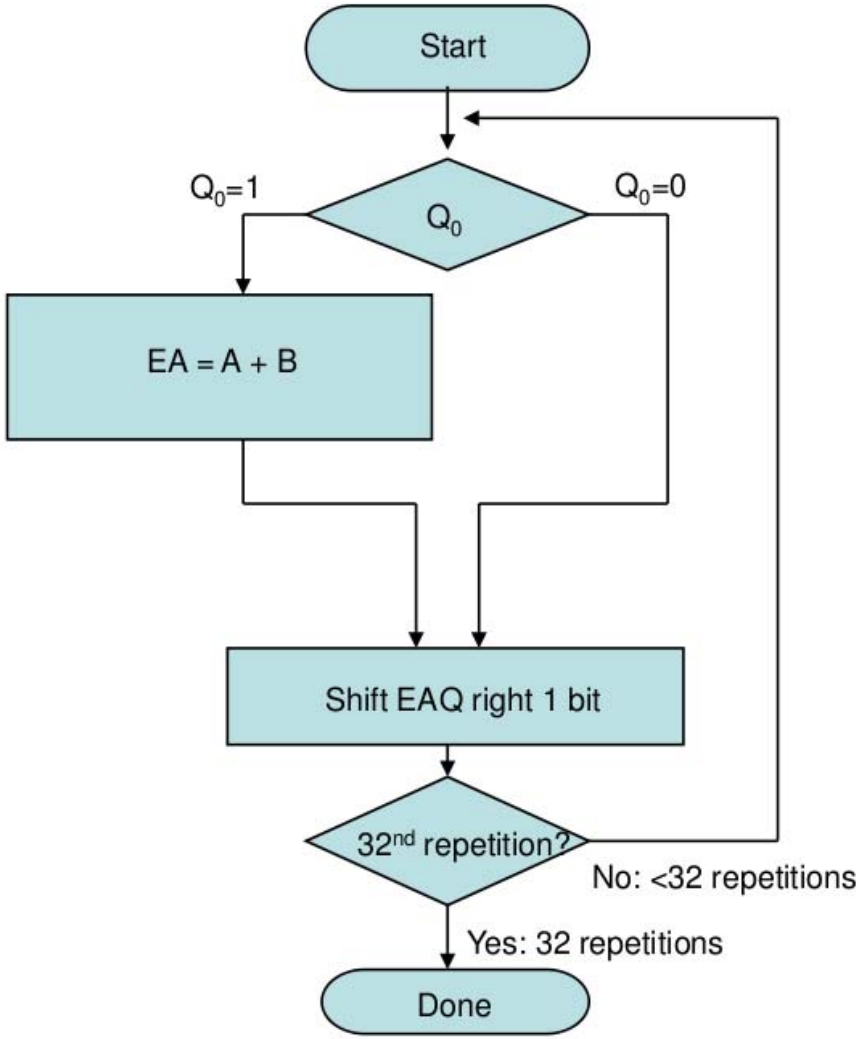
ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ



ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ

- Βελτιωμένη έκδοση
- Λιγότερο υλικό
  - Μόνο το γινόμενο 64 bits
  - Δεν χρειάζεται ο καταχωρητής του πολλαπλασιαστή
- Ολίσθηση δεξιά





ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ

```
      10111     multiplicand(23)
    × 10011     multiplier(19)
    --------
      10111
     10111
    00000
   00000
  10111
  ---------
  110110101     product(437)
```

multiplicand(B)=10111

E	A	Q	
0	00000	10011	*multiplier*
	10111		*Q0=1; add B*
0	10111		*1st partial product*
0	01011	11001	*Shift right EAQ*
	10111		*Q0=1; add B*
1	00010		*2nd partial product*
0	10001	01100	*Shift right EAQ*
0	01000	10110	*Q0=0; shift right EAQ*
0	00100	01011	*Q0=0; shift right EAQ*
	10111		*Q0=1; add B*
0	11011		*5th partial product*
0	01101	10101	*Shift right EAQ*
	final product		

ΑΛΓΟΡΙΘΜΟΣ BOOTH

- Σχεδιάστηκε από τον Andrew Booth το 1951
- Πολλαπλασιάζει 2 προσημασμένους αριθμούς σε παράσταση συμπληρώματος ως προς 2
- Στηρίζεται στην παρακάτω παρατήρηση :
  - Έστω δυαδικός αριθμός με τα bits m εώς k ίσα με 1. Τότε ο αριθμός αυτός είναι ίσος με $\mathbf{2^{k+1} - 2^{m}}$
- Επομένως μπορώ να μετατρέψω τον πολλαπλασιασμό ως εξής :

$$\mathbf{M * Multiplier = M * 2^{k+1} - M * 2^{m}}$$

ΑΛΓΟΡΙΘΜΟΣ BOOTH

- Παράδειγμα :

  Multiplier = 14 = 001110

  → k = 3, m = 1 → $2^4 - 2^1 = 16 - 2 = 14$

  → M * Multiplier = M * 2^4 - M * 2^1

  → 4 αριστερές ολισθήσεις του M – 1 αριστερή ολίσθηση του M

ΑΛΓΟΡΙΘΜΟΣ BOOTH

- Πολλαπλασιασμός B (multiplicand) * Q * (multiplier)

  - Έλεγχος του τρέχοντος και του προηγούμενου bit :

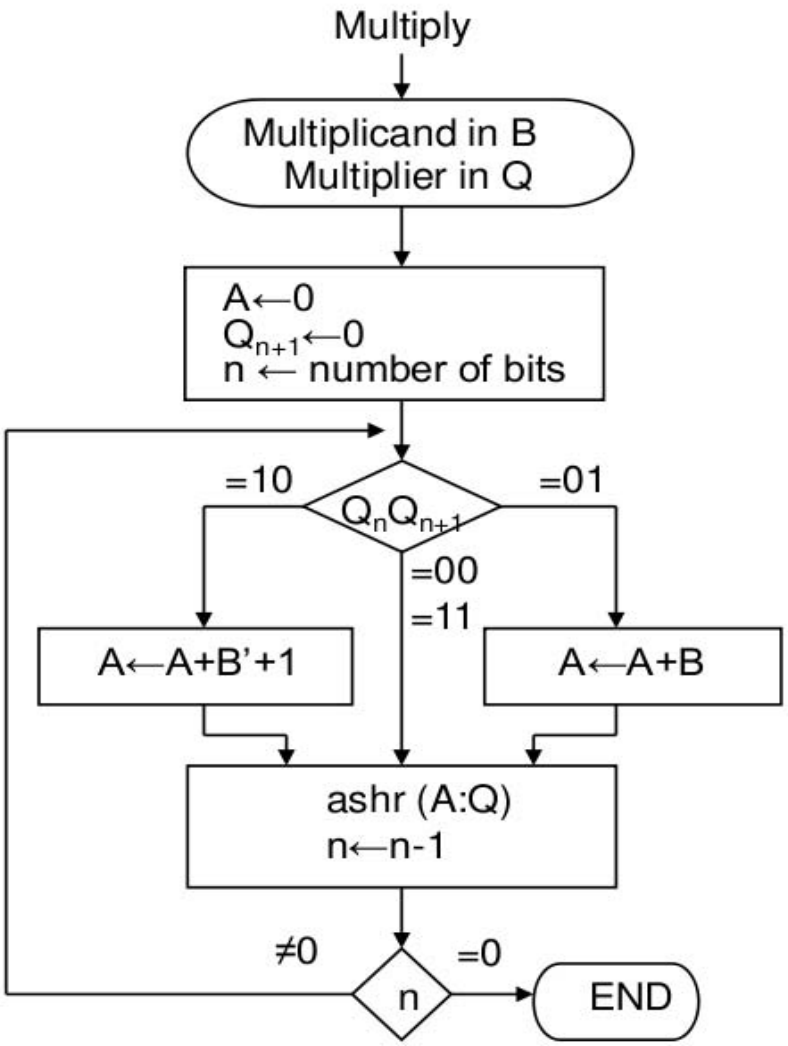
    01 : Τέλος μιας σειράς από 1 → Πρόσθεση του πολλαπλασιαστέου στο αριστερό μισό του γινομένου

    10 : Αρχή μιας σειράς από 1 → Αφαίρεση του πολλαπλασιαστέου από το αριστερό μισό του γινομένου

    00, **11** : Καμία αριθμητική πράξη

  - Αριθμητική ολίσθηση του γινομένου δεξιά κατά 1 bit και επανάληψη του αλγορίθμου (επέκταση του προσήμου για να διατηρηθεί το πρόσημο κατά τον πολλαπλασιασμό προσημασμένων αριθμών)

ΑΛΓΟΡΙΘΜΟΣ BOOTH



(-9) * (-13) = +117

10111 10011 00011 10101

B'+1=01001

Q_nQ_{n+1}	B=10111	A	Q	Q_{n+1}	n
	initial	00000	10011	0	101
1 0	subtract B	01001			
		01001			
	ashr	00100	11001	1	100
1 1	ashr	00010	01100	1	011
0 1	add B	10111			
		11001			
	ashr	11100	10110	0	010
0 0	ashr	11110	01011	0	001
1 0	subtract B	01001			
		00111			
	ashr	00011	10101	1	000

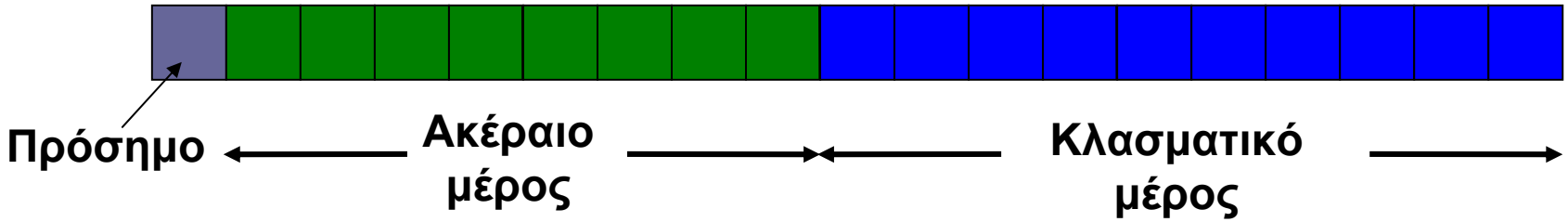
ΠΡΑΓΜΑΤΙΚΟΙ ΑΡΙΘΜΟΙ

- n διαθέσιμα bits
- Πεπερασμένος αριθμός από αριθμούς που αναπαρίστανται (όπως στους ακέραιους)
- Μπορούν να αναπαρασταθούν συγκεκριμένος αριθμός πραγματικών με συγκεκριμένη απόσταση (ακρίβεια)

- 2 προσεγγίσεις:
  - Αναπαράσταση σταθερής υποδιαστολής
  - Αναπαράσταση κινητής υποδιαστολής

ΑΝΑΠΑΡΑΣΤΑΣΗ ΣΤΑΘΕΡΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

- n διαθέσιμα bits : $n=1+ n_1 + n_2$
- MSB διατίθεται για πρόσημο
- n_1 bits διατίθενται για παράσταση ακέραιου μέρους (καθορίζουν το εύρος)
- n_2 bits διατίθενται για παράσταση κλασματικού μέρους (καθορίζουν την ακρίβεια)
- Όλοι οι αριθμοί ισαπέχουν



ΑΝΑΠΑΡΑΣΤΑΣΗ ΣΤΑΘΕΡΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

- Π.χ. 8 bits, συμπλήρωμα ως προς 1
- 4 bits ακέραιο μέρος
- Μεγαλύτερος αριθμός 0 1111 , 111 = $15{,}875_{(10)}$
- Μικρότερος αριθμός 1 0000 , 000 = $-15{,}875_{(10)}$
- Άρα παρίστανται 256 πραγματικοί αριθμοί στο [-15,875 15,875]
- Ομοιόμορφη απόσταση $2^{-3} = 0{,}001_{(2)} = 0{,}125_{(10)}$
- Σφάλμα στρογγυλοποίησης: 2^{-4}

ΔΕΝ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΓΙΑΤΙ ΕΧΕΙ ΠΟΛΥ ΜΕΓΑΛΟ ΣΧΕΤΙΚΟ ΣΦΑΛΜΑ

ΑΝΑΠΑΡΑΣΤΑΣΗ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

- Εκθετική μορφή αναπαράστασης πραγματικών αριθμών:

$$N = (-1)^{sign} \times \sigma \times 2^{e}$$

- **σ = συντελεστής (mantissa / significant)**
- **e = εκθέτης (exponent)**

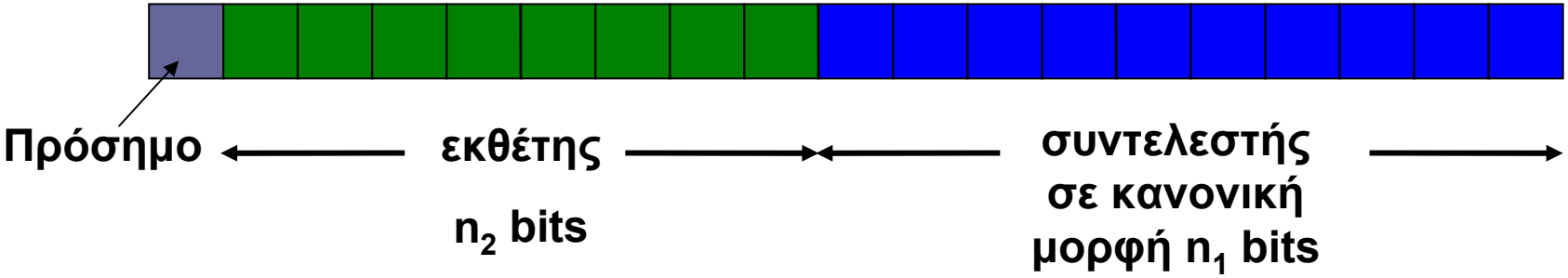
- Πολλοί τρόποι για την παράσταση του ίδιου αριθμού:

$101{,}011 = 101{,}011 \times 2^{0} = 1{,}01011 \times 2^{2} = 0{,}101011 \times 2^{3} = 101011 \times 2^{-3}$

- Κανονική μορφή (normal form):

$$1{,}xxxxxxxx \times 2^{e}$$

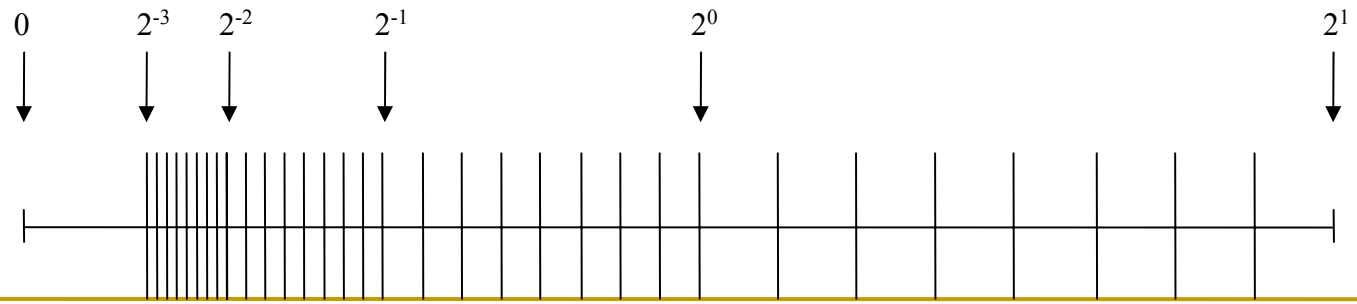
ΑΝΑΠΑΡΑΣΤΑΣΗ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ



- Εξασφαλίζει μεγάλο εύρος για την παράσταση μεγάλων αριθμών και μεγάλη ακρίβεια για τους μικρούς αριθμούς

- Απόλυτο σφάλμα στρογγυλοποίησης: $2^{-n1-1} \times 2^{e}$
- Σχετικό σφάλμα στρογγυλοποίησης: 2^{-n1-1}

- Δεν ισοκατανέμει τους αριθμούς



ΑΝΑΠΑΡΑΣΤΑΣΗ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

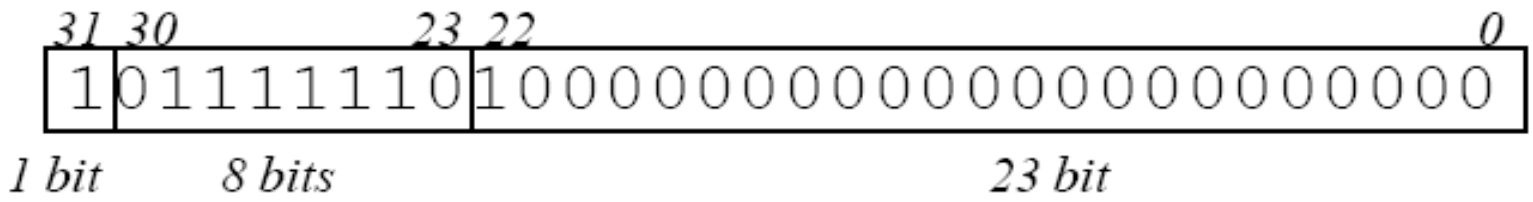
- Αύξηση των bits του συντελεστή → αύξηση της ακρίβειας
- Αύξηση των bits του εκθέτη → αύξηση του εύρους των αριθμών που μπορούν να παρασταθούν
- Υπερχείλιση → Ο εκθέτης είναι πολύ μεγάλος για να χωρέσει στο πεδίο (floating point overflow)
- Ανεπάρκεια → Ο αρνητικός εκθέτης είναι πολύ μικρός για να χωρέσει στο πεδίο (floating point underflow)
- IEEE 754 floating point standard :
  - Single precision : 8 bits εκθέτης, 23 bits συντελεστής
  - Double precision : 11 bits εκθέτης, 52 bits συντελεστής
  - Το αρχικό bit 1 των κανονικοποιημένων συντελεστών υπονοείται (είναι κρυφό) → κέρδος 1 ακόμα bit για το συντελεστή

ΑΝΑΠΑΡΑΣΤΑΣΗ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

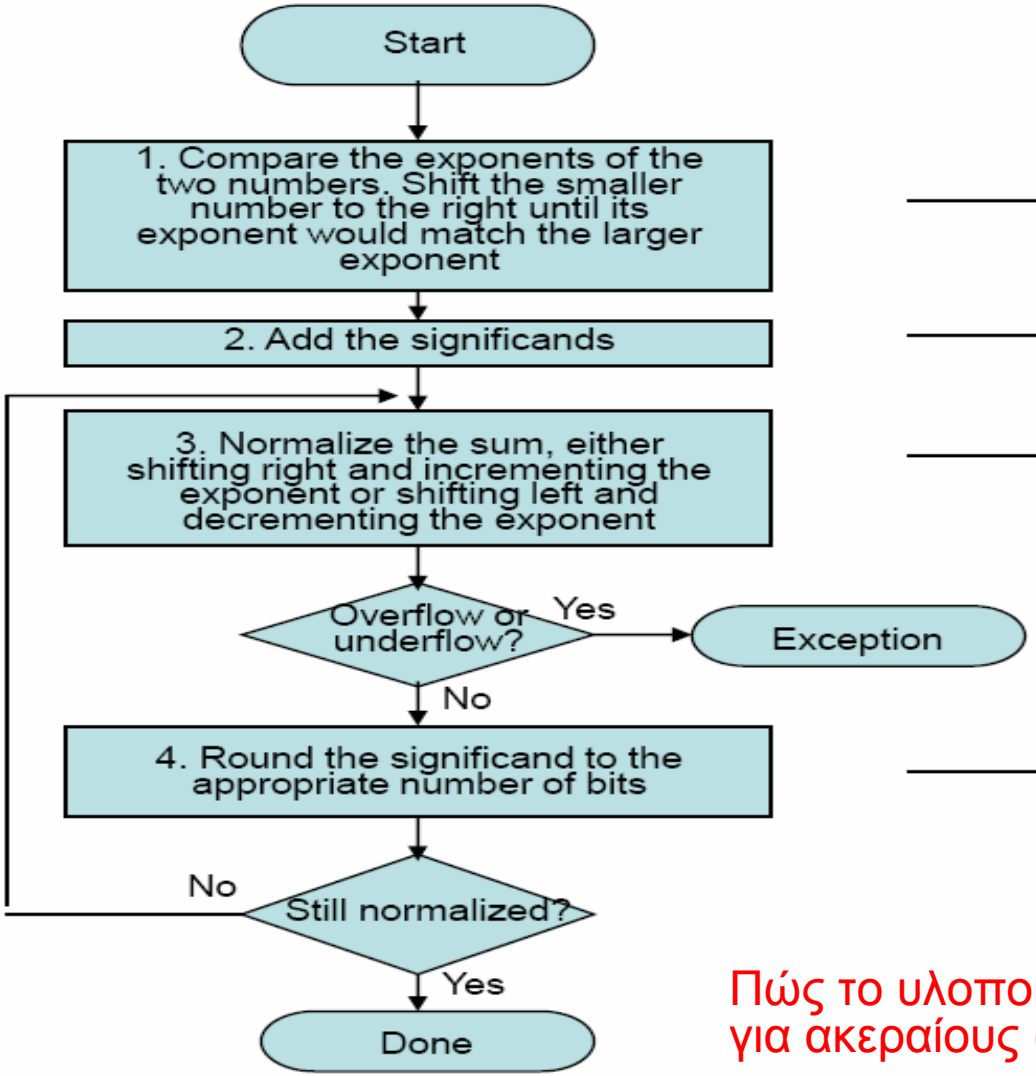
- Πόλωση εκθέτη
  - Min = 000…00
  - Max = 111..11
  - Κάνει ευκολότερη τη σύγκριση και την ταξινόμηση των εκθετών
- IEEE 754
  - πόλωση = 127 για single precision
  - πόλωση = 1023 για double precision
- $\mathbf{N = (-1)^{sign} \times (1 + \sigma) \times 2^{e\text{-}bias}}$
- $e = \sigma = 0 \rightarrow N = 0$
- $e = 0, \sigma \neq 0 \rightarrow$ N μη κανονικοποιημένος αριθμός
- $e = 255/2047, \sigma = 0 \rightarrow N = \pm\infty$
- $e = 255/2047, \sigma \neq 0 \rightarrow N = NaN$ (όχι αριθμός)

ΑΝΑΠΑΡΑΣΤΑΣΗ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

- Decimal : - 0.75 = - (½ + ¼)
- Binary : - 0.11 = - 1.1 x 2^{-1}
- Floating point :
  - Εκθέτης = 126 = 01111110
  - Συντελεστής = 1



ΠΡΟΣΘΕΣΗ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ



Assume: 4 digits of precision
Add 0.5 and -0.4375

$1.000*2^{-1}$ $-1.110*2^{-2}$

$1.000*2^{-1}$ $-0.111*2^{-1}$

$0.001*2^{-1}$

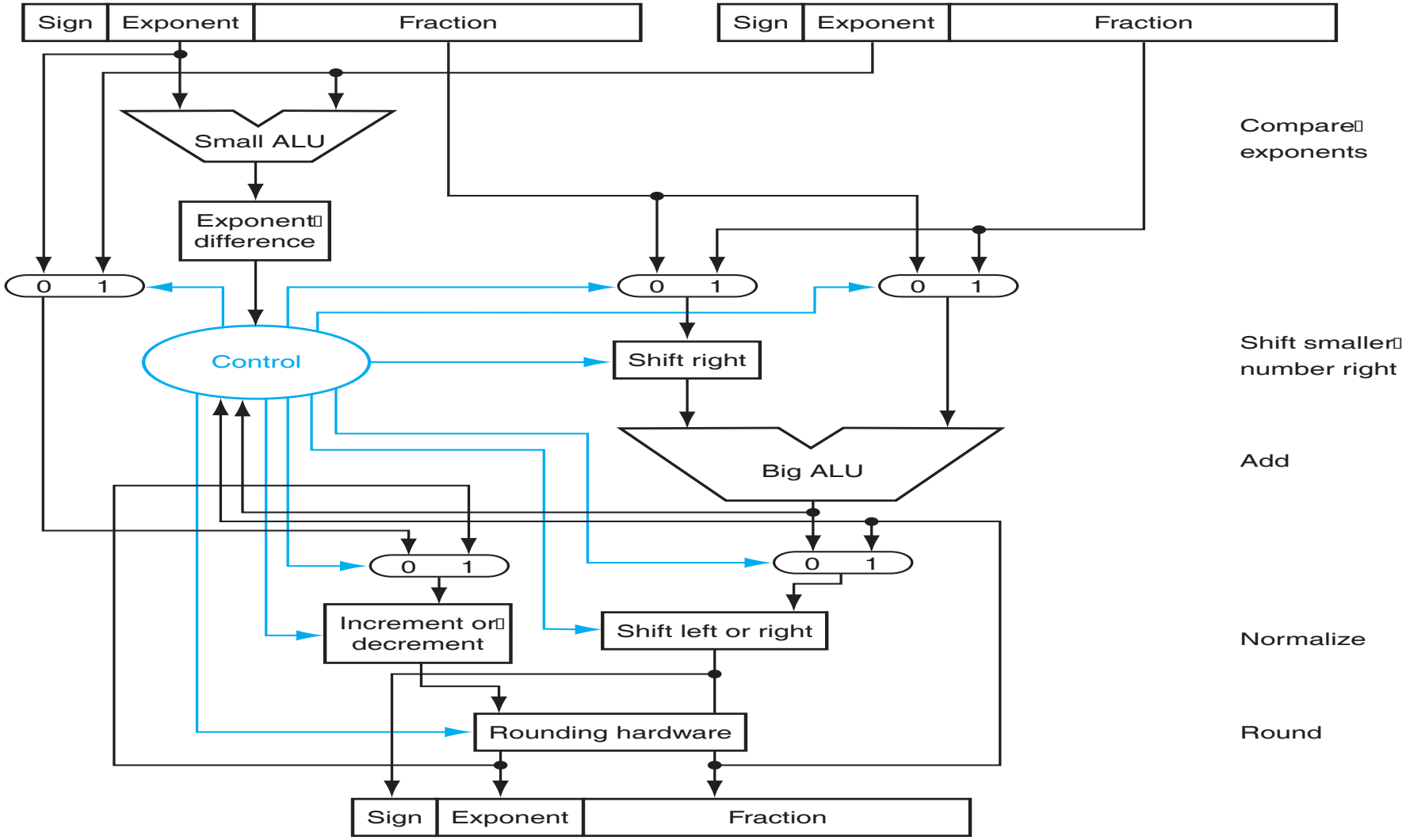
$1.000*2^{-4}$

$1.000*2^{-4}$

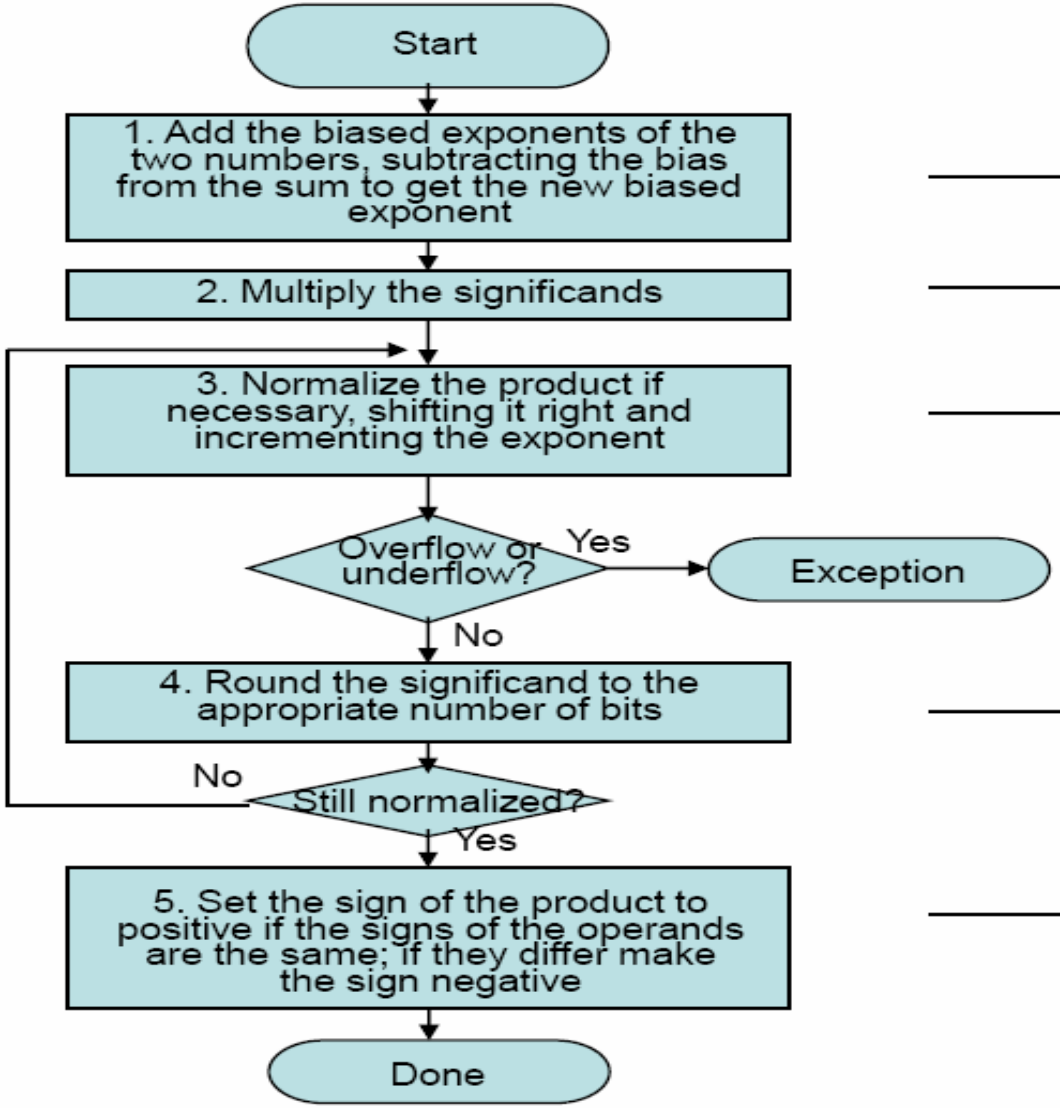
0.0625

Πώς το υλοποιούμε με την μονάδα πράξεων για ακεραίους (integer arithmetic unit) ???

ΠΡΟΣΘΕΣΗ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ



ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ



Assume: 4 digits of precision

Multiply 0.5 and -0.4375

$1.000*2^{-1}$ $-1.110*2^{-2}$

E=-3

F=1.110

$1.110*2^{-3}$

$1.110*2^{-3}$

$-1.110*2^{-3}$

-0.21875