Κεφάλαιο 3

Αριθμητική για υπολογιστές

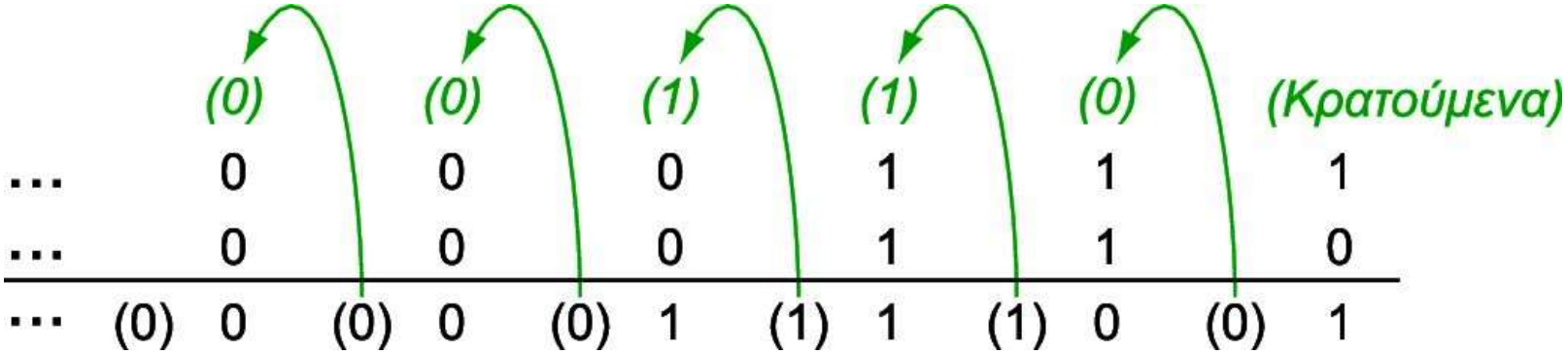
Διαφάνειες διδασκαλίας του πρωτότυπου βιβλίου μεταφρασμένες στα ελληνικά και εμπλουτισμένες (μετάφραση, επιμέλεια, προσθήκες: Δημήτρης Γκιζόπουλος, Πανεπιστήμιο Αθηνών)

Αριθμητική για υπολογιστές

- Πράξεις με ακέραιους αριθμούς
  - Πρόσθεση και αφαίρεση
  - Πολλαπλασιασμός και διαίρεση
  - Πώς αντιμετωπίζεται η υπερχείλιση
- Πραγματικοί αριθμοί κινητής υποδιαστολής
  - Αναπαράσταση και πράξεις

Πρόσθεση ακεραίων

- Παράδειγμα: 7 + 6



- Αν το αποτέλεσμα είναι εκτός εύρους, υπερχείλιση
  - Στην πρόσθεση ενός θετικού και ενός αρνητικού τελεστέου: δεν παρατηρείται υπερχείλιση
  - Στην πρόσθεση δύο θετικών τελεστέων
    - Υπερχείλιση αν το πρόσημο του αποτελέσματος είναι 1
  - Στην πρόσθεση δύο αρνητικών τελεστέων
    - Υπερχείλιση αν το πρόσημο του αποτελέσματος είναι 0

Αφαίρεση ακεραίων

- Πρόσθεση του αντιθέτου του δεύτερου τελεστέου
- Παράδειγμα: 7 – 6 = 7 + (–6)

+7:	0000 0000 … 0000 0111
–6:	1111 1111 … 1111 1010
+1:	0000 0000 … 0000 0001

- Αν το αποτέλεσμα είναι εκτός εύρους, υπερχείλιση
  - Στην αφαίρεση δύο θετικών ή δύο αρνητικών τελεστέων: δεν παρατηρείται υπερχείλιση
  - Στην πρόσθεση ενός θετικού από έναν αρνητικό τελεστέο
    - Υπερχείλιση αν το πρόσημο του αποτελέσματος είναι 0
  - Στην αφαίρεση ενός αρνητικού από έναν θετικό τελεστέο
    - Υπερχείλιση αν το πρόσημο του αποτελέσματος είναι 1

Αριθμητική για πολυμέσα

- Τα γραφικά και η επεξεργασία πολυμέσων γίνεται σε διανύσματα δεδομένων των 8 bit και των 16 bit
  - Χρήση αθροιστή 64 bit, με διαμέριση των αλυσίδων των κρατουμένων
    - Πράξεις σε διανύσματα 8×8 bit, 4×16 bit, ή 2×32 bit
  - SIMD (single-instruction, multiple-data –μία εντολή, πολλά δεδομένα)
- Λειτουργίες κορεσμού
  - Στην υπερχείλιση, το αποτέλεσμα είναι η μεγαλύτερη τιμή που μπορεί να αναπαρασταθεί
    - πρβλ. αριθμητική modulo συμπληρώματος ως προς δύο
  - π.χ. περικοπή ηχητικών αποσπασμάτων, κορεσμός σε βίντεο

Πολλαπλασιασμός

- Πολλαπλασιασμός αριθμών με το χέρι

Πολλαπλα-σιαστέος → 1000

Πολλαπλα-σιαστής → × 1001

1000

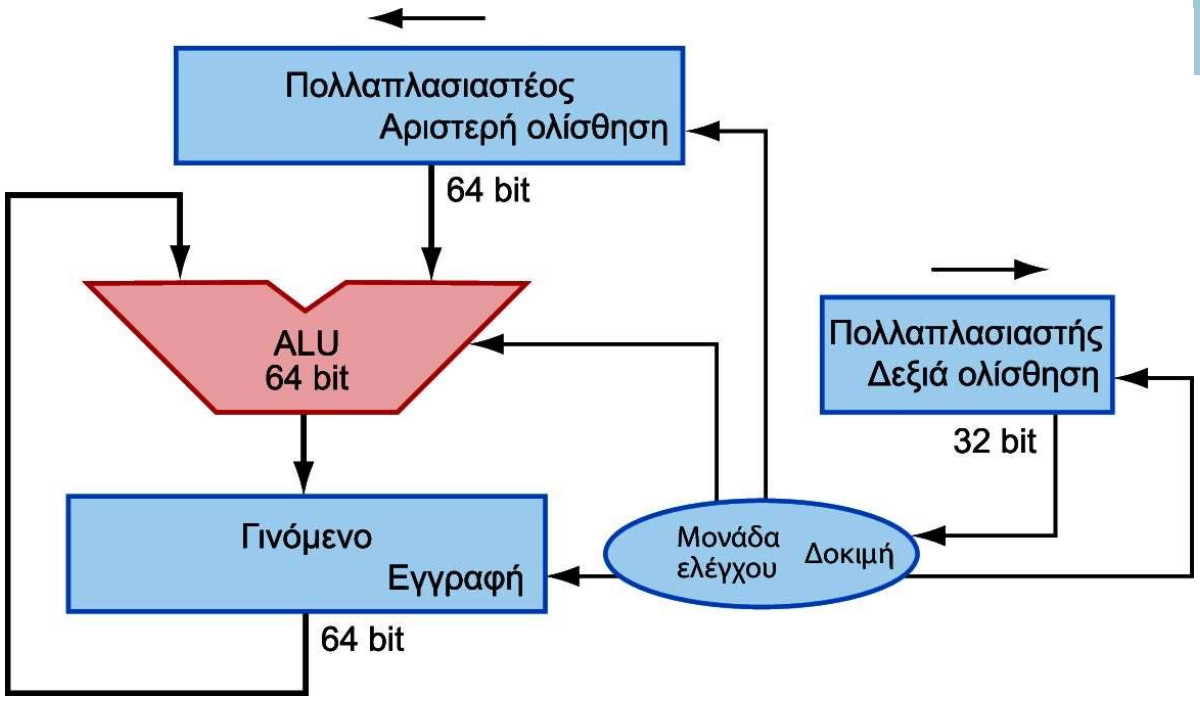
0000

0000

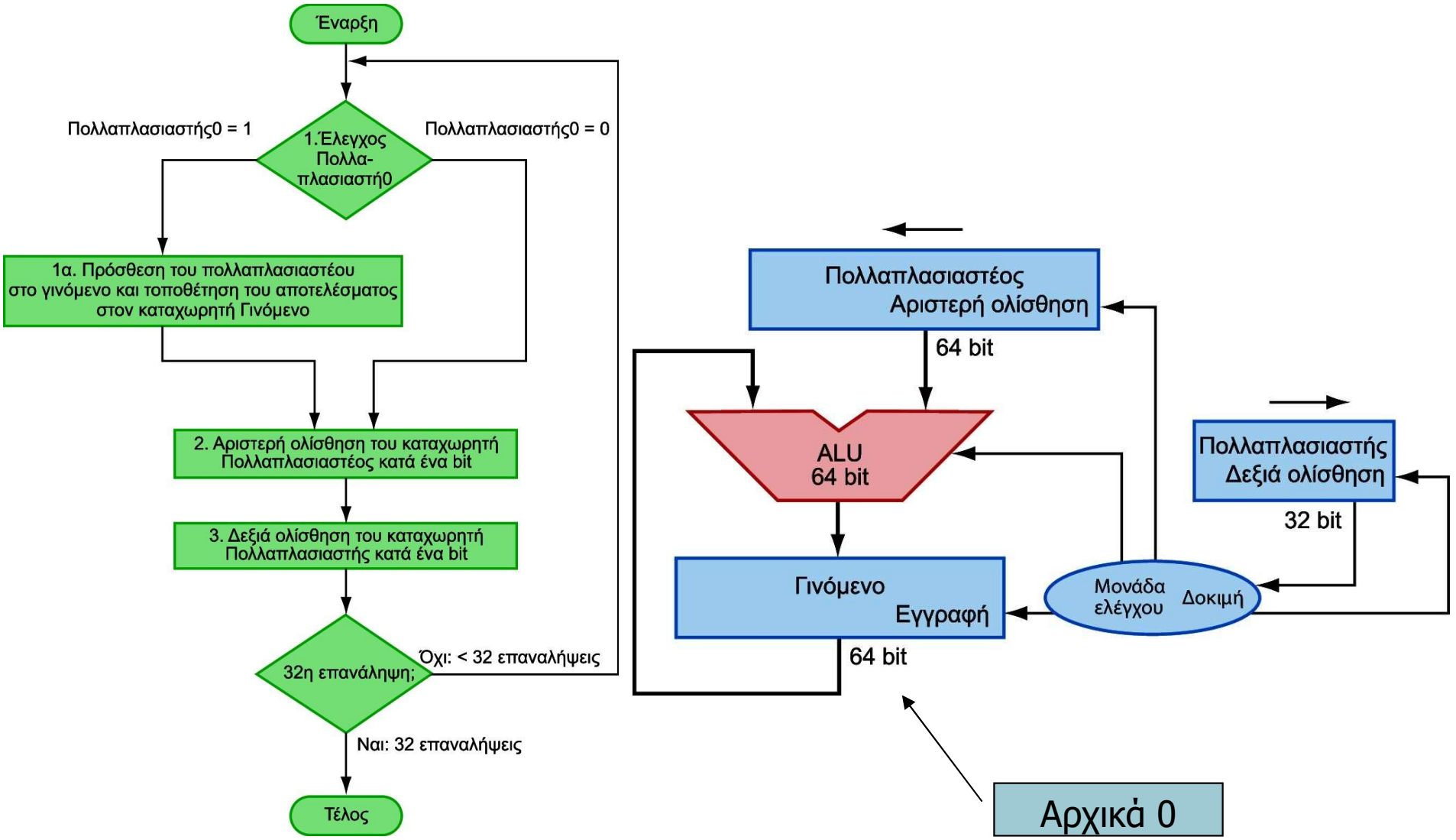
1000

Γινόμενο → 1001000

Το μήκος του γινομένου είναι το άθροισμα των μηκών των τελεστέων

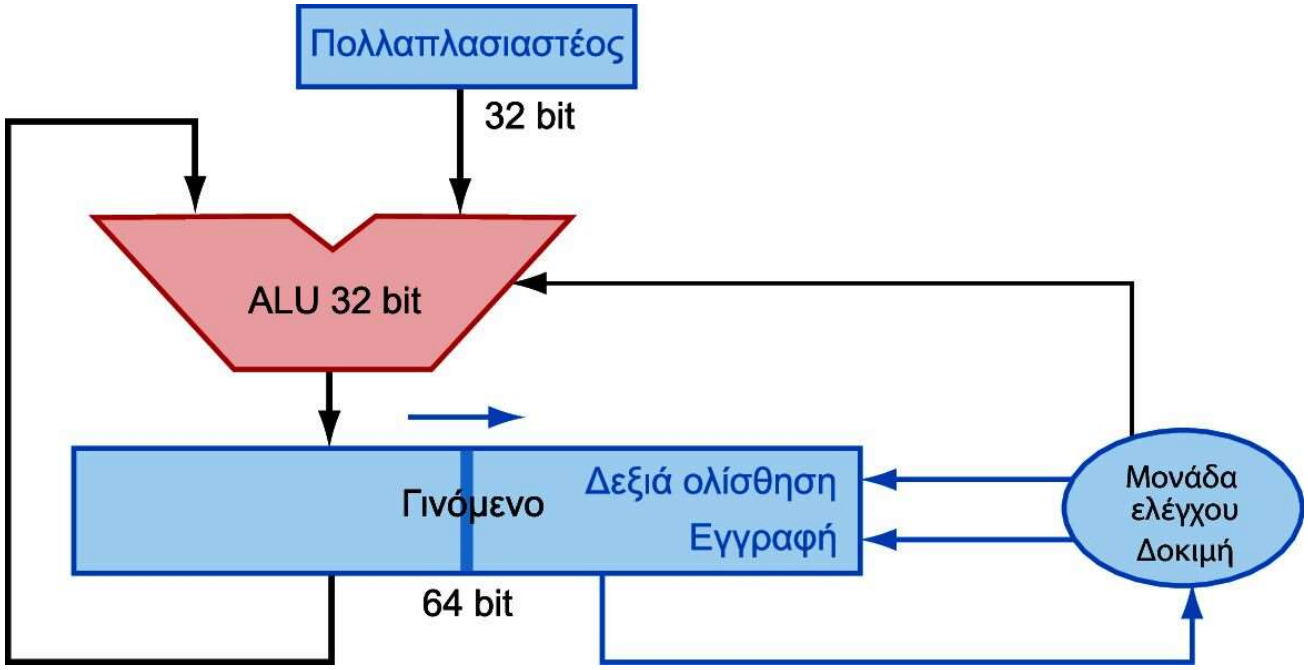


Υλικό πολλαπλασιασμού



Βελτιστοποιημένος πολλαπλασιασμός

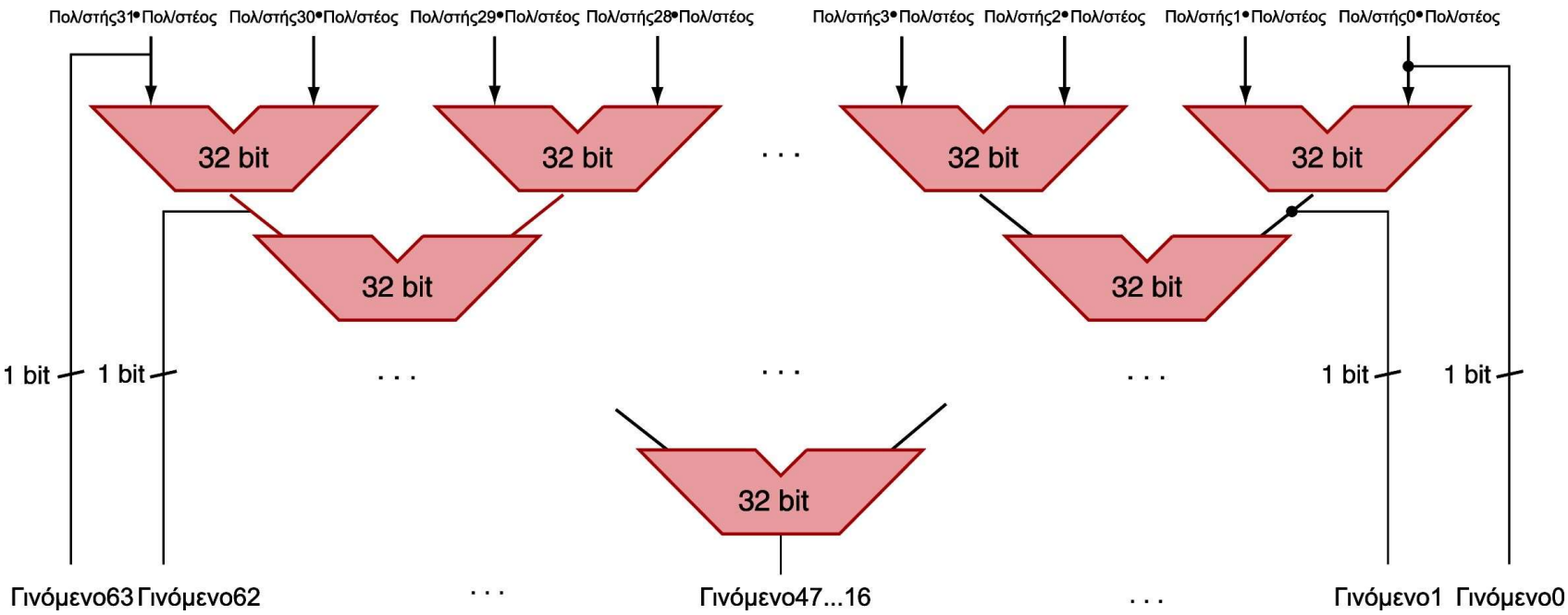
- Παράλληλη εκτέλεση των βημάτων: πρόσθεση/ολίσθηση



- Ένας κύκλος για την πρόσθεση κάθε μερικού γινομένου
  - Αποδεκτό αν γίνονται λίγες πράξεις πολλαπλασιασμού

Ταχύτερος πολλαπλασιασμός

- Χρήση περισσότερων του ενός αθροιστών
  - Συμβιβασμός ανάμεσα σε κόστος και απόδοση

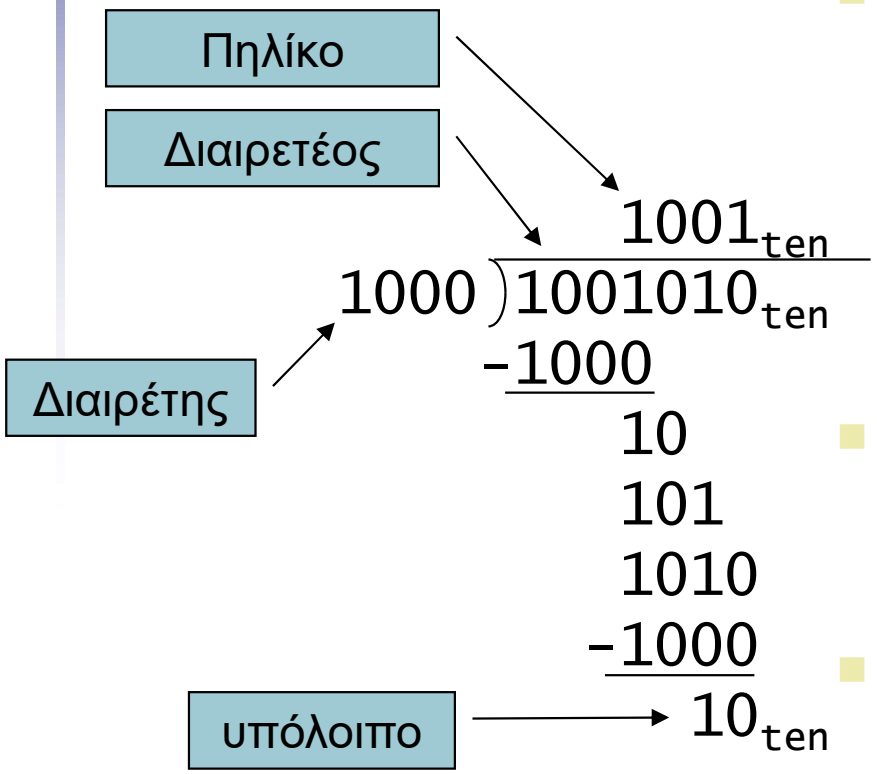


- Υλοποιείται και με διοχέτευση
  - Παράλληλη εκτέλεση πολλών πράξεων πολλαπλασιασμού

Πολλαπλασιασμός RISC-V

- Τέσσερις εντολές multiply:
  - mul: multiply
    - Δίνει τα 32 χαμηλότερα bit του γινομένου
  - mulh: multiply high
    - Δίνει τα 32 υψηλότερα bit του γινομένου, εφόσον οι τελεστέοι είναι προσημασμένοι
  - mulhu: multiply high unsigned
    - Δίνει τα 32 υψηλότερα bit του γινομένου, εφόσον οι τελεστέοι είναι μη προσημασμένοι
  - mulhsu: multiply high signed/unsigned
    - Δίνει τα 32 υψηλότερα bit του γινομένου, εφόσον ο ένας τελεστέος είναι προσημασμένος και ο άλλος μη προσημασμένος
  - Χρήση του αποτελέσματος της mulh για έλεγχο υπερχείλισης από τον πολλαπλασιασμό των 32 bit

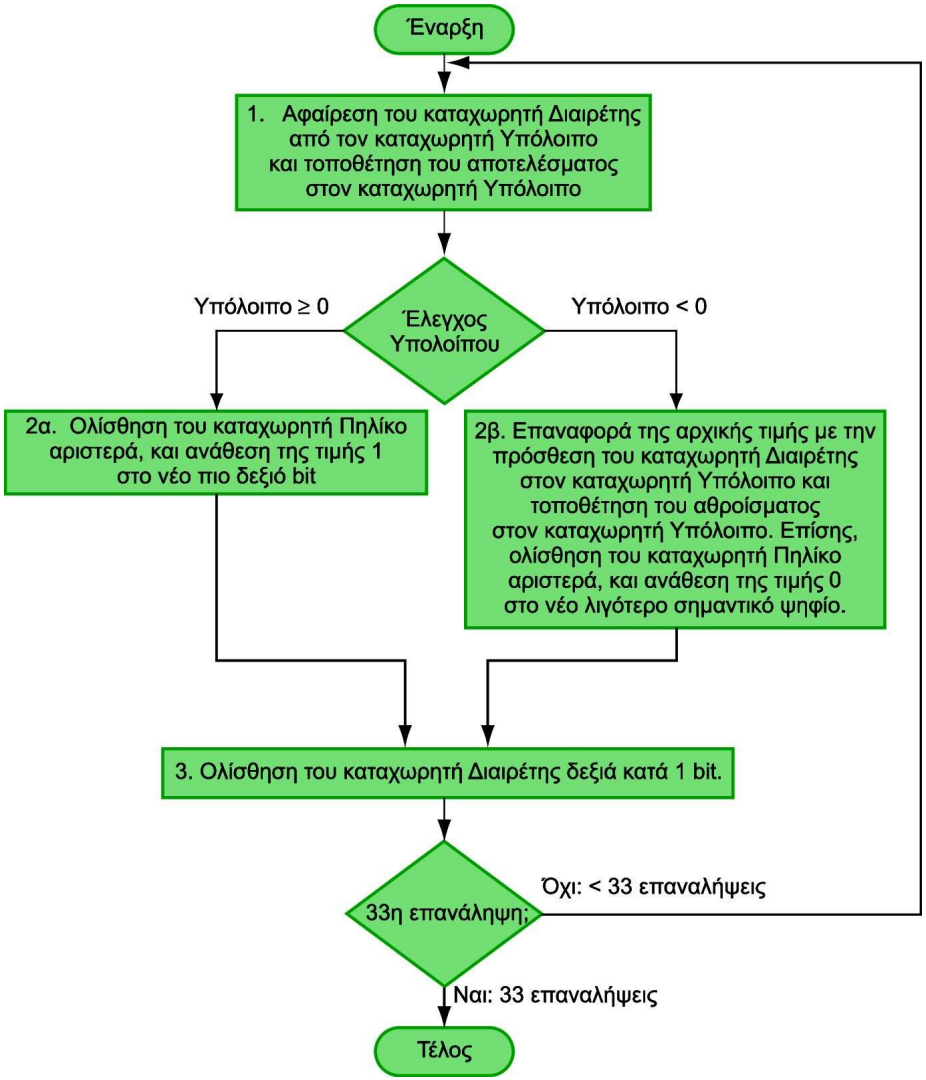
Διαίρεση

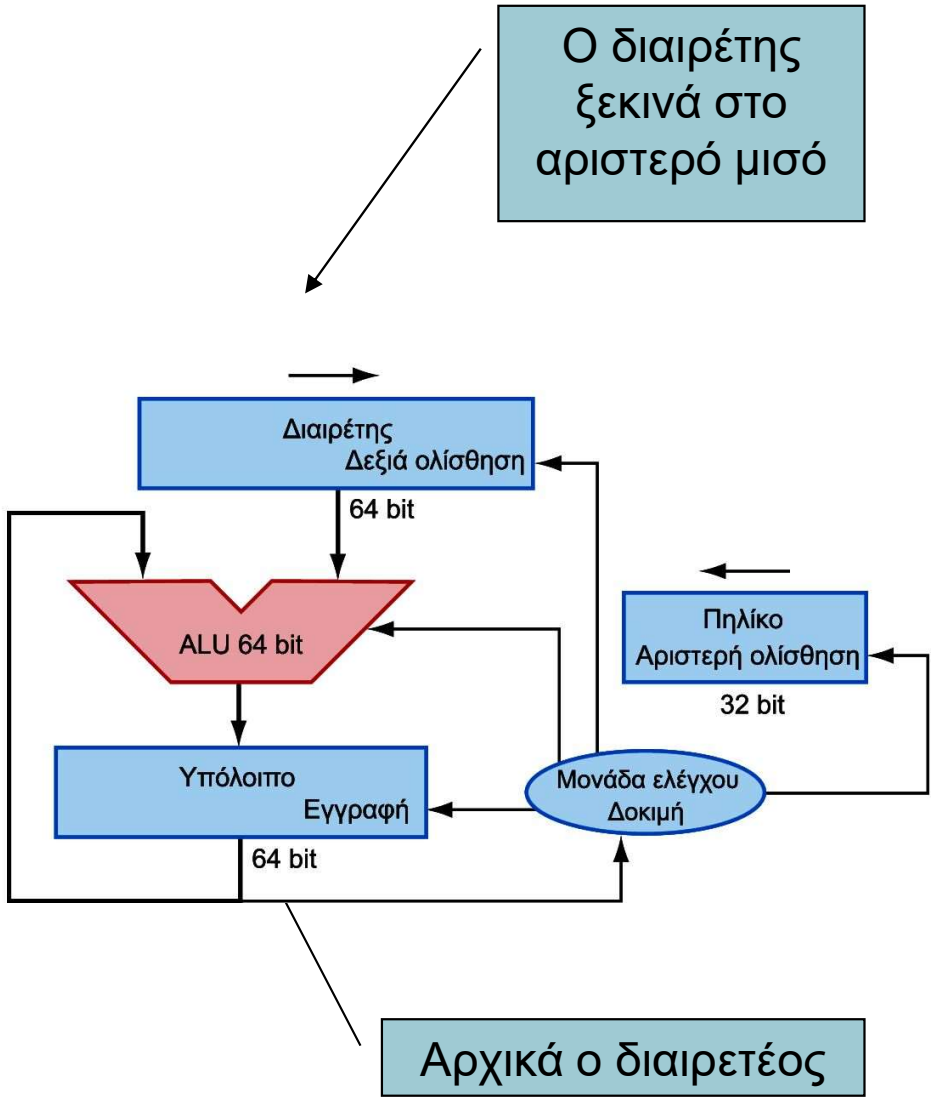


τελεστέοι των *n* bit δίνουν πηλίκο και υπόλοιπο των *n*-bit

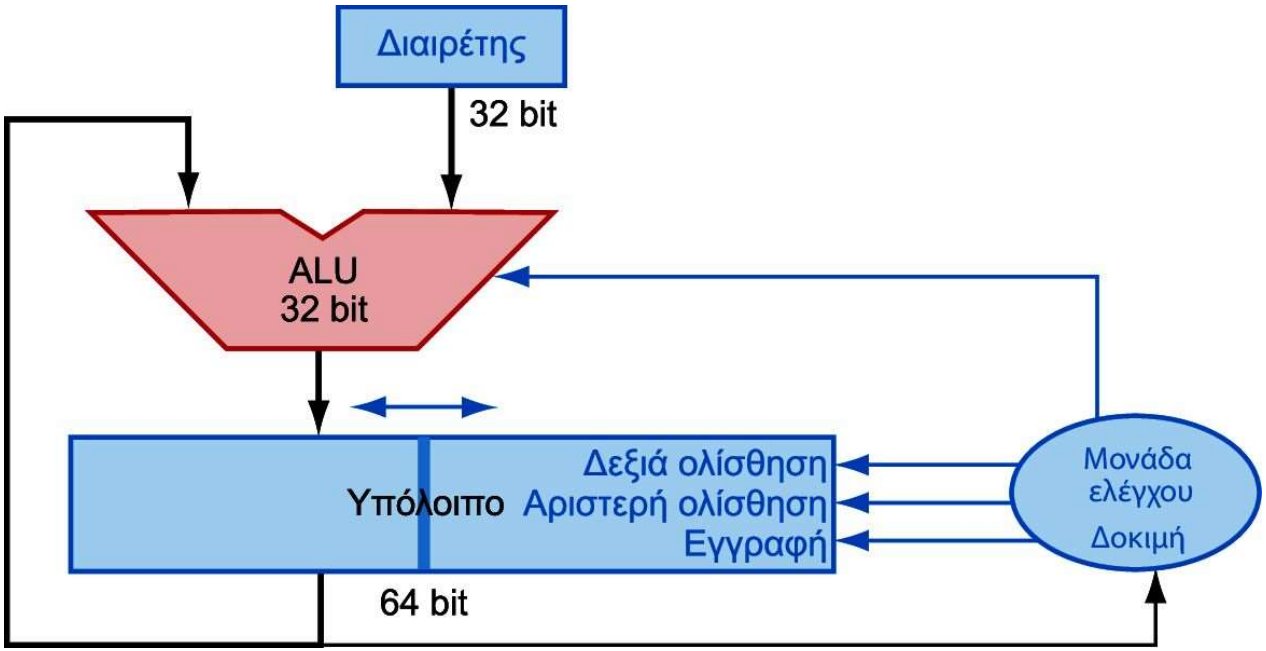
- Έλεγχος για διαίρεση με το 0
- Διαίρεση αριθμών με το χέρι
  - Αν bit διαιρέτη ≤ bit διαιρετέου
    - 1 bit στο πηλίκο, αφαίρεση
  - Ειδάλλως
    - 0 bit στο πηλίκο, κατέβασμα του επόμενου bit του διαιρετέου
- Διαίρεση με επαναφορά
  - Γίνεται η αφαίρεση· αν υπόλοιπο < 0, πρόσθεση ξανά του διαιρέτη
- Προσημασμένη διαίρεση
  - Διαίρεση με απόλυτες τιμές
  - Κατάλληλη προσαρμογή του προσήμου του πηλίκου και του υπολοίπου

Υλικό διαίρεσης





Βελτιστοποιημένη διαίρεση



- Ένας κύκλος για την αφαίρεση κάθε μερικού υπολοίπου
- Μοιάζει πολύ με πολλαπλασιασμό!
  - Μπορεί να χρησιμοποιηθεί το ίδιο υλικό τόσο για πολλαπλασιασμό όσο και για διαίρεση

Ταχύτερη διαίρεση

- Δεν μπορεί να χρησιμοποιηθεί υλικό παράλληλης εκτέλεσης όπως στον πολλαπλασιασμό
  - Η αφαίρεση εξαρτάται από το πρόσημο του υπολοίπου
- Στις ταχύτερες τεχνικές διαίρεσης (π.χ. διαίρεση SRT) παράγονται περισσότερα από ένα bit του πηλίκου σε κάθε βήμα
  - Αλλά, και πάλι, απαιτούνται περισσότερα από ένα βήματα

Διαίρεση RISC-V

- Τέσσερις εντολές:
  - div, rem: διαίρεση προσημασμένων ακεραίων με υπόλοιπο
  - divu, remu: διαίρεση μη προσημασμένων ακεραίων με υπόλοιπο
- Η υπερχείλιση και η διαίρεση με το μηδέν δεν προκαλούν σφάλμα
  - Επιστρέφουν καθορισμένα αποτελέσματα
  - Ταχύτερη εκτέλεση για την πιο κοινή (συνηθισμένη) περίπτωση της διαίρεσης χωρίς σφάλμα

Κινητή υποδιαστολή

- Αναπαράσταση αριθμών με κλασματικά μέρη
  - Τόσο πολύ μικρών όσο και πολύ μεγάλων αριθμών
- Μοιάζει με την επιστημονική σημειογραφία
  - -2.34×10^{56} ← κανονικοποιημένος
  - $+0.002 \times 10^{-4}$ ← μη κανονικοποιημένοι
  - $+987.02 \times 10^{9}$ ← μη κανονικοποιημένοι
- Σε δυαδική αναπαράσταση
  - $\pm 1.xxxxxxx_2 \times 2^{yyyy}$
- Οι τύποι `float` και `double` στη γλώσσα C

Το πρότυπο της κινητής υποδιαστολής

- Ορίστηκε στο πρότυπο 754-1985 της IEEE
- Αναπτύχθηκε λόγω της διαφοροποίησης των αναπαραστάσεων
  - Ζητήματα φορητότητας του κώδικα επιστημονικών εφαρμογών
- Πλέον, σχεδόν καθολικά καθιερωμένο
- Δύο αναπαραστάσεις
  - Απλής ακρίβειας (32 bit)
  - Διπλής ακρίβειας (64 bit)

Μορφή κινητής υποδιαστολής κατά IEEE

απλή ακρίβεια: 8 bit απλή ακρίβεια: 23 bit
διπλή ακρίβεια: 11 bit διπλή ακρίβεια: 52 bit

S	Εκθέτης	Κλάσμα

$$(-1)^{\text{πρόσημο}} \times (1 + \text{κλάσμα}) \times 2^{(\text{εκθέτης} - \text{πόλωση})}$$

- S: bit προσήμου (0 ⇒ μη αρνητικό, 1 ⇒ αρνητικό)
- Κανονικοποίηση σημαντικού: 1.0 ≤ |σημαντικό| < 2.0
  - Το αρχικό 1 bit υπονοείται πάντα στο σημαντικό, οπότε δεν είναι ανάγκη να αναπαρασταθεί ρητά (κρυφό bit)
  - Το σημαντικό είναι το κλάσμα αφού γίνει η επαναφορά του “1.”
- Εκθέτης: αναπαράσταση με μορφή υπέρβασης: ο πραγματικός εκθέτης + πόλωση
  - Διασφαλίζει ότι ο εκθέτης είναι μη προσημασμένος
  - απλή ακρίβεια: πόλωση = 127, διπλή ακρίβεια: πόλωση = 1203

Εύρος τιμών απλής ακρίβειας

- Οι εκθέτες 00000000 και 11111111 είναι δεσμευμένοι
- Η μικρότερη τιμή
  - Εκθέτης: 00000001
    $\Rightarrow$ πραγματικός εκθέτης = 1 – 127 = -126
  - Κλάσμα: 000…00 $\Rightarrow$ σημαντικό = 1.0
  - $\pm 1.0 \times 2^{-126} \approx \pm 1.2 \times 10^{-38}$
- Η μεγαλύτερη τιμή
  - Εκθέτης: 11111110
    $\Rightarrow$ πραγματικός εκθέτης = 254 – 127 = +127
  - Κλάσμα: 111…11 $\Rightarrow$ σημαντικό $\approx$ 2.0
  - $\pm 2.0 \times 2^{+127} \approx \pm 3.4 \times 10^{+38}$

Εύρος τιμών διπλής ακρίβειας

- Οι εκθέτες 0000…00 και 1111…1111 είναι δεσμευμένοι
- Η μικρότερη τιμή
  - Εκθέτης: 00000000001
    $\Rightarrow$ πραγματικός εκθέτης = 1 – 1023 = -1022
  - Κλάσμα: 000…00 $\Rightarrow$ σημαντικό = 1.0
  - $\pm 1.0 \times 2^{-1022} \approx \pm 2.2 \times 10^{-308}$
- Η μεγαλύτερη τιμή
  - Εκθέτης: 11111111110
    $\Rightarrow$ πραγματικός εκθέτης = 2046 – 1023 = +1023
  - Κλάσμα: 111…11 $\Rightarrow$ σημαντικό $\approx$ 2.0
  - $\pm 2.0 \times 2^{+1023} \approx \pm 1.8 \times 10^{+308}$

Ακρίβεια τιμών κινητής υποδιαστολής

- Σχετική ακρίβεια
  - Όλα τα κλασματικά bit είναι σημαντικά
  - Απλής ακρίβειας: ~ 2^{-23}
    - Ισοδυναμούν με $23 \times \log_{10}2 \approx 23 \times 0.3 \approx$ ακρίβεια 6 δεκαδικών ψηφίων
  - Διπλής ακρίβειας: ~ 2^{-52}
    - Ισοδυναμούν με $52 \times \log_{10}2 \approx 52 \times 0.3 \approx$ ακρίβεια 16 δεκαδικών ψηφίων

Ένα παράδειγμα με αριθμούς κινητής υποδιαστολής

- Να αναπαρασταθεί ο –0.75
  - $-0.75 = (-1)^1 \times 1.1_2 \times 2^{-1}$
  - S = 1
  - κλάσμα = $1000\ldots00_2$
  - Εκθέτης = –1 + πόλωση
    - απλή ακρίβεια: –1 + 127 = 126 = 01111110_2
    - διπλή ακρίβεια: –1 + 1023 = 1022 = 01111111110_2
- απλή ακρίβεια: 10111111101000…00
- διπλή ακρίβεια: 10111111111101000…00

Ένα παράδειγμα με αριθμούς κινητής υποδιαστολής

- Ποιος δεκαδικός αριθμός αναπαρίσταται από τον παρακάτω αριθμό κινητής υποδιαστολής απλής ακρίβειας;

  1100000010100 0…00
  - S = 1
  - κλάσμα = $01000\ldots00_2$
  - Εκθέτης = $10000001_2 = 129$
- $x = (-1)^1 \times (1 + 01_2) \times 2^{(129 - 127)}$

  $= (-1) \times 1.25 \times 2^2$

  $= -5.0$

Μη κανονικοποιημένοι αριθμοί

- Εκθέτης = 000...0 $\Rightarrow$ το κρυφό bit είναι το 0

$$(-1)^{\text{πρόσημο}} \times (0 + \text{κλάσμα}) \times 2^{-\text{πόλωση}}$$

- Μικρότεροι από τους κανονικούς αριθμούς
  - Επιτρέπεται η βαθμιαία ανεπάρκεια, με φθίνουσα ακρίβεια
- Μη κανονικοποιημένοι με κλάσμα = 000...0

$$(-1)^{\text{πρόσημο}} \times (0 + 0) \times 2^{-\text{πόλωση}} = \pm 0.0$$

Δύο αναπαραστάσεις του 0.0!

Θετικό/αρνητικό άπειρο και NaN

- Εκθέτης = 111...1, Κλάσμα = 000...0
  - ±άπειρο
  - Μπορεί να χρησιμοποιηθεί σε επόμενους υπολογισμούς, οπότε δεν υπάρχει ανάγκη για έλεγχο υπερχείλισης
- Εκθέτης = 111...1, Κλάσμα ≠ 000...0
  - Σύμβολο Not-a-Number (NaN)
  - Υποδεικνύει μη έγκυρο ή μη ορισμένο αποτέλεσμα
    - π.χ. 0.0 / 0.0
  - Μπορεί να χρησιμοποιηθεί σε επόμενους υπολογισμούς

Εύρος & ακρίβεια κινητής υποδιαστολής

- Συζητούμε για το εύρος αναπαραστάσεων και την ακρίβεια της αναπαράστασης κινητής υποδιαστολής με βάση το πρότυπο IEEE 754.
- Συγκρίνουμε με τους ακεραίους.

Εύρος αναπαράστασης (1)

- Στα 32 bit μπορούμε να αναπαραστήσουμε ακεραίους (προσημασμένους και απρόσημους) και πραγματικούς απλής ακρίβειας.

Αναπαράσταση	**Εύρος τιμών**
Απρόσημοι ακέραιοι	0 ... +4 294 967 295 *4 δισεκατομμύρια +*
Προσημασμένοι ακέραιοι (συμπλήρωμα ως προς 2)	–2 147 483 648 ... +2 147 483 647 *2 δισεκατομμύρια +*
Πραγματικοί (απλή ακρίβεια) – μέγιστος	± 340 282 346 638 528 859 811 704 183 484 516 925 440 *340 ενδεκάκις εκατομμύρια ...*
Πραγματικοί (απλή ακρίβεια) – ελάχιστος	0.00000000000000000000000000000000000000023509885615147285834557659820715330266457179855179808553659262368500061299303460771170648513361811637878417968 75

Εύρος αναπαράστασης (2)

- Στα 32 bit μπορούμε να αναπαραστήσουμε ακεραίους (προσημασμένους και απρόσημους) και πραγματικούς απλής ακρίβειας. Στα 64 bit πραγματικούς αριθμούς διπλής ακρίβειας

Αναπαράσταση	**Εύρος τιμών**
Απλή ακρίβεια – μέγιστος	± 340 282 346 638 528 859 811 704 183 484 516 925 440 *340 ενδεκάκις εκατομμύρια ...*
Απλή ακρίβεια – ελάχιστος	0.00000000000000000000000000000000000023509885615147285834557659820715330266457179855179808553659262368500061299303460771170648513361811637878417968 75
Διπλή ακρίβεια – μέγιστος	$1.797693134862315708145274237 32 \times 10^{308}$
Διπλή ακρίβεια – ελάχιστος	$4.450147717014402272114819593 42 \times 10^{-308}$

Απόσταση μεταξύ αριθμών (1)

- Στην αναπαράσταση κινητής υποδιαστολής απλής ή διπλής ακρίβειας, η απόσταση μεταξύ διαδοχικών πραγματικών αριθμών που αναπαρίστανται εξαρτάται από τον εκθέτη τους. Όσο μεγαλύτερος ο εκθέτης τόσο μεγαλύτερη η απόσταση μεταξύ διαδοχικών αριθμών.

Αναπαράσταση κινητής υποδιαστολής	**Αριθμός**
0 11000011 10000000000000000000**00**	**442 721 857 769 029 238 784**
S=1 (+) Exponent = 195 – 127 (68), Fraction = 0.1_2 (=0.5_{10}), Significant = 1.1_2 (=1.5_{10})	*442+ πεντάκις εκατομ.*
0 11000011 10000000000000000000**01**	**442 721 892 953 401 327 616**
S=1 (+) Exponent = 195 – 127 (68), Fraction = $0.10000000000000000000001_2$ (=$0.50000011920928 96_{10}$), Significant = $1.\ 10000000000000000000001_2$, (=$1.50000011920928 96_{10}$)	*«απόσταση» 35 τρις! ↓*
Διαφορά (απόσταση)	**+35 184 372 088 832**

Απόσταση μεταξύ αριθμών (2)

- Ας μικρύνουμε τον εκθέτη.

Αναπαράσταση κινητής υποδιαστολής	**Αριθμός**
0 10011111 10000000000000000000000	6 442 450 944
S=1 (+) Exponent = 159– 127 (32), Fraction = 0.1_2 (=0.5_{10}), Significant = 1.1_2 (=1.5_{10})	
0 10011111 10000000000000000000001	6 442 451 456
S=1 (+) Exponent = 159 – 127 (32), Fraction = $0.10000000000000000000001_2$ (=$0.50000011920928 96_{10}$), Significant = 1. $10000000000000000000001_2$, (=$1.50000011920928 96_{10}$)	*«απόσταση» μόνο λίγες εκατοντάδες* ↓
Διαφορά (απόσταση)	+512
0 10011111 10000000000000000000010	6 442 451 968
	+512
	…

Απόσταση μεταξύ αριθμών (3)

- Ας μικρύνουμε κι άλλο τον εκθέτη.

Αναπαράσταση κινητής υποδιαστολής	Αριθμός
0 11000011 10000000000000000000**00**	**442 721 857 769 029 238 784**
S=1 (+) Exponent = 195 – 127 (68), Fraction = 0.1_2 (=0.5_{10}), Significant = 1.1_2 (=1.5_{10})	*442+ πεντάκις εκατομ.*
0 11000011 10000000000000000000**01**	**442 721 892 953 401 327 616**
S=1 (+) Exponent = 195 – 127 (68), Fraction = $0.10000000000000000000001_2$ (=$0.50000011920928 96_{10}$), Significant = $1.\ 10000000000000000000001_2$, (=$1.50000011920928 96_{10}$)	*«απόσταση» 35 τρις! ↓*
Διαφορά (απόσταση)	**+35 184 372 088 832**

Πρόσθεση κινητής υποδιαστολής

- Θεωρήστε, για παράδειγμα, τους 4ψήφιους δεκαδικούς
  - $9.999 \times 10^{1} + 1.610 \times 10^{-1}$
- 1. Ευθυγράμμιση υποδιαστολών δεκαδικών μερών
  - Ολίσθηση του αριθμού με τον μικρότερο εκθέτη
  - $9.999 \times 10^{1} + 0.016 \times 10^{1}$
- 2. Πρόσθεση των σημαντικών
  - $9.999 \times 10^{1} + 0.016 \times 10^{1} = 10.015 \times 10^{1}$
- 3. Κανονικοποίηση του αποτελέσματος και έλεγχος για υπερχείλιση ή ανεπάρκεια
  - 1.0015×10^{2}
- 4. Στρογγυλοποίηση και, αν είναι απαραίτητο, εκ νέου κανονικοποίηση
  - 1.002×10^{2}

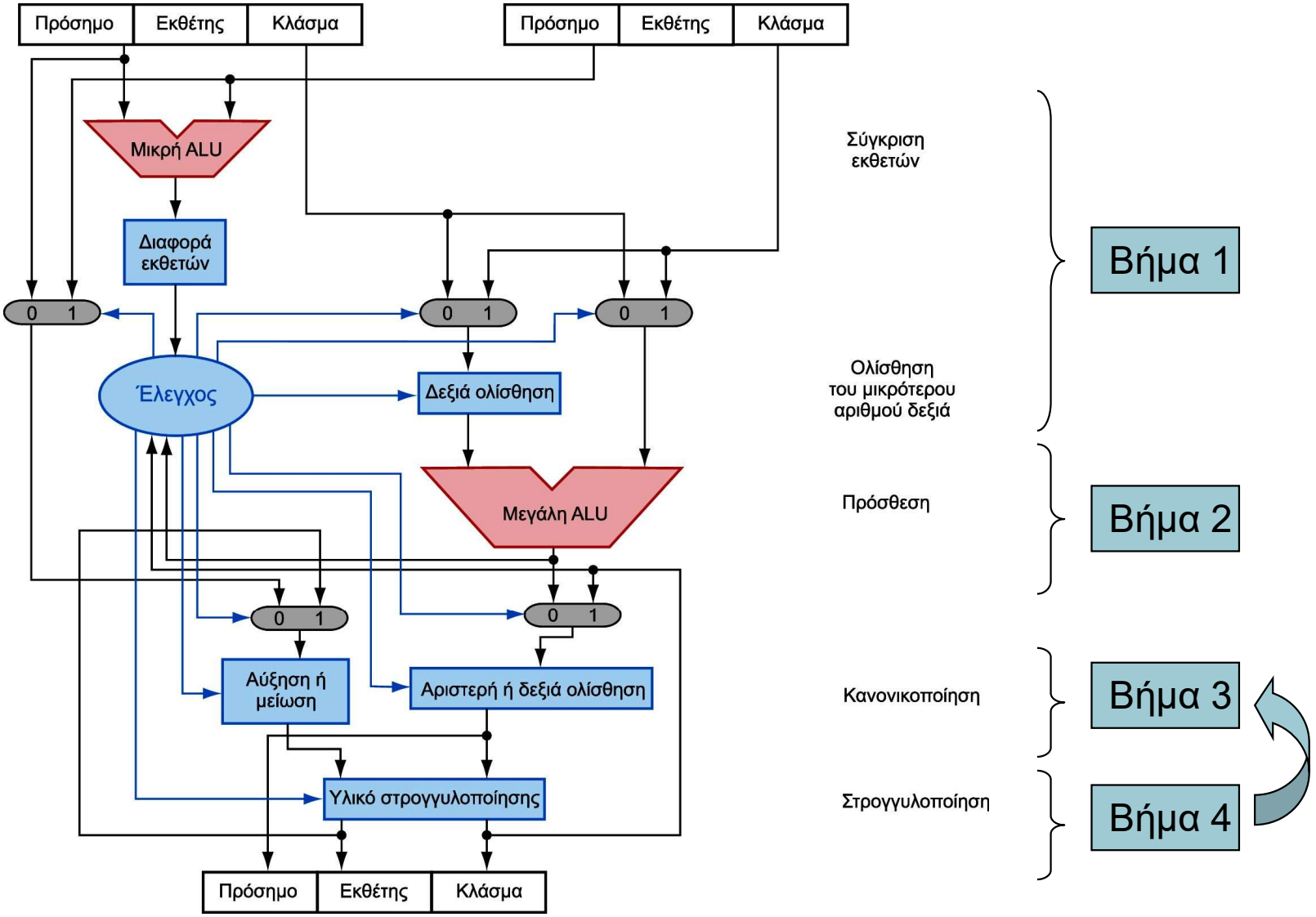
Πρόσθεση κινητής υποδιαστολής

- Τώρα θεωρήστε, για παράδειγμα, τους 4ψήφιους δυαδικούς
  - $1.000_2 \times 2^{-1} + -1.110_2 \times 2^{-2}$ (0.5 + –0.4375)
- 1. Ευθυγράμμιση υποδιαστολών δυαδικών μερών
  - Ολίσθηση του αριθμού με τον μικρότερο εκθέτη
  - $1.000_2 \times 2^{-1} + -0.111_2 \times 2^{-1}$
- 2. Πρόσθεση των σημαντικών
  - $1.000_2 \times 2^{-1} + -0.111_2 \times 2^{-}1 = 0.001_2 \times 2^{-1}$
- 3. Κανονικοποίηση του αποτελέσματος και έλεγχος για υπερχείλιση ή ανεπάρκεια
  - $1.000_2 \times 2^{-4}$, χωρίς υπερχείλιση ή ανεπάρκεια
- 4. Στρογγυλοποίηση και, αν είναι απαραίτητο, εκ νέου κανονικοποίηση
  - $1.000_2 \times 2^{-4}$ (καμία αλλαγή) = 0.0625

Υλικό για την πρόσθεση κινητής υποδιαστολής

- Πολύ πιο σύνθετο από τον αντίστοιχο υλικό πρόσθεσης ακεραίων
- Η εκτέλεση σε έναν κύκλο ρολογιού θα διαρκούσε πάρα πολύ
  - Πολύ περισσότερο σε σχέση με τις πράξεις με ακέραιους αριθμούς
  - Με το πιο αργό ρολόι θα επιβαρύνονταν όλες οι εντολές
- Συνήθως η πρόσθεση κινητής υποδιαστολής χρειάζεται αρκετούς κύκλους
  - Υλοποιείται και με διοχέτευση

Υλικό για την πρόσθεση κινητής υποδιαστολής



Πολλαπλασιασμός κινητής υποδιαστολής

- Θεωρήστε, για παράδειγμα, τους 4ψήφιους δεκαδικούς
  - $1.110 \times 10^{10} \times 9.200 \times 10^{-5}$
- 1. Πρόσθεση εκθετών
  - Όταν έχουμε πολωμένους εκθέτες, αφαιρείται η πόλωση από το άθροισμα
  - Νέος εκθέτης = 10 + –5 = 5
- 2. Πολλαπλασιασμός των σημαντικών
  - $1.110 \times 9.200 = 10.212 \Rightarrow 10.212 \times 10^{5}$
- 3. Κανονικοποίηση του αποτελέσματος και έλεγχος για υπερχείλιση ή ανεπάρκεια
  - 1.0212×10^{6}
- 4. Στρογγυλοποίηση και, αν είναι απαραίτητο, εκ νέου κανονικοποίηση
  - 1.021×10^{6}
- 5. Προσδιορισμός του προσήμου του αποτελέσματος από τα πρόσημα των τελεστέων
  - $+1.021 \times 10^{6}$

Πολλαπλασιασμός κινητής υποδιαστολής

- Τώρα θεωρήστε, για παράδειγμα, τους 4ψήφιους δυαδικούς
  - $1.000_2 \times 2^{-1} \times -1.110_2 \times 2^{-2}$ (0.5 × –0.4375)
- 1. Πρόσθεση εκθετών
  - Μη πολωμένοι: –1 + –2 = –3
  - Πολωμένοι: (–1 + 127) + (–2 + 127) = –3 + 254 – 127 = –3 + 127
- 2. Πολλαπλασιασμός των σημαντικών
  - $1.000_2 \times 1.110_2 = 1.1102 \Rightarrow 1.110_2 \times 2^{-3}$
- 3. Κανονικοποίηση του αποτελέσματος και έλεγχος για υπερχείλιση ή ανεπάρκεια
  - $1.110_2 \times 2^{-3}$ (καμία αλλαγή) χωρίς υπερχείλιση ή ανεπάρκεια
- 4. Στρογγυλοποίηση και, αν είναι απαραίτητο, εκ νέου κανονικοποίηση
  - $1.110_2 \times 2^{-3}$ (καμία αλλαγή)
- 5. Προσδιορισμός προσήμου: (θετικός) × (αρνητικός) ⇒ (αρνητικός)
  - $-1.110_2 \times 2^{-3} = -0.21875$

Υλικό αριθμητικής κινητής υποδιαστολής

- Ο πολλαπλασιασμός κινητής υποδιαστολής είναι παρόμοιας πολυπλοκότητας με την πρόσθεση κινητής υποδιαστολής
  - Αλλά γίνεται πολλαπλασιασμός των σημαντικών, όχι πρόσθεση
- Κατά κανόνα, το υλικό αριθμητικής κινητής υποδιαστολής εκτελεί
  - πρόσθεση, αφαίρεση, πολλαπλασιασμό, διαίρεση, υπολογισμό αντιστρόφων, υπολογισμό τετραγωνικών ριζών
  - Μετατροπή κινητής υποδιαστολής ↔ ακεραίου
- Συνήθως οι πράξεις χρειάζονται αρκετούς κύκλους
  - Υλοποιούνται και με διοχέτευση

Εντολές κινητής υποδιαστολής στον RISC-V

- Ξεχωριστοί καταχωρητές κινητής υποδιαστολής: f0, …, f31
  - διπλής ακρίβειας
  - οι τιμές απλής ακρίβειας περιέχονται στα 32 χαμηλοτερα bit
- Οι εντολές κινητής υποδιαστολής επενεργούν μόνο σε καταχωρητές κινητής υποδιαστολής
  - Γενικώς, τα προγράμματα δεν εκτελούν πράξεις/λειτουργίες ακεραίων σε δεδομένα κινητής υποδιαστολής, ή το αντίθετο
  - Περισσότεροι καταχωρητές με ελάχιστη επίπτωση στο μέγεθος του κώδικα
- Εντολές φόρτωσης (load) και αποθήκευσης (store) κινητής υποδιαστολής
  - `flw, fld`
  - `fsw, fsd`

Εντολές κινητής υποδιαστολής στον RISC-V

- Αριθμητικές πράξεις απλής ακρίβειας
  - `fadd.s, fsub.s, fmul.s, fdiv.s, fsqrt.s`
    - π.χ. `fadds.s f2, f4, f6`
- Αριθμητικές πράξεις διπλής ακρίβειας
  - `fadd.d, fsub.d, fmul.d, fdiv.d, fsqrt.d`
    - π.χ. `fadd.d f2, f4, f6`
- Σύγκριση πράξεων απλής και διπλής ακρίβειας
  - `feq.s, flt.s, fle.s`
  - `feq.d, flt.d, fle.d`
  - Αποτέλεσμα 0 ή 1 στον ακέραιο καταχωρητή προορισμού
    - Χρήση των εντολών beq, bne για διακλάδωση με βάση το αποτέλεσμα της σύγκρισης
- Διακλάδωση με βάση το αν η συνθήκη κινητής υποδιαστολής έχει τιμή true ή false
  - `b.cond`

Ένα παράδειγμα κινητής υποδιαστολής: Μετατροπή βαθμών °F σε βαθμούς °C

- Κώδικας C:

```
float f2c (float fahr) {
  return ((5.0/9.0)*(fahr - 32.0));
}
```

  - fahr στον f10, αποτέλεσμα στον f10, λεκτικά στην καθολική μνήμη

- Μεταγλωττισμένος κώδικας RISC-V:

```
f2c:
 flw    f0,const5(x3)  // f0 = 5.0f
 flw    f1,const9(x3)  // f1 = 9.0f
 fdiv.s f0, f0, f1  // f0 = 5.0f / 9.0f
 flw    f1,const32(x3) // f1 = 32.0f
 fsub.s f10,f10,f1  // f10 = fahr - 32.0
 fmul.s f10,f0,f10  // f10 = (5.0f/9.0f) * (fahr-32.0f)
 jalr   x0,0(x1)    // επιστροφή
```

Ένα παράδειγμα κινητής υποδιαστολής: Πολλαπλασιασμός πινάκων

- C = C + A × B
  - Όλοι είναι πίνακες 32 × 32, με στοιχεία διπλής ακρίβειας 64 bit
- Κώδικας C:

```
void mm (double c[][],
         double a[][], double b[][]) {
  size_t i, j, k;
  for (i = 0; i < 32; i = i + 1)
    for (j = 0; j < 32; j = j + 1)
      for (k = 0; k < 32; k = k + 1)
        c[i][j] = c[i][j]
                + a[i][k] * b[k][j];
}
```

  - Διευθύνσεις των c, a, b στους x10, x11, x12, και των i, j, k στους x5, x6, x7

Ένα παράδειγμα κινητής υποδιαστολής: Πολλαπλασιασμός πινάκων

- Κώδικας RISC-V:

```
mm:...
        li    x28,32        // x28 = 32 (μέγεθος γραμμής/τέλος βρόχου)
        li    x5,0          // i = 0, αρχικοποίηση 1ου βρόχου for
L1:     li    x6,0          // j = 0, αρχικοποίηση 2ου βρόχου for
L2:     li    x7,0          // k = 0; αρχικοποίηση 3ου βρόχου for
        slli  x30,x5,5      // x30 = i * 2**5 (μέγεθος της γραμμής του c)
        add   x30,x30,x6    // x30 = i * μέγεθος(γραμμής) + j
        slli  x30,x30,3     // x30 = σχετική απόσταση byte του [i][j]
        add   x30,x10,x30   // x30 = διεύθυνση byte του c[i][j]
        fld   f0,0(x30)     // f0 = c[i][j]
L3:     slli  x29,x7,5      // x29 = k * 2**5 (μέγεθος της γραμμής του b)
        add   x29,x29,x6    // x29 = k * μέγεθος(γραμμής) + j
        slli  x29,x29,3     // x29 = σχετική απόσταση byte του [k][j]
        add   x29,x12,x29   // x29 = διεύθυνση byte του b[k][j]
        fld   f1,0(x29)     // f1 = b[k][j]
```

Ένα παράδειγμα κινητής υποδιαστολής: Πολλαπλασιασμός πινάκων

```
...
        slli    x29,x5,5      // x29 = i * 2**5 (μέγεθος της γραμμής του a)
        add     x29,x29,x7    // x29 = i * μέγεθος(γραμμής) + k
        slli    x29,x29,3     // x29 = σχετική απόσταση byte του [i][k]
        add     x29,x11,x29   // x29 = διεύθυνση byte του a[i][k]
        fld     f2,0(x29)     // f2 = a[i][k]
        fmul.d  f1, f2, f1    // f1 = a[i][k] * b[k][j]
        fadd.d  f0, f0, f1    // f0 = c[i][j] + a[i][k] * b[k][j]
        addi    x7,x7,1       // k = k + 1
        bltu    x7,x28,L3     // αν (k < 32), μετάβαση στην L3
        fsd     f0,0(x30)     // c[i][j] = f0
        addi    x6,x6,1       // j = j + 1
        bltu    x6,x28,L2     // αν (j < 32), μετάβαση στην L2
        addi    x5,x5,1       // i = i + 1
        bltu    x5,x28,L1     // αν (i < 32), μετάβαση στην L1
```

Ακριβής αριθμητική

- Στο πρότυπο 754 της IEEE καθορίζονται πρόσθετοι έλεγχοι στρογγυλοποίησης
  - Επιπλέον bit ακρίβειας: bit-φρουρός (guard), bit στρογγύλευσης (round), επίμονο bit (sticky)
  - Επιλογή τρόπων στρογγυλοποίησης
  - Ο προγραμματιστής μπορεί να επιλέξει την επιθυμητή αριθμητική συμπεριφορά ενός υπολογισμού
- Δεν υλοποιούνται όλες οι επιλογές σε όλες τις μονάδες κινητής υποδιαστολής
  - Στις περισσότερες γλώσσες προγραμματισμού και βιβλιοθήκες κινητής υποδιαστολής χρησιμοποιούνται απλώς οι προκαθορισμένες επιλογές
- Αντισταθμίζονται η πολυπλοκότητα και η απόδοση του υλικού με τις απαιτήσεις της αγοράς

Παραλληλία υπολέξης

- Οι εφαρμογές ήχου και γραφικών μπορούν να εκμεταλλευτούν τη δυνατότητα εκτέλεσης ταυτόχρονων λειτουργιών σε μικρά διανύσματα
  - Παράδειγμα: Αθροιστής των 128 bit:
    - Δεκαέξι τελεστέοι των 8 bit
    - Οκτώ τελεστέοι των 16 bit
    - Τέσσερις τελεστέοι των 32 bit
- Γνωστή και ως παραλληλία επιπέδου δεδομένων (data-level parallelism), παραλληλία διανυσμάτων (vector parallelism), ή επεκτάσεις μίας εντολής-πολλών δεδομένων (Single Instruction, Multiple Data –SIMD)

Αρχιτεκτονική x86 κινητής υποδιαστολής

- Βασίστηκε αρχικά στον συνεπεξεργαστή κινητής υποδιαστολής 8087
  - 8 × καταχωρητές επεκταμένης ακρίβειας των 80 bit
  - Χρησιμοποιείται ως στοίβα που επεκτείνεται προς τα κάτω
  - Δεικτοδότηση των καταχωρητών από TOS: ST(0), ST(1), …
- Οι τιμές κινητής υποδιαστολής είναι 32 bit ή 64 bit στη μνήμη
  - Μετατρέπονται κατά τη φόρτωση/αποθήκευση του τελεστέου μνήμης
  - Αλλά και οι ακέραιοι τελεστέοι μπορούν να μετατρέπονται κατά τη φόρτωση/αποθήκευση
- Δύσκολο να παραχθεί και να βελτιστοποιηθεί ο κώδικας
  - Το αποτέλεσμα: κακές επιδόσεις στις λειτουργίες κινητής υποδιαστολής

Εντολές κινητής υποδιαστολής στην αρχιτεκτονική x86

Μεταφορά δεδομένων	Αριθμητικές πράξεις	Σύγκριση	Υπερβατικές συναρτήσεις
FILD mem/ST(i) FISTP mem/ST(i) FLDPI FLD1 FLDZ	FIADDP mem/ST(i) FISUBRP mem/ST(i) FIMULP mem/ST(i) FIDIVRP mem/ST(i) FSQRT FABS FRNDINT	FICOMP FIUCOMP FSTSW AX/mem	FPATAN F2XMI FCOS FPTAN FPREM FPSIN FYL2X

- Προαιρετικές παραλλαγές
  - I: ακέραιος τελεστέος
  - P: εξαγωγή τελεστέου από τη στοίβα
  - R: αντίστροφη σειρά τελεστέων
  - Δεν επιτρέπονται όλοι οι συνδυασμοί

Επέκταση Streaming SIMD Extension 2 (SSE2)

- Πρόσθεση 4 καταχωρητών των 128 bit
  - Στην αρχιτεκτονική AMD64/EM64T, το πλήθος τους επεκτείνεται στους 8
- Μπορεί να χρησιμοποιηθεί για πολλούς τελεστέους κινητής υποδιαστολής
  - 2 τελεστέους των 64 bit, διπλής ακρίβειας
  - 4 τελεστέους των 32 bit, διπλής ακρίβειας
  - Οι εντολές επενεργούν σε αυτούς ταυτόχρονα
    - Μία εντολή, πολλά δεδομένα (Single-Instruction Multiple-Data)

Πολλαπλασιασμός πινάκων

- Μη βελτιστοποιημένος κώδικας:

```
1. void dgemm (int n, double* A, double* B, double* C)
2. {
3.  for (int i = 0; i < n; ++i)
4.    for (int j = 0; j < n; ++j)
5.    {
6.     double cij = C[i+j*n]; /* cij = C[i][j] */
7.     for(int k = 0; k < n; k++ )
8.      cij += A[i+k*n] * B[k+j*n]; /* cij += A[i][k]*B[k][j] */
9.     C[i+j*n] = cij; /* C[i][j] = cij */
10.   }
11. }
```

Πολλαπλασιασμός πινάκων

- Βελτιστοποιημένος κώδικας σε C:

```
1. #include <x86intrin.h>
2. void dgemm (int n, double* A, double* B, double* C)
3. {
4. for (int i = 0; i < n; i+=8)
5.    for (int j = 0; j < n; ++j)
6.     {
7.       __m512d c0 =  _mm512_load_pd(C+i+j*n); // c0 = C[i][j]
8.          for( int k = 0; k < n; k++ )
9.            {  // c0 += A[i][k]*B[k][j]
10.            __m512d bb = _mm512_broadcastsd_pd(_mm_load_sd(B+j*n+k));
11.            c0 = _mm512_fmadd_pd(_mm512_load_pd(A+n*k+i), bb, c0);
12.           }
13.       _mm512_store_pd(C+i+j*n, c0); // C[i][j] = c0
14.    }
15.}
```

Πολλαπλασιασμός πινάκων

- Βελτιστοποιημένος κώδικας συμβολικής γλώσσας για αρχιτεκτονική x86:

```
vmovapd (%r11),%zmm1                  # Φόρτωση 8 στοιχείων του C στον %zmm1
mov     %rbx,%rcx                     # Καταχωρητής %rcx = %rbx
xor     %eax,%eax                     # Καταχωρητής %eax = 0
vbroadcastsd (%rax,%r8,8),%zmm0       # Κάνε 8 αντίγραφα του στοιχείου του B στον %zmm0
add     $0x8,%rax                     # Καταχωρητής %rax = %rax + 8
vfmadd231pd (%rcx),%zmm0,%zmm1        # Παράλληλος πολλαπλασιασμός & πρόσθεση %zmm0, %zmm1
add     %r9,%rcx                      # Καταχωρητής %rcx = %rcx
cmp     %r10,%rax                     # Σύγκρινε τον %r10 με τον %rax
jne     50 <dgemm+0x50>               # άλμα αν δεν ισχύει %r10 != %rax
add     $0x1, %esi                    # Καταχωρητής % esi = % esi + 1
vmovapd %zmm1, (%r11)                 # Αποθήκευση του %zmm1 στα 8 στοιχεία του C
```

Δεξιά ολίσθηση και διαίρεση

- Μια εντολή αριστερής ολίσθησης μπορεί να αντικαταστήσει έναν ακέραιο πολλαπλασιασμό με μια δύναμη του 2
- Μια εντολή δεξιάς ολίσθησης είναι το ίδιο με μια ακέραια διαίρεση με μια δύναμη του 2;
  - Αυτό ισχύει μόνο για μη προσημασμένους ακέραιους
- Για προσημασμένους ακέραιους αριθμούς
  - Αριθμητική δεξιά ολίσθηση: αναπαράγει το bit προσήμου
  - π.χ. –5 / 4
    - 11111011_2 >> 2 = 11111110_2 = –2
    - Στρογγυλοποίηση προς το –∞
  - πρβλ. 11111011_2 >>> 2 = 00111110_2 = +62

Προσεταιριστικότητα

- Τα προγράμματα που γράφονται με γνώμονα την παράλληλη εκτέλεση ενδέχεται να διαπλέκουν τις λειτουργίες και πράξεις σε απρόβλεπτη σειρά
  - Ενδέχεται να μην ισχύει η παραδοχή της προσεταιριστικότητας

		(x+y)+z	x+(y+z)
x	-1.50E+38		-1.50E+38
y	1.50E+38	0.00E+00	
z	1.0	1.0	1.50E+38
		1.00E+00	0.00E+00

- Τα προγράμματα παράλληλης εκτέλεσης πρέπει να επικυρώνονται υπό διάφορους βαθμούς παραλληλίας

Ποιοι ενδιαφέρονται για την ακρίβεια των πράξεων κινητής υποδιαστολής;

- Σημαντική για τον κώδικα επιστημονικών εφαρμογών
  - Αλλά όσον αφορά τις καθημερινές εφαρμογές ευρείας χρήσης;
    - «Το υπόλοιπο του τραπεζικού μου λογαριασμού διαφέρει κατά 0.0002 σεντ!» ☹
- Το σφάλμα FDIV στους επεξεργαστές Pentium της Intel
  - Η καταναλωτική αγορά θεωρεί δεδομένη την ακρίβεια στις πράξεις
  - Παραπομπή σε: Colwell, *The Pentium Chronicles*

Συμπερασματικές παρατηρήσεις

- Οι σειρές των bit δεν έχουν κάποιο εσωτερικό νόημα
  - Η ερμηνεία εξαρτάται από τις εντολές που εκτελούνται
- Αναπαραστάσεις των αριθμών στους υπολογιστές
  - Πεπερασμένο εύρος, πεπερασμένη ακρίβεια
  - Αυτό πρέπει να λαμβάνεται υπόψη στα προγράμματα

Συμπερασματικές παρατηρήσεις

- Οι αρχιτεκτονικές συνόλου εντολών (ISA) υποστηρίζουν την αριθμητική
  - Προσημασμένοι και μη προσημασμένοι αριθμοί
  - Προσέγγιση των πραγματικών αριθμών με αριθμούς κινητής υποδιαστολής
- Εύρος και ακρίβεια εντός ορίων
  - Ενδεχόμενο υπερχείλισης και ανεπάρκειας στις πράξεις