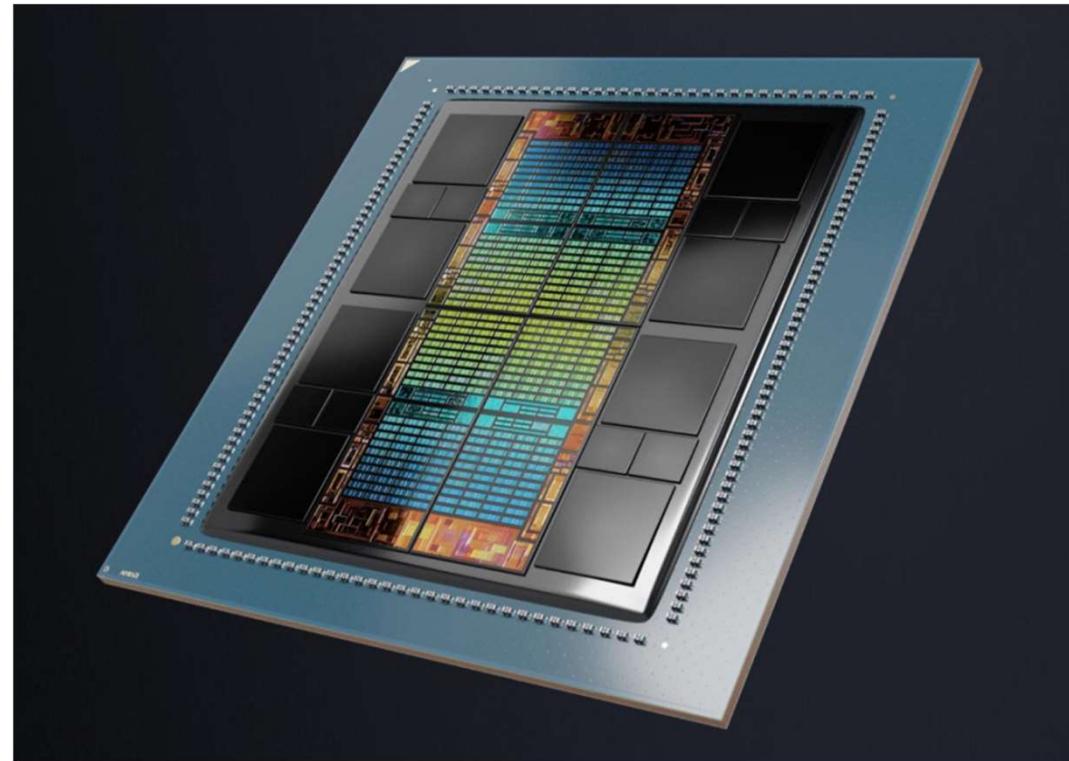
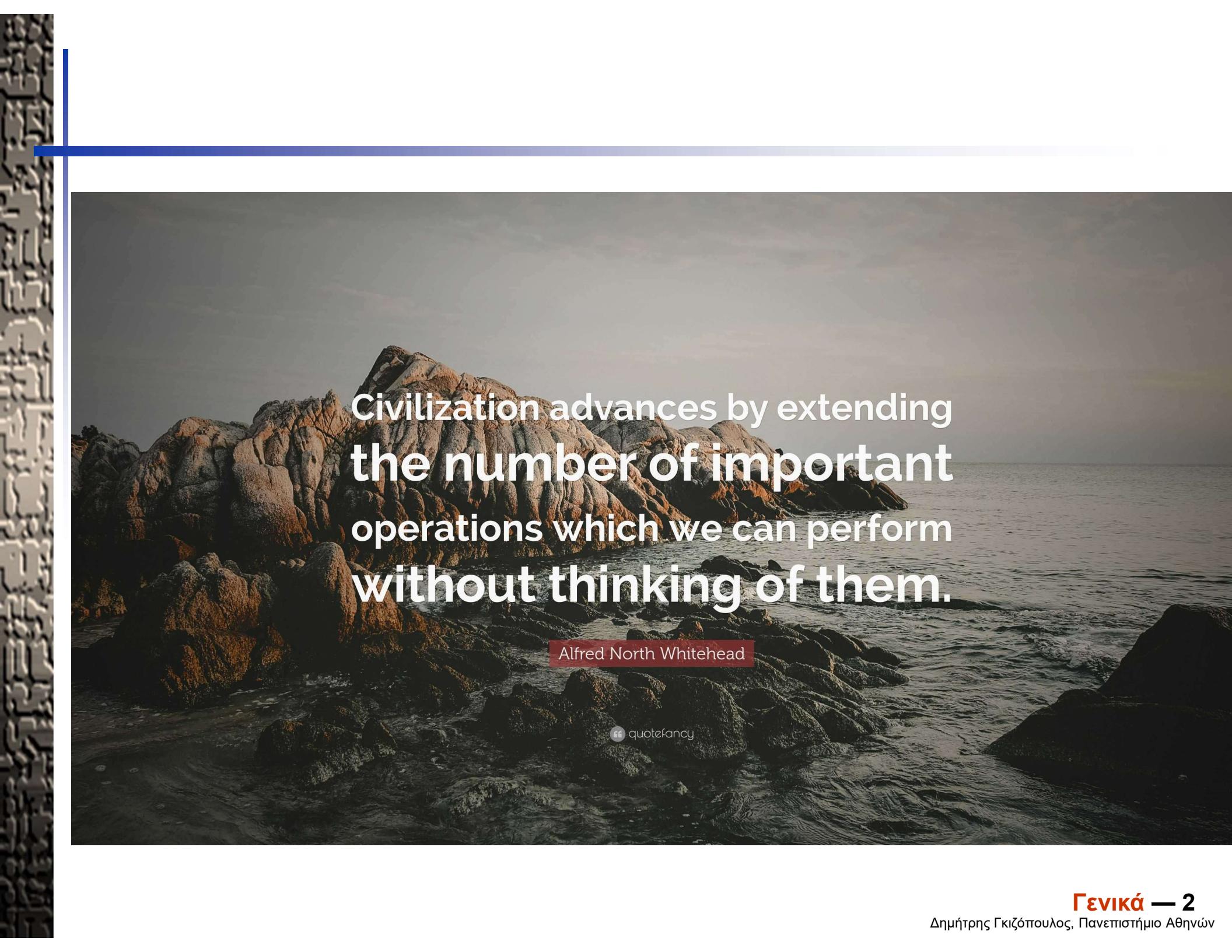


Αρχιτεκτονική Υπολογιστών

Δημήτρης
Γκιζόπουλος

Καθηγητής





Civilization advances by extending
the number of important
operations which we can perform
without thinking of them.

Alfred North Whitehead

quotefancy

Η Βασική Εξίσωση Απόδοσης

Χρόνος προγράμματος =

$$= \frac{\text{εντολές}}{\text{πρόγραμμα}} \times \frac{\text{κύκλοι}}{\text{εντολή}} \times \frac{\text{sec}}{\text{κύκλος}}$$

Παράδειγμα:

πρόγραμμα $50 \cdot 10^9$ (50 δις) εντολών

2 κύκλους ρολογιού ανά εντολή

0.5 ns ($=0.5 \cdot 10^{-9}$ s) διάρκεια κύκλου, (ή ρυθμός ρολογιού 2 GHz)

$$\text{Χρόνος} = 50 \cdot 10^9 \times 2 \times 0.5 \cdot 10^{-9} = 50 \text{ sec}$$

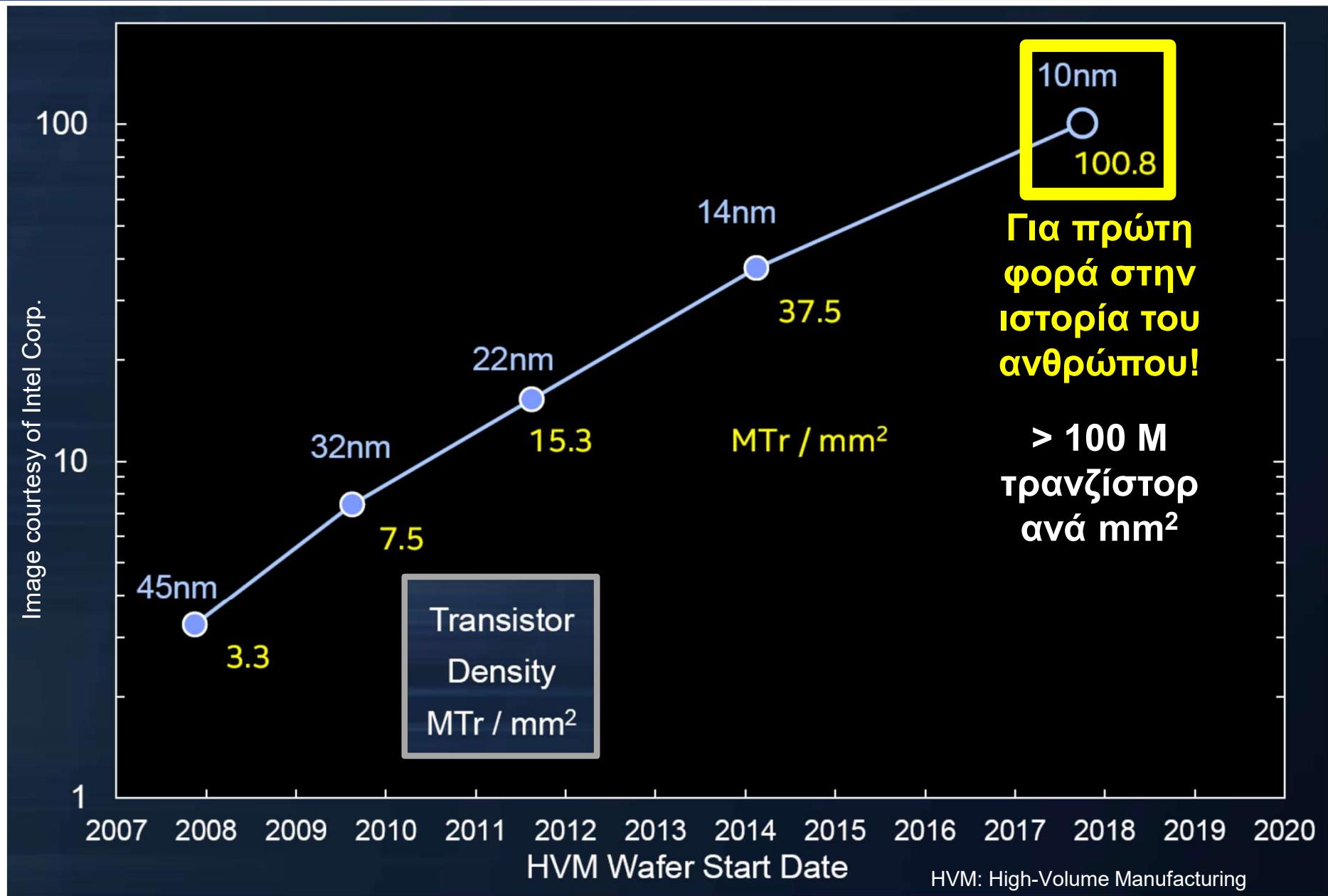
Γενικά

- Δημήτρης Γκιζόπουλος, Καθηγητής
 - <http://www.di.uoa.gr/~dgizop>
 - γραφείο Α36
- Διδασκαλία
 - Δευτέρα και Τρίτη 13⁰⁰-15⁰⁰
- Φροντιστήριο (κατόπιν ενημέρωσης)
 - Δευτέρα 17⁰⁰-19⁰⁰
- Εργαστήριο
 - Έναρξη σύντομα – δήλωση συμμετοχής στο eclass
- E-class
 - <http://eclass.uoa.gr/courses/D19/>
 - διαφάνειες, υλικό, ασκήσεις, επικοινωνία

Βαθμολογία μαθήματος

- **Θεωρία 70% + Εργαστήριο 30%**
 - RARS + QtRVSIM simulators αρχιτεκτονικής RISC-V
 - Δηλώσεις για εγγραφή στο εργαστήριο
 - παρακολούθηση + εξέταση
 - Ισχύει για όλους τους φοιτητές με ΑΜ 2010... +
- **Kouίζ** (μάθημα της εβδομάδας) – συστηματική συμμετοχή δίνει 1 επιπλέον βαθμό (bonus) στην τελική εξέταση της θεωρίας

2018 – Χρονιά «Σταθμός»

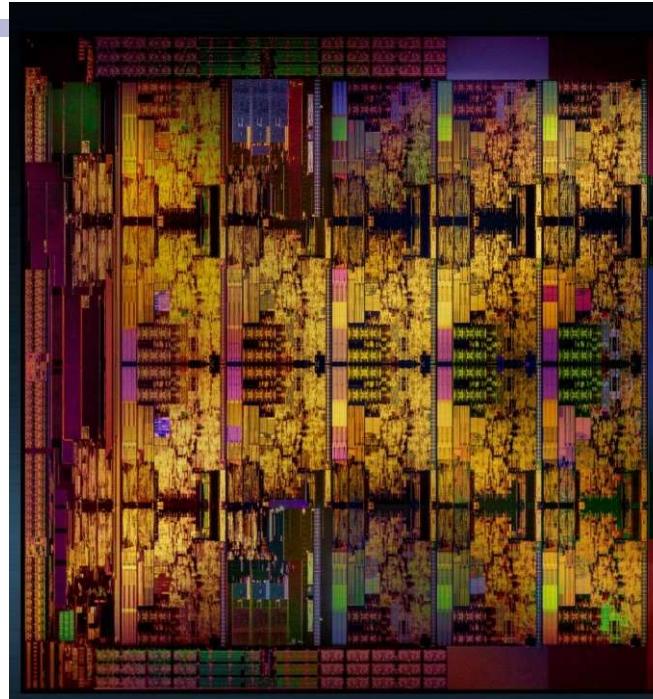


Central Processing Units (CPUs)

High-End Desktop (HEDT) CPU

Intel Core i9-7980XE (Skylake-X),
September 2017

14 nm
18 cores →
6–7 B transistors
484 mm² die area



Images courtesy of Intel Corp.

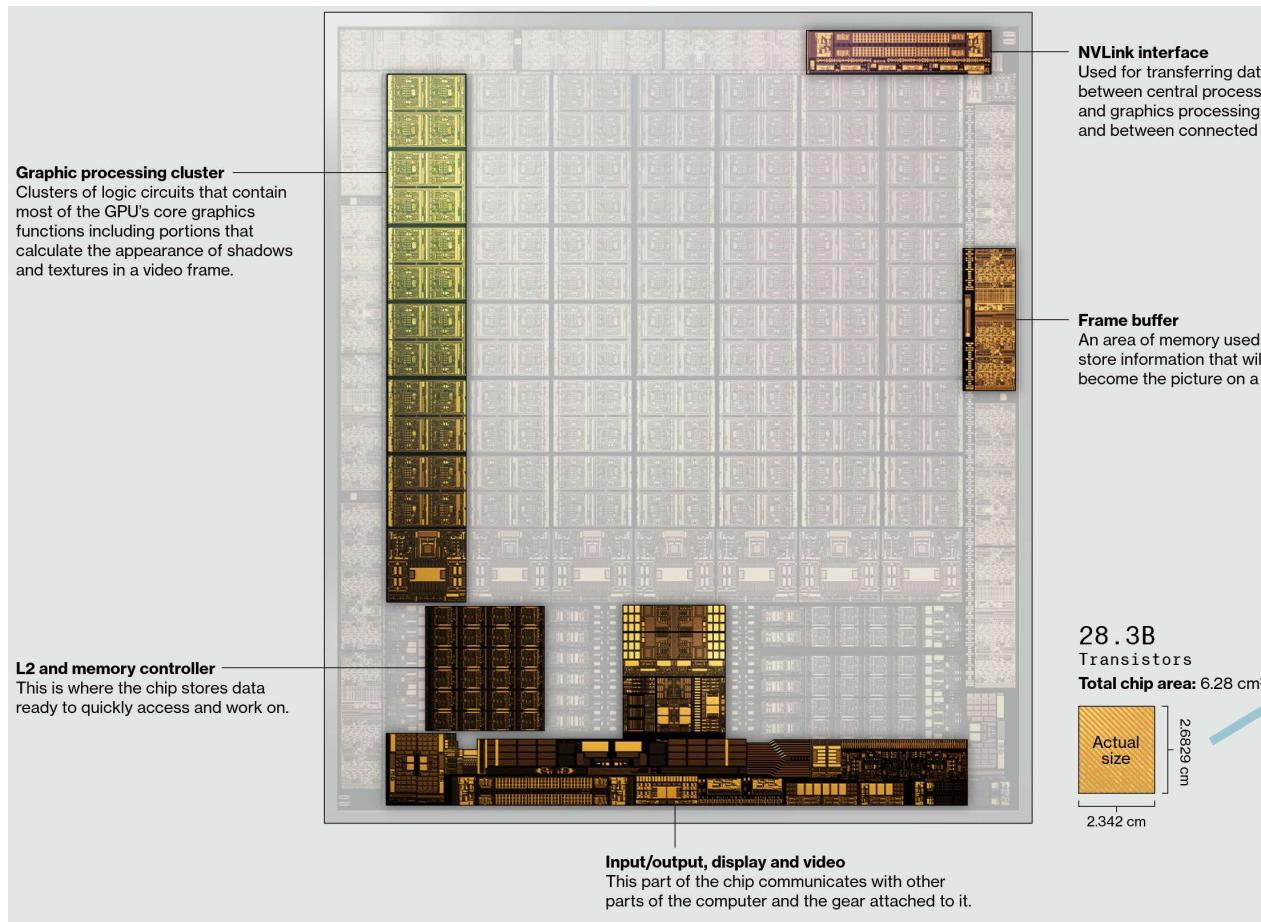
Server CPU

Intel Xeon Platinum 8180
[Skylake Purley Extreme
Core Count (XCC)],
July 2017

14 nm
28 cores →
~8.4–10 B transistors
698 mm² die area



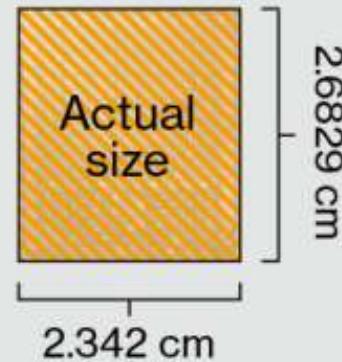
GPUs



28.3B
Transistors
Total chip area: 6.28 cm²

A diagram showing the total chip area as a large square divided into four quadrants. The top-left quadrant is labeled "Actual size" and has dimensions of 2.6829 cm by 2.342 cm. A blue arrow points from this diagram to the text above it.

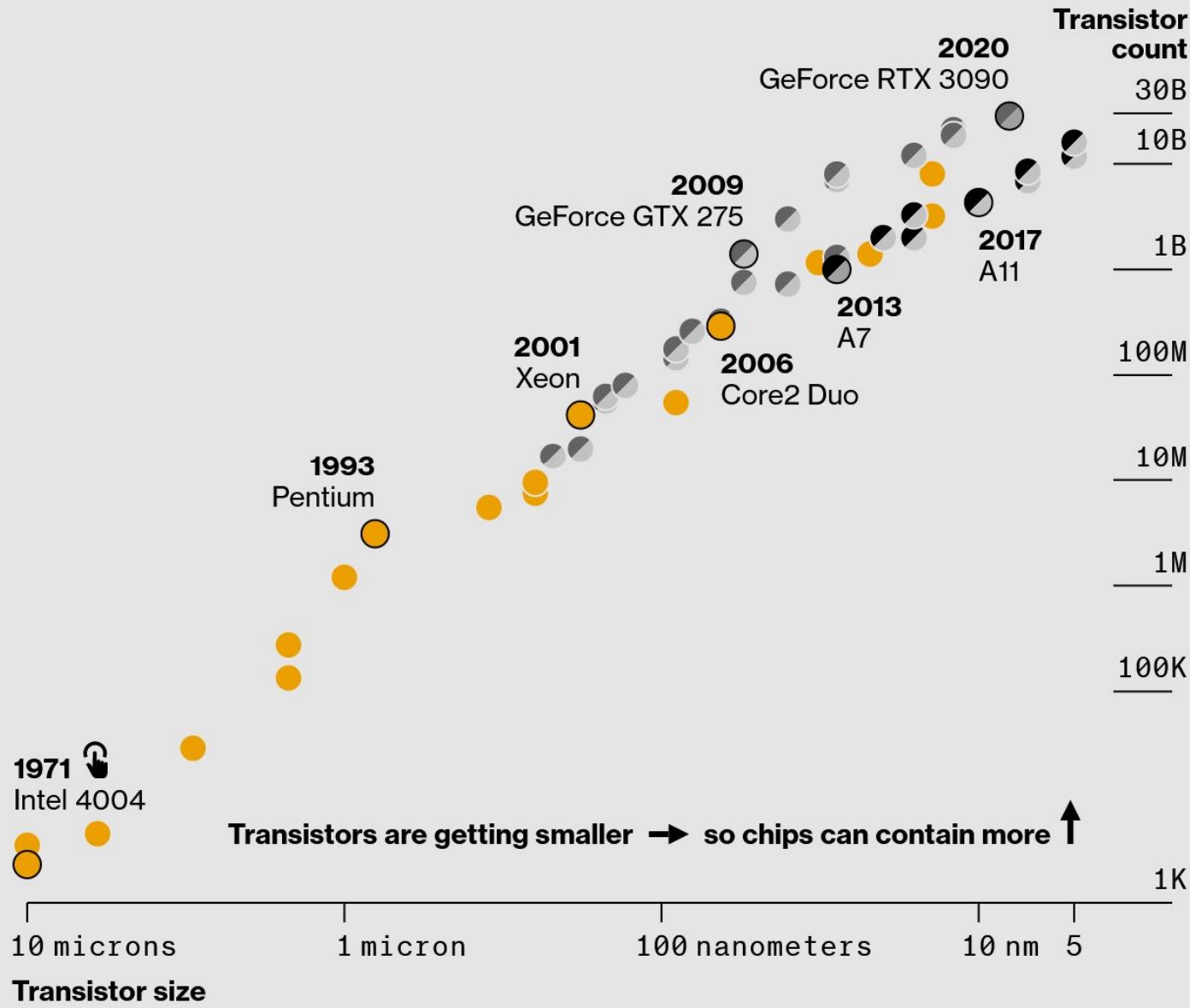
28.3B
Transistors
Total chip area: 6.28 cm²



Εξέλιξη κατασκευής CPU/GPU

Designer and manufacturer: ● Intel

Designer: ● Apple ● Nvidia Manufacturer: ● Samsung ● TSMC



Αρχιτεκτονική Υπολογιστών;

- Ο ευρύτερος ορισμός:
 - **Αρχιτεκτονική Υπολογιστών** είναι η σχεδίαση των επιπέδων αφαίρεσης (abstraction layers) που μας επιτρέπει να υλοποιήσουμε εφαρμογές επεξεργασίας πληροφορίας (information processing applications) με αποδοτικό τρόπο χρησιμοποιώντας τις διαθέσιμες τεχνολογίες κατασκευής.
- Γιατί χρειάζονται επίπεδα αφαίρεσης;
 - Διότι η απόσταση μεταξύ της εφαρμογής και της φυσικής συσκευής που θα την εκτελέσει είναι πολύ μεγάλη

Επίπεδα αφαίρεσης

■ Στα σύγχρονα υπολογιστικά συστήματα

Application

Εφαρμογή

Algorithm

Αλγόριθμος

Programming Language

Γλώσσα προγραμματισμού

Operating System/Virtual Machines

Λειτουργικό Σύστημα/Εικονικές Μηχ.

Instruction Set Architecture (ISA)

Αρχιτεκτονική Συνόλου Εντολών

Microarchitecture

Μικροαρχιτεκτονική

Gates/Register-Transfer Level (RTL)

Πύλες/Επίπεδο Μεταφοράς Καταχ/τύ

Circuits

Κυκλώματα

Devices

Συσκευές (στοιχεία κυκλώματος)

Physics

Φυσική

Γιατί τα χρειαζόμαστε;

[~ 10^1] **32 / 60 C/C++ keywords**

[~ 10^2] **300 system calls**

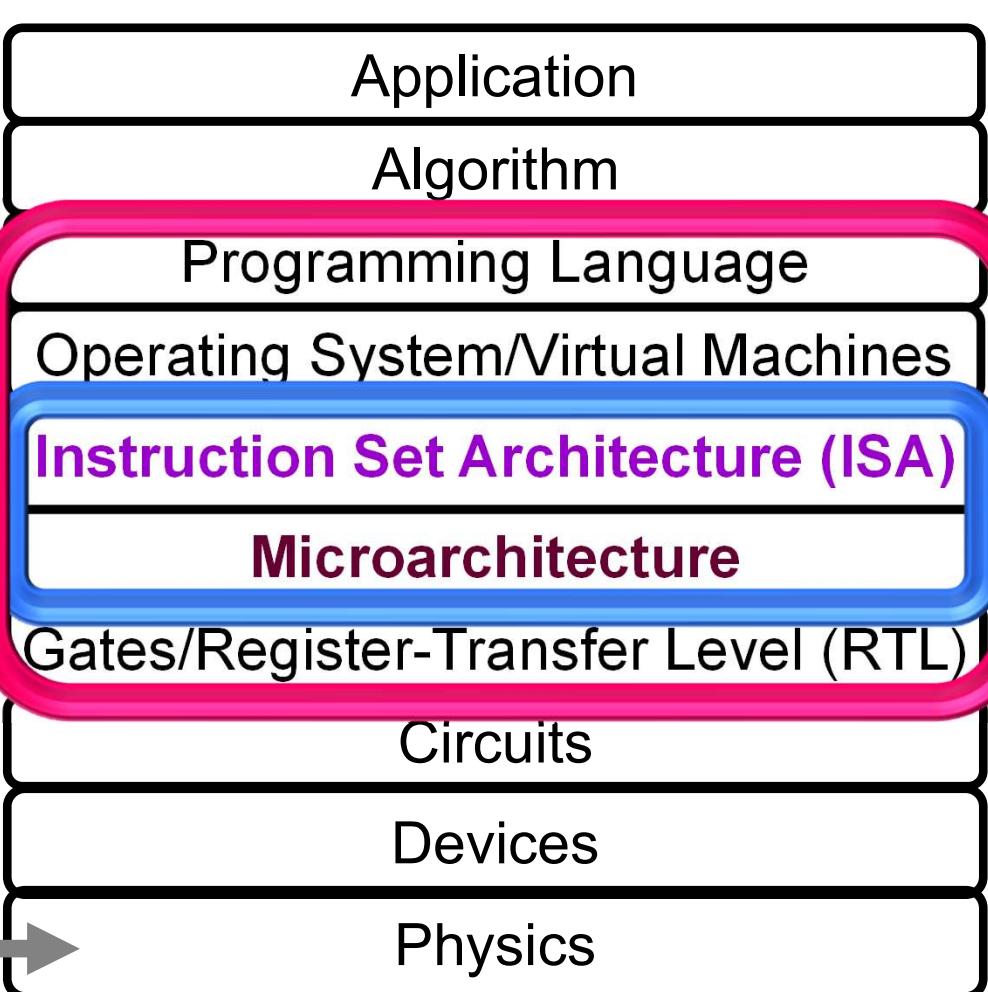
[~ 10^3] **3684 x86 instructions**

[~ 10^2] **354 Arm instructions**

[~ 10^4] **10 000 hardware blocks**

[~ 10^9] **1 billion gates**

[~ 10^{10}] **6.7 billion transistors**



Το όνομα του μαθήματος

- Εμείς έχουμε τα μαθήματα
 - **Αρχιτεκτονική Υπολογιστών I**
 - βασικές έννοιες οργάνωσης υπολογιστών, αρχιτεκτονικής υπολογιστών, αριθμητικής υπολογιστών, σχεδίασης υπολογιστών
 - **Αρχιτεκτονική Υπολογιστών II**
 - προηγμένες έννοιες αρχιτεκτονικής υπολογιστών και μέθοδοι σχεδίασης υψηλών επιδόσεων
- áλλα τμήματα τα ονομάζουν
 - Οργάνωση Υπολογιστών
 - Αρχιτεκτονική Υπολογιστών

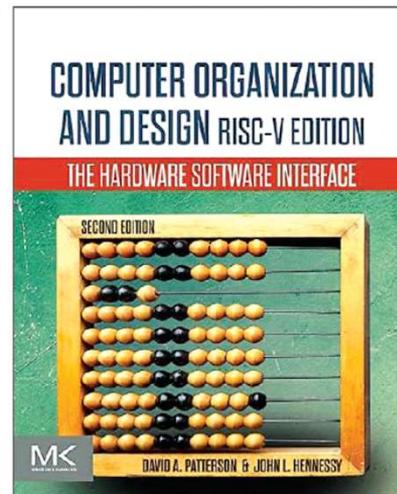
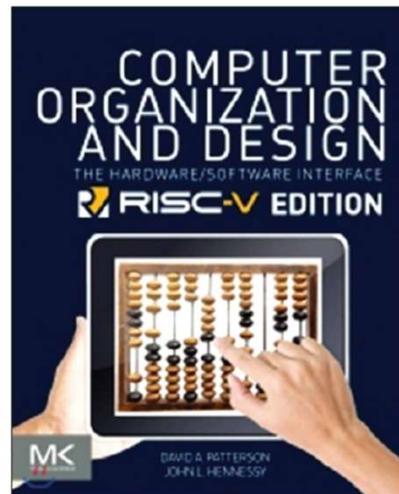
Βιβλίο του μαθήματος

- **Οργάνωση και Σχεδίαση Υπολογιστών: η Διασύνδεση Υλικού και Λογισμικού, 2η έκδοση RISC-V, (Computer Organization and Design: the Hardware/Software Interface, 2e RISC-V)**
- D.A.Patterson – CS@Berkeley U
- J.L.Hennessy – Stanford U
- Elsevier/Morgan Kaufmann
 - Μετάφραση, επιστημονική επιμέλεια στα ελληνικά: Δ.Γκιζόπουλος (εκδόσεις Κλειδάριθμος)

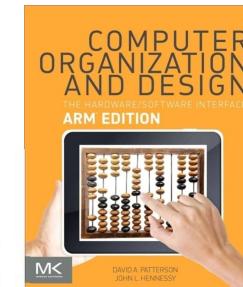


Η ιστορία του βιβλίου

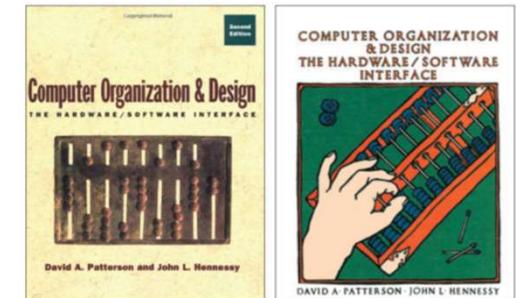
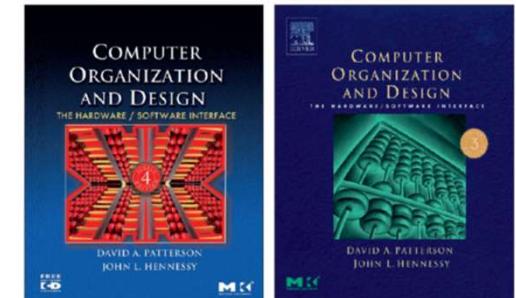
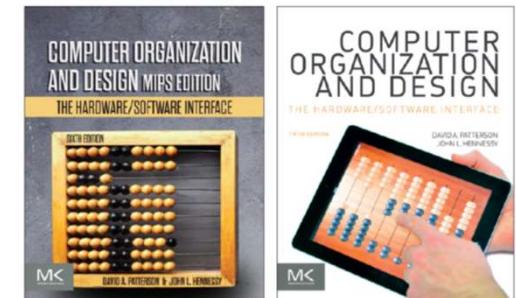
- Με βάση τους επεξεργαστές
MIPS, Arm, RISC-V



arm



MIPS
TECHNOLOGIES



Ξενάγηση στο βιβλίο

Κεφάλαιο ή παράρτημα	Ενότητες	Έμφαση στο λογισμικό	Έμφαση στο υλικό
1. Αφηρημένες έννοιες και τεχνολογία υπολογιστών	1.1 έως 1.12		
	1.13 (Ιστορική προοπτική)		
2. Εντολές: Η γλώσσα του υπολογιστή	2.1 έως 2.14		
	2.15 (Μεταγλωττιστές και Java)		
	2.16 έως 2.22		
	2.23 (Ιστορική προοπτική)		
D. Αρχιτεκτονικές συνόλου εντολών RISC	D.1 έως D.6		
3. Αριθμητική για υπολογιστές	3.1 έως 3.5		
	3.6 έως 3.8 (Παραλληλία υπολέξης)		
	3.9 έως 3.10 (Πλάνες)		
	3.11 (Ιστορική προοπτική)		
A. Τα βασικά της λογικής σχεδίασης	A.1 έως A.13		
4. Ο επεξεργαστής	4.1 (Επισκόπηση)		
	4.2 (Συμβάσεις λογικής)		
	4.3 έως 4.4 (Απλή υλοποίηση)		
	4.5 (Υλοποίηση πολλών κύκλων)		
	4.6 (Γενικά για τη διοχέτευση)		
	4.7 (Διαδρομή δεδομένων με διοχέτευση)		
	4.9 έως 4.10 (Κίνδυνοι, εξαιρέσεις)		
	4.11 έως 4.13 (Παραλληλία, πραγματικότητα)		

Ξενάγηση στο βιβλίο (συνεχ.)

	4.14 (Ελεγκτής διοχέτευσης Verilog)		
	4.15 έως 4.16 (Πλάνες)		
	4.17 (Ιστορική προοπτική)		
C. Αντιστοίχιση του ελέγχου στο υλικό	C.1 έως C.6		
5. Μεγάλη και γρήγορη: Αξιοποίηση της ιεραρχίας τής μνήμης	5.1 έως 5.10		
	5.11 (Πλεονασματικές Συστοιχίες Φθηνών Δίσκων)		
	5.12 (Ελεγκτές κρυφής μνήμης στη Verilog)		
	5.13 έως 5.16		
	5.17 (Ιστορική προοπτική)		
6. Παράλληλοι επεξεργαστές από τις συσκευές έως το νέφος	6.1 έως 6.0		
	6.10 (Συστοιχίες)		
	6.11 έως 6.15		
	6.16 (Ιστορική προοπτική)		
B. Μονάδες επεξεργασίας γραφικών	B.1 έως B.11		

Διαβάστε με προσοχή
Επισκόπηση ή ανάγνωση



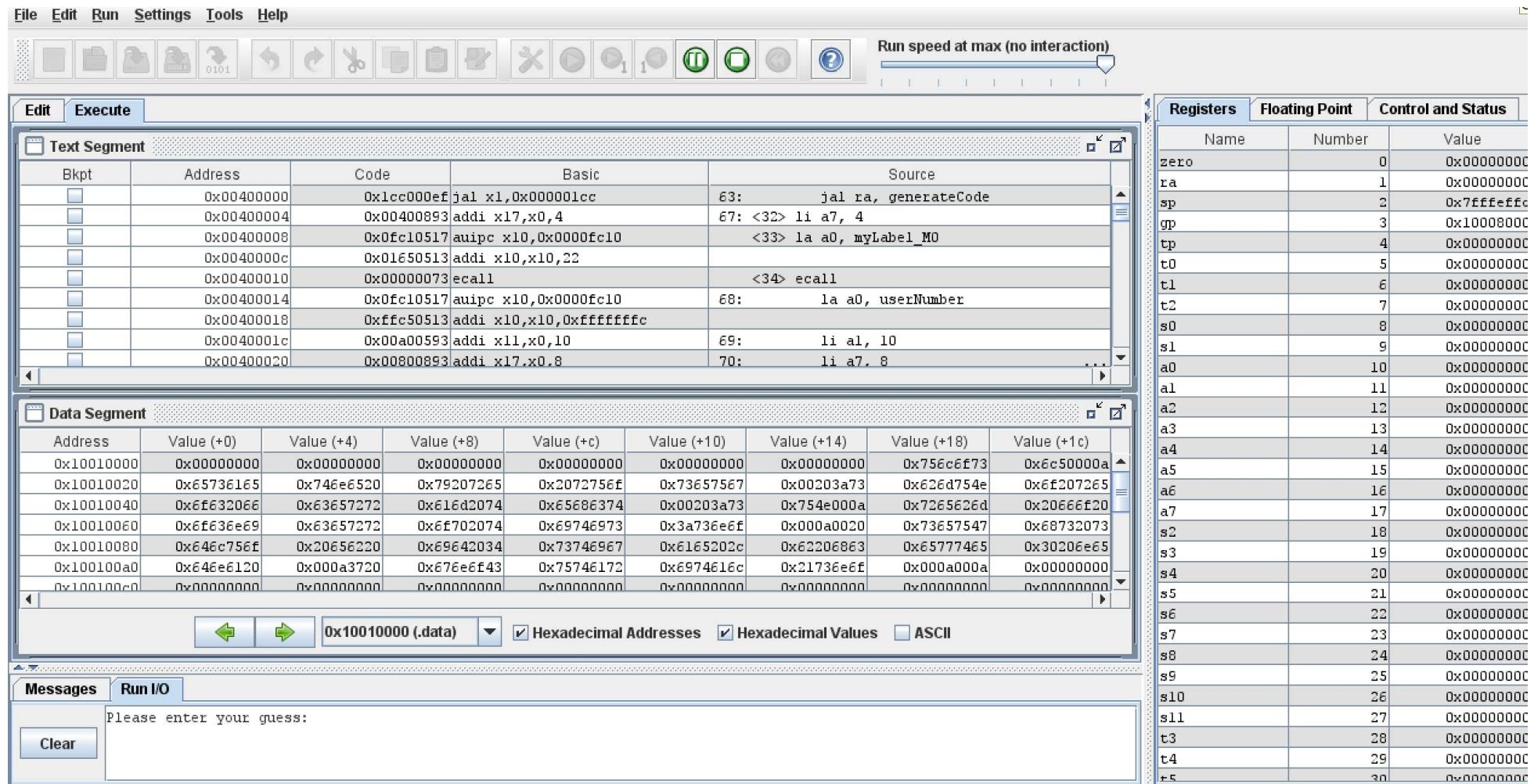
Διαβάστε αν έχετε χρόνο
Διαβάστε για επιμόρφωση



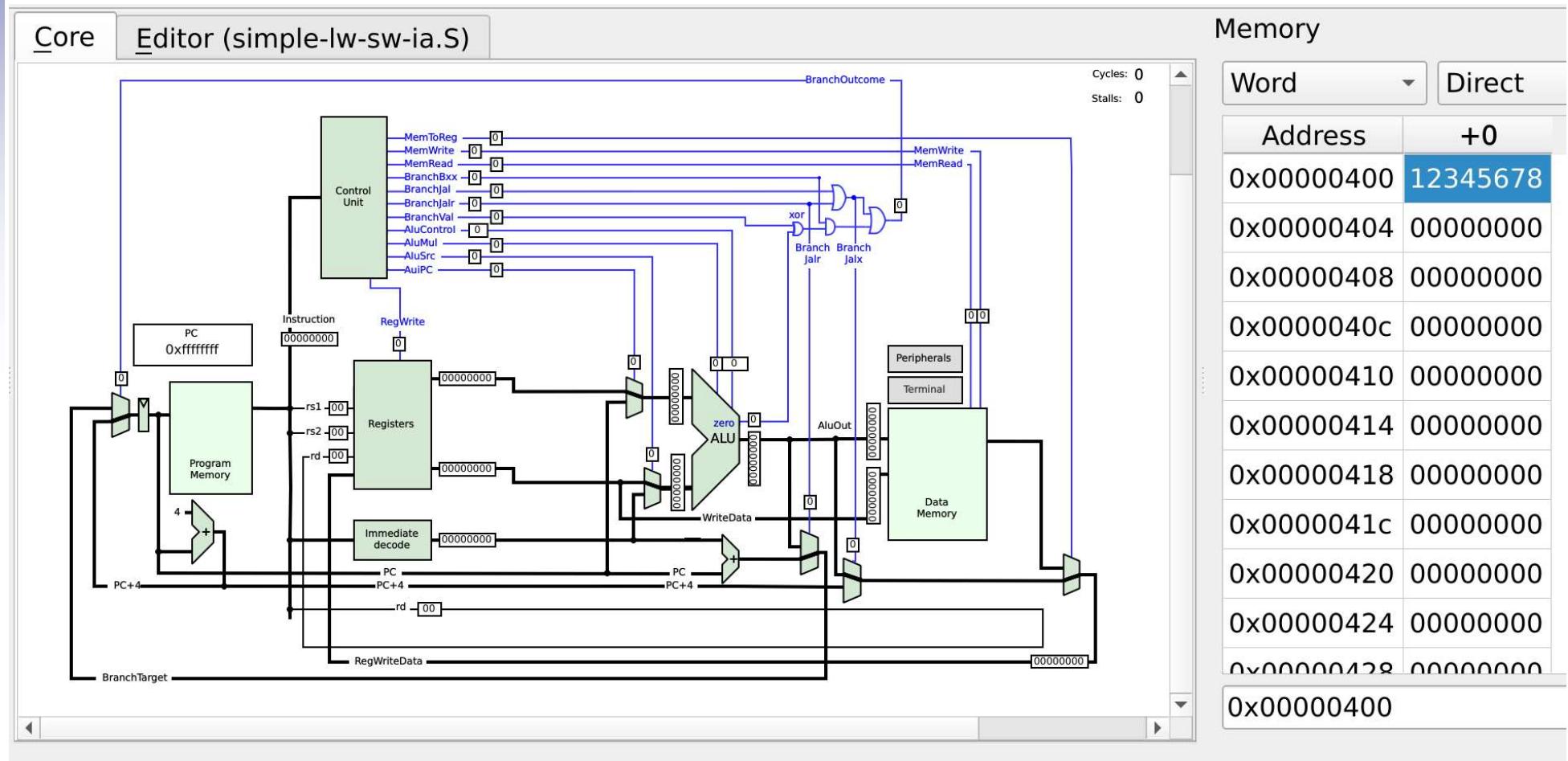
Αναφορά



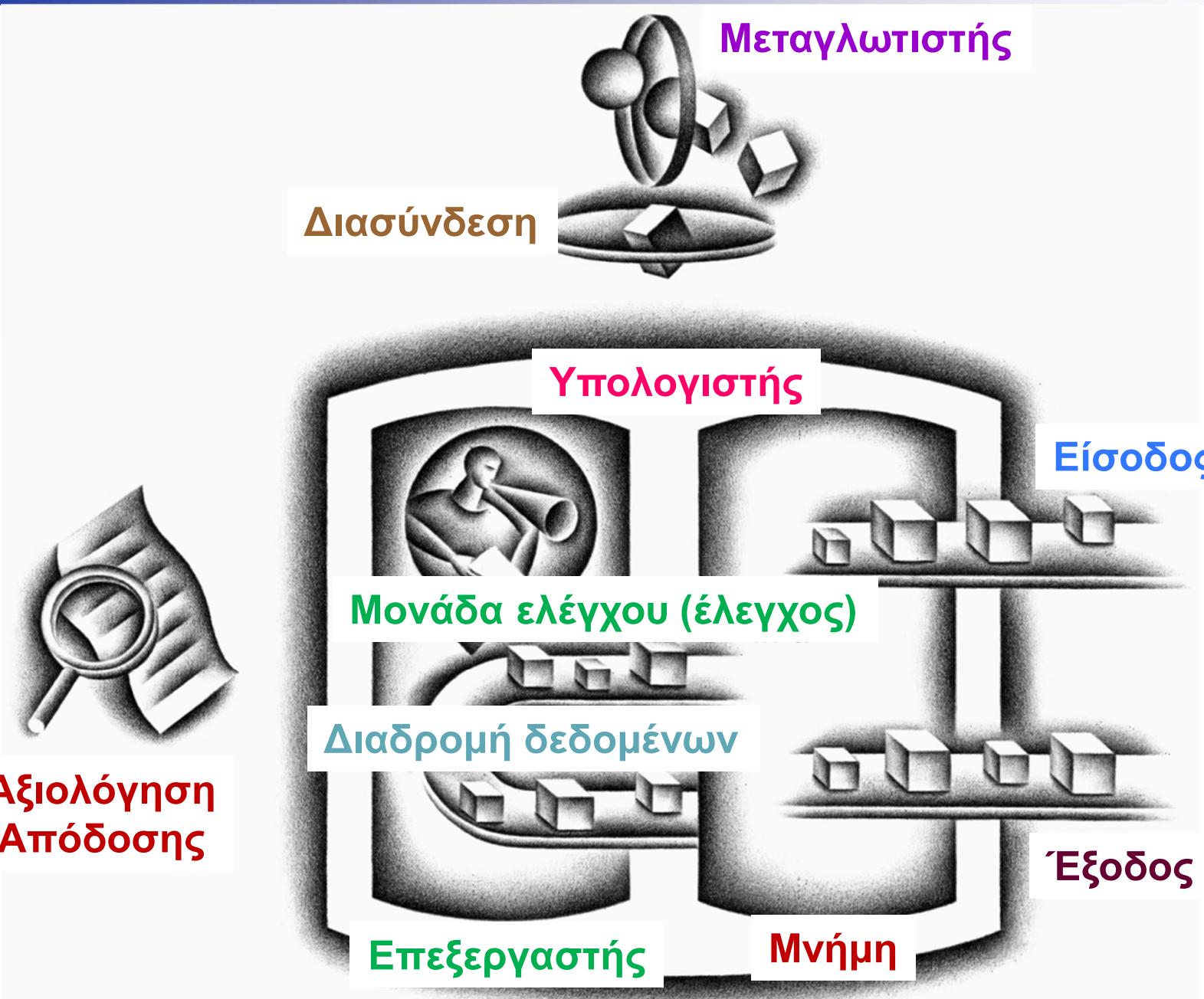
Simulators – RARS



Simulators – QtRVSim



Περιεχόμενο μαθήματος



Τι απασχολεί την Αρχιτεκτονική

■ **Σχεδίαση**

- Πόσο εύκολα και σωστά σχεδιάζεται;



■ **Απόδοση – Performance**

- Πόσο γρήγορα εκτελείται;



■ **Ενέργεια/Ισχύς – Energy/Power**

- Πόσο καταναλώνει;



■ **Αξιοπιστία/Ασφάλεια – Reliability/Security**

- Προστασία δεδομένων/πληροφορίας;



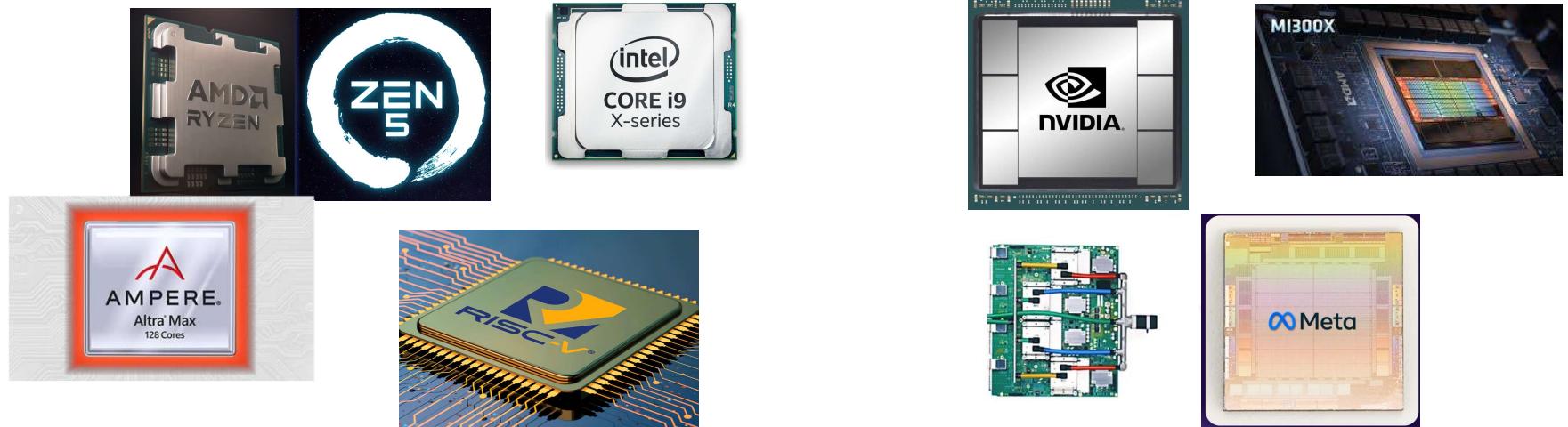
■ **Προγραμματισμότητα – Programmability**

- Πόσο εύκολα προγραμματίζεται;



Κατηγορίες «επεξεργαστών»

- CPU – central processing unit
- GPU – graphics processing unit
- DSA – domain-specific accelerator
 - TPU – tensor processing unit
 - AIA – artificial intelligence accelerator



Επίγνωση Αρχιτεκτονικής

- Γνωρίζοντας και χρησιμοποιώντας την αρχιτεκτονική της μηχανής **βελτιώνουμε την απόδοση – γρήγορα προγράμματα!**
- Παράδειγμα: **πολλαπλασιασμός πινάκων**
 - Βήμα 1 – Python
 - Βήμα 2 – C
 - Βήμα 3 – Παραλληλία δεδομένων
 - Βήμα 4 – Παραλληλία εντολών
 - Βήμα 5 – Βελτιστοποιήσεις μνήμης
 - Βήμα 6 – Παραλληλία νήματος

Πολλαπλασιασμός Πινάκων (1)

4 γραμμές κώδικα

- Γλώσσα Python

```
for i in xrange(n) :  
    for j in xrange(n) :  
        for k in xrange(n) :  
            C[i][j] += A[i][k] * B[k][j]
```

Πολλαπλασιασμός Πινάκων (2)

11 γραμμές κώδικα

- Γλώσσα C

```
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 (3)

- Παραλληλία επιπέδου δεδομένων (Data Level Parallelism)
- (Διανυσματικές επεκτάσεις - Vector extensions)
- AVX512 (x86)

15 γραμμές κώδικα

```
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 (4)

- Παραλληλία επιπέδου εντολών (Instruction Level Parallelism)
- Ξετύλιγμα βρόχου (loop unrolling)

22 γραμμές κώδικα

```
1 #include <x86intrin.h>
2 #define UNROLL (4)
3
4 void dgemm (int n, double* A, double* B, double* C)
5 {
6     for (int i = 0; i < n; i+=UNROLL*8)
7         for (int j = 0; j < n; ++j) {
8             m512d c[UNROLL];
9             for (int r=0;r<UNROLL;r++)
10                 c[r] = _mm512_load_pd(C+i+r*8+j*n); // [ UNROLL];
11
12             for( int k = 0; k < n; k++ )
13             {
14                 m512d bb = _mm512_broadcastsd_pd(_mm_load_sd(B+j*n+k));
15                 for (int r=0;r<UNROLL;r++)
16                     c[r] = _mm512_fmadd_pd(_mm512_load_pd(A+n*k+r*8+i), bb, c[r]);
17             }
18
19             for (int r=0;r<UNROLL;r++)
20                 _mm512_store_pd(C+i+r*8+j*n, c[r]);
21         }
22 }
```

Πολλαπλασιασμός Πινάκων (5)

- Βελτιστοποίηση ιεραρχίας μνήμης
- Blocking (διαχωρισμός κρυφής μνήμης – cache – σε μπλοκ

31 γραμμές κώδικα

```
1 #include <x86intrin.h>
2 #define UNROLL (4)
3 #define BLOCKSIZE 32
4 void do_block (int n, int si, int sj, int sk,
5                 double *A, double *B, double *C)
6 {
7     for ( int i = si; i < si+BLOCKSIZE; i+=UNROLL*8 )
8         for ( int j = sj; j < sj+BLOCKSIZE; j++ ) {
9             m512d c[UNROLL];
10            for (int r=0;r<UNROLL;r++)
11                c[r] = _mm512_load_pd(C+i+r*8+j*n); // [ UNROLL];
12
13            for( int k = sk; k < sk+BLOCKSIZE; k++ )
14            {
15                m512d bb = _mm512_broadcastsd_pd(_mm_load_sd(B+j*n+k));
16                for (int r=0;r<UNROLL;r++)
17                    c[r] = _mm512_fmadd_pd(_mm512_load_pd(A+n*k+r*8+i), bb, c[r]);
18            }
19
20            for (int r=0;r<UNROLL;r++)
21                _mm512_store_pd(C+i+r*8+j*n, c[r]);
22        }
23    }
24
25 void dgemm (int n, double* A, double* B, double* C)
26 {
27     for ( int sj = 0; sj < n; sj += BLOCKSIZE )
28         for ( int si = 0; si < n; si += BLOCKSIZE )
29             for ( int sk = 0; sk < n; sk += BLOCKSIZE )
30                 do_block(n, si, sj, sk, A, B, C);
31 }
```

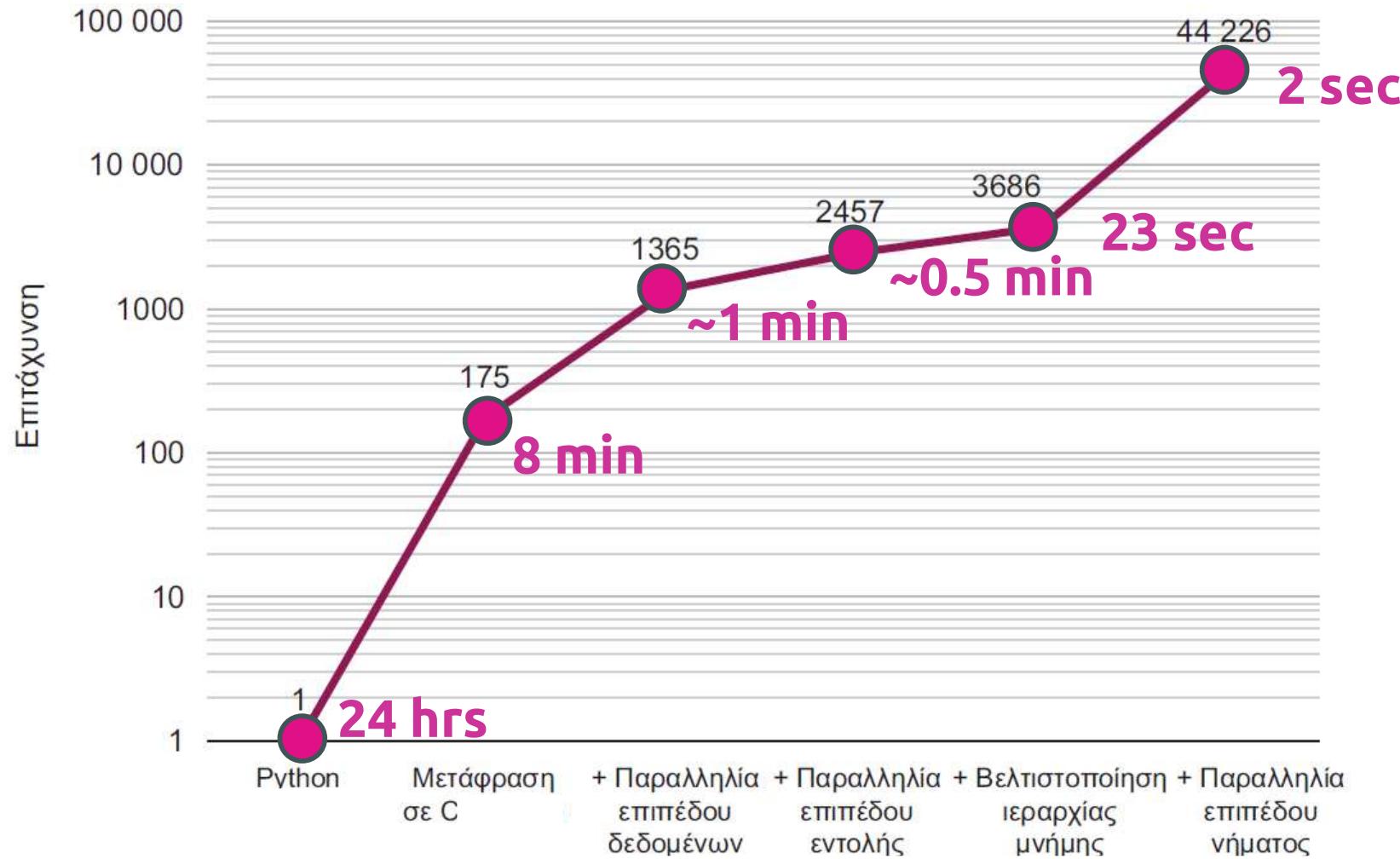
Πολλαπλασιασμός Πινάκων (6)

- Παραλληλία επιπέδου νήματος (Thread Level Parallelism)
- Πολλαπλοί πυρήνες (OpenMP – open multi-processor)

32 γραμμές κώδικα

```
1 #include <x86intrin.h>
2 #define UNROLL (4)
3 #define BLOCKSIZE 32
4 void do_block (int n, int si, int sj, int sk,
5                 double *A, double *B, double *C)
6 {
7     for ( int i = si; i < si+BLOCKSIZE; i+=UNROLL*8 )
8         for ( int j = sj; j < sj+BLOCKSIZE; j++ ) {
9             m512d c[UNROLL];
10            for (int r=0;r<UNROLL;r++)
11                c[r] = mm512_load_pd(C+i+r*8+j*n); // [ UNROLL];
12
13            for( int k = sk; k < sk+BLOCKSIZE; k++ )
14            {
15                m512d bb = mm512_broadcastsd_pd( mm_load_sd(B+j*n+k));
16                for (int r=0;r<UNROLL;r++)
17                    c[r] = _mm512_fmadd_pd(_mm512_load_pd(A+n*k+r*8+i), bb, c[r]);
18            }
19
20            for (int r=0;r<UNROLL;r++)
21                _mm512_store_pd(C+i+r*8+j*n, c[r]);
22        }
23    }
24
25 void dgemm (int n, double* A, double* B, double* C)
26 {
27 #pragma omp parallel for
28     for ( int sj = 0; sj < n; sj += BLOCKSIZE )
29         for ( int si = 0; si < n; si += BLOCKSIZE )
30             for ( int sk = 0; sk < n; sk += BLOCKSIZE )
31                 do_block(n, si, sj, sk, A, B, C);
32 }
```

Βελτίωση Απόδοσης – 44 226 x



Η Βασική Εξίσωση Απόδοσης

Χρόνος προγράμματος =

$$\begin{aligned} &= \frac{\text{εντολές}}{\text{πρόγραμμα}} \times \frac{\text{κύκλοι}}{\text{εντολή}} \times \frac{\text{sec}}{\text{κύκλος}} \\ &= \frac{\text{sec}}{\text{πρόγραμμα}} \end{aligned}$$