ΠΑΡΑΛΛΗΛΗ ΕΠΕΞΕΡΓΑΣΙΑ

Ε Ι Σ Α Γ Ω Γ Η

ΠΑΡΑΛΛΗΛΕΣ ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΕΡΓΑΣΤΗΡΙΟ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ ΥΨΗΛΩΝ ΕΠΙΔΟΣΕΩΝ

Εισαγωγή

- **“[1] Future computers of all sizes will embrace parallelism even more that they today … those who understand applications, algorithms, and architectures will be prepared for this opportunity”**
- «Οι υπολογιστές όλων των μεγεθών θα ενσωματώσουν των παραλληλισμό ακόμα περισσότερο από ότι σήμερα … αυτοί οι οποίοι κατανοούν από εφαρμογές, αλγόριθμούς και αρχιτεκτονικές θα βρεθούν προετοιμασμένοι γι’ αυτή την ευκαιρία.»

[1] Computer Organization & Design the Hardware/Software Interface, David Patterson & John Hennessy, Morgan Kaufmann Publishers, 4th ed. 2008.

Τι είναι Παράλληλη Επεξεργασία

- Η **Wikipedia** για το τι είναι Παράλληλη Επεξεργασία:

- **Parallel processing in Humans**: The ability of the brain to simultaneously process incoming stimuli. This becomes most important in vision, as the brain divides and conquers what it sees. It breaks up a scene into four components: color, motion, form, and depth. These are individually analysed and then compared to stored memories, which helps the brain identify what you are viewing. The brain then combines all of these into one image that you see and comprehend. This is a continual and seamless operation.
- **Parallel Processing in Computers:** Parallel computing is a form of computation in which many calculations are carried out simultaneously, operating on the principle that large problems can often be divided into smaller ones, which are then solved concurrently ("in parallel"). There are several different forms of parallel computing: bit-level, instruction level, data, and task parallelism. Parallelism has been employed for many years, mainly in high-performance computing, but interest in it has grown lately due to the physical constraints preventing frequency scaling. As power consumption (and consequently heat generation) by computers has become a concern in recent years, parallel computing has become the dominant paradigm in computer architecture, mainly in the form of multicore processors.
- Note that parallel processing differs from multitasking, in which a single CPU executes several programs at once???.
- Parallel processing is also called parallel computing.

Παράλληλος Υπολογισμός - Παράλληλο Σύστημα

- **Παράλληλος Υπολογισμός**

Παράλληλη υπολογισμός/επεξεργασία είναι η ταυτόχρονη πραγματοποίηση δύο ή περισσοτέρων “διαφορετικών ροών εκτέλεσης” κατά το ίδιο χρονικό διάστημα.

- **Παράλληλο Σύστημα**

Παράλληλο σύστημα έχουμε όταν συνυπάρχουν πολλαπλά αντίτυπα μονάδων υλικού (Hardware) στο ίδιο υπολογιστικό σύστημα, τα οποία έχουν τη δυνατότητα να παράγουν έργο ταυτόχρονα.

Παράλληλοι Υπολογιστές/Παράλληλοι Επεξεργαστές

Βασική ορολογία που συναντάται συχνά στην περιοχή της “Παράλληλης Επεξεργασίας”
-Τι σημαίνουν οι όροι και τι εννοούμε -

Το Υλικό – **The Hardware**

- Παράλληλος Υπολογιστής – Parallel Computer
- Παράλληλος Επεξεργαστής – Parallel Processor
- Υπερυπολογιστής – Supercomputer
- Πολυεπεξεργαστικό σύστημα – Multiprocessor
- Πολυπύρηνος επεξεργαστής – Multi-core
- Επεξεργαστής πολλαπλών – Many-core

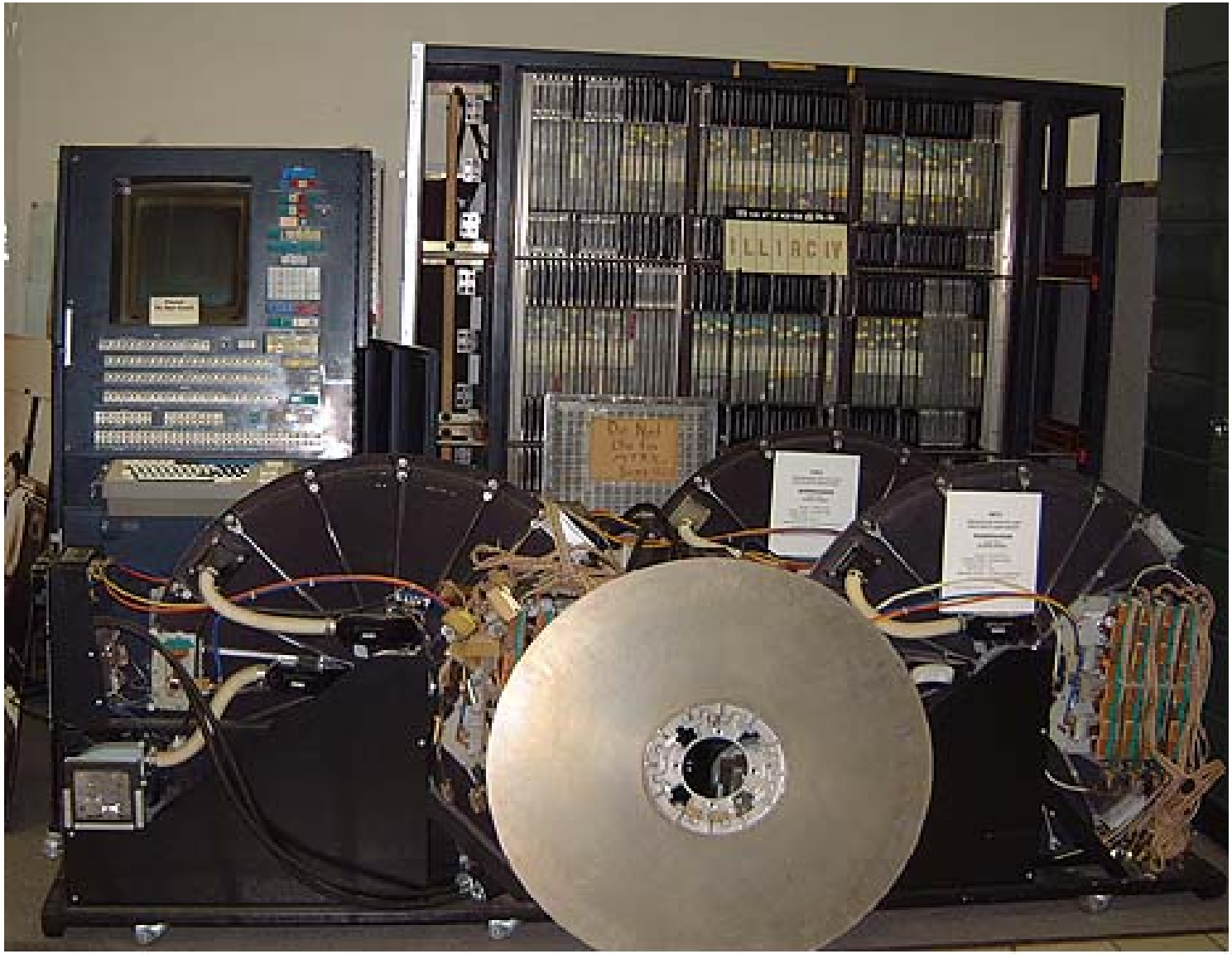
Το Λογισμικό – **The Software**

- Παράλληλος αλγόριθμος – Parallel algorithm
- Παράλληλο πρόγραμμα – Parallel program
- Παραλληλοποιητής Μεταφραστής – Parallelizing Compiler

Μερικά Ιστορικά Στοιχεία

- Οι πρώτες ιδέες για single instruction multiple data (SIMD) υπολογιστές εντοπίζονται χρονολογικά στην κατασκευή του Illiac IV.
- Ο Iliiac IV ήταν ένα από τα πρώτα ερευνητικά παράλληλα συστήματα των supercomputer projects της δεκαετίας του ‘70. Το έργο άρχισε το 1965 και “έτρεξε” την πρώτη πραγματική εφαρμογή του το 1976 (έντεκα χρόνια!!!). Το σύστημα σχεδιάστηκε για να αποδίδει 1000MFLOPS και όταν ολοκληρώθηκε απέδιδε μόνο 15MFLOPS (δηλαδή μόνο το 1,5% του αρχικού σχεδιασμού !!!). Το κόστος αρχικής εκτίμησης για την κατασκευή του ήταν 8$ εκατομμύρια και ξεπέρασε τα 31$ εκατομμύρια το 1966, τελικά όμως κατασκευάστηκε μόνο το **ένα τέταρτο!!!** της αρχικά σχεδιασμένης μηχανής. Το σύστημα χρησιμοποιούσε 64 επεξεργαστές στα 13MHz και η συνολική του μνήμη ήταν 1MB!!!.

ILLIAC IV



ILLIAC IV computer system layout composite

Photographer: Lee Jones; Date: July 13, 1972

Η Παράλληλη Επεξεργασία Σήμερα (1/4)

- **Υλικό - Hardware**

  Η πρόσκρουση στο φράγμα της μνήμης (memory wall) + στο φράγμα της ισχύος (power wall) ...

  οδήγησαν στην επανάσταση των πολυπύρηνων επεξεργαστών

  (the multi-core revolution)

Ο νόμος του Moore επαναδιατυπώνεται:

- Ο αριθμός των πυρήνων σε κάθε επεξεργαστή διπλασιάζεται περίπου κάθε δυο χρόνια (ενώ η ταχύτητα ρολογιού παραμένει σχεδόν σταθερή)

Η Παράλληλη Επεξεργασία Σήμερα (2/4)

- **Λογισμικό - Software**

  Αναζήτηση για νέα προγραμματιστικά μοντέλα.

  Νέες, ταυτόχρονες (concurrent) γλώσσες προγραμματισμού

  Παραλληλοποιητές Μεταφραστές

  Ετερογενή περιβάλλοντα: CPUs+GPUs

  Περιβάλλοντα κατανεμημένης επεξεργασίας: MPI, SDSM, PGAS.

  Πολυνηματισμός και Διεκπεραιωτική Μνήμη (transactional memory).

Η Παράλληλη Επεξεργασία Σήμερα (3/4)

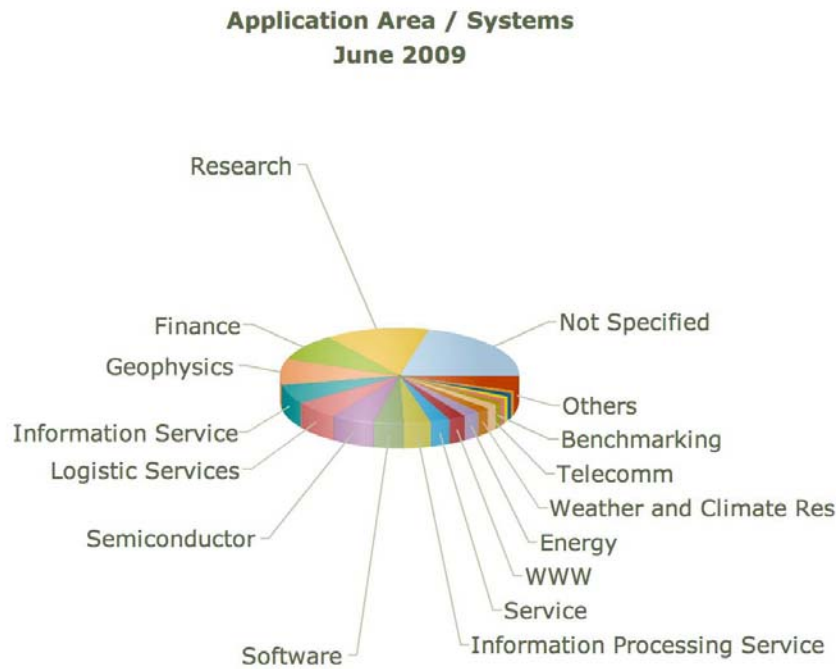
- Εφαρμογές

Από τον Υπολογισμό Υψηλών Επιδόσεων (HPC):

- Υπολογιστική Ρευστομηχανική (CFD)
- Περιβάλλον και Ενέργεια
- Υπολογιστική Χημεία
- Εξόρυξη Δεδομένων
- Παράλληλη Επεξεργασία Σημάτων
- Μοριακή Βιολογία . . .

αλλά και:

- Βάσεις δεδομένων
- Γραφικά
- Εφαρμογές ενσωματωμένων συστημάτων
- **Web 2.0**



Η Παράλληλη Επεξεργασία Σήμερα(4/4)

Οι Υπερυπολογιστές Σήμερα: (http://www.top500.org)

TOP 5

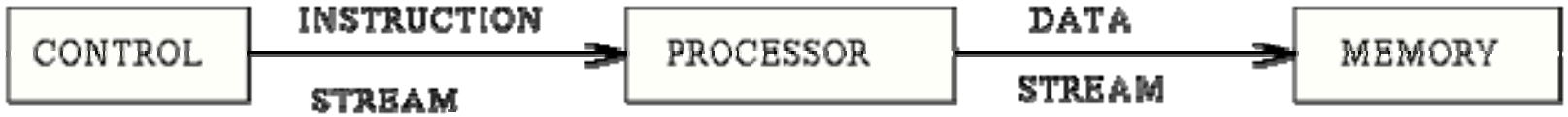
- Jaguar, USA
  - Cores: 224,256
- Roadrunner, USA
  - Cores: 122,400
- Kraken, USA
  - Cores: 99,072
- Jugene, Germany
  - Cores: 65,536
- Tianhe, China
  - CPUs: 6,144 Intel
  - GPUs: 5,120 AMD

Γενικά χαρακτηριστικά Αρχιτεκτονικών Παράλληλων Υπολογιστών

Ταξινόμηση κατά **Flynn**

- Single Instruction Stream/Single Data Stream (SISD)
  - a sequential computer.
- Multiple instruction Stream/Single Data Stream (MISD)
  - unusual.
- Single Instruction Stream/Multiple Data Stream (SIMD)
  - array processors.
- Multiple Instruction Stream/Multiple Data Stream (MIMD)
  - multiple autonomous processors simultaneously executing different instructions on different data.

Single Instruction Single Data Stream (SISD)



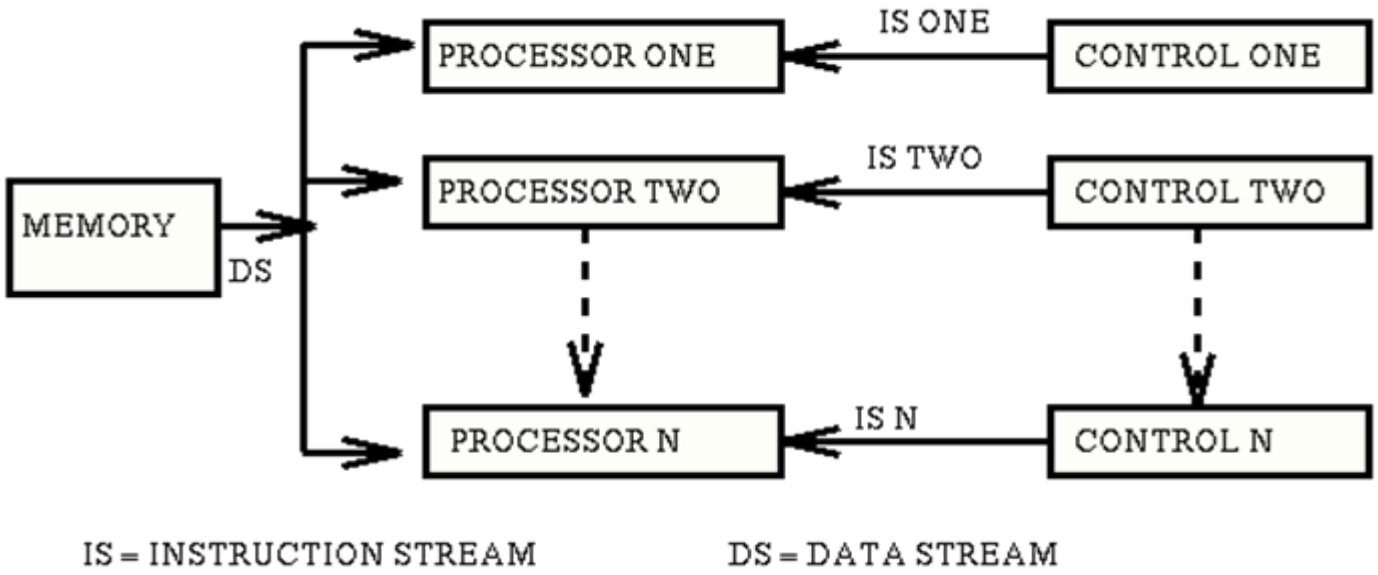
- Το κλασικό ακολουθιακό υπολογιστικό σύστημα (ένας επεξεργαστής).

Παράδειγμα

- Για τον υπολογισμό ενός αθροίσματος N = **{1, 2, …, n}** σε μια ακολουθιακή μηχανή, ο επεξεργαστής χρειάζεται να προσπελάσει τη μνήμη του συστήματος **n** φορές, και να υλοποιήσει **n-1** προσθέσεις σε **(n-1)*t_0** χρόνο. **t_0** είναι ο στοιχειώδεις χρόνος που απαιτείται για μια πρόσθεση **a+a,** όπου **a= 1, 2, …, n.**

Multiple instruction/Single Data Stream (MISD)

- Στις MISD μηχανές υπάρχουν N Processors και N Control Units και η ίδια κοινή μνήμη (Memory module). Εφαρμόζονται N Instruction Streams IS στο ίδιο Stream of Data

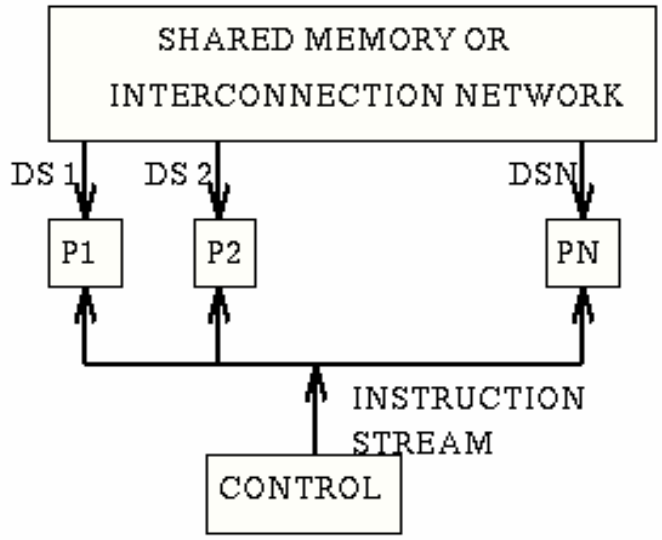


Παράδειγμα

Ο έλεγχος για το αν κάποιος αριθμός Z, είναι πρώτος.

Single Instruction/Multiple Data (SIMD)

Στις SIMD μηχανές υπάρχουν πολλά ίδια επεξεργαστικά στοιχεία (N Processors), κάτω από τον έλεγχο μίας μονάδας ελέγχου (Control Unit). Οι επεξεργαστές λειτουργούν συγχρονισμένα κάτω από τον έλεγχο ενός κεντρικού ρολογιού (global clock).



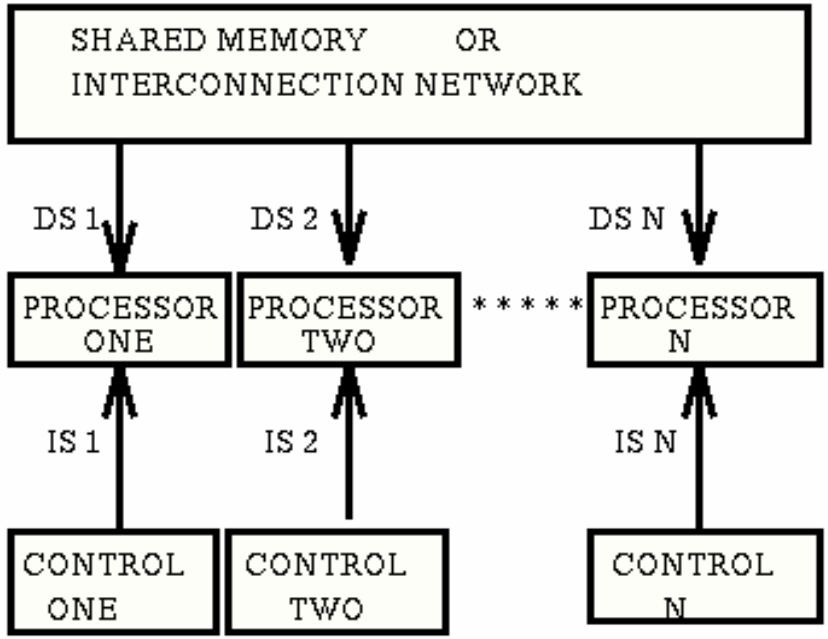
Παράδειγμα

Πρόσθεση δύο πινάκων A, B. A+B=C

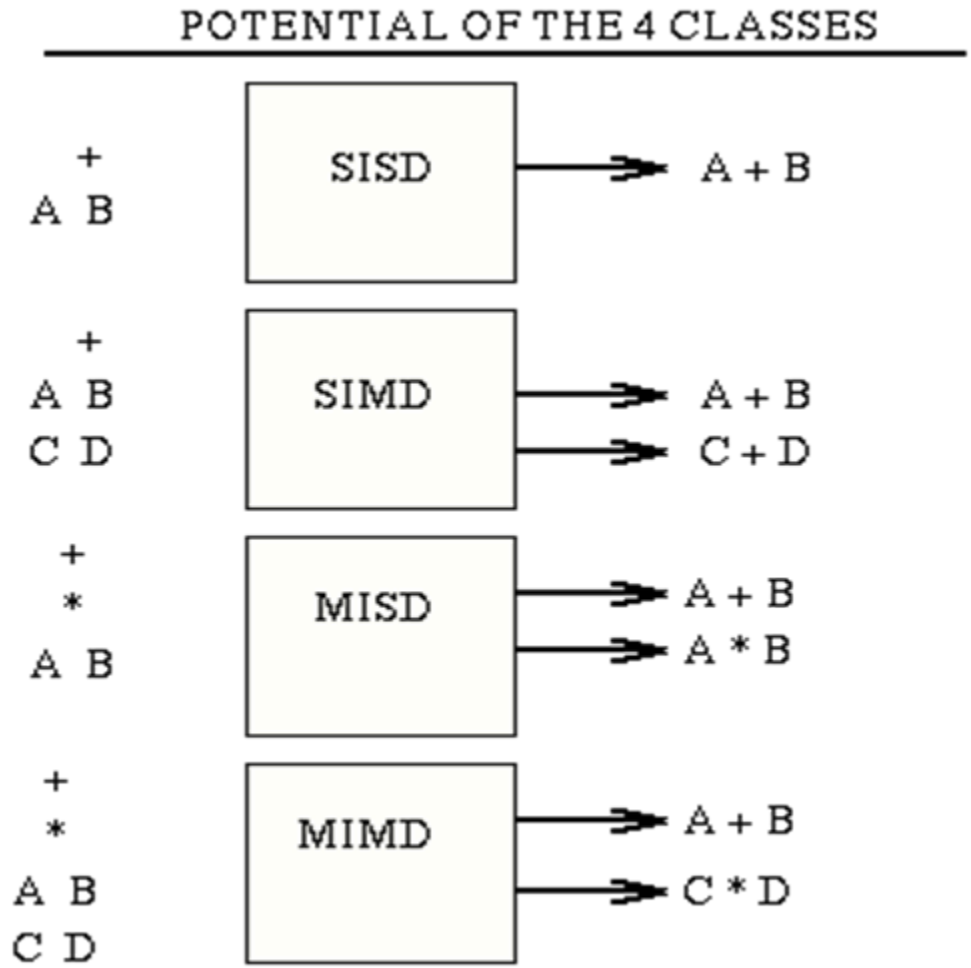
Multiple Instruction/Multiple Data (MIMD)

(MIMD) Η ταξινόμηση κατά flynn ενός παράλληλου υπολογιστικού συστήματος, όπου πολλές λειτουργικές μονάδες (functional units) εκτελούν διαφορετικές λειτουργίες σε διαφορετικά δεδομένα (different operations on different data), κατά το ίδιο όμως χρονικό διάστημα.

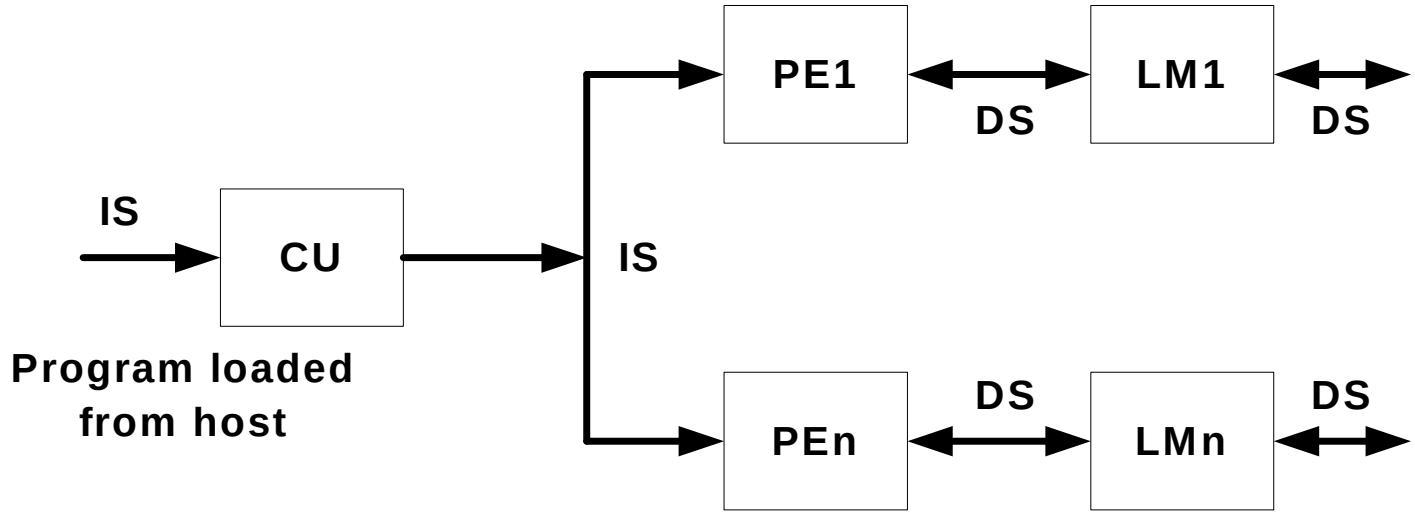
Ένα απτό παράδειγμα, αποτελεί ένα δίκτυο από σταθμούς εργασίας.



Απλή αναπαράσταση της κατά **Flynn** Ταξινόμησης



SIMD Multiprocessor Computers



SIMD Architecture

Παράδειγμα (1/6)

- Έστω ότι έχουμε να αθροίσουμε 100000 αριθμούς σε ένα SIMD σύστημα με 100 επεξεργαστικά στοιχεία (PE).
  - Το πρώτο βήμα είναι να μοιράσουμε τους 100000 αριθμούς σε 100 ανεξάρτητες υποομάδες, μια υποομάδα ανά PE.
  - Ο ένας από τους επεξεργαστές τοποθετεί κάθε ομάδα αριθμών στην τοπική μνήμη του κάθε PE.

Παράδειγμα (2/6)

- Αν οι 100000 αριθμοί είναι αρχικά τοποθετημένοι στον host i, στον πίνακα A, ονομάζουμε A_i τον πίνακα στην τοπική μνήμη του PE_i στοιχείου και τοποθετούμε τους 1000 αριθμούς που του αντιστοιχούν στον τοπικό του πίνακα, A_i.
- Το επόμενο βήμα είναι να υπολογίσουμε το άθροισμα για κάθε υποσύνολο αριθμών. Αυτό το βήμα, που είναι και το πρώτο κομμάτι του SIMD κώδικα, είναι απλά ένας βρόγχος (loop) τον οποίο κάθε PE θα πρέπει να εκτελέσει. «Διάβασε μια λέξη (τιμή) από την τοπική μνήμη και πρόσθεσε την σε μία **τοπική μεταβλητή**».

Παράδειγμα (3/6)

```
sum = 0;
for (i = 0; i < 1000; i = i + 1)
  /*loop over each array*/
  sum = sum + A1[i];/*sum the local arrays*/
```

Παράδειγμα (4/6)

- Το τελευταίο βήμα είναι η άθροιση των 100 επιμέρους μερικών αθροισμάτων.
- Κάθε μερικό άθροισμα βρίσκεται σε διαφορετική επεξεργαστική μονάδα. Έτσι, θα πρέπει να χρησιμοποιήσουμε το διασυνδετικό δίκτυο (ΔΔ) του συστήματος για να μετακινήσουμε τα μερικά αθροίσματα σε ένα από τα επεξεργαστικά στοιχεία του συστήματος, ώστε να εκτελεστεί ο υπολογισμός του τελικού αθροίσματος.
- Αντί όμως να στείλουμε τα μερικά αθροίσματα σε μία επεξεργαστική μονάδα υλοποιώντας τελικά ακολουθιακό τρόπο άθροισης, μοιράζουμε την διαδικασία αυτή μεταξύ των PE διαιρώντας κάθε φορά των αριθμό των εμπλεκομένων επεξεργαστικών στοιχείων δια δύο, PE/2.

Παράδειγμα (5/6)

- Έτσι με P_n θα συμβολίσουμε τον αριθμό των PEs.
- Η `send(x,y)` θα υποθέσουμε ότι είναι μία συνάρτηση η οποία αναλαμβάνει να στείλει μέσω του ΔΔ στο PE_x, (`x=1,…,n`) την τιμή y, και `receive(y)` θα είναι μία άλλη συνάρτηση η οποία παραλαμβάνει από το δίκτυο μία τιμή, από το PE_y για το PE στο οποίο εκτελείται (PE_x).

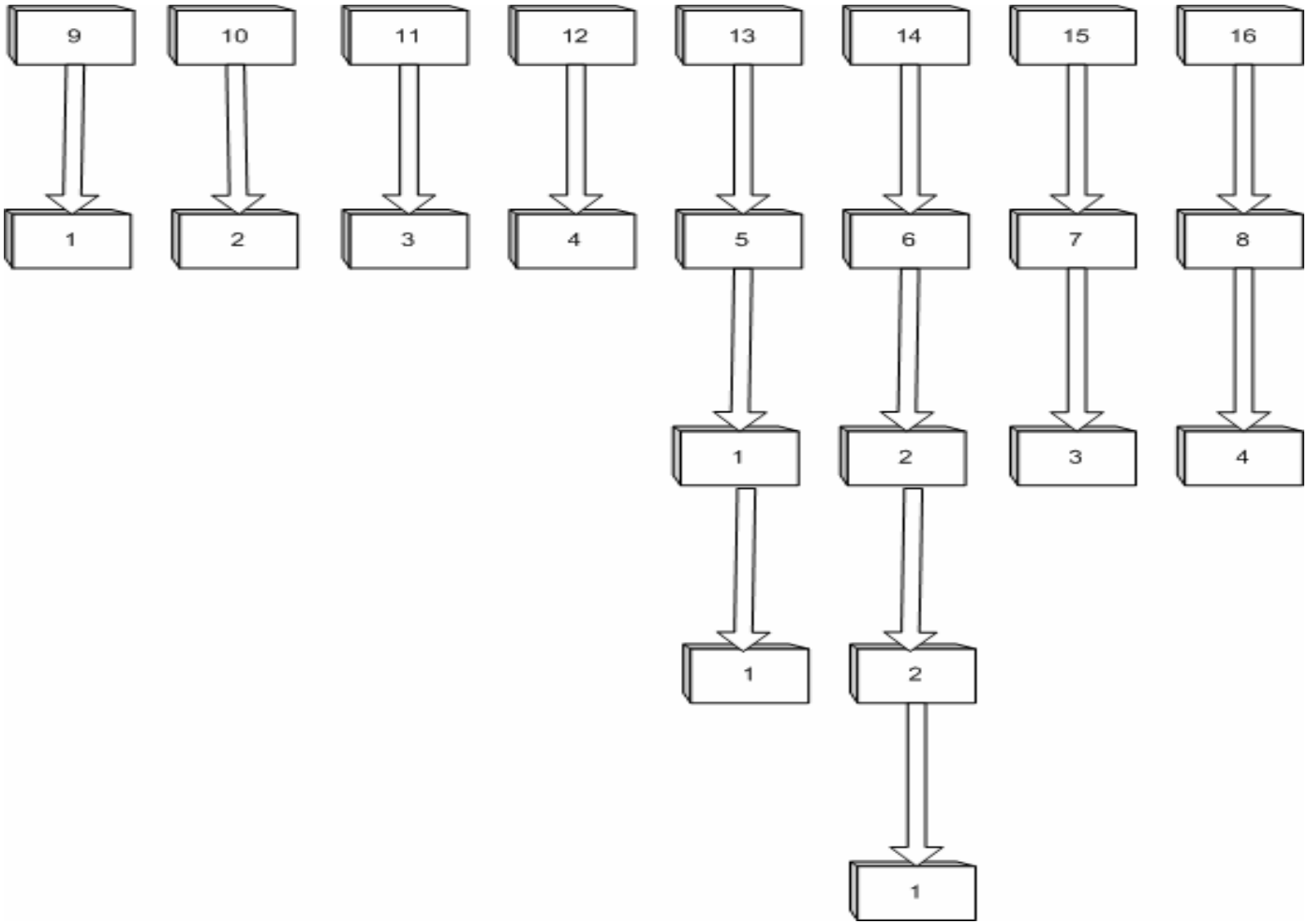
Παράδειγμα (6/6)

Συνάρτηση για τον υπολογισμό των επιμέρους αθροισμάτων

```
limit = 100; /*100 execution units in SIMD*/
half = 0;
repeat
  half = limit/2; /*send vs receive dividing line*
  if (Pn >= half && Pn < limit)
     send(Pn%half,sum);
  if (Pn < half) sum = sum + receive();
  limit = half; /*upper limit of senders*/
until (half == 1); /*exit with final sum*/
```

Σχηματική αναπαράσταση μεταφοράς μερικών αθροισμάτων μεταξύ των **PE** του συστήματος

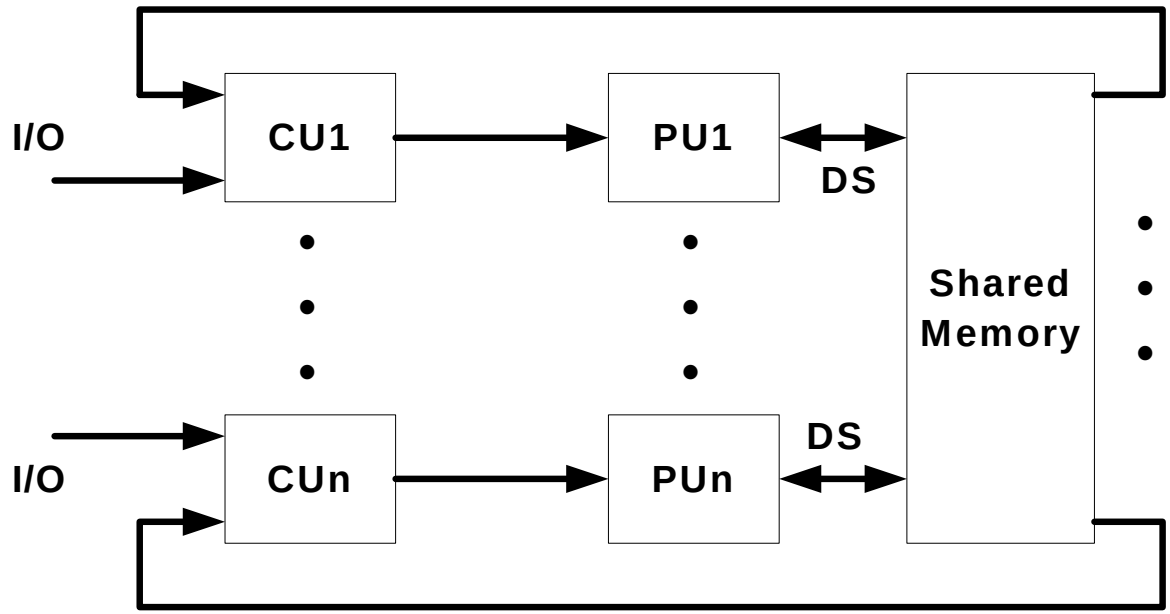
Για σύστημα με 16 επεξεργαστικά στοιχεία



Trade-off

- Το βασικό trade-off στις SIMD μηχανές, είναι η απόδοση του επεξεργαστή σε σχέση με τον αριθμό των επεξεργαστών που διαθέτει το σύστημα.
- Για παράδειγμα η Connection Machine 2 (CM-2) διέθετε 65.536 single-bit-wide processors, ενώ ο Illiac IV διέθετε 64 επεξεργαστές των 64-bit.

MIMD Multiprocessor Systems



MIMD Architecture

Παράδειγμα (1/4)

- Ας θεωρήσουμε το προηγούμενο παράδειγμα μας, της πρόσθεσης 100.000 αριθμών, και ας υποθέσουμε ότι θέλουμε να υλοποιήσουμε την πρόσθεση σε MIMD σύστημα με 10 επεξεργαστές.
- Το πρώτο βήμα πάλι είναι να διαχωριστούν οι 100.000 αριθμοί σε υποσύνολα του ιδίου πλήθους και να κατανεμηθούν στους 10 επεξεργαστές του συστήματος.
- Τώρα όμως δεν χρειάζεται να μεταφέρουμε τα δεδομένα μας μεταξύ των επεξεργαστών του συστήματος αφού οι MIMD μηχανές έχουν κοινή μνήμη. Απλά δίνουμε διαφορετική αρχική διεύθυνση μνήμης στους επιμέρους επεξεργαστές.

Παράδειγμα (2/4)

- Συμβολίζουμε ξανά με P_n τον αριθμό των επεξεργαστών και τους αριθμούμε από 0 έως 9.
- Όλοι οι επεξεργαστές αρχίζουν την εκτέλεση του προγράμματος με την εκτέλεση του βρόχου (loop) που αθροίζει το σύνολο των αριθμών που αντιστοιχεί σε κάθε επεξεργαστή.

Παράδειγμα (3/4)

```
sum[Pn] = 0;
for (i=10000*Pn; i < 10000*(Pn+1); i = i +1)
  sum[Pn] = sum[Pn] + A1[i]; /*sum the assigned
  memory areas*/
```

Παράδειγμα (4/4)

```
half = 100; /*10 processors in single-bus MIMD*/
repeat
  synch(); /*wait for completion of parallel sums*/
  half = half/2; /*dividing line of who sums*/
  if (Pn <= half) sum[Pn] = sum[Pn] + sum[(2*Pn)-1];
until (half == 1); /*exit with final sum in Sum[0]*/
```