

ΗΥ 134

Εισαγωγή στην Οργάνωση και  
στον Σχεδιασμό Υπολογιστών Ι

Διάλεξη 14

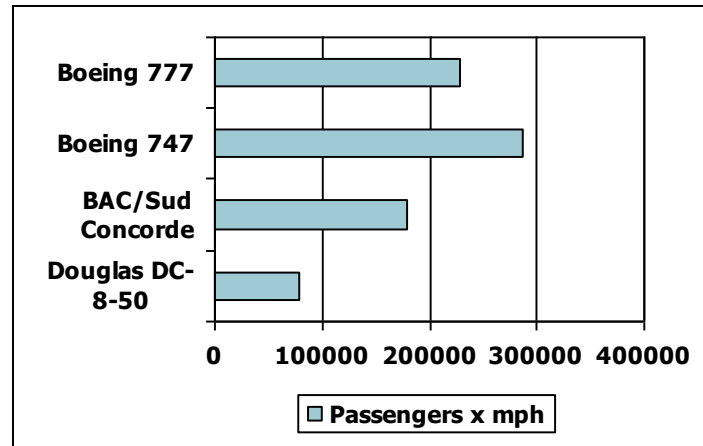
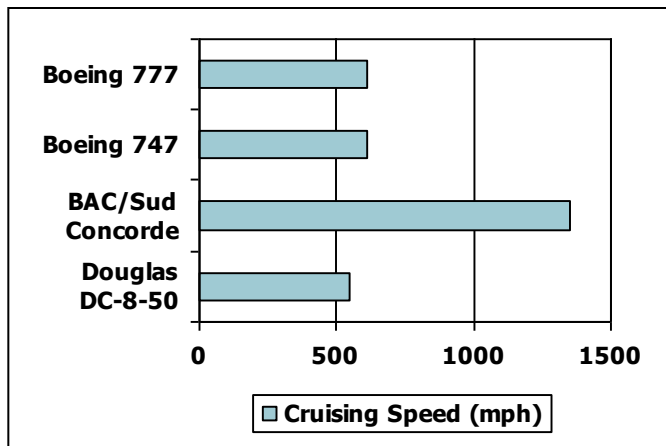
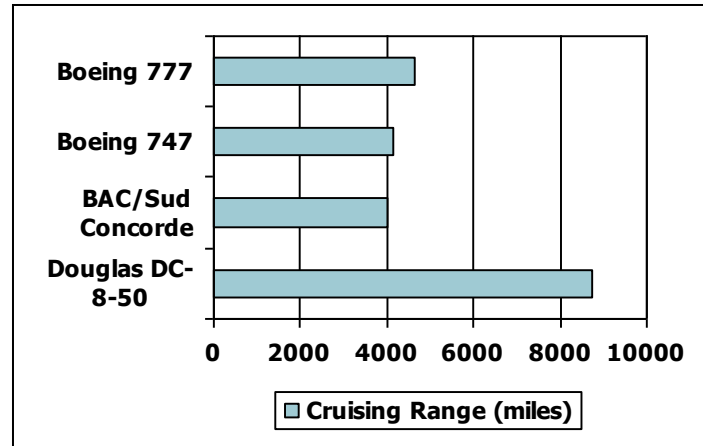
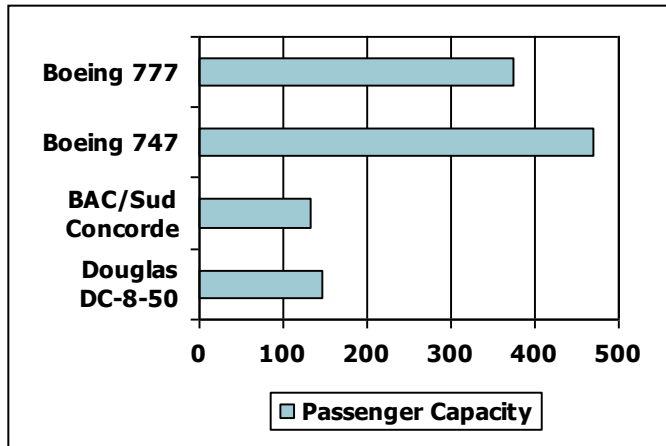
Απόδοση Υπολογιστικών Συστημάτων

Νίκος Μπέλλας

Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων

# Τι σημαίνει απόδοση;

- Ποιό αεροπλάνο έχει καλύτερη απόδοση?



# Μετρικές Απόδοσης

- *Χρόνος Απόκρισης (ή Χρόνος Εκτέλεσης)*
  - Latency (Execution time)
  - Πόσος χρόνος απαιτείται για την εκτέλεση μιας εργασίας
- *Throughput*
  - Συνολική ποσότητα εργασίας που διεκπεραιώνεται σε ένα δεδομένο χρόνο
    - πχ εργασίες/ώρα, bytes/sec, κοκ.
- *Ο Χρόνος Απόκρισης και το Throughput δεν είναι το ίδιο σημαντικές μετρικές για διαφορετικά συστήματα*
  - Database transaction systems (latency)
  - Graphics rendering system (GPU) (throughput)

# Συγκριτική Απόδοση

- Ορισμός:  $\text{Απόδοση} = 1 / \text{Χρόνος Εκτέλεσης}$
- “Το ότι το σύστημα  $X$  είναι  $n$  φορές πιο γρήγορο από το σύστημα  $Y$  σημαίνει: ”

$$\text{Απόδοση}_X / \text{Απόδοση}_Y =$$

$$\text{Χρόνος Εκτέλεσης}_Y / \text{Χρόνος Εκτέλεσης}_X = n$$

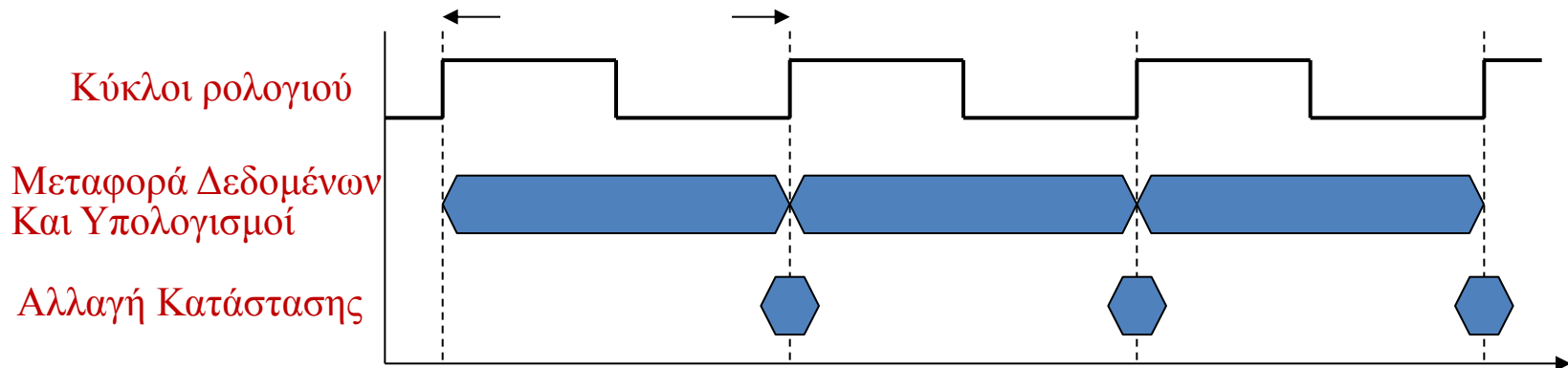
- Για παράδειγμα, ο χρόνος εκτέλεσης ενός προγράμματος:
  - 10s on A, 15s on B
  - $\text{Execution Time}_B / \text{Execution Time}_A$   
 $= 15s / 10s = 1.5$
  - Ο A είναι 1.5 φορές γρηγορότερος από τον B

# Μέτρηση Χρόνου Εκτέλεσης

- **Παρελθών Χρόνος (Elapsed Time)**
  - Συνολικός χρόνος εκτέλεσης μιάς εργασίας που περιλαμβάνει: επεξεργασία, Input/Output (πχ προσπέλαση στον σκληρό δίσκο), κλήσεις στο λειτουργικό σύστημα, κοκ.
- **Χρόνος CPU (CPU time)**
  - Περιλαμβάνει μόνο τον χρόνο που η CPU επεξεργάζεται μια διεργασία
    - Δεν περιλαμβάνει Input/Output ή χρόνο που η CPU ασχολείται με άλλες διεργασίες.
  - $\text{CPU time} = \text{User CPU time} + \text{System CPU time}$

# Ρολόι CPU

- Η λειτουργία κάθε ψηφιακού συστήματος βασίζεται σε ένα ρολόι που τρέχει με σταθερό ρυθμό και προσδιορίζει πότε προκύπτουν διάφορα συμβάντα στο hardware



- Περίοδος Ρολογιού (clock period): ο χρόνος για έναν πλήρη κύκλο ρολογιού
  - πχ,  $250\text{ps} = 0.25\text{ns} = 250 \times 10^{-12}\text{s}$
- Συχνότητα ρολογιού (clock frequency or rate): κύκλοι ρολογιού ανά μονάδα χρόνου =  $1/\text{Περίοδος}$ 
  - πχ,  $4.0\text{GHz} = 4000\text{MHz} = 4.0 \times 10^9\text{Hz}$
  - Αντιστοιχεί σε περίοδο  $1/4\text{GHz} = 0.25\text{ns}$

# Χρόνος CPU

$$\begin{aligned}\text{Χρόνος CPU} &= \text{Αριθμός Κύκλων CPU} \times \text{Περίοδος Ρολογιού} \\ &= \frac{\text{Αριθμός Κύκλων CPU}}{\text{Συχνότητα Ρολογιού}}\end{aligned}$$

- Η απόδοση του συστήματος μπορεί να βελτιωθεί με το να:
  - Μειώσουμε τον αριθμό των κύκλων CPU που απαιτούνται για να εκτελεσθεί το πρόγραμμα
  - Αυξήσουμε την συχνότητα ρολογιού
  - Αυτά τα δύο δεν είναι όμως ανεξάρτητα

# Παράδειγμα χρόνου CPU

- Υπολογιστής A: Ρολόι 2GHz, Χρόνος CPU 10s
- Θέλουμε να σχεδιάσουμε έναν Υπολογιστή B
  - Στόχος μας να πέσουμε στα 6s χρόνου CPU
  - Πιο γρήγορο ρολόι απαιτεί 1.2X περισσότερους κύκλους
  - **Ερώτημα:** Πόσο πρέπει να είναι το ρολόι του Υπολογιστή B;

$$\text{Συχνότητα}_B = \frac{\text{Αριθμός Κύκλων CPU}_B}{\text{Χρόνος CPU}_B}$$

$$= \frac{1.2 \times \text{Αριθμός Κύκλων CPU}_A}{6s}$$

$$\begin{aligned} \text{Αριθμός Κύκλων CPU}_A &= \text{Χρόνος CPU}_A \times \text{Συχνότητα}_A \\ &= 10s \times 2\text{GHz} = 20 \times 10^9 \end{aligned}$$

$$\text{Συχνότητα}_B = \frac{1.2 \times 20 \times 10^9}{6s} = \frac{24 \times 10^9}{6s} = 4\text{GHz}$$



# Με λίγη περισσότερη λεπτομέρεια

$$\begin{aligned}\text{Χρόνος CPU} &= \text{Αριθμός Εντολών στο Πρόγραμμα} \times \text{CPI} \times \text{Περίοδος Ρολογιού} \\ &= \frac{\text{Αριθμός Εντολών} \times \text{CPI}}{\text{Συχνότητα Ρολογιού}}\end{aligned}$$

- **Αριθμός Εντολών στο Πρόγραμμα**
  - Εξαρτάται από το πρόγραμμα, την αρχιτεκτονική συνόλου εντολών (πχ MIPS, x86, ARM) και τον compiler.
- **CPI ή Cycles per Instruction**
  - Πόσες εντολές μπορεί να εκτελέσει ένας επεξεργαστής σε έναν κύκλο μηχανής.
  - Πάντα θεωρούσαμε  $\text{CPI}=1$  στο HY134.
  - Πιο πολύπλοκες εντολές μπορεί να έχουν  $\text{CPI}>1$

# Παράδειγμα

- Δύο υπολογιστές MIPS τρέχουν το ίδιο πρόγραμμα
- Υπολογιστής A: Περίοδος = 250ps, CPI = 2.0
- Υπολογιστής B : Περίοδος = 500ps, CPI = 1.2
- Ποιος είναι πιο γρήγορος και κατά πόσο;

$$\begin{aligned}\text{ΧρόνοςCPU}_A &= \text{ΑριθμόςΕντολ.} \times \text{CPI}_A \times \text{Περίοδος}_A \\ &= AE \times 2.0 \times 250\text{ps} = AE \times 500\text{ps}\end{aligned}$$

A πιο  
γρήγορο

$$\begin{aligned}\text{ΧρόνοςCPU}_B &= \text{ΑριθμόςΕντολ.} \times \text{CPI}_B \times \text{CycleTime}_B \\ &= AE \times 1.2 \times 500\text{ps} = AE \times 600\text{ps}\end{aligned}$$

$$\frac{\text{ΧρόνοςCPU}_B}{\text{ΧρόνοςCPU}_A} = \frac{AE \times 600\text{ps}}{AE \times 500\text{ps}} = 1.2$$

κατά 1.2X

# Παράδειγμα Υπολογισμού Μέσου CPI

- Έστω ότι τα προγράμματα MIPS που τρέχουμε αποτελούνται από τρεις κατηγορίες εντολών A, B, C.

Κατηγορία Εντολών	A	B	C
CPI για κατηγορία	1	2	3
ΑΕ στο Προγρ. 1	2	1	2
ΑΕ στο Προγρ. 2	4	1	1

## ■ Πρόγραμμα 1: $AE = 5$

- Αριθμός Κύκλων  
 $= 2 \times 1 + 1 \times 2 + 2 \times 3$   
 $= 10$
- Μέσο CPI =  $10/5 = 2.0$

## ■ Πρόγραμμα 2 : $AE = 6$

- Αριθμός Κύκλων  
 $= 4 \times 1 + 1 \times 2 + 1 \times 3$   
 $= 9$
- Μέσο CPI =  $9/6 = 1.5$

# Παράδειγμα κώδικα MIPS (I)

Συνάρτηση καθαρισμού (*clear*) μίας περιοχής μνήμης

```
clear1(int array[], int size) {  
    int i;  
    for (i = 0; i < size; i += 1)  
        array[i] = 0;  
}
```

```
clear2(int *array, int size) {  
    int *p;  
    for (p = &array[0]; p < &array[size];  
        p = p + 1)  
        *p = 0;  
}
```

```
    move $t0,$zero    # i = 0  
loop1: sll $t1,$t0,2   # $t1 = i * 4  
    add $t2,$a0,$t1   # $t2 =  
                    # &array[i]  
    sw $zero, 0($t2) # array[i] = 0  
    addi $t0,$t0,1   # i = i + 1  
    slt $t3,$t0,$a1  # $t3 =  
                    # (i < size)  
    bne $t3,$zero,loop1 # if (...)  
                    # goto loop1
```

```
    move $t0,$a0     # p = & array[0]  
    sll $t1,$a1,2    # $t1 = size * 4  
    add $t2,$a0,$t1 # $t2 =  
                    # &array[size]  
loop2: sw $zero,0($t0) # Memory[p] = 0  
    addi $t0,$t0,4   # p = p + 4  
    slt $t3,$t0,$t2 # $t3 =  
                    # (p<&array[size])  
    bne $t3,$zero,loop2 # if (...)  
                    # goto loop2
```

Αριθμός εντολών που εκτελούνται, *clear1*:  $6N+1$ , *clear2*:  $4N+3$

# Παράδειγμα κώδικα MIPS (II)

- Έστω ότι θέλουμε να τρέξουμε *clear1* και *clear2* για  $N=10^6$  σε δύο υπολογιστές MIPS
- Υπολογιστής A (*clear1*): Συχνότητα = 1 GHz, CPI = 2.0
- Υπολογιστής B (*clear2*) : Περίοδος = 500 MHz, CPI = 2.0
- Ποιος τρέχει πιο γρήγορα το πρόγραμμα και κατά πόσο;

$$\text{ΧρόνοςCPU}_A = \text{ΑριθμόςΕντολών}_A \times \text{CPI}_A \times \text{Περίοδος}_A$$

$$= (6 * 10^6 + 1) \times 2.0 \times 1 \text{ ns} = 12 \text{ us}$$

Α πιο  
γρήγορος

$$\text{ΧρόνοςCPU}_B = \text{ΑριθμόςΕντολών}_B \times \text{CPI}_B \times \text{CycleTime}_B$$

$$= (4 * 10^6 + 3) \times 2.0 \times 2 \text{ ns} = 16 \text{ us}$$

κατά 1.33X

$$\frac{\text{ΧρόνοςCPU}_B}{\text{ΧρόνοςCPU}_A} = \frac{16}{12} = 1.33$$

# Απόδοση Συστήματος

- Η απόδοση ενός συστήματος είναι ίση με

$$\text{Χρόνος CPU} = \text{Αριθμός Εντολών} \times \text{CPI} \times \text{Περίοδος Ρολογιού}$$

και εξαρτάται από τα παρακάτω:

- Αλγόριθμο
  - Καθορίζει τον αριθμό των εντολών που εκτελούνται
- Compiler, Γλώσσα Προγραμματισμού (πχ C, Assembly)
  - Καθορίζει τον αριθμό των εντολών που εκτελούνται ΚΑΙ το CPI
- Hardware, Αρχιτεκτονική
  - Καθορίζει την περίοδο του ρολογιού