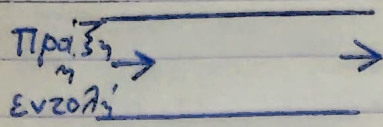
Αρχιτεκτονική Διάλεξη 13
Pipeline → Διασωλήνωση



• Δίνεται η πράξη
$(A_i \cdot B_i) + C_i$, $i = 1$ εως 7 Αρα έχουμε 7 tasks 1-7. Τα A, B, C βρίσκονται στη μνήμη

Δουλειες
1) Φόρτωση των A, B απο την μνήμη
2) Πολλαπλασιασμος των A, B και φόρτωση C απο την μνήμη
3) Προσθεση $(A \cdot B)$ με το C
4) Αποθήκευση στη μνήμη

Segments - pipeline

Σειριακά: Segments · tasks $= 4 \cdot 7 = 28$

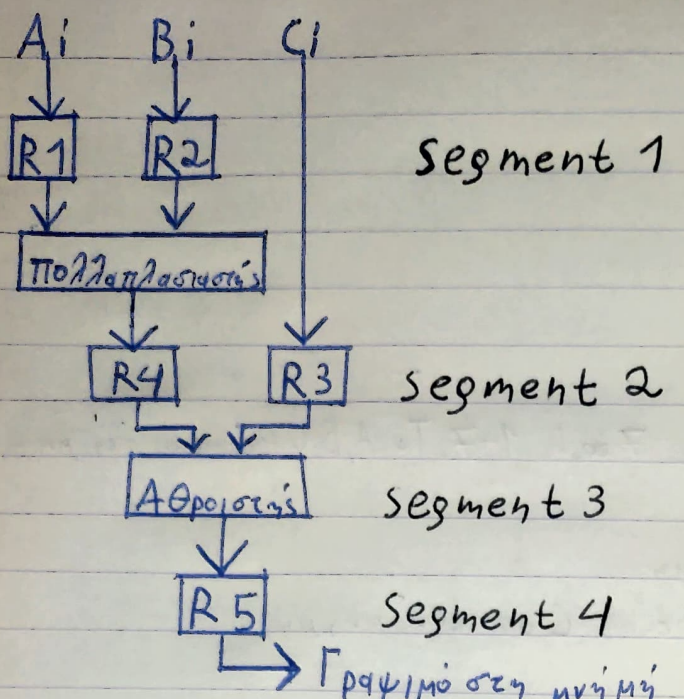
1) Segment: Τμήμα hardware αυτόνομο R1 ← διαβάζει τιμές του διανύσματος A
R2 ← διαβάζει τιμές του διανύσματος B

Segment 1 (R1 R2) Δηλαδή segment 1 διαβάζει σε ίσα όρια χρονικό παλμό 2 τιμές A_i, B_i

Segment 2 (R3 R4) R3 διαβάζει το C_i
ο πολλαπλασιαστης εκτελεί πολλαπλασιασμούς
R4 υποδεχεται το αποτελεσμα πολλαπλασιασμού

Segment 3 (αθροιστης R5) αποτέλεσμα προσθεσης

Segment 4 (R6) Επιστροφή αποτελεσμάτων στην μνήμη



segment	1	2	3	4	5	6	7	8	9	10 t
S1	T1	T2	T3	T4	T5	T6	T7			
S2		T1	T2	T3	T4	T5	T6	T7		
S3			T1	T2	T3	T4	T5	T6	T7	
S4				T1	T2	T3	T4	T5	T6	T7

Εργασίες $T_1 - T_7$

$T_1 = A_1 \cdot B_1 + C_1$

$T_2 = A_2 \cdot B_2 + C_2$

$T_3 = A_3 \cdot B_3 + C_3$

$T_4 = A_4 \cdot B_4 + C_4$

$T_5 = A_5 \cdot B_5 + C_5$

$T_6 = A_6 \cdot B_6 + C_6$

$T_7 = A_7 \cdot B_7 + C_7$

Το pipeline έχει χαμηλή. Από εδώ και κάτω ισθε χρονικος παλμος (t) ολοκληρώνει μια πράξη

- Τα T που αναγραφονται στο χρονοδιάγραμμα αφορουν τις εργασίες που χάνει ισθε Segment για ενα task.

Π.Χ. χρονος = 5*

1) Το S1 διαβάζει τα A5, B5 (αναγνωση στοιχειών για το T5)
2) Το S2 υπολογίζει A4, B4 και διαβάζει C4
3) Το S3 προσθέτει τα $A_3 \cdot B_3 + C_3$
4) Το S4 γράφει το αποτελεσμα $A_2 \cdot B_2 + C_2$

Θεωρουμε οτί οι χρόνοι τον T είναι ίση (αυτο δεν ισχύει)

πλήθος task + πλήθος segment − 1 = t

7 + 4 − 1 = 10 ← στο παράδειγμα μας

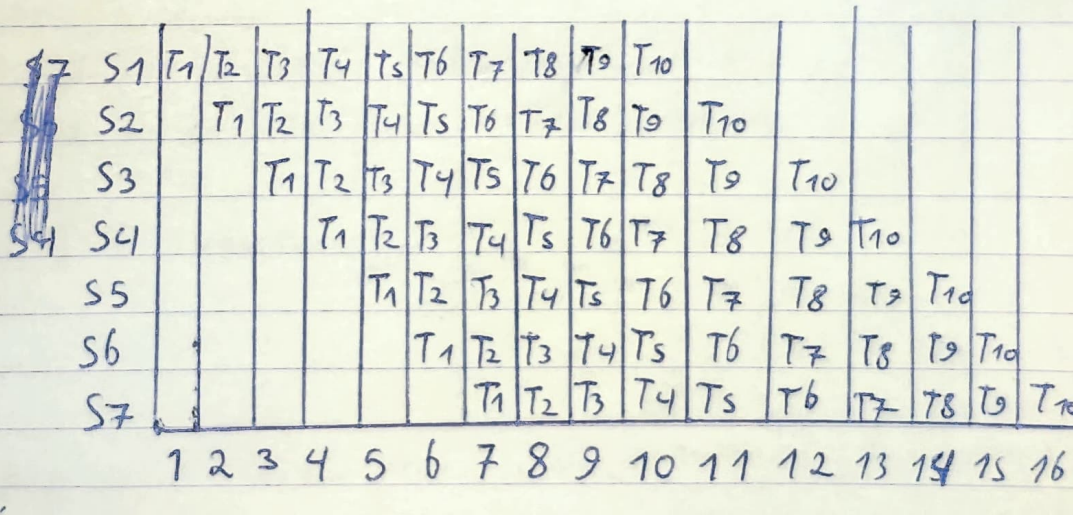
$$\text{Speedup} = \frac{\text{Χρόνος Pipeline}}{\text{Χρόνος με σειριακή εκτέλεση}} = \frac{10}{28}$$

- Δίνονται 10 αριθμοί A_i και 10 αριθμοί B_i κινητής υποδιαστολής και θέλουμε να εκτελέσουμε τις πράξεις $C_i = A_i = B_i$

1) Να ορίσετε segments (πλήθος/hardware)
2) Να δείξετε το διάγραμμα χρονισμού
3) Να βρείτε το Speedup

	Hardware
1) Ανάγνωση των A_i, B_i	R1, R2
2) Συγκριση εκθετών	συγκριτής, R3 (αποθηκεύει το αποτέλεσμα της σύγκρισης 0 αν οι εκθέτες διαφέρουν, 1 αν είναι ίσοι)
3) Εξίσωση εκθετών, αλλαγή κλασματικού μέρους (ολίσθηση)	Αθροιστής — κύκλωμα ολίσθησης που προσθέτει μηδενικά στο άλλο τμήμα
4) Πρόσθεση κλασματικών τμημάτων	
5) Κανονικοποίηση αποτελέσματος	(Εκθέτης = Εκθέτης + 1) αθροιστής
6) Αποθήκευση — R4	
7) Γράψιμο στη μνήμη	

$T_1 - T_{10}$
$S_1 - S_7$
$10 + 7 - 1 = 16$
$\text{Speedup} = \frac{16}{70}$



Αριθμητικό Pipeline
(Pipeline Arithmetic)
Instruction Pipeline

Ανάκληση

T_0: MAR ← PC, Z ← PC+1 | S1: προσκομιδή εντολής

T_1: MDR ← M[MAR], PC ← Z

T_2: IR ← MDR[OPCODE] S2: Αποκωδικοποίηση

• STA

T_3: MDR ← ACC | S1: εκτέλεση εντος CPU

T_4: M[MAR] ← MDR | S4: Εγγραφή αποτελέσματος στη μνήμη

• ADD (Προσθέτω το ACC στα περιεχόμενα της θέσης μνήμης A)

T_3: MAR ← MDR[Adress] | Προσκομιδή απο την μνήμη

T_4: MDR ← M[MAR]

T_5: Z ← ACC + MDR | πράξεις εντος CPU

T_6: MDR ← Z

T_7: M[MAR] ← MDR Αποθήκευσή στη μνήμη

1) Ανομοιογένεια εντολών

2) Εντολές εκτελούνται σειριακά

Z ← PC+1 Αν συμβεί μετάβαση σε ρουτίνα τοτε το τμήμα προσκόμισης εντολής πρέπει να σταματήσει! Δεν ξέρει που θα πάει το πρόγραμμα

3) Αν τα αποτελέσματα μιας εντολής χρησιμοποιούνται στην επόμενη το Pipeline πρέπει να καθυστερήσει μέχρι να παραχθουν τα αποτελέσματα