Αρχιτεκτονική Υπολογιστών

Ενότητα 7: Πρόγνωση διακλαδώσεων.
Εξαρτήσεις και εκτέλεση εκτός σειράς.

Δρ. Μηνάς Δασυγένης
mdasyg@ieee.org
Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών
http://arch.ece.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα στο Πανεπιστήμιο Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σκοπός της Ενότητας

- Η κατανόηση των προβλημάτων των διακλαδώσεων και οι εναλλακτικές τεχνικές πρόγνωσης αυτών.

- Η κατανόηση των τεχνικών αύξησης της απόδοσης, που χρησιμοποιούνται στους σημερινούς επεξεργαστές.

Πρόγνωση Διακλαδώσεων

Η απόδοση της διοχέτευσης καταστρέφεται από 2 προβλήματα

- Η διοχέτευση αυξάνει την απόδοση ενός επεξεργαστή.
- Η διοχέτευση λειτουργεί άψογα σε γραμμικό/ακολουθιακό κώδικα.
- Όμως, υπάρχουν 2 καταστάσεις προβληματικές:
  - Διακλαδώσεις.
  - Εξαρτήσεις.

Το παρακάτω τμήμα κώδικα προκαλεί πρόβλημα στη διοχέτευση

- Οι 2 από τις πέντε εντολές είναι διακλαδώσεις.
- Μια διακλάδωση υπό συνθήκη (**BNE**: branch if not equal ή **JNE** στη x86).
- Μια διακλάδωση χωρίς συνθήκη (**BR**: branch ή **JMP** στη x86).

```
if (i == 0)
    k = 1;
else
    k = 2;
```

(a)

```
        CMP i,0; compare i to 0
        BNE Else; branch to Else if not equal
Then:   MOV k,1; move 1 to k
        BR Next; unconditional branch to Next
Else: MOV k,2; move 2 to k
Next:
```

(b)

Γιατί οι διακλαδώσεις χωρίς συνθήκη προκαλούν πρόβλημα στη διοχέτευση;

Το πρόβλημα βρίσκεται στη φύση της διοχέτευσης. Η αποκωδικοποίηση της εντολής γίνεται σε προχωρημένο στάδιο. Έτσι **η μονάδα προσκόμισης εντολών δε γνωρίζει για τη διακλάδωση**, παρά μόνο όταν αποκωδικοποιηθεί η εντολή. Τότε θα είναι αργά όμως, αφού θα έχει ήδη γεμίσει το υπόλοιπο pipeline με εντολές που ίσως δε θα εκτελεστούν.

Τι γίνεται στις διακλαδώσεις με συνθήκη;

- Οι διακλαδώσεις με συνθήκη είναι πολύ χειρότερες από τις διακλαδώσεις χωρίς συνθήκη.
- Δε γνωρίζει από που να διαβάσει την επόμενη εντολή, παρά μόνο πολύ αργότερα στη γραμμή διοχέτευσης.
- Οι πρώτες ΚΜΕ έκαναν στάση (**stall**) μέχρι να εκτελεστεί η εντολή διακλάδωσης.
- Είναι απαιτητικό να υπάρχει πρόβλεψη διακλαδώσεων, γιατί η στάση της διοχέτευσης καταστρέφει την απόδοση.

Υπάρχουν πολλαπλές προσεγγίσεις για το χειρισμό διακλαδώσεων

- Μην κάνεις τίποτα.
- Πολλές συνεχείς ροές.
- Προ-προσαγωγή στόχου διακλάδωσης.
- Προσωρινή μνήμη βρόχου.
- Πρόβλεψη διακλάδωσης.
- Καθυστερημένη διακλάδωση.

Περιγράψτε την προσέγγιση πολλές συνεχείς ροές

- Όταν υπάρχει διακλάδωση αντιγράφονται τα αρχικά τμήματα της σωλήνωσης (κλωνοποίηση σωλήνωσης) και προσάγονται και οι 2 εντολές (και αυτή που θα εκτελεστεί και αυτή που δε θα εκτελεστεί).
- Απαιτείται κλωνοποίηση σε hardware του πρώτου σταδίου και τμήματος του 2ου.
- Υπάρχουν 2 προβλήματα:
  - Καθυστέρηση λόγω διεκδίκησης προσπέλασης καταχωρητών.
  - Αν υπάρξει και επιπρόσθετη διακλάδωση τότε δημιουργείται καθυστέρηση.

Περιγράψτε την προσέγγιση προ-προσαγωγή στόχου διακλάδωσης

- Προ-προσάγεται ο στόχος της διακλάδωσης μαζί με την εντολή που ακολουθεί τη διακλάδωση.
- Αν ακολουθηθεί η διακλάδωση τότε ο στόχος έχει ήδη έρθει και εκτελείται.
- Διαφορετικά δε λαμβάνεται υπόψιν.

Περιγράψτε την προσέγγιση προσωρινή μνήμη βρόχου

- Υπάρχει μια μικρή μνήμη πολύ μεγάλης ταχύτητας μέσα στον επεξεργαστή.
- Διατηρούνται οι **n** εντολές διακλάδωσης που έχουν προσαχθεί.
- Αν πρόκειται να ακολουθηθεί μια διακλάδωση, ελέγχεται αυτή η μνήμη. Αν υπάρχει ο στόχος της διακλάδωσης μέσα στη μνήμη, τότε χρησιμοποιείται αυτός.
- Η τεχνική αυτή ευνοεί της επαναλήψεις ή τους βρόχους (από όπου παίρνει και το όνομα).
- Διαφέρει από την κρυφή μνήμη εντολών, γιατί έχει μόνο τις **n** τελευταίες εντολές διακλάδωσης.

Η προσέγγιση της προσωρινής μνήμης βρόχου

Παράδειγμα:

(Εντολή)	Διεύθυνση Εντολής	Στόχος
ja stoxos	0040h	0060h
jb again	0030h	

Υπάρχουν πολλές τεχνικές πρόβλεψης διακλαδώσεων

- Η διακλάδωση προς τα πίσω γίνεται πάντοτε.
- Η διακλάδωση δε γίνεται ποτέ.
- Η διακλάδωση γίνεται πάντοτε.
- Πρόβλεψη με τον opcode.
- Πραγματοποίηση/μη πραγματοποίηση μεταγωγής.
- Πίνακας προϊστορίας της διακλάδωσης.

Τι ισχύει για τη διακλάδωση προς τα πίσω;

- Η διακλάδωση προς τα πίσω γίνεται συνήθως στο τέλος του βρόχου επανάληψης.
- Οι βρόχοι εκτελούνται πολλές φορές. Μας συμφέρει να ακολουθούμε αυτήν την τεχνική.
- Οι περισσότερες διακλαδώσεις προς τα εμπρός εμφανίζονται όταν υπάρχουν σφάλματα *(π.χ. Ένα αρχείο δεν είναι διαθέσιμο)*.
  Τα σφάλματα είναι σπάνια για αυτό και αυτές δεν πραγματοποιούνται.

```
add ax,bx
jc telos ;πηγαίνε
στο τέλος αν
έχουμε overflow
```

```
mov cx,10
again: add si,cx
loop again
```

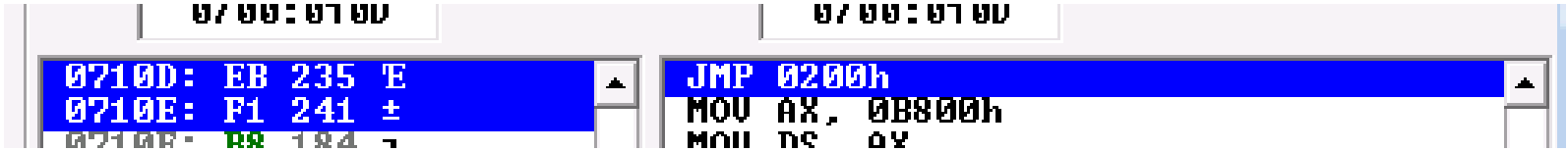
Τι συμβαίνει αν γίνει σωστή πρόγνωση και τι αν δε γίνει;

- Αν μια διακλάδωση προβλεφθεί σωστά δε χρειάζεται να γίνει τίποτα ιδιαίτερο.
- Αν μια διακλάδωση προβλεφθεί λάθος, τότε πρέπει να αναιρεθούν οι εντολές που εκτελέστηκαν ήδη, ενώ δεν έπρεπε. Αντιμετώπιση:
  - **(α)** Επιτρέπονται οι εντολές που προσκομίζονται στην πρόβλεψη, έως να επιχειρήσουν να μεταβάλουν την κατάσταση της μηχανής (π.χ. εγγραφή σε καταχωρητή). Τότε η νέα τιμή τοποθετείται σε ένα κρυφό καταχωρητή. Αν είναι σωστή η πρόγνωση τότε μεταφέρεται πίσω.
  - **(β)** Εκτελούνται οι εντολές, άλλα όταν υπάρχει μεταβολή σε καταχωρητή, φυλάσσεται σε μια περιοχή η παλαιά τιμή, ώστε αν χρειαστεί να γίνει αναίρεση.

Τι γνωρίζετε για την τεχνική πρόβλεψης με τον opcode

- Είναι μια στατική προσέγγιση.
- Με μετρήσεις έχουν βρεθεί κάποιοι opcodes διακλάδωσης που έχουν μεγαλύτερη πιθανότητα να γίνουν και μερικοί όχι.
- Έχει αναφερθεί επιτυχία 75%.

Για παράδειγμα, opcode **EB**: πρέπει να γίνεται πάντα η διακλάδωση.

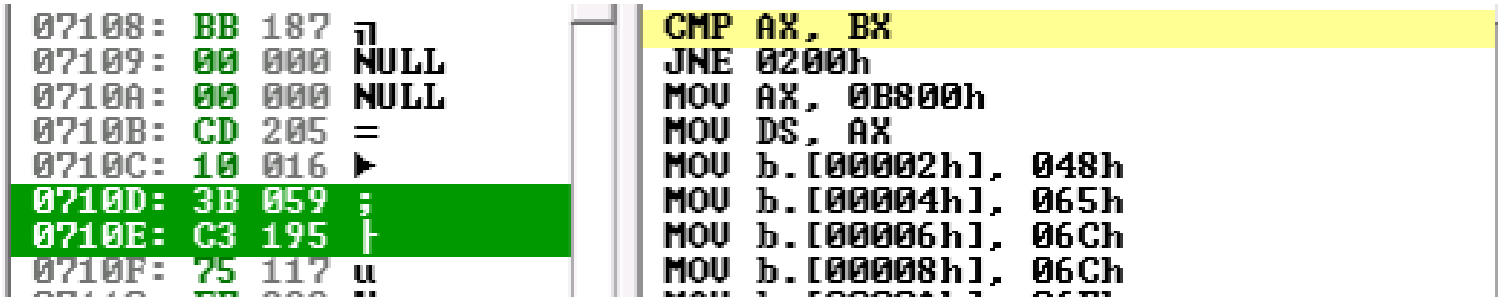


Τι γνωρίζετε για τη πραγματοποίηση/ μη πραγματοποίηση μεταγωγής;

- Δυναμική πρόγνωση διακλαδώσεων.
- Μέσα στον επεξεργαστή υπάρχει ειδική μνήμη.
- Καταγράφονται οι διακλαδώσεις υπό συνθήκη *(διεύθυνση),* ένα bit που δείχνει αν πραγματοποιήθηκε η διακλάδωση ή όχι. Η παραδοχή είναι ότι θα γίνει ότι έγινε την προηγούμενη φορά στη διακλάδωση αυτή.
- Η οργάνωση γίνεται **όπως στις κρυφές μνήμες**. Δηλαδή η διεύθυνση που βρίσκεται η εντολή διακλάδωσης χωρίζεται σε 2 μέρη n+k. Τα n bits διεθυνσιοδοτούν 2^n γραμμές της ειδικής μνήμης. Σε κάθε γραμμή υπάρχουν k bits.
- Δημιουργείται διένεξη στις διευθύνσεις όπως στην κρυφή μνήμη. Μπορεί να αυξηθεί η συσχετικότητα.

Ειδική Μνήμη για πρόγνωση διακλαδώσεων (1/2)

- Έστω ότι έχουμε το παρακάτω απόσπασμα κώδικα.
- Στη διεύθυνση 0710Dh ή **011 1000100001101** υπάρχει διακλάδωση.



Ειδική Μνήμη για πρόγνωση διακλαδώσεων (2/2)

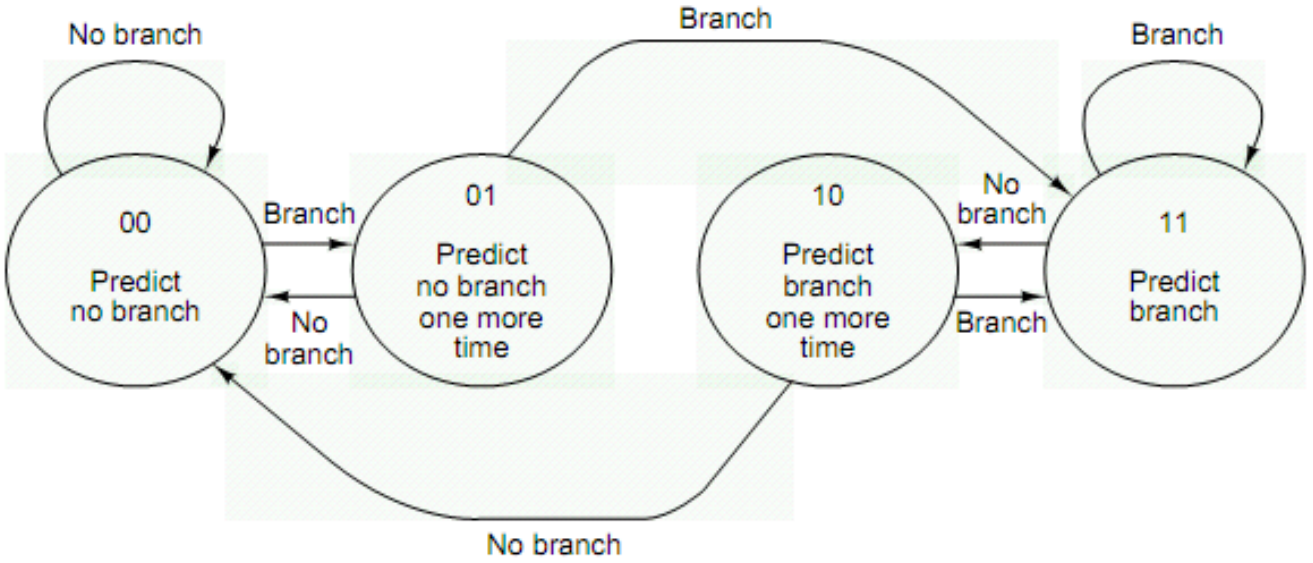
- Χωρίζουμε τα bit σε **n, κ**, οπότε **n+k=σύνολο bit**.
- Πίνακας n γραμμών, έστω n=3 bit=> 8 γραμ.
- Τα 3 MSB είναι ο αρ.γραμμής, υπόλοιπα tag.

tag	**Πραγματοποίηση;**
1000100001101	1

Τι γνωρίζετε για την τεχνική πρόγνωσης με πίνακα προϊστορίας της διακλάδωσης;

- Ονομάζεται και πρόγνωση δεύτερης ευκαιρίας.
- Με την προηγούμενη απλή καταγραφή, πάντα στο τέλος του βρόχου θα έχουμε λάθος πρόγνωση, οπότε σε εμφωλιασμένο βρόχο έχουμε σημαντικό πρόβλημα.
- Με την τεχνική αυτή δίνεται μια δεύτερη ευκαιρία.
- Η πρόγνωση αλλάζει μόνο σε δυο διαδοχικές λάθος προγνώσεις.
- Στην πιο απλή περίπτωση υπάρχουν 2 bit:
  - Είτε για τις 2 προηγούμενες εκτελέσεις.
  - Είτε για μηχανή πεπερασμένων καταστάσεων.

Υλοποίηση δυναμικής πρόγνωσης με μηχανή πεπερασμένων καταστάσεων



Το αριστερό bit είναι η πρόγνωση και το δεξιό τι έγινε τελικά.

tag	BR	PR
1000100001101	1	0

Γιατί απαιτείται να αποθηκεύεται η πραγματική διεύθυνση στις διακλαδώσεις;

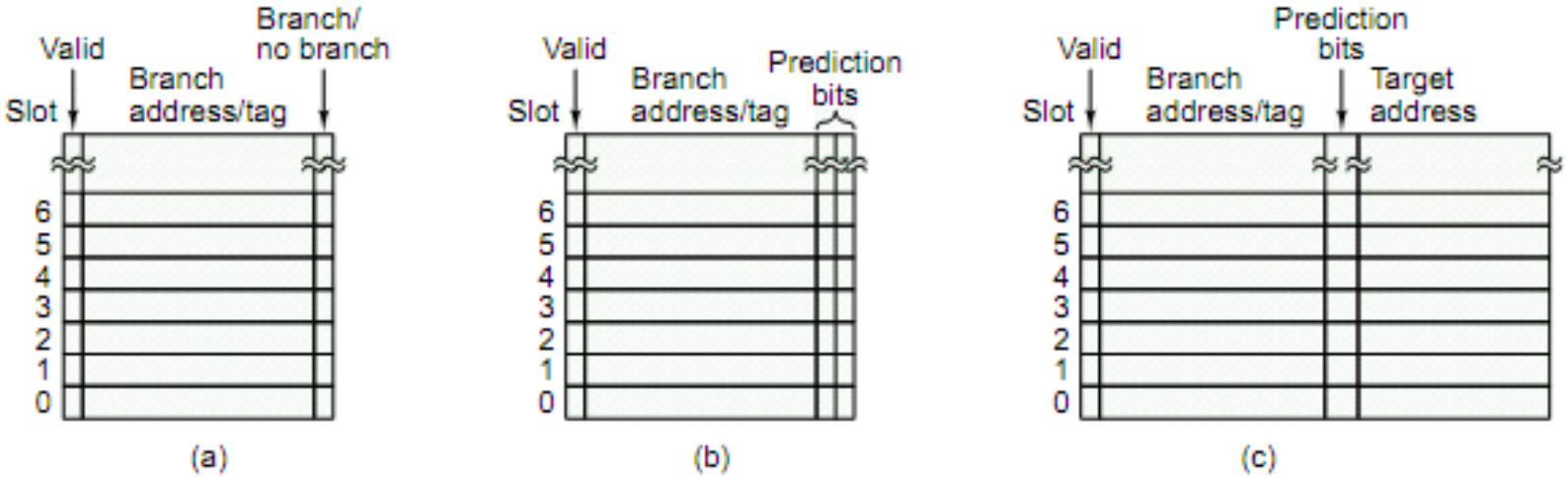
- Μερικές φορές δεν αρκεί να γνωρίζουμε αν μια διακλάδωση θα γίνει ή όχι.
- Είναι σημαντικό να γνωρίζουμε που οδηγεί η διακλάδωση *(ποια εντολή θα εκτελεστεί).*
- Ο στόχος της διακλάδωσης υπολογίζεται μετά από κάποιες εντολές *(δεν είναι γνωστός εκ των προτέρων)*. (Π.χ. **jne [di+bp]**).
- Απαιτείται λοιπόν στον πίνακα ιστορικού να υπάρχει μια στήλη που να δείχνει τη διεύθυνση που ακολουθήθηκε την προηγούμενη φορά.

Η απλή τεχνική των k-διακλαδώσεων

- Κρατάμε σε έναν πίνακα τις **k** προηγούμενες διακλαδώσεις. (**k bit**).
- Ο αριθμός των **k bit** συγκρίνεται μετά παράλληλα με όλες τις καταχωρήσεις ενός πίνακα ιστορικού με κλειδί των **k bit**.
- Αν εμφανιστεί μια επιτυχία χρησιμοποιείται η πρόγνωση που βρίσκεται εκεί.

Ακολουθεί συνοπτικό σχήμα.

Ειδικές μνήμες για δυναμική πρόγνωση διακλαδώσεων



Τι ισχύει για την διακλάδωση βοηθούμενη από το μεταφραστή;

- Κάποιοι επεξεργαστές δέχονται οδηγίες από το μεταφραστή σχετικά με το αν μια διακλάδωση θα γίνει ή όχι.
- Παράδειγμα στην εντολή **for (i=0;i<10000;i++) {..}** η διακλάδωση στο τέλος του βρόχου θα πραγματοποιείται σχεδόν πάντα.
- Ο μεταφραστής μπορεί να υποδείξει στο υλικό ότι η συγκεκριμένη διακλάδωση θα γίνει σχεδόν πάντα με το να θέσει κάποια bit στην εντολή (machine code).
- Απαιτείται λιγότερο υλικό, πιο εξελιγμένο μεταφραστή.
- Μια παραλλαγή είναι να χρησιμοποιηθεί profiling για να βρεθούν ποιες διακλαδώσεις ακολουθούνται σχεδόν πάντα *(πάνω από 90%).*

Τι είναι η τεχνική της καθυστερημένης διακλάδωσης;

- Είναι μια τεχνική που αυξάνει την απόδοση σε μια διασωλήνωση.
- Χρησιμοποιεί μια διακλάδωση που δεν ενεργοποιείται παρά μετά την εκτέλεση της επόμενης εντολής *(“καθυστερημένη”).*
- Η θέση εντολής αμέσως μετά τη διακλάδωση ονομάζεται **θυρίδα καθυστέρησης** *(delay slot).*
- Τοποθετείται μια εντολή μετά τη διακλάδωση στο delay slot και έτσι έχουμε αύξηση της απόδοσης.
- Λειτουργεί με επιτυχία στις διακλαδώσεις χωρίς συνθήκη, στις κλήσεις και στις επιστροφές από διαδικασίες.

Τι ονομάζεται θυρίδα καθυστέρησης; (1/2)

Τι ονομάζεται θυρίδα καθυστέρησης;

- Μερικοί επεξεργαστές για να αντιμετωπίσουν το πρόβλημα της διακλάδωσης χωρίς συνθήκη, τοποθετούν μια θυρίδα καθυστέρησης *(delay slot),* για να προλάβει το pipeline να αποκωδικοποιήσει την εντολή.
- Οι μεταγλωττιστές προσπαθούν να τοποθετήσουν κάποια εντολή εκεί, όμως συχνά αυτό είναι αδύνατο και έτσι τοποθετείται η NOP.

Τι ονομάζεται θυρίδα καθυστέρησης; (2/2)

- Για παράδειγμα, έστω ο κώδικας:

```
mov ax,bx
cmp dx,cx
jne label:
```

- Ένας compiler **με βελτιστοποίηση** θα χρησιμοποιήσει τη θυρίδα καθυστέρησης και θα δημιουργήσει τον κώδικα:

```
cmp dx,cx
mov ax,bx
jne label
```

- Με αυτόν τον τρόπο θα γίνει η σύγκριση των dx,cx. θα μπει στο pipeline η επόμενη εντολή και όταν μπει η jne label, τότε θα γνωρίζουμε ποια θα είναι η επόμενη εντολή *(αν θα γίνει αλλαγή ροής εκτέλεσης ή όχι)* αφού η εντολή cmp θα έχει ολοκληρώσει τη φάση εκτέλεσης.
- *(Η θυρίδα καθυστέρησης συνήθως τοποθετείται μετά την εντολή διακλάδωσης, σε αντίθεση με το παράδειγμα).*

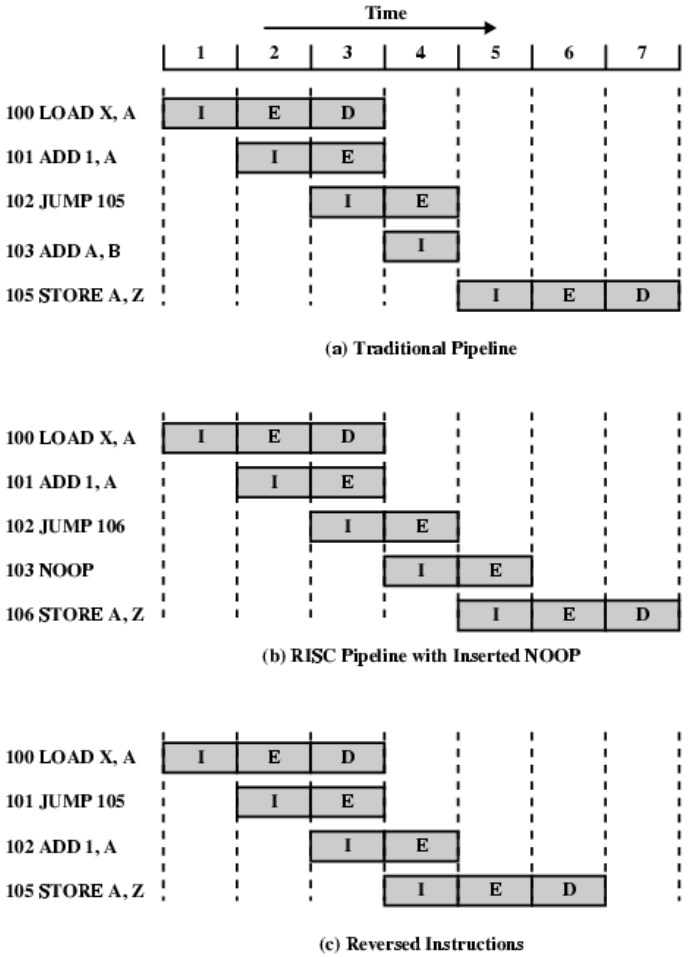
Παράδειγμα κανονικής και καθυστερημένης διακλάδωσης

Address	Normal	Delayed	Optimized
100	LOAD X,A	LOAD X,A	LOAD X,A
101	ADD 1,A	ADD 1,A	JUMP 105
102	JUMP 105	JUMP 105	ADD 1,A
103	ADD A,B	NOOP	ADD A,B
104	SUB C,B	ADD A,B	SUB C,B
105	STORE A,Z	SUB C,B	STORE A,Z
106		STORE A,Z	

Καθυστερημένη διακλάδωση στους σύγχρονους επεξεργαστές

- Δε χρησιμοποιείται γιατί είναι περιττή λόγω:
  - Διοχετεύσεων μεγαλύτερου μήκους.
  - Εκτέλεσης με υπερβάθμωση.
  - Δυναμικής πρόβλεψης διακλαδώσεων.

Η καθυστερημένη διακλάδωση αυξάνει την απόδοση



Τι είναι η τεχνική της καθυστερημένης φόρτωσης;

- Στις εντολές **LOAD** *(μεταφορά από μνήμη σε καταχωρητή),* ο καταχωρητής που πρόκειται να αποτελέσει στόχο της φόρτωσης κλειδώνεται από τον επεξεργαστή.
- Ύστερα ο επεξεργαστής συνεχίζει την εκτέλεση της ροής εντολών.
- Αν δει μια εντολή που χρειάζεται τον καταχωρητή ακινητοποιείται ***(stall)*** μέχρι να ολοκληρωθεί η φόρτωση.

Πότε υπάρχει παραλληλισμός στάθμης εντολής;

- Ο παραλληλισμός στάθμης εντολής *(instruction level parallelism)* υπάρχει όταν οι εντολές σε σειρά είναι ανεξάρτητες και έτσι μπορούν να εκτελεστούν παράλληλα.

Παράδειγμα:

LOAD R1<=R2

ADD R3<=R3+1

ADD R4<=R4+R2

Όλες οι εντολές είναι ανεξάρτητες, μπορούν να εκτελεστούν παράλληλα.

- Ενώ, δεν ισχύει το ίδιο για:

ADD R3<=R3+1

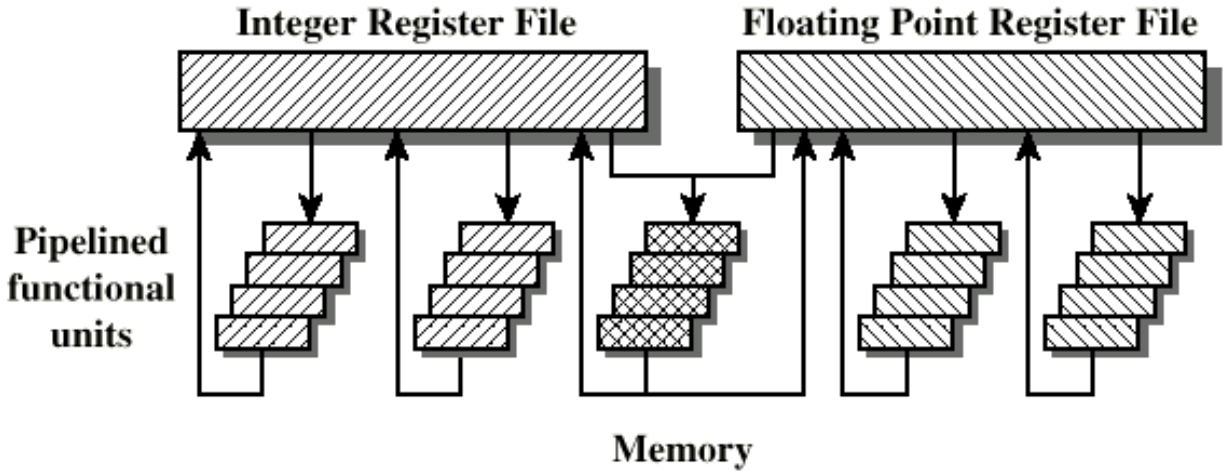
ADD R4<=R3+R2

Υπάρχουν εξαρτήσεις δεδομένων που απαιτούν οι εντολές να εκτελεστούν σειριακά.

STORE [R4]<=R0

Υπερβαθμωτοί επεξεργαστές

- Γενική οργάνωση Superscalar:
  - Πολλαπλές λειτουργικές μονάδες *(ALU, FPU, …).*



Ποια είναι τα βασικά στοιχεία που πρέπει να έχει ένας υπερβαθμωτός επεξεργαστής;

Ένας επεξεργαστής superscalar πρέπει να έχει:

- Στρατηγικές προσαγωγής εντολών.
- Πρόβλεψη διακλαδώσεων.
- Πολλαπλά στάδια διασωλήνωσης.
- Λογική καθορισμού εξαρτήσεων.
- Παράλληλη εκτέλεση εντολών.
- Πολλαπλούς πόρους για παράλληλη εκτέλεση.
- Μηχανισμό απόσυρσης κατάστασης επεξεργασίας.

Εξαιρέσεις και Διακοπές (1/2)

- Ένα από τα δυσκολότερα μέρη της μονάδας ελέγχου.
- **Εξαίρεση** *(exception)*: ένα μη προγραμματισμένο συμβάν που διακόπτει την κανονική εκτέλεση του προγράμματος. Προέρχεται από το εσωτερικό του υπολογιστή. Π.χ. Διαίρεση με το 0.
- **Διακοπή** *(interrupt)*: ένα μη προγραμματισμένο συμβάν που διακόπτει την κανονική εκτέλεση του προγράμματος. Προέρχεται από το εξωτερικό του υπολογιστή. Π.χ. Ολοκλήρωση εισόδου/εξόδου.

Εξαιρέσεις και Διακοπές (2/2)

- Η ανίχνευση συνθηκών εξαίρεσης και η λήψη κατάλληλων μέτρων βρίσκονται στην κρίσιμη διαδρομή του χρονισμού.
- Απαιτείται λοιπόν ένας πολύ γρήγορος χειρισμός αυτών, γιατί επηρεάζει την απόδοση.
- Μπορεί να υπάρχει διαβάθμιση εξαιρέσεων *(προτεραιότητες).*

Ενέργειες υλικού κατά την εξαίρεση/διακοπή

- Αποθήκευση της διεύθυνσης εντολής του προγράμματος που εκτελείται (PC ή CS:IP).
- Μεταφορά ελέγχου στο Λειτουργικό Σύστημα.
- Γνωστοποίηση του λόγου που δημιούργησε την διακοπή:
  - Μέσω καταχωρητή αιτίας διακοπής και μεταφορά ελέγχου σε **μια** συγκεκριμένη διεύθυνση εκτέλεσης για όλες τις διακοπές.
  - Με διανυσματική μεταφορά, αφού υπάρχει για κάθε διακοπή **ειδική** διεύθυνση χειρισμού.

Ενέργειες υλικού κατά την εξαίρεση

- Η CPU σταματάει προβληματική εντολή.
- Επιτρέπει τις προηγούμενες εντολές να ολοκληρωθούν.
- Εκκενώνει τις εντολές που ακολουθούν.
- Ενημερώνει κατάλληλα καταχωρητή αιτίας διακοπής.
- Αποθήκευση δ/νσης της προβληματικής εντολής.
- Πραγματοποίηση άλματος σε προκαθορισμένη διεύθυνση.

Ενέργειες λογισμικού κατά την εξαίρεση

- Το ΛΣ βρίσκει την αιτία της διακοπής.
- Κατάλληλη αντίδραση:
  - π.χ. για αστοχία υλικού, μη ορισμένη εντολή, ➔ τερματισμός της διαδικασίας.
  - π.χ. για Ε/Ε ➔ αποθήκευση κατάστασης προγράμματος και πραγματοποίηση Ε/Ε, και μετά στο μέλλον συνέχεια της εκτέλεσης.
- Η πιο συχνή εξαίρεση είναι το σφάλμα σελίδας.
- Η πιο συχνή διακοπή είναι timer χρονοδρομολόγησης.

Εξαιρέσεις και διασωλήνωση

- Οι εξαιρέσεις είναι κίνδυνοι διασωλήνωσης.
- Απαιτείται:
  - Εκκένωση διασωλήνωσης.
  - Προσκόμιση εντολών από νέα διεύθυνση.
- Όταν συμβεί μια εξαίρεση πρέπει να σταματήσει οποιαδήποτε εκτέλεση άλλης εντολής.
- Πρέπει να μπορεί να προσδιοριστεί ακριβώς η εντολή που προκάλεσε την εξαίρεση.

Διακοπές

- Ανακριβής διακοπή *(imprecise interrupt):*

  Δε σχετίζεται η διακοπή με την εντολή που την προκάλεσε. Πιο φθηνή υλοποίηση, κυρίως χαμηλής πολυπλοκότητας CPU.

- Ακριβής διακοπή *(precise interrupts/exceptions):*

  Σχετίζεται πάντοτε με τη σωστή εντολή στους υπολογιστές με διοχέτευση.

  Οι σύγχρονοι επεξεργαστές κυρίως υποστηρίζουν τις ακριβές διακοπές.

Επεξεργαστές πολλαπλής εκκίνησης

- Εκκίνηση πολλών εντολών ανά στάδιο προκειμένου να αυξηθεί το IPC *(superscalars).*
- Υπάρχουν πολλοί περιορισμοί στους τύπους των εντολών που μπορούν να εκτελεστούν ταυτόχρονα.
- Συνήθως 3 έως 6 εντολές εκκινούν οι σύγχρονοι επεξεργαστές σε κάθε κύκλο ρολογιού.
- Υλοποίηση CPU πολλαπλής εκκίνησης:
  - **Στατική** *(οι αποφάσεις λαμβάνονται από το μεταφραστή).*
  - **Δυναμική** *(οι αποφάσεις λαμβάνονται κατά τη διάρκεια της εκτέλεσης από τον επεξεργαστή).*

Στατική Πολλαπλή Εκκίνηση

- Χρησιμοποιείται ο μεταγλωττιστής για να βοηθήσει στη συσκευασία των εντολών και στο χειρισμό των κινδύνων.
- Το σύνολο των εντολών που ξεκινούν σε ένα δεδομένο κύκλο ρολογιού, ονομάζεται πακέτο εκκίνησης *(issue packet).*
- Μοιάζει σαν μια μεγάλη εντολή με πολλές λειτουργίες.
- Ονομάζεται και τεχνική VLIW *(very long instruction word):*

"**Ένα στυλ αρχιτεκτονικής συνόλου εντολών που ξεκινάει πολλές λειτουργίες, οι οποίες έχουν οριστεί ως ανεξάρτητες, σε μια μοναδική ευρεία εντολή, με πολλά opcodes**".

Στατική Πολλαπλή Εκκίνηση (VLIW) (1/2)

- Παράδειγμα:
  2 issue MIPS Processor VLIW.
  - Πρώτη εντολή: ALU ή διακλάδωση.
  - Δεύτερη εντολή: Φόρτωση ή αποθήκευση.
  - Απαιτούνται περισσότερες θύρες στο αρχείο των καταχωρητών *(για να μην υπάρχει δομικός κίνδυνος).*
  - Ο compiler τοποθετεί τις εντολές σε ζεύγη, αν δε μπορεί να τοποθετήσει εντολή σε ζεύγος τότε την αφήνει ως nop.
- Ο έλεγχος των εξαρτήσεων γίνεται **(α)** αποκλειστικά από το compiler ή **(b)** από το υλικό, κάνοντας stall όταν υπάρχει λόγος.

Στατική Πολλαπλή Εκκίνηση (VLIW) (2/2)

- Η χρονοδρομολόγηση των εντολών γίνεται αποκλειστικά από το compiler.
- Ο compiler παράγει κώδικα αποκλειστικά για μια συγκεκριμένη αρχιτεκτονική ενός επεξεργαστή, με συγκεκριμένους περιορισμούς.
- Δεν είναι εύκολο να μεταφερθεί ο μεταγλωττισμένος κώδικας από μια VLIW αρχιτεκτονική σε μια άλλη.

Δυναμική Πολλαπλή Εκκίνηση

- Είναι οι επεξεργαστές Superscalar.
- Έχουν hardware που ελέγχει τις εξαρτήσεις και αποφασίζει ποιες εντολές εκκινούν σε κάθε γύρο, γιατί είναι ανεξάρτητες.
- Μοιάζει με τους VLIW, επειδή ομοίως έχει πολλαπλούς πόρους *(θύρες στο αρχείο καταχωρητών, ALU μονάδες κ.α.)* για την παράλληλη εκτέλεση εντολών.
- Διαφέρει από τους VLIW, επειδή ο έλεγχος των εξαρτήσεων και η χρονοδρομολόγηση γίνεται από το υλικό.

Οι 3 κύριες μονάδες μιας δυναμικά χρονοπρογραμματιζόμενης διοχέτευσης (1/2)

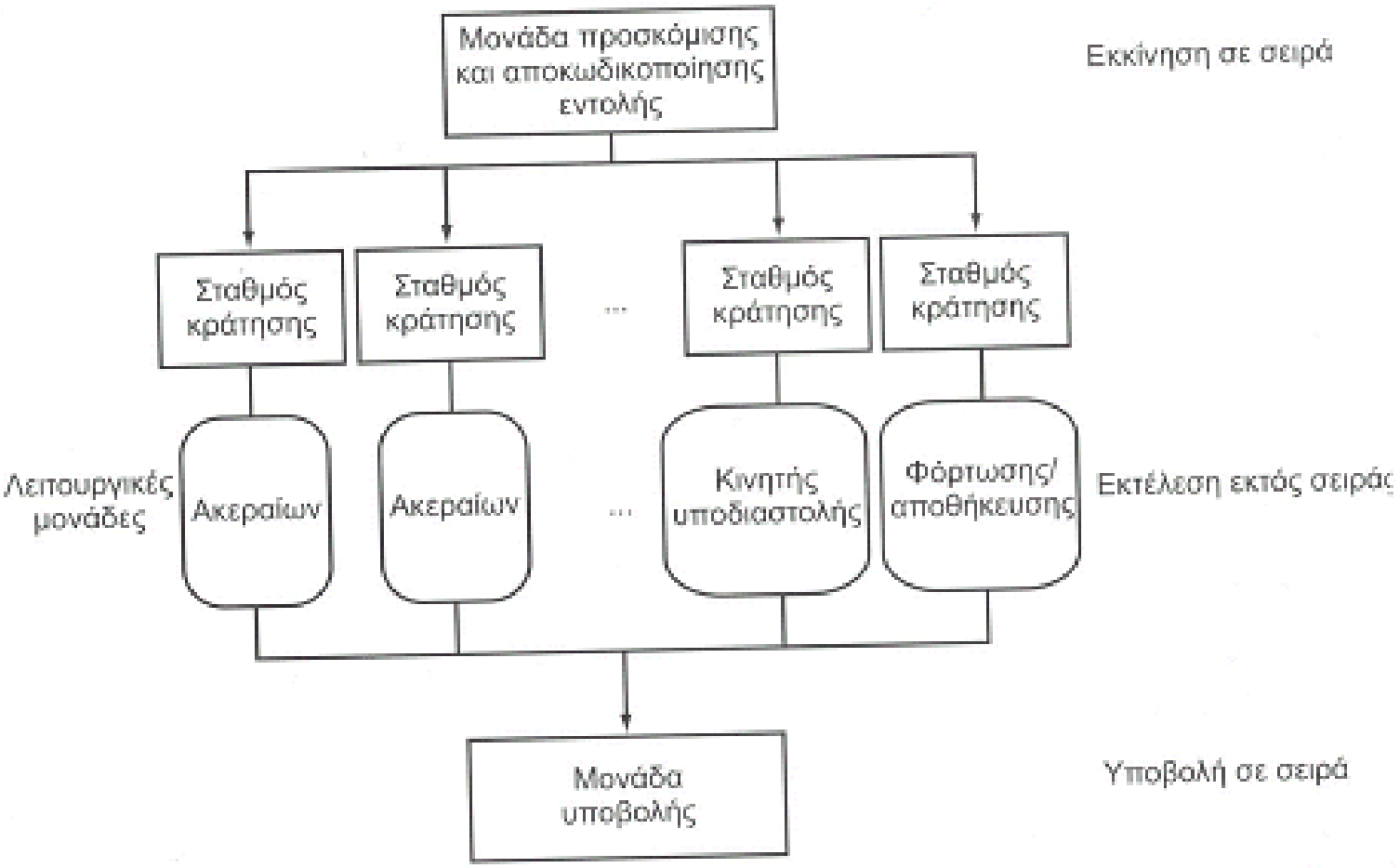
- Η διοχέτευση διαιρείται σε τρεις κύριες μονάδες:
  - Μονάδα προσκόμισης και εκκίνησης εντολής:

    προσκομίζει, αποκωδικοποιεί και στέλνεται σε λειτουργική μονάδα.
  - Πολλαπλές λειτουργικές μονάδες:

    έχει προσωρινές μνήμες που ονομάζονται σταθμοί κράτησης.
  - Μονάδα υποβολής:

    αποθηκεύει προσωρινά το αποτέλεσμα μέχρι να είναι ασφαλής η τοποθέτηση του στο αρχείο καταχωρητών. Η προσωρινή μνήμη, ονομάζεται μνήμη αναδιάταξης (reoder buffer).

Οι 3 κύριες μονάδες μιας δυναμικά χρονοπρογραμματιζόμενης διοχέτευσης (2/2)



VLIW ή Superscalar

- Υπάρχουν προβλήματα από την στατική χρονοδρομολόγηση των εντολών γιατί:
  - Δεν είναι προβλέψιμες όλες οι καθυστερήσεις.
  - Δε μπορεί να γνωρίζει ο compiler την ακριβή σειρά των εντολών κατά το χρόνο μεταγλώττισης, ιδιαίτερα όταν υπάρχουν διακλαδώσεις.
  - Ο λανθάνων χρόνος της διοχέτευσης (ο χρόνος στον οποίο είναι ένα αποτέλεσμα διαθέσιμο για χρήση σε μια εντολή που ακολουθεί) αλλάζει από υλοποίηση σε υλοποίηση. Έτσι, ο καλύτερος τρόπος μεταγλώττισης μιας ακολουθίας κώδικα αλλάζει.
- **Και οι 2 τεχνικές όμως αυξάνουν την IPC, αξιοποιώντας το ILP.**

Διοχέτευση πολλαπλής εκκίνησης

- 2 αρμοδιότητες:
  - Συγκέντρωση εντολών σε υποδοχές εκκίνησης *(issue slots):* Προσδιορίζονται πόσες και ποιες εντολές μπορούν να ξεκινήσουν σε ένα δεδομένο κύκλο ρολογιού. Οι εντολές μπορούν να αναδιαταχθούν για να αποφευχθούν καθυστερήσεις.
  - Αντιμετώπιση κινδύνων δεδομένων και ελέγχου *(εξαρτήσεις).*

Σύγχρονοι επεξεργαστές και πολλαπλή εκκίνηση (1/2)

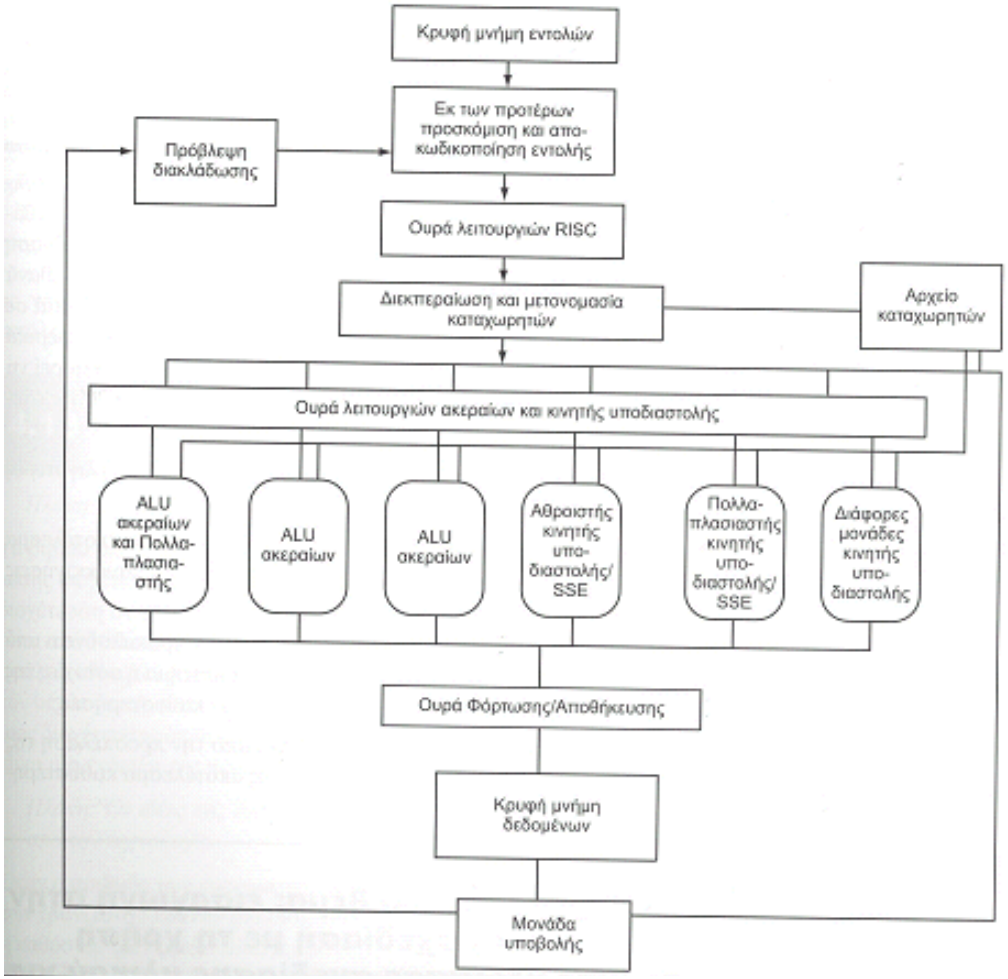
Οι σύγχρονοι επεξεργαστές μπορούν να εκκινούν έως 6 εντολές ανά πυρήνα. Τις περισσότερες όμως φορές χρησιμοποιούνται 1-2.

- Λόγοι που μειώνουν την πολλαπλή εκκίνηση:
  - Εξαρτήσεις που δε μπορούν να εξαλειφθούν.
  - Απώλειες (=καθυστέρηση) στο σύστημα μνήμης.

Σύγχρονοι επεξεργαστές και πολλαπλή εκκίνηση (2/2)

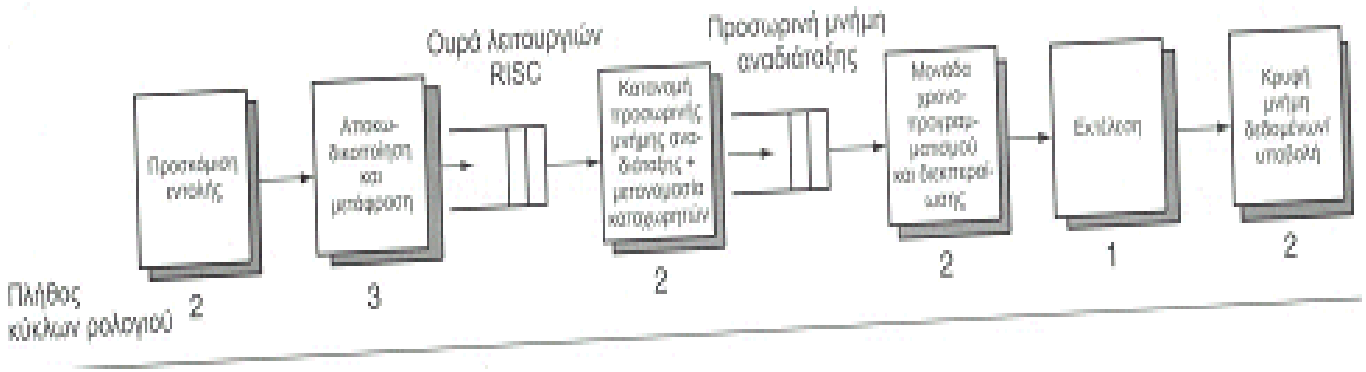
Μικροεπεξεργαστής	Έτος	Ρυθμός ρολογιού	Στάδια διοχέτευσης	Εύρος εκκίνησης	Εκτός σειράς /εικασία	Πυρήνες /τσιπ	Ισχύς
Intel 486	1989	25MHz	5	1	Όχι	1	5W
Intel Pentium	1993	66MHz	5	2	Όχι	1	10W
Intel Pentium Pro	1997	200MHz	10	3	Ναι	1	29W
Intel Pentium4 Willamette	2001	2000MHz	22	3	Ναι	1	75W
Intel Pentium 4 Prescott	2004	3600MHz	31	3	Ναι	1	103W
Intel Core	2006	2930MHz	14	4	Ναι	2	75W
Sun UltraSPARC III	2003	1950MHz	14	4	Όχι	1	90W
Sun UltraSPARC T1 (Niagara)	2005	1200MHz	6	1	Όχι	8	70W

AMD Opteron (1/2)



- Μετονομασία των 16 ορατών καταχωρητών σε 72.
- 12 στάδια pipeline.

AMD Opteron (2/2)



- **Εικόνα 4.75 Η διοχέτευση του Operton X4 με τη ροή της διοχέτευσης για μία τυπική εντολή και τον αριθμό των κύκλων ρολογιού για τα κυριότερα βήματα στη διοχέτευση 12 σταδίων για λειτουργίες RISC ακέραιων.**
- Η ουρά εκτέλεσης των πράξεων κινητής υποδιαστολής έχει μήκος 17 σταδίων. Παρουσιάζονται επίσης οι κυριότερες προσωρινές μνήμες όπου περιμένουν οι λειτουργίες RISC.

Εκτέλεση Εντολών εκτός σειράς

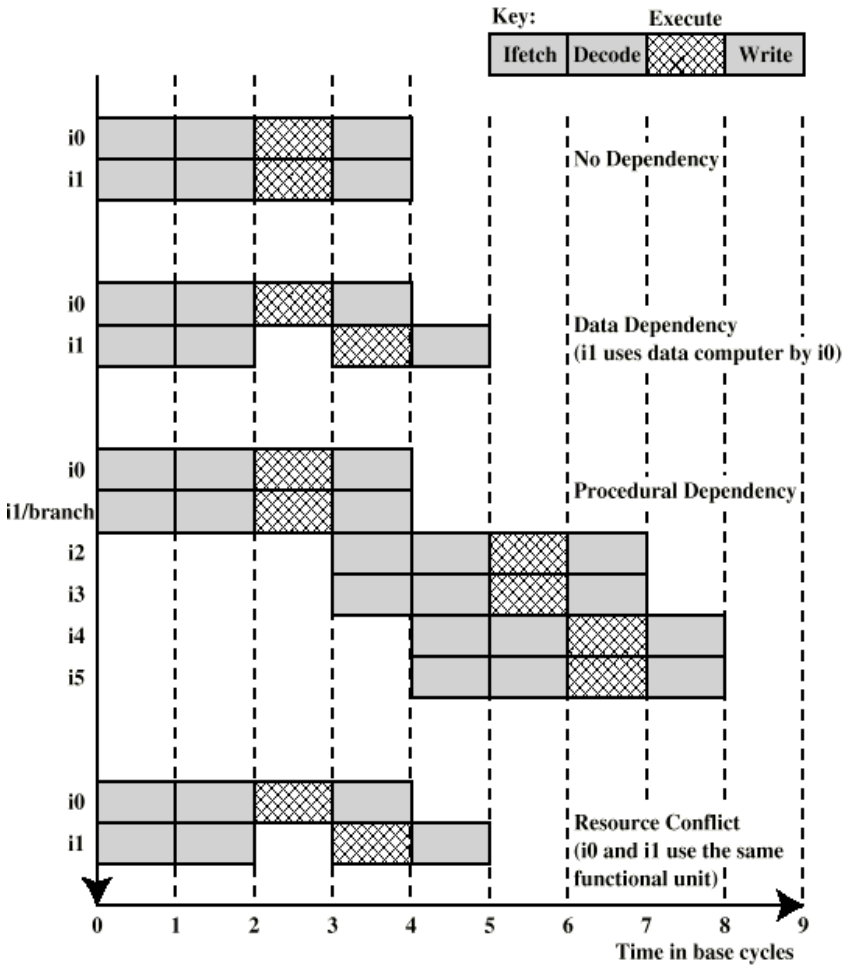
Τι ονομάζεται παραλληλισμός μηχανής;

- Ο παραλληλισμός μηχανής (**machine parallelism**) αποτελεί μέτρο της ικανότητας του επεξεργαστή να εκμεταλλευτεί τον παραλληλισμό στάθμης εντολής.
- Καθορίζεται από:
  - Το πλήθος των εντολών που μπορούν να προσαχθούν.
  - Το πλήθος των εντολών που μπορούν να εκτελεστούν παράλληλα.
  - Την ταχύτητα και την ευφυΐα των μηχανισμών που χρησιμοποιεί ο επεξεργαστής για να βρει ανεξάρτητες εντολές.

Ποιοι περιορισμοί υπάρχουν που μειώνουν τον παραλληλισμό;

- Εξάρτηση από δεδομένα ή ροής (π.χ. **add r1,r2 ; mov r3,r1**).
- Εξαρτήσεις από διακλαδώσεις.
- Συγκρούσεις πόρων *(καταχωρητές, λειτουργικές μονάδες,...).*
- Εξάρτηση εξόδου *(εγγραφής αποτελέσματος,* ***WAW****).*
- Αντιεξάρτηση *(εγγραφής αποτελέσματος* ***RAW****).*

Η επίδραση των εξαρτήσεων στις επιδόσεις



Γιατί η εκτέλεση με σειρά δε δίνει πάντα βέλτιστα αποτελέσματα;

- Υπάρχουν εξαρτήσεις ανάμεσα στις εντολές.
- Αν μια εντολή για παράδειγμα απαιτεί μια τιμή που υπολογίζεται από την προηγούμενη εντολή, τότε αυτή η εντολή δε μπορεί να εκτελεστεί αν δεν υπολογιστεί η προηγούμενη εντολή. Η εξάρτηση αυτή είναι RAW (**read-after-write**).
  - Παράδειγμα:

    add ax,bx

    sub dx,ax

    Απαιτεί να υπολογιστεί πρώτα η **add (AX)** και μετά η **sub (DX).**

Τι είδους εξαρτήσεις πρέπει να ελέγχονται; (1/2)

Για την παράλληλη εκτέλεση σε μια υπερβαθμωτή μηχανή υπάρχουν οι εξής έλεγχοι:

- Αν οποιοσδήποτε τελεστέος γράφεται, η εντολή δεν υποβάλλεται *(εξάρτηση RAW, read-after-write).*
- Αν ο καταχωρητής του αποτελέσματος διαβάζεται, η εντολή δεν υποβάλλεται *(εξάρτηση WAR, write-after-read).*
- Αν ο καταχωρητής του αποτελέσματος γράφεται, η εντολή δεν υποβάλλεται *(εξάρτηση WAW, write-after-write).*

Τι είδους εξαρτήσεις πρέπει να ελέγχονται; (2/2)

Παραδείγματα εξαρτήσεων:

- RAW

  MOV AX,40

  ADD AX,10

- WAR

  ADD AX,10

  MOV AX,40

- WAW

  MOV DI,40

  MOV DI,[DI]

Ποιες εξαρτήσεις μπορούμε να απομακρύνουμε;

- Οι εξαρτήσεις WAR και WAW μπορούν να αποφευχθούν με το να τοποθετηθούν τα αποτελέσματα κάπου προσωρινά και όταν ολοκληρωθεί η προηγούμενη εντολή να μεταφερθούν τα αποτελέσματα στην πραγματική τους θέση.

- Αυτή η τεχνική ονομάζεται "**μετονομασία καταχωρητών**" (register renaming).

- Οι σημερινές CPU έχουν δεκάδες μυστικούς καταχωρητές που χρησιμοποιούνται στη μετονομασία καταχωρητών.

Δώστε ένα παράδειγμα της χρήσης της τεχνικής μετονομασίας καταχωρητών

- Έστω το τμήμα κώδικα:

R3<= R3 +R5

R4<=R3+1

R3<=R5+1 (εδώ υπάρχει εξάρτηση WAR)

R7<=R3+R4

- Η τεχνική της μετονομασίας καταχωρητών χρησιμοποιώντας τους κρυφούς S καταχωρητές:

R3<= R3 +R5

R4<=R3+1

S3<=R5+1

R7<=S3+R4

***** Η παραλληλία του κώδικα αυξήθηκε *****

Ποιες είναι οι κατηγορίες επεξεργαστών ως προς τη σειρά προσαγωγής/εξαγωγής εντολών;

- Προσαγωγή εντολής σε σειρά με ολοκλήρωση σε σειρά.
- Προσαγωγή εντολής σε σειρά με ολοκλήρωση εκτός σειράς.
- Προσαγωγή εντολής εκτός σειράς με ολοκλήρωση εκτός σειράς.

Αυτονόητο είναι ότι πάντα πρέπει να **δίνεται το ίδιο αποτέλεσμα** σαν να εκτελείται το πρόγραμμα με τη σειρά που είναι γραμμένο.

Προσαγωγή εντολής σε σειρά με ολοκλήρωση σε σειρά

- Απλούστερη τεχνική.
- Καθόλου αποδοτική.
- Στις εξαρτήσεις οι εντολές καθυστερούν.
- Ακολουθείται τόσο η σειρά προσαγωγής εντολών όσο και η σειρά εξαγωγής των αποτελεσμάτων.

Παραδείγματα περιπτώσεων

Στις επόμενες διαφάνειες θεωρούμε ότι:

- Η I1 χρειάζεται δύο κύκλους για να εκτελεστεί.
- Οι I3 και I4 συγκρούονται για την ίδια λειτουργική μονάδα.
- Η I5 εξαρτάται από την τιμή που παράγεται από την I4.
- Οι I5 και I6 συγκρούονται για μια λειτουργική μονάδα.

Οι εντολές προσάγονται ανά δύο.

Προσαγωγή εντολής σε σειρά με ολοκλήρωση σε σειρά

Decode		Execute			Write		Cycle
I1	I2						1
I3	I4	I1	I2				2
I3	I4	I1					3
	I4			I3	I1	I2	4
I5	I6			I4			5
	I6		I5		I3	I4	6
			I6				7
					I5	I6	8

Προσαγωγή εντολής σε σειρά με ολοκλήρωση εκτός σειράς (1/2)

- Οι εντολές προσάγονται με τη σειρά.
- Αν υπάρχουν εξαρτήσεις κάποιες εντολές μπορούν να ολοκληρωθούν πιο πριν αν και έπονται.

Παράδειγμα:

R3:= R3 + R5; (I1)

R4:= R3 + 1; (I2)

R3:= R5 + 1; (I3)

I2 depends on result of I1 - data dependency.

If I3 completes before I1, the result from I1 will be wrong - output (read-write) dependency.

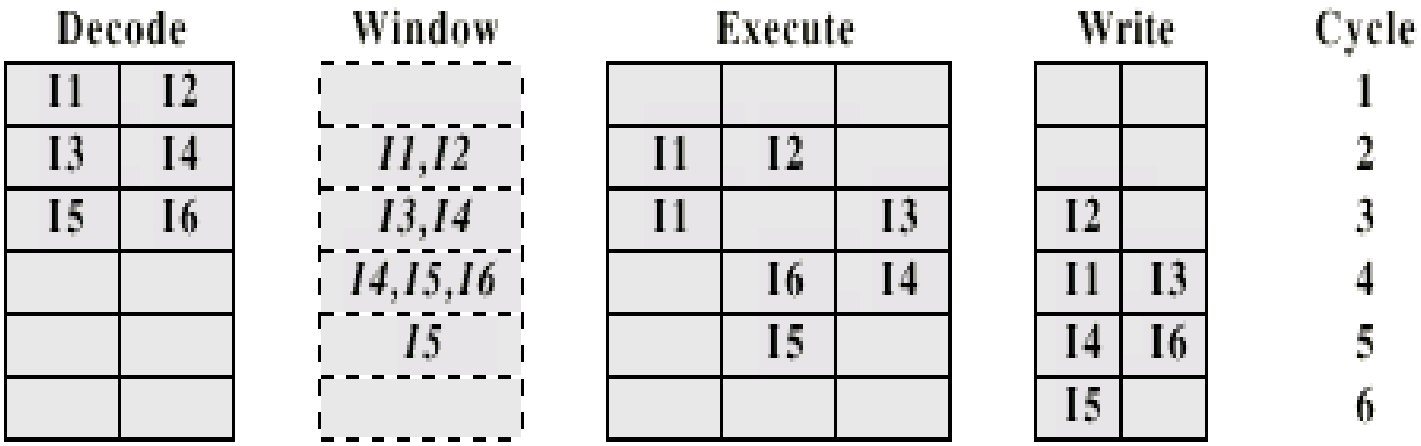
Προσαγωγή εντολής σε σειρά με ολοκλήρωση εκτός σειράς (2/2)

Decode		Execute			Write		Cycle
I1	I2						1
I3	I4	I1	I2				2
	I4	I1		I3	I2		3
I5	I6			I4	I1	I3	4
	I6		I5		I4		5
			I6		I5		6
					I6		7

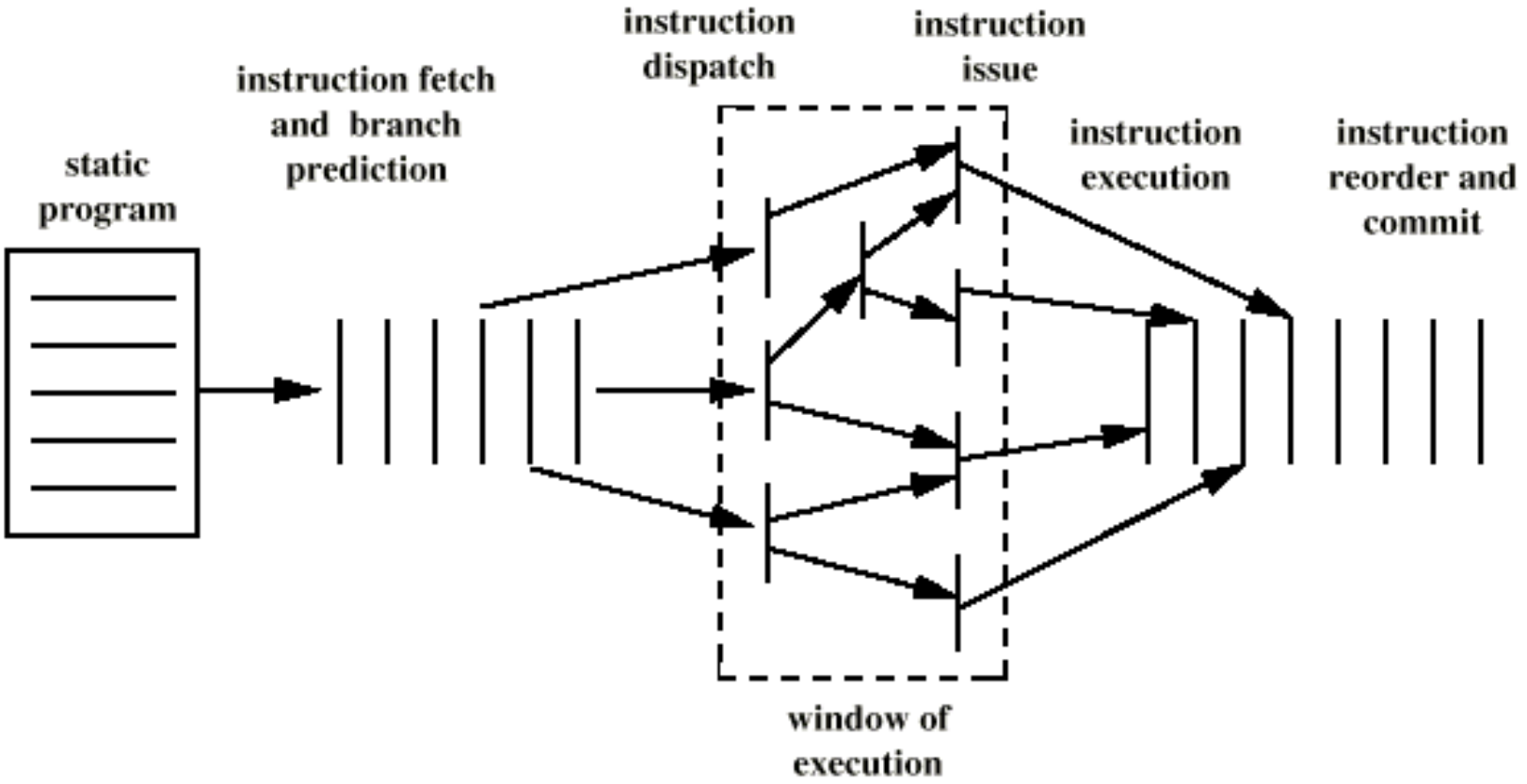
Προσαγωγή εντολής εκτός σειράς με ολοκλήρωση εκτός σειράς (1/2)

- Στην έκδοση με σειρά ο επεξεργαστής **μόλις συναντήσει εξαρτήσεις θα σταματήσει** μέχρι να διευθετηθούν.
- Αν γίνει αποσύνδεση των σταδίων αποκωδικοποίησης και εκτέλεσης τότε μπορούν να εκτελούνται εντολές εκτός σειράς.
- Οι εντολές αποκωδικοποιούνται και τοποθετούνται σε ένα παράθυρο εντολών.
- Όταν υπάρχει διαθέσιμη μια λειτουργική μονάδα τότε ο επεξεργαστής επιλέγει μια αποκωδικοποιημένη εντολή και την εκτελεί.
- **Μπορεί να διατεθεί οποιαδήποτε εντολή με την προϋπόθεση ότι (1) χρειάζεται τη συγκεκριμένη μονάδα και (2) η εντολή δεν εμποδίζεται από συγκρούσεις ή εξαρτήσεις.**
- Ο επεξεργαστής βλέπει “εμπρός” και ψάχνει για ανεξάρτητες εντολές.

Προσαγωγή εντολής εκτός σειράς με ολοκλήρωση εκτός σειράς (2/2)



Η εκτέλεση εντολών στους επεξεργαστές



Τι πρόβλημα εισάγει η εκτέλεση εντολών εκτός σειράς

- Ο λόγος είναι λεπτός, αλλά σημαντικός:
  - Αν οι εντολές εκτελούνται εκτός σειράς και **εμφανίζεται μια διακοπή** *(interrupt)* είναι πολύ δύσκολο *(αλλά όχι ακατόρθωτο)* να αποθηκευτεί η κατάσταση της μηχανής για να αποκατασταθεί αργότερα.
  - Μάλιστα δε θα μπορεί να προσδιοριστεί εύκολα αν οι εντολές μέχρι κάποια διεύθυνση έχουν εκτελεστεί και ότι οι εντολές μετά από αυτή τη διεύθυνση δεν έχουν εκτελεστεί.

Γιατί απαιτείται το βήμα της απόσυρσης (retiring) εντολής;

- Η **απόσυρση** *(retiring)* ή **διάπραξη** *(commiting)* είναι το τελευταίο βήμα μιας εντολής αφού εκτελεστεί.
- Είναι πολύ σημαντικό βήμα.
- Επειδή οι εντολές μπορεί να ολοκληρωθούν με διαφορετική σειρά από ότι ήταν αρχικά ή επειδή λόγω λάθος πρόγνωσης διακλάδωσης εκτελέστηκαν λάθος εντολές, τα αποτελέσματα αποθηκεύονται σε προσωρινούς καταχωρητές *(μη ορατούς από το χρήστη).*
- Όταν **καθοριστεί ότι το ακολουθιακό μοντέλο εκτέλεσε ορθά την εντολή**, τότε μεταφέρονται οι τιμές στους κανονικούς καταχωρητές.

Η τεχνική της εικασίας

- Είναι μια προσέγγιση όπου ο μεταγλωττιστής ή ο επεξεργαστής εικάζουν (μαντεύουν) το αποτέλεσμα μιας εντολής για να την αφαιρέσουν ως μια εξάρτηση κατά την εκτέλεση άλλων εντολών.
- Π.χ. Αποτέλεσμα διακλάδωσης.
- Η δυσκολία είναι ότι μπορεί να είναι λανθασμένη.
- Απαιτείται:
  - Μηχανισμός εικασίας.
  - Μηχανισμός ελέγχου εικασίας αν ήταν σωστή ή αν ήταν λάθος *(και επαναφορά των ορθών αποτελεσμάτων).*

Μπορεί η εικασία να προκαλέσει εξαιρέσεις που δε θα γίνονταν διαφορετικά.

Ξετύλιγμα βρόχου για αύξηση απόδοσης

- Μια τεχνική αύξηση της απόδοσης βρόχων που προσπελάζουν πίνακες, στην οποία δημιουργούνται πολλά αντίγραφα του σώματος του βρόχου, και εντολές από διαφορετικές επαναλήψεις χρονοπρογραμματίζονται μαζί.

παράδειγμα:

Αρχικός βρόχος

```
mov cx,5
again: add buffer[si],cx
loop again
```

Ξετιλυγμένος βρόχος

```
add buffer[1],1
add buffer[2],2
add buffer[3],3
add buffer[4],4
add buffer[5],5
```

Πλεονέκτημα: Αύξηση παραλληλοποίησης (επιδόσεων)

Μειονέκτημα:_____;

Τέλος Ενότητας