Προηγμένη Αρχιτεκτονική Υπολογιστών

Τεχνική διοχέτευσης (pipelining)

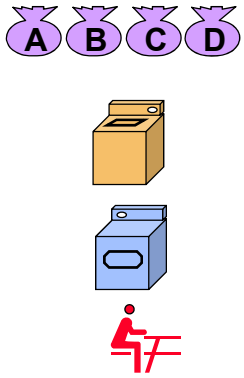
Μιχάλης Ψαράκης

Τεχνική διοχέτευσης (pipelining)

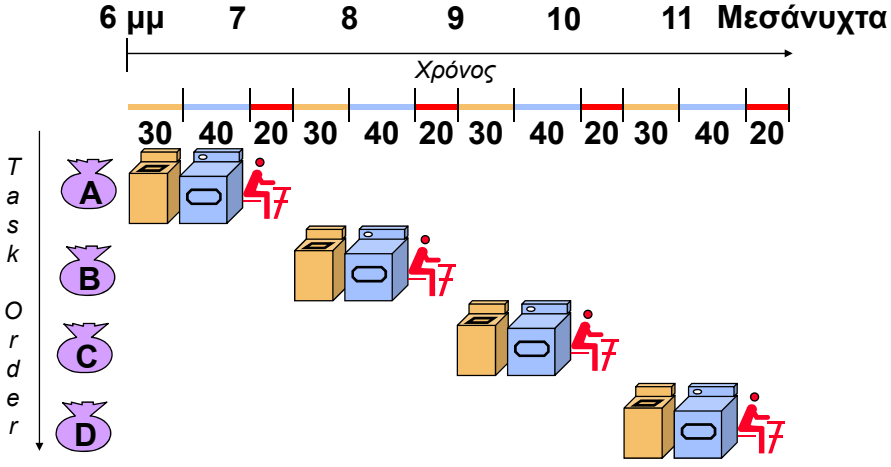
- Τεχνική για την αύξηση της απόδοσης των επεξεργαστών
- Ιδέα: η εκτέλεση των εντολών γίνεται με επικάλυψη

Παράδειγμα: το πλυντήριο ρούχων

- Για να κατανοήσουμε το κίνητρο της χρήσης της τεχνικής διοχέτευσης θα μελετήσουμε ένα παράδειγμα από τον πραγματικό κόσμο
- Τέσσερα άτομα (A, B, C, D) έχουν ένα φορτίο ρούχων για πλύσιμο
- Το πλύσιμο διαρκεί 30'
- Το στέγνωμα διαρκεί 40'
- Το δίπλωμα διαρκεί 20'

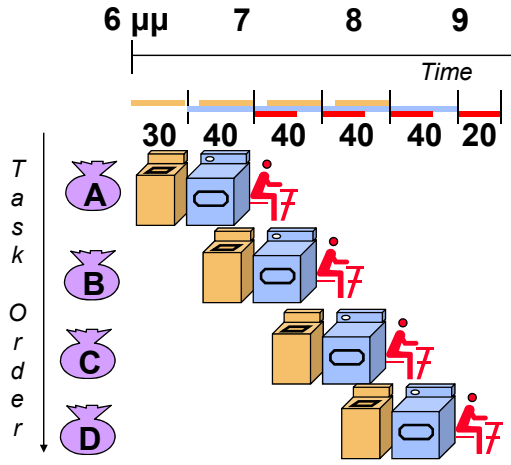


Το πλυντήριο χωρίς διοχέτευση



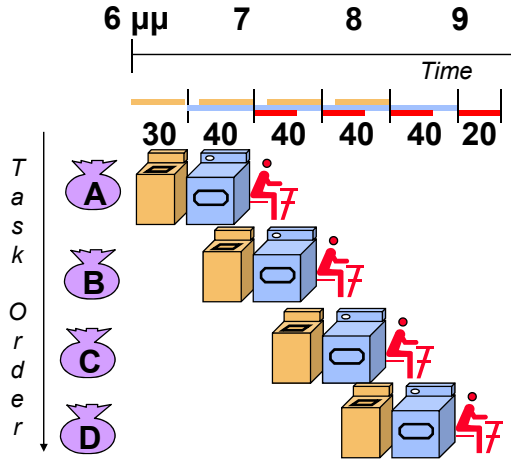
- Επειδή δεν γνωρίζουν την τεχνική διοχέτευσης πλένουν τα άπλυτά τους (!!!) για 6 ώρες

Το πλυντήριο με διοχέτευση



- Χρησιμοποιώντας την τεχνική διοχέτευσης το πλύσιμο των ρούχων διαρκεί 3,5 ώρες αντί για 6 ώρες

Το πλυντήριο με διοχέτευση



- Συνολικός χρόνος πλυσίματος μεγαλύτερος ή μικρότερος;
- Συνολικός χρόνος αναμονής μεγαλύτερος ή μικρότερος;
- Ποιο είναι το ποσοστό χρησιμοποίησης του στεγνωτηρίου;
- Ποιο είναι το ποσοστό χρησιμοποίησης του τραπεζιού διπλώματος;

Στάδια εκτέλεσης των εντολών ενός επεξεργαστή MIPS

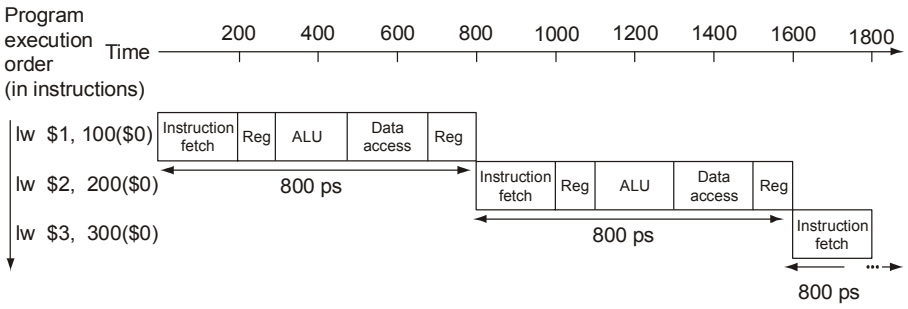
- Instruction Fetch (IF):
  - Προσκόμιση (fetch) της εντολής από τη μνήμη
- Reg ή Instruction Decode (ID):
  - Ανάγνωση των καταχωρητών κατά την αποκωδικοποίηση της εντολής
    Η μορφή των εντολών του MIPS επιτρέπει να γίνεται ταυτόχρονα ανάγνωση και αποκωδικοποίηση
- ALU ή Execute (EX):
  - Εκτέλεση της λειτουργίας (πράξης) ή υπολογισμός μιας διεύθυνσης
- Data access ή Memory (MEM):
  - Προσπέλαση ενός τελεστέου στη μνήμη δεδομένων
- Reg ή Write Back (WB):
  - Εγγραφή του αποτελέσματος σε έναν καταχωρητή

Επεξεργαστής MIPS χωρίς διοχέτευση

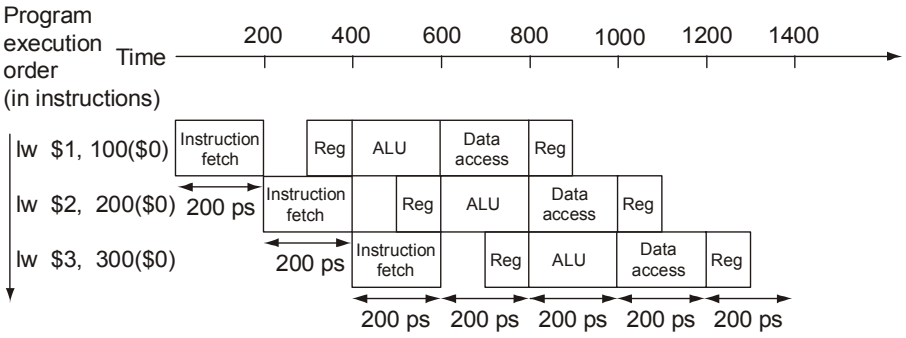
- Χρόνος εκτέλεσης (σε ps) των εντολών σε έναν επεξεργαστή MIPS χωρίς διοχέτευση (υλοποίηση ενός κύκλου)

Κατηγορία εντολής	IF	ID	EX	MEM	WB	Συνολικός χρόνος
Load word (lw)	200	100	200	200	100	800
Store word (sw)	200	100	200	200		700
R-format (add, sub, and, or, slt)	200	100	200		100	600
Branch (beq)	200	100	200			500

Εκτέλεση ενός κύκλου χωρίς διοχέτευση



Εκτέλεση με διοχέτευση



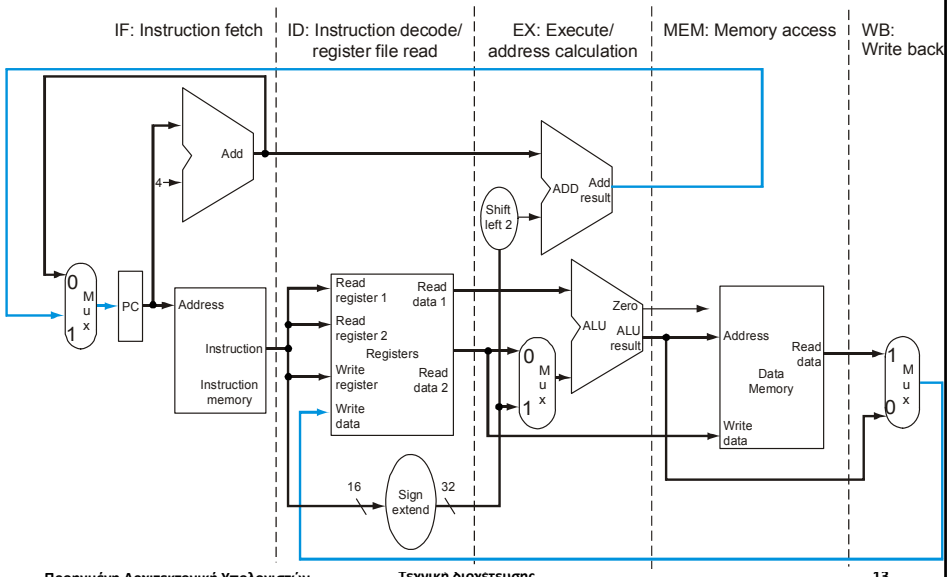
Απόδοση διοχέτευσης

- Βελτιώνει τη διεκπεραιωτική ικανότητα (throughput) σε εντολές και όχι το χρόνο εκτέλεσης (latency) μίας μεμονωμένης εντολής
- Οι χρόνοι των σταδίων περιορίζονται από τον πιο αργό πόρο
- Μέγιστη επιτάχυνση = αριθμός των σταδίων
- Μη ισορροπημένα στάδια μειώνουν την επιτάχυνση
- Ο απαιτούμενος χρόνος για να «γεμίσουν» και να «αδειάσουν» τα στάδια της διοχέτευσης μειώνει την επιτάχυνση

Απόδοση διοχέτευσης

- Απαιτεί ξεχωριστές εργασίες/στάδια
- Απαιτεί ξεχωριστούς πόρους
- Επιτυγχάνει παραλληλία στην εκτέλεση των εντολών χωρίς πολλαπλασιασμό των πόρων
- Η αποδοτικότητα της διοχέτευσης (δηλ. να είναι «γεμάτος» ο μηχανισμός της διοχέτευσης) είναι κρίσιμη για την απόδοση του συστήματος

Διαδρομή δεδομένων ενός κύκλου



Σχεδίαση συνόλου εντολών για διοχέτευση

- Όλες οι εντολές έχουν το ίδιο μήκος
  - Ευκολότερη προσκόμιση (1ο στάδιο) και αποκωδικοποίηση εντολών (2ο στάδιο)
- Τα πεδία των καταχωρητών βρίσκονται στην ίδια θέση σε κάθε εντολή
  - Ανάγνωση του αρχείου καταχωρητών ταυτόχρονα με τον προσδιορισμό του τύπου της εντολής
- Οι τελεστέοι μνήμης εμφανίζονται μόνο στις εντολές φόρτωσης (load) και αποθήκευσης (store)
  - Υπολογισμός της διεύθυνσης μνήμης στο στάδιο εκτέλεσης και προσπέλαση της μνήμης στο επόμενο στάδιο
- Οι τελεστέοι είναι ευθυγραμμισμένοι στη μνήμη
  - Τα δεδομένα της μνήμης προσπελαύνονται σε ένα στάδιο

MIPS: τυπικός RISC επεξεργαστής

Πεδίο	0	rs	rt	rd	shamt	funct
Θέσεις bit	31:26	25:21	20:16	15:11	10:6	5:0

α. Εντολή τύπου R

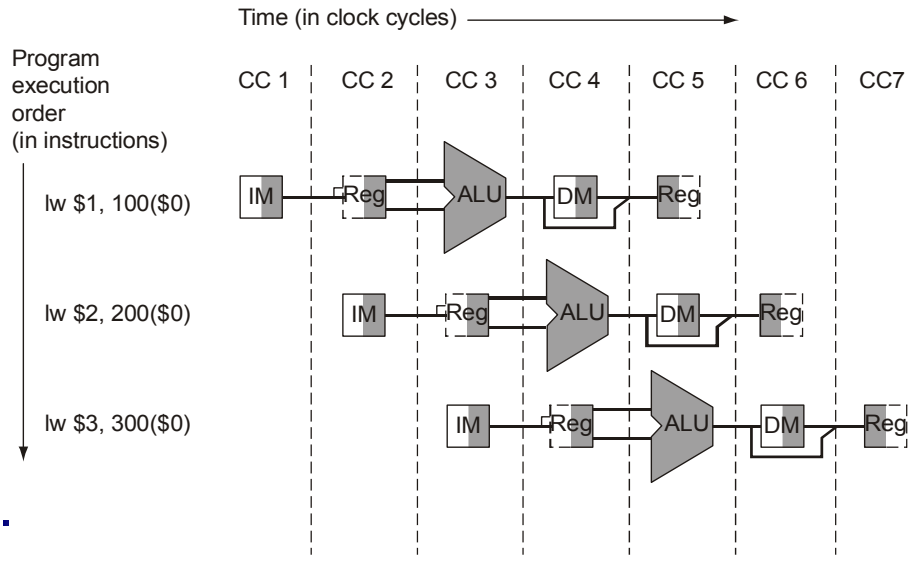
Πεδίο	35 or 43	rs	rt	Διεύθυνση
Θέσεις bit	31:26	25:21	20:16	15:0

β. Εντολή load ή store

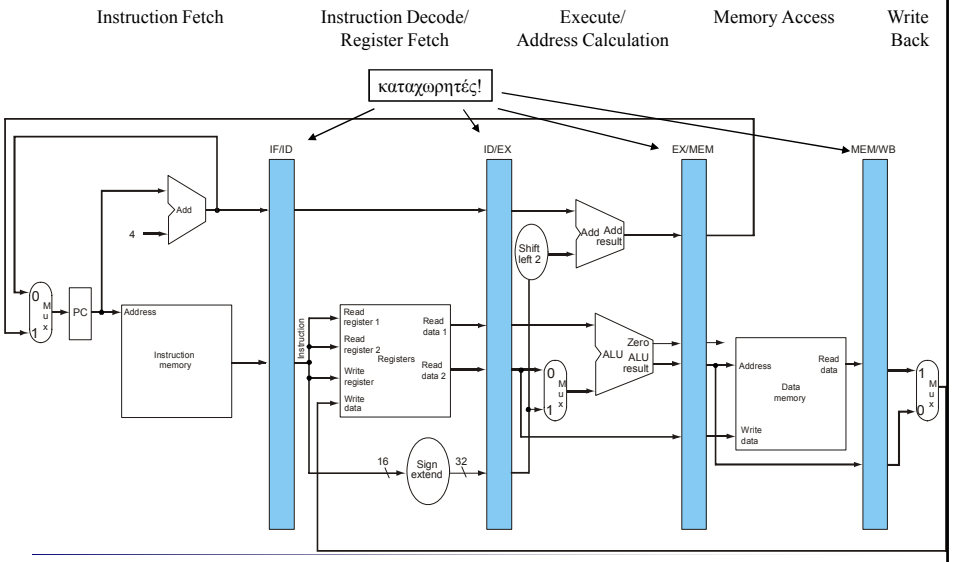
Πεδίο	4	rs	rt	Διεύθυνση
Θέσεις bit	31:26	25:21	20:16	15:0

γ. Εντολή διακλάδωσης

Εκτέλεση εντολών με χρήση διοχέτευσης



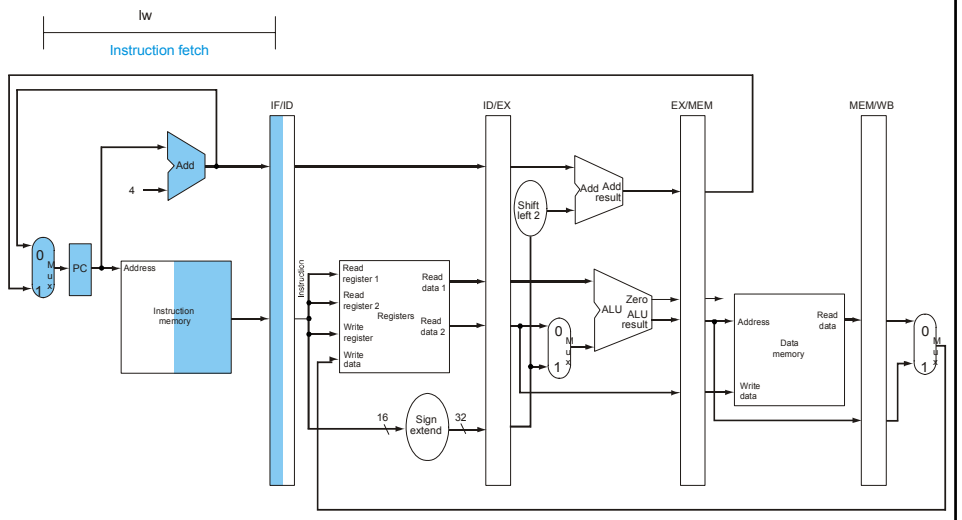
Διαδρομή δεδομένων με διοχέτευση



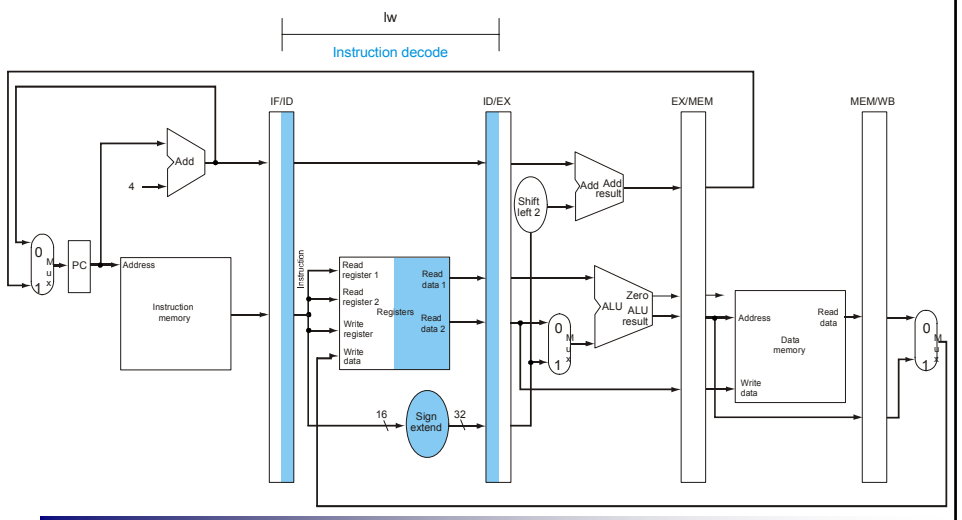
Παράδειγμα

- Ένας επεξεργαστής που χρειάζεται 5ns για να εκτελέσει μια εντολή υλοποιεί το μηχανισμό διοχέτευσης με 5 ίσα στάδια. Οι καταχωρητές μεταξύ των σταδίων έχουν μια καθυστέρηση των 0.25 ns.
  1. Ποιος είναι ο ελάχιστος χρόνος κύκλου ρολογιού του επεξεργαστή με διοχέτευση;
  2. Ποια είναι η μέγιστη επιτάχυνση που μπορεί να επιτευχθεί από τον επεξεργαστή με διοχέτευση (συγκρινόμενος με τον αρχικό επεξεργαστή χωρίς διοχέτευση);

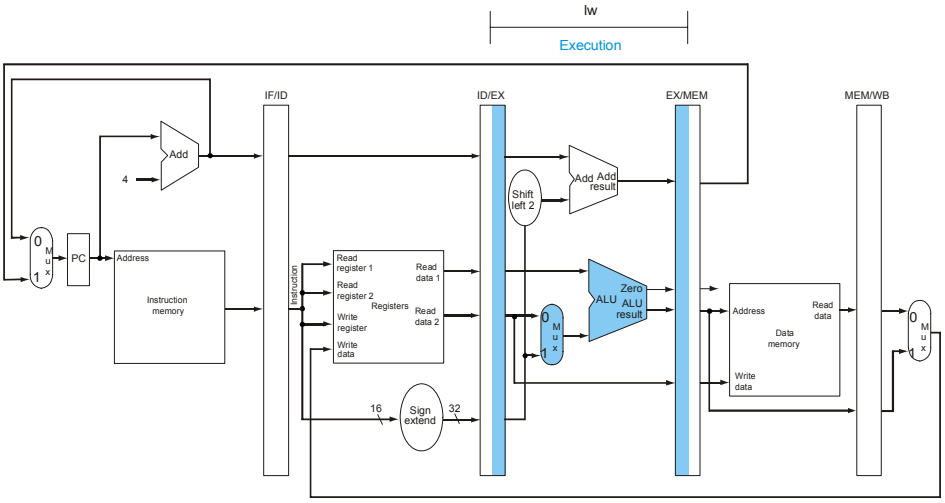
Στάδιο IF μιας εντολής load



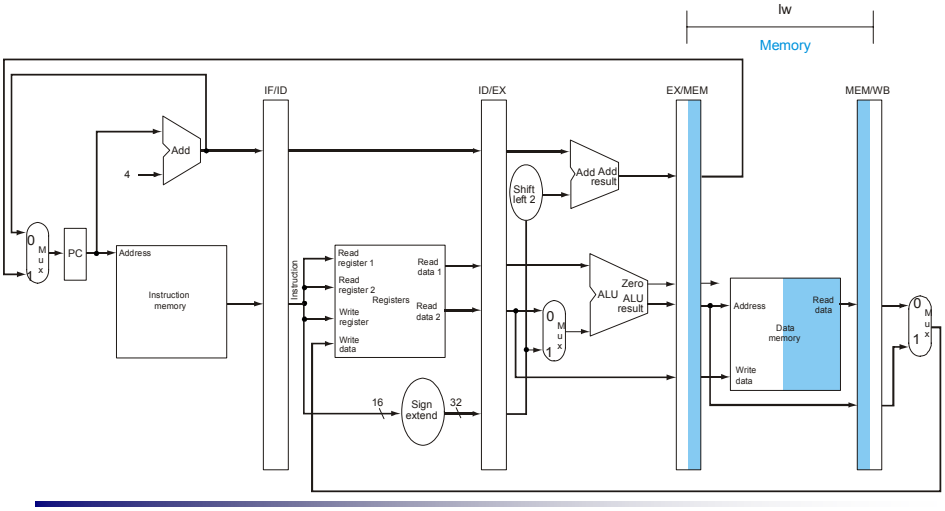
Στάδιο ID μιας εντολής load



Στάδιο EX μιας εντολής load

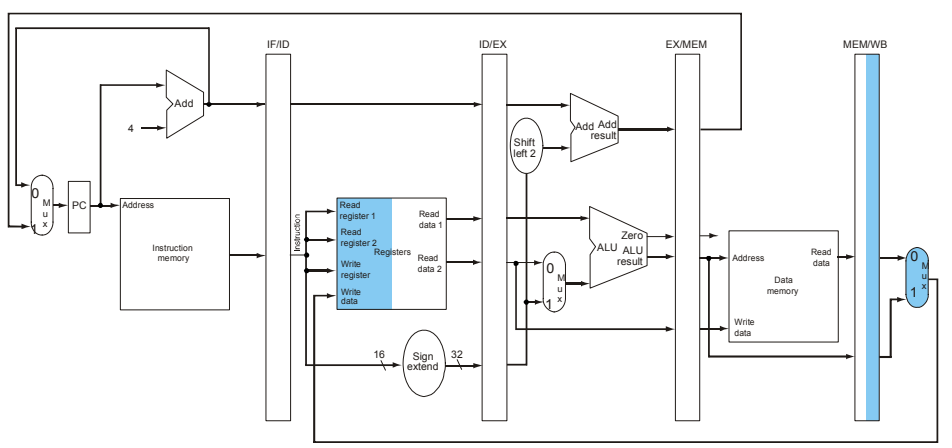


Στάδιο MEM μιας εντολής load



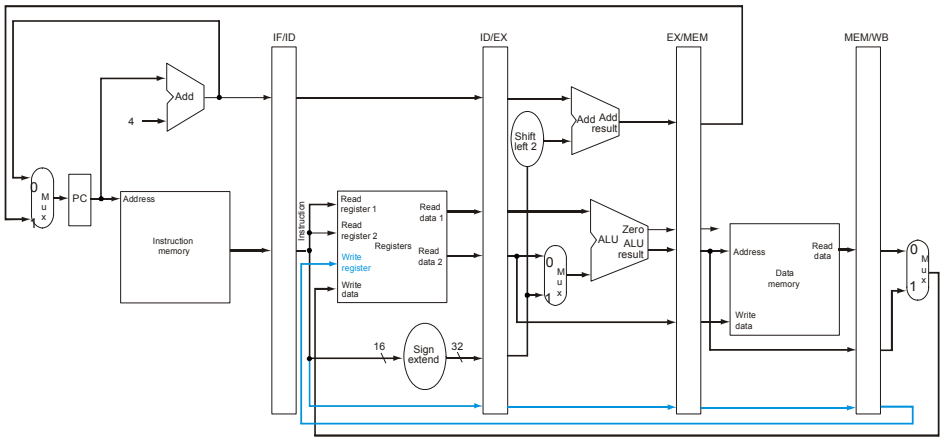
Στάδιο WB μιας εντολής load

Ποιος πρόσεξε το λάθος στο μηχανισμό διοχέτευσης;

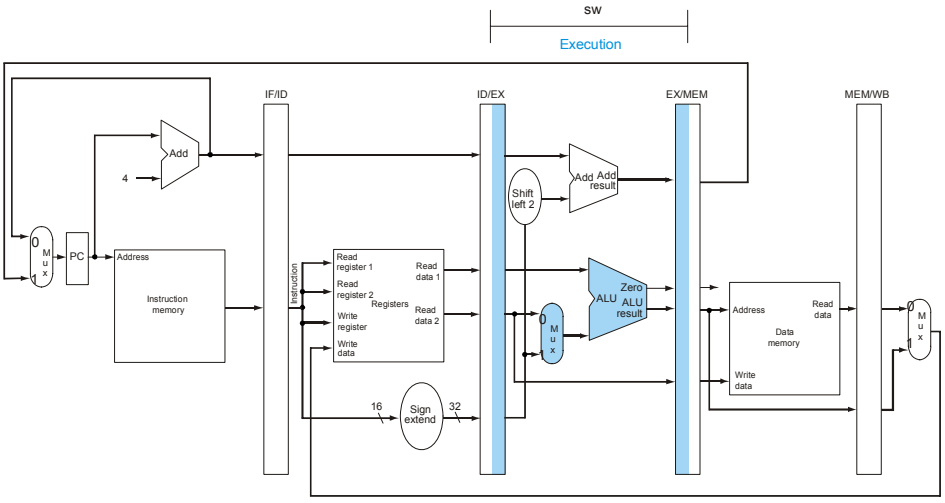


Διαδρομή δεδομένων με διοχέτευση

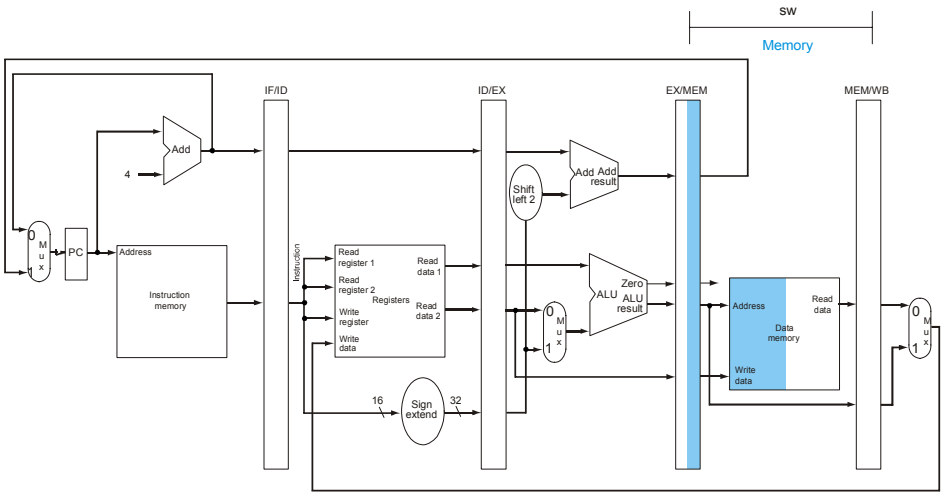
- Ο αριθμός καταχωρητή εγγραφής περνάει από το στάδιο ID της διοχέτευσης μέχρι να φτάσει στον καταχωρητή διοχέτευσης MEM/WB



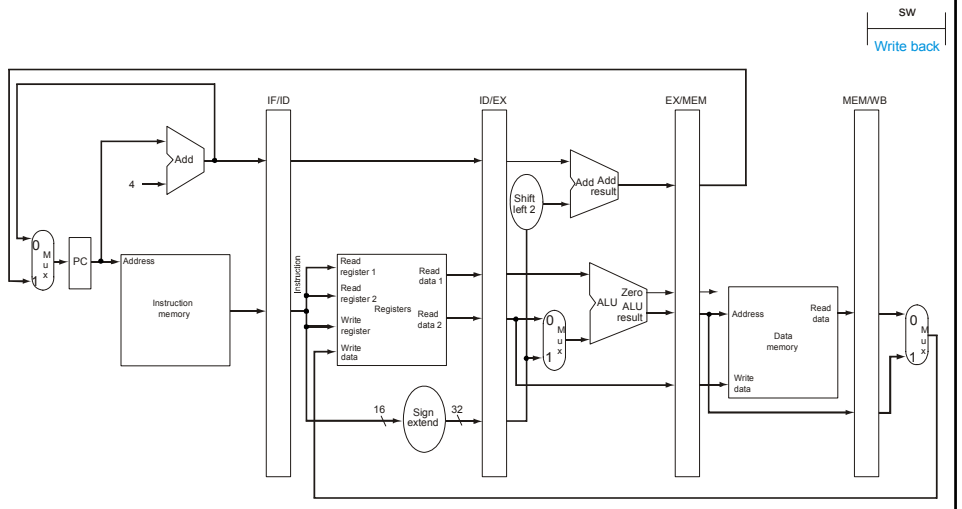
Στάδιο EX μιας εντολής store



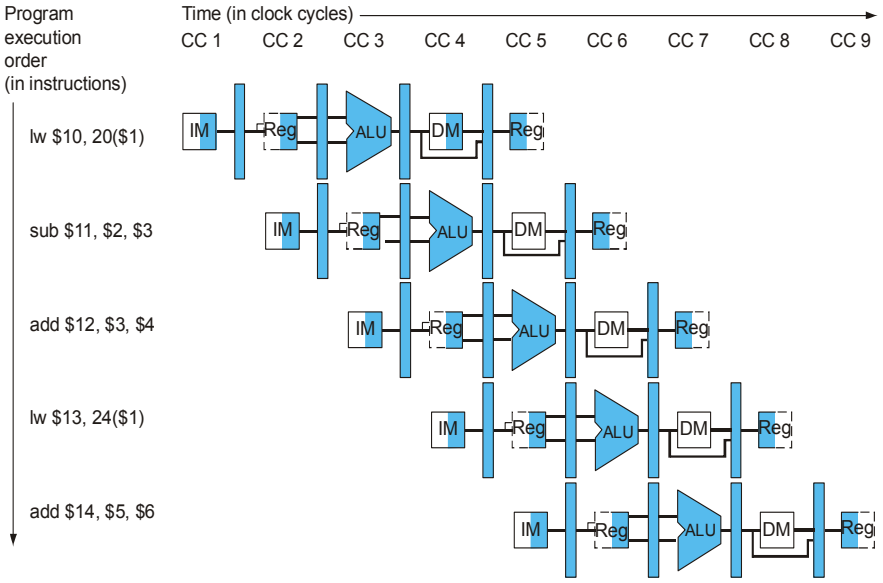
Στάδιο MEM μιας εντολής store



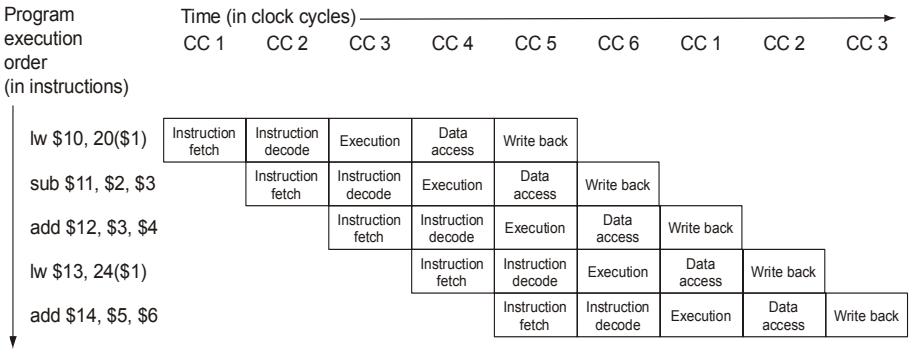
Στάδιο WB μιας εντολής store



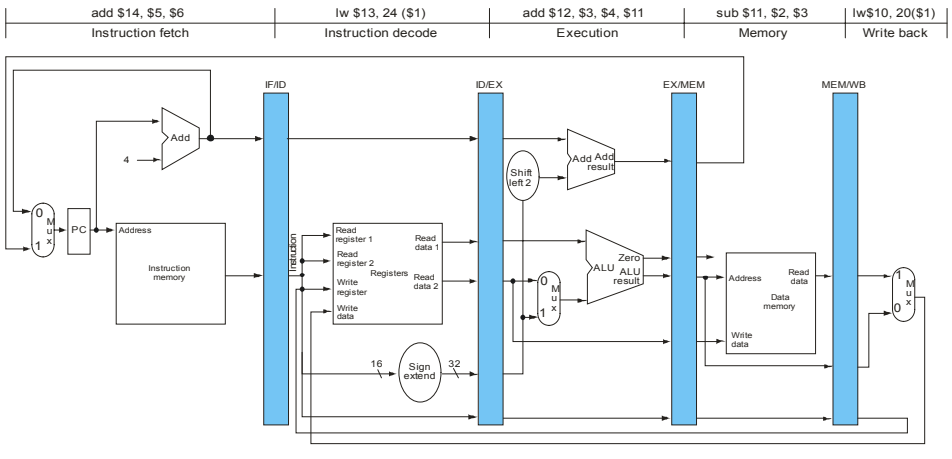
Διάγραμμα διοχέτευσης πολλών κύκλων



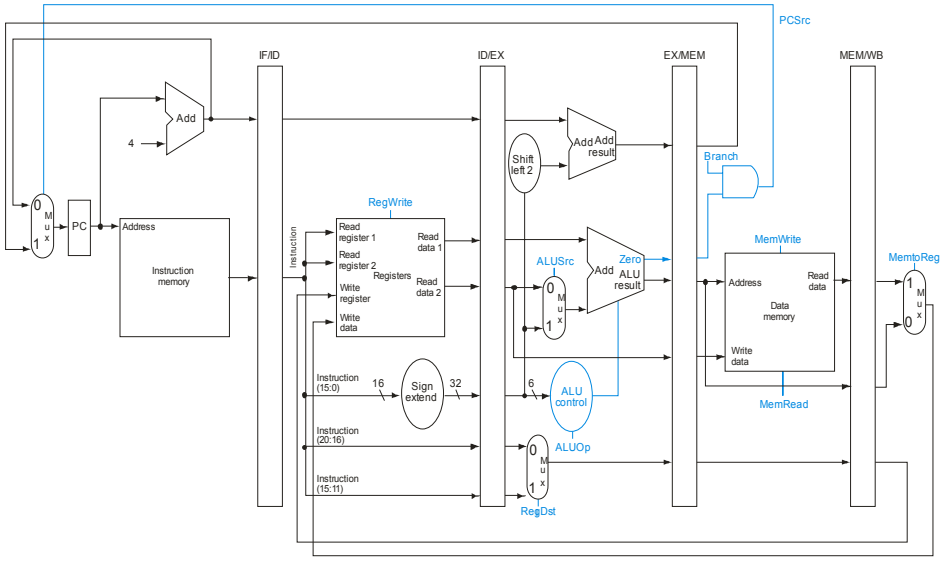
Διάγραμμα διοχέτευσης πολλών κύκλων



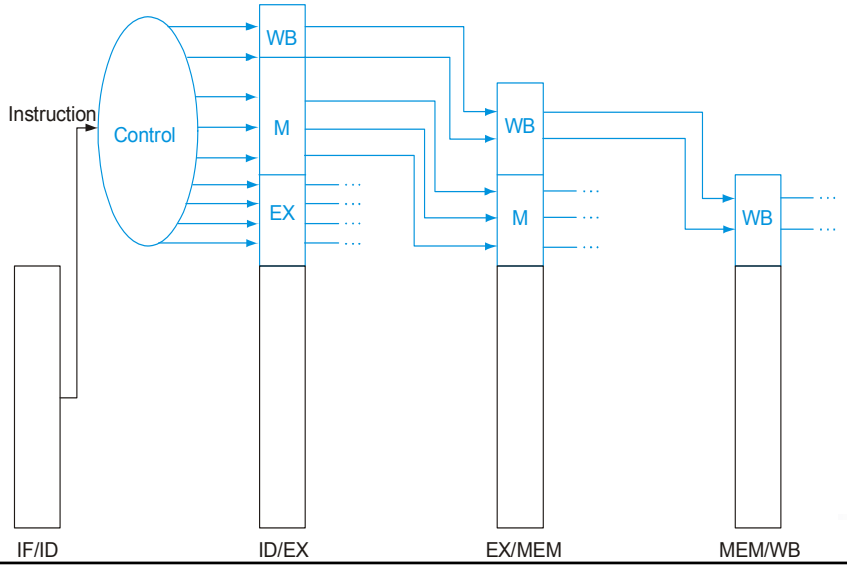
Διάγραμμα διοχέτευσης ενός κύκλου



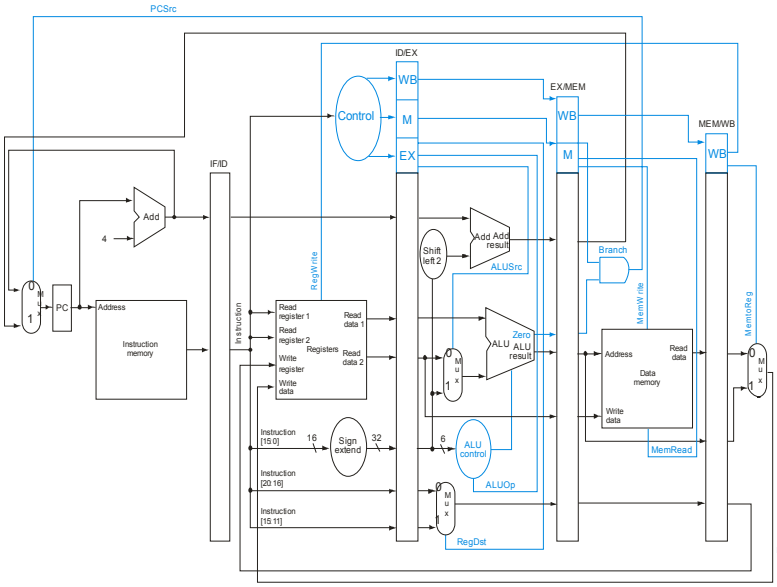
Μονάδα ελέγχου διοχέτευσης



Σήματα ελέγχου για τα στάδια: EX,MEM,WB



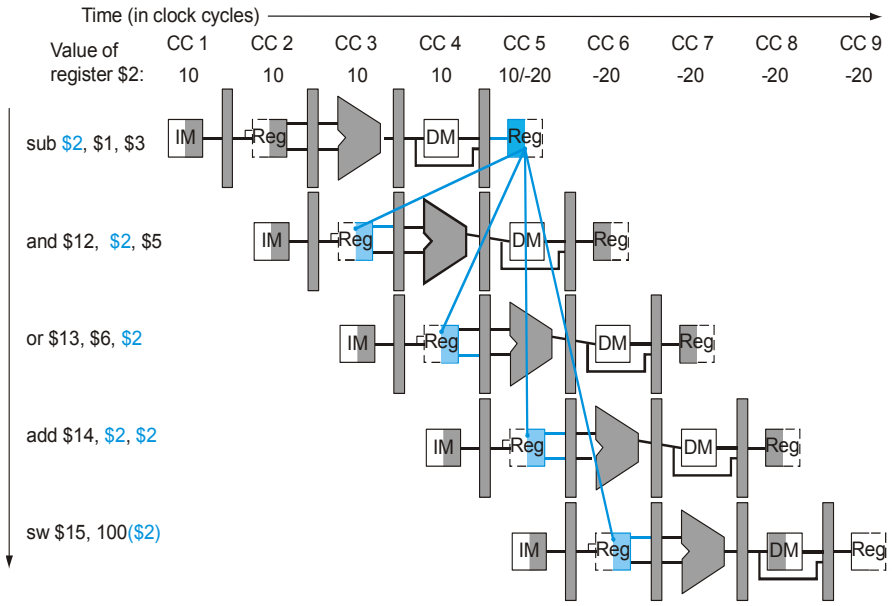
Έλεγχος διοχέτευσης



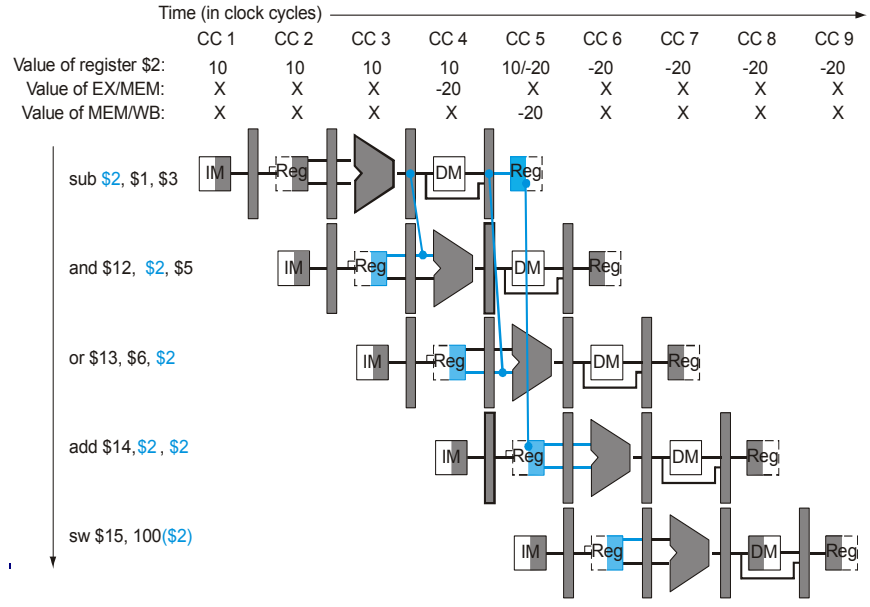
Κίνδυνοι διοχέτευσης (hazards)

- Η επόμενη εντολή δεν μπορεί να εκτελεστεί στον κύκλο ρολογιού που ακολουθεί
- Δομικοί κίνδυνοι (structural hazards):
  - Το υλικό δεν μπορεί να υποστηρίξει το συνδυασμό εντολών που έχουν οριστεί να εκτελεστούν σε έναν κύκλο
- Κίνδυνοι δεδομένων (data hazards):
  - Μία εντολή δεν μπορεί να εκτελεστεί στον κατάλληλο κύκλο επειδή χρειάζεται δεδομένα που δεν είναι ακόμα διαθέσιμα
- Κίνδυνοι ελέγχου (control hazards):
  - Δεν μπορεί να εκτελεστεί η κατάλληλη εντολή στον κατάλληλο κύκλο επειδή η ροή των εντολών δεν είναι αυτή που περίμενε η διοχέτευση

Εξαρτήσεις (dependencies)



Προώθηση (forwarding)



Ανίχνευση εξαρτήσεων

```
sub $2, $1, $3      # Ο καταχωρητής $2 γράφεται από τη sub
and $12, $2, $5     # Ο πρώτος τελεστέος ($2) εξαρτάται από τη sub
or $13, $6, $2      # Ο δεύτερος τελεστέος ($2) εξαρτάται από τη sub
add $14, $2, $2     # Ο πρώτος ($2) και ο δεύτερος ($2) εξαρτώνται από
                    # τη sub
sw $15, 100($2)     # Η βάση ($2) εξαρτάται από τη sub
```

- Συνθήκες κινδύνου (Εξάρτηση 1)
  - 1α. EX/MEM.RegisterRd = ID/EX.RegisterRs
  - 1β. EX/MEM.RegisterRd = ID/EX.RegisterRt
- Συνθήκες κινδύνου (Εξάρτηση 2)
  - 2α. MEM/WB.RegisterRd = ID/EX.RegisterRs
  - 2β. MEM/WB.RegisterRd = ID/EX.RegisterRt

Ανίχνευση εξαρτήσεων

- Εξάρτηση υπάρχει μόνο εάν η εντολή που προωθεί πρόκειται να γράψει σε κάποιον καταχωρητή, δηλαδή
  - EX/MEM.RegWrite ή
  - MEM/WB.RegWrite
- και ο καταχωρητής προορισμού δεν είναι ο $0
  - EX/MEM.RegisterRd ≠ 0 ή
  - MEM/WB.RegisterRd ≠ 0

Συνθήκες προώθησης

- Κίνδυνος EX
  (πολυπλέκτες στις εισόδους της ALU):

  αν (EX/MEM.RegWrite)
  και (EX/MEM.RegisterRd ≠ 0)
  και (EX/MEM.RegisterRd = ID/EX.RegisterRs))
  ForwardA = 10

  αν (EX/MEM.RegWrite)
  και (EX/MEM.RegisterRd ≠ 0)
  και (EX/MEM.RegisterRd = ID/EX.RegisterRt))
  ForwardB = 10

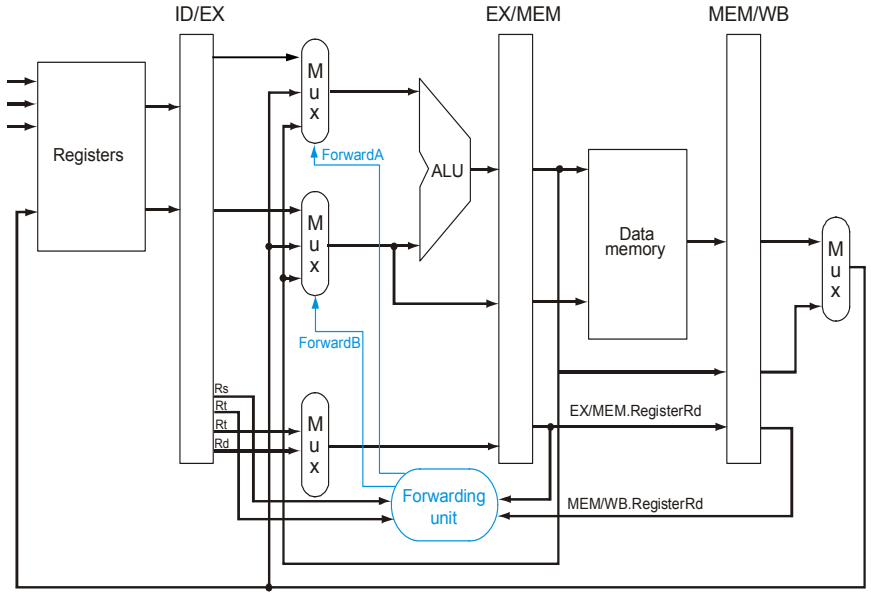
Συνθήκες προώθησης

- Κίνδυνος MEM
  (πολυπλέκτες στις εισόδους της ALU) :

  αν (MEM/WB.RegWrite)
  και (MEM/WB.RegisterRd ≠ 0)
  και (MEM/WB.RegisterRd = ID/EX.RegisterRs))
  ForwardA = 01

  αν (MEM/WB.RegWrite)
  και (MEM/WB.RegisterRd ≠ 0)
  και (MEM/WB.RegisterRd = ID/EX.RegisterRt))
  ForwardB = 01

Μονοπάτια προώθησης



Διπλός κίνδυνος δεδομένων

```
add $1,$1,$2
add $1,$1,$3
add $1,$1,$4
```

- Συμβαίνουν και οι δύο κίνδυνοι
  - Θέλουμε να χρησιμοποιήσουμε τον πιο πρόσφατο
- Αναθεώρηση της συνθήκης προώθησης του MEM
  - Προώθηση μόνο αν δεν υπάρχει εξάρτηση στο στάδιο EX

Συνθήκες προώθησης

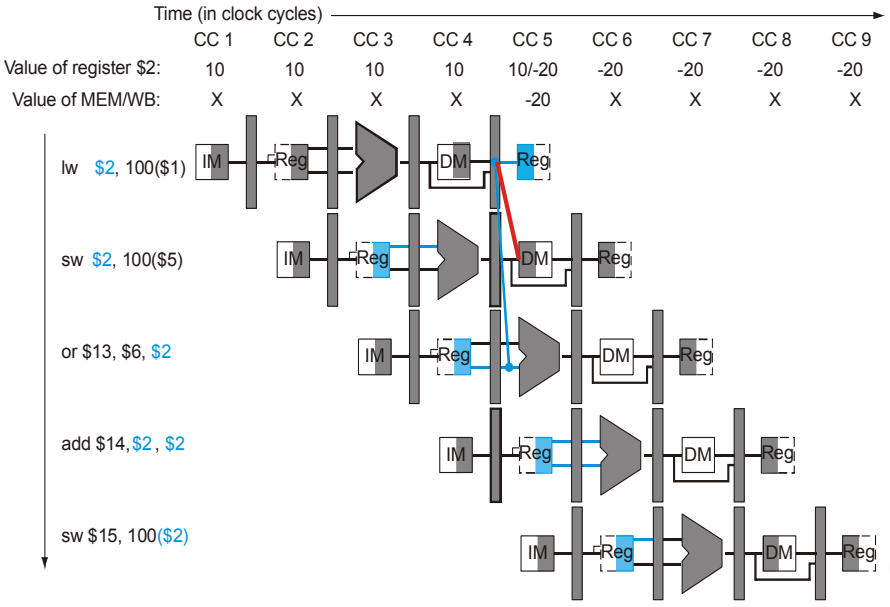
add $1,$1,$2
add $1,$1,$3
add $1,$1,$4

- Κίνδυνος MEM (πολυπλέκτες στις εισόδους της ALU) :

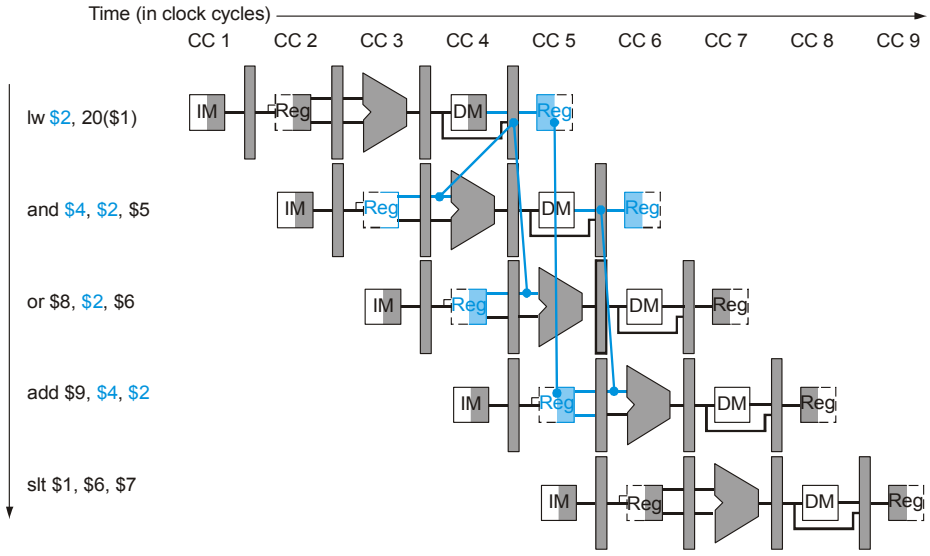
αν (MEM/WB.RegWrite)
και (MEM/WB.RegisterRd ≠ 0)
και (EX/MEM.RegisterRd ≠ ID/EX.RegisterRs)
και (MEM/WB.RegisterRd = ID/EX.RegisterRs))
ForwardA = 01

αν (MEM/WB.RegWrite)
αν (MEM/WB.RegisterRd ≠ 0)
και (EX/MEM.RegisterRd ≠ ID/EX.RegisterRt)
και (MEM/WB.RegisterRd = ID/EX.RegisterRt))
ForwardB = 01

Προώθηση στο στάδιο MEM



Κίνδυνος φόρτωσης-χρήσης

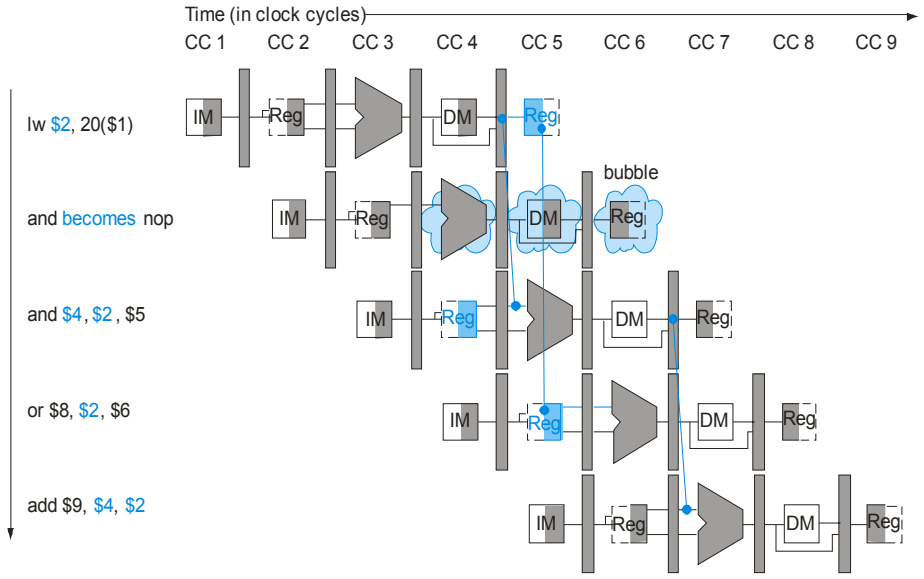


Μονάδα ανίχνευσης κινδύνου (hazard detection unit)

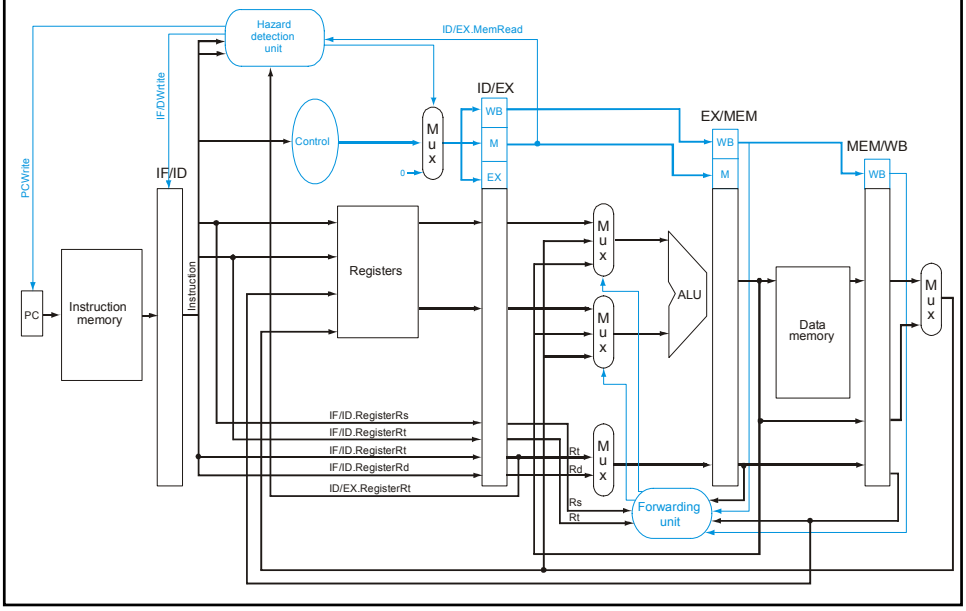
- Συνθήκες ανίχνευσης κινδύνου

  αν (ID/EX.MemRead) και
  ((ID/EX.RegisterRt = IF/ID.RegisterRs) ή
  (ID/EX.RegisterRt = IF/ID.RegisterRt))
  καθυστέρηση της διοχέτευσης

Καθυστερήσεις (stalls)



Μονάδα ανίχνευσης κινδύνου



Αναδιάταξη κώδικα

- Κώδικας σε C:

  A = B + E;

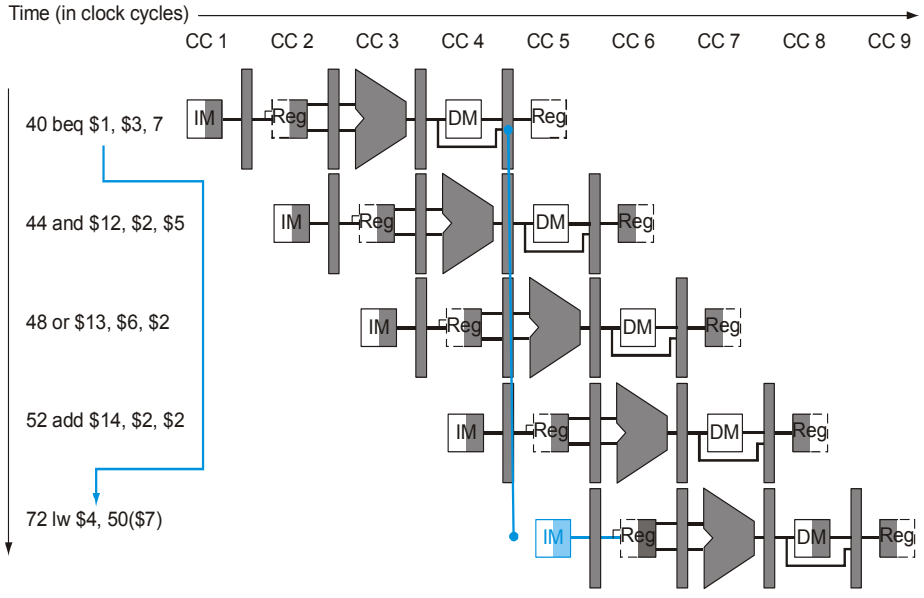
  C = B + F;

- Κώδικας του MIPS:

```
lw    $t1, 0($t0)
lw    $t2, 4($t0)
add   $t3, $t1,$t2
sw    $t3, 12($t0)
lw    $t4, 8($t0)
add   $t5, $t1,$t4
sw    $t5, 16($t0)
```

- Σχεδιάστε το διάγραμμα διοχέτευσης
- Βρείτε τους κινδύνους
- Αναδιατάξτε τις εντολές για να αποφύγετε τις καθυστερήσεις

Κίνδυνοι ελέγχου (control hazards)



Τεχνικές για την αντιμετώπιση των κινδύνων ελέγχου

- Μείωση της καθυστέρησης των διακλαδώσεων
  - Υπολογισμός της διακλάδωσης νωρίτερα (MIPS: στο στάδιο ID αντί στο στάδιο MEM)
- Πρόβλεψη διακλάδωσης (branch prediction)
  - Στατική πρόβλεψη διακλάδωσης
  - Δυναμική πρόβλεψη διακλάδωσης
- Καθυστερημένη διακλάδωση (delayed branch)
  - MIPS: καθυστέρηση διακλάδωσης ενός κύκλου

Στατική πρόβλεψη διακλάδωσης

- Υπόθεση μη ληφθείσας διακλάδωσης
- Σωστή πρόβλεψη: η διακλάδωση δεν λαμβάνεται
  - Καμία καθυστέρηση
- Λάθος πρόβλεψη: η διακλάδωση λαμβάνεται
  - Καθυστέρηση και απόρριψη των εντολών
  - Εάν η διακλάδωση υπολογίζεται στο στάδιο MEM, εκκένωση των εντολών (flush instructions) στα στάδια IF, ID, EX

Μείωση της καθυστέρησης

- Μετακίνηση της απόφασης διακλάδωσης στο στάδιο ID
  - Υπολογισμός της διεύθυνσης προορισμού
  - Αξιολόγηση της απόφασης
- Μετακίνηση του αθροιστή/συγκριτή διακλάδωσης στο στάδιο ID
- Επιπλέον υλικό:
  - Μονοπάτια προώθησης
  - Ανίχνευση καθυστερήσεων

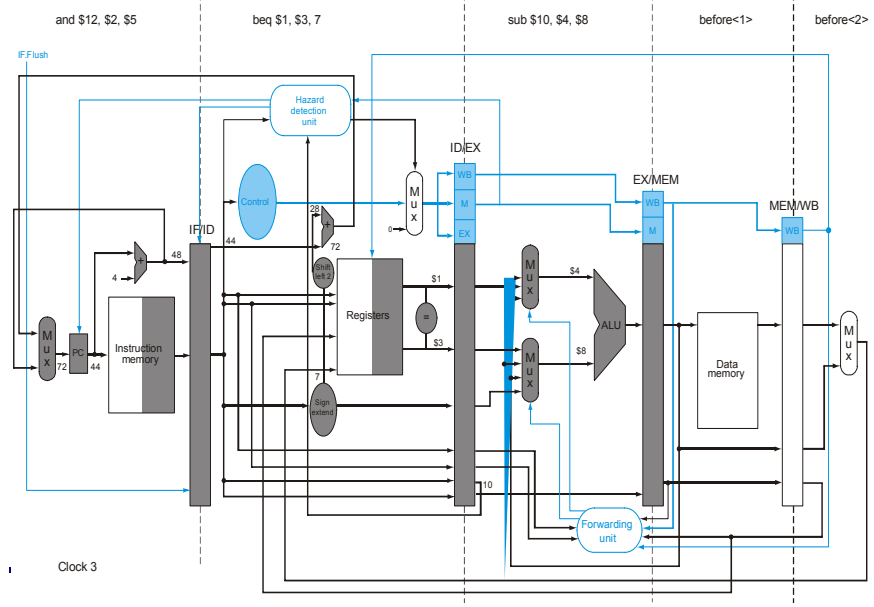
Παράδειγμα

- Διάγραμμα διοχέτευσης ενός κύκλου για την ακολουθία εντολών:

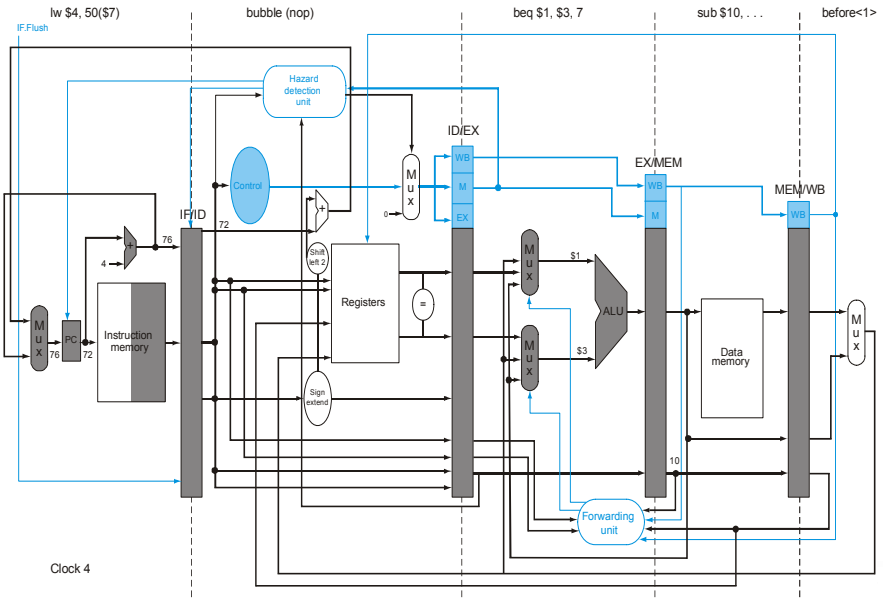
```
36  sub $10,$4,$8
40  beq $1,$3,7
44  and $12,$2,$5
48  or $13,$2,$6
52  add $14,$4,$2
56  slt $15,$6,$7
...
72  lw $4,50($7)
```

- Η διακλάδωση λαμβάνεται
- Ο μηχανισμός διοχέτευσης
  - Χρησιμοποιεί την τεχνική μη ληφθείσας διακλάδωσης
  - Υπολογίζει τη διακλάδωση στο στάδιο ID

Διάγραμμα ενός κύκλου (κύκλος 3)



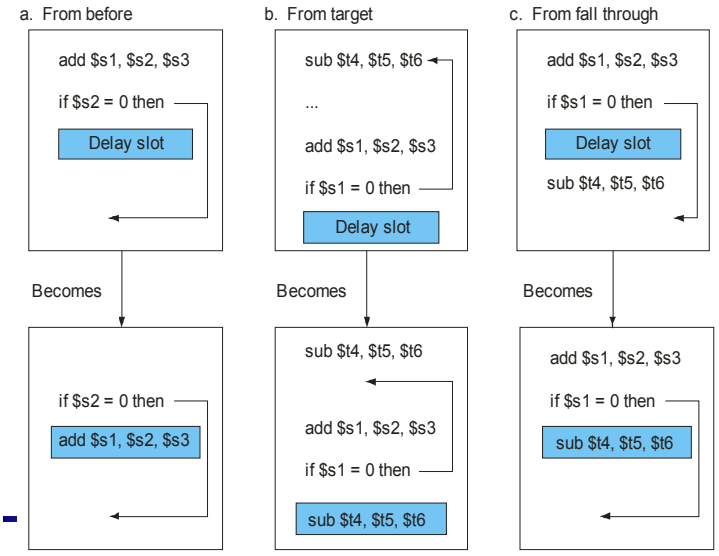
Διάγραμμα ενός κύκλου (κύκλος 4)



Καθυστερημένη διακλάδωση

- Εκτελεί πάντοτε την επόμενη στη σειρά εντολή
- Υποδοχή καθυστέρησης διακλάδωσης (branch delay slot)
  - Οι μεταγλωττιστές και οι συμβολομεταφραστές τοποθετούν εντολές στην υποδοχή
- Αποδοτική λύση για την διοχέτευση 5-σταδίων

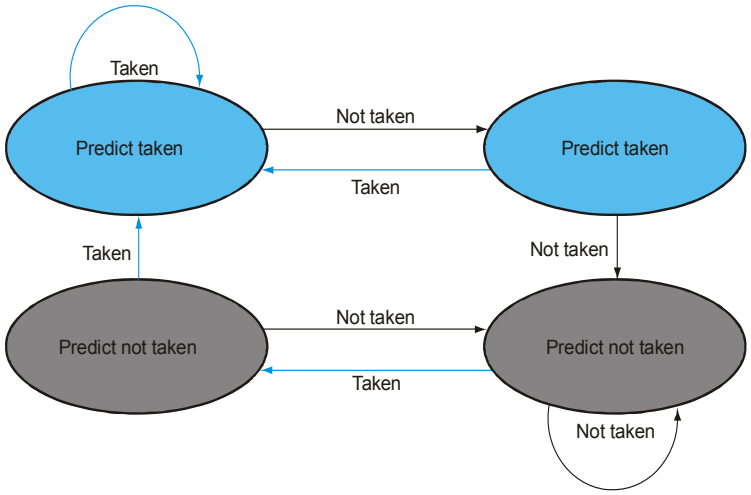
Χρονοπρογραμματισμός της υποδοχής καθυστέρησης διακλάδωσης



Δυναμική πρόβλεψη διακλάδωσης

- Πρόβλεψη διακλαδώσεων κατά το χρόνο εκτέλεσης με τη χρήση πληροφοριών που συλλέγονται κατά το χρόνο εκτέλεσης
- Προσωρινή μνήμη πρόβλεψης διακλάδωσης (branch prediction buffer)
  - Δεικτοδοτείται από το χαμηλότερο τμήμα της διεύθυνσης της εντολής διακλάδωσης
  - Περιέχει ένα bit που λέει αν η διακλάδωση λήφθηκε πρόσφατα ή όχι
  - Μειονέκτημα απόδοσης: στην πρόβλεψη βρόγχων

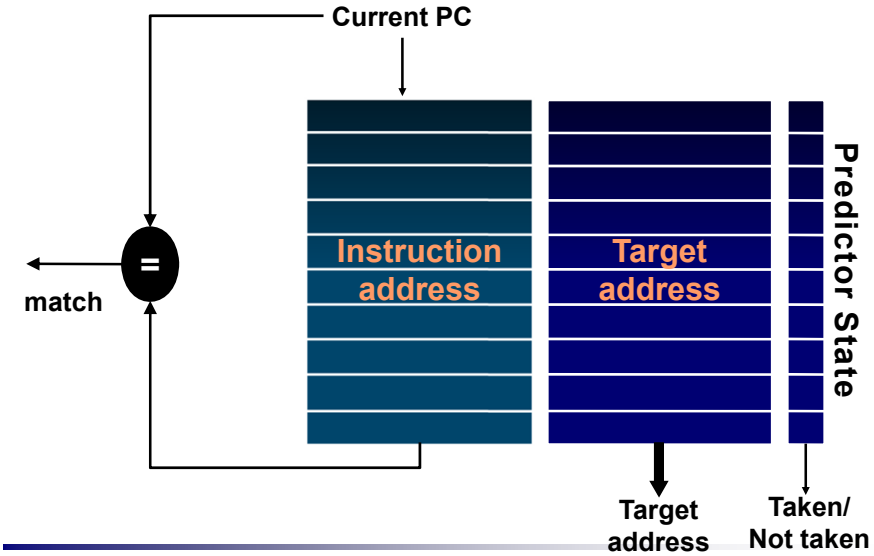
Μέθοδος πρόβλεψης 2 bit



Επιπλέον τεχνικές πρόβλεψης

- Προσωρινή μνήμη προορισμού διακλάδωσης (branch target buffer)
  - Αποθηκεύει την εντολή προορισμού για μια διακλάδωση
- Διάταξη πρόβλεψης συσχέτισης (correlating predictor)
  - Συνδυάζει την τοπική (local) συμπεριφορά μιας διακλάδωσης με την καθολική (global) συμπεριφορά διακλαδώσεων που εκτελέστηκαν πρόσφατα
- Επιλεκτική διάταξη πρόβλεψης διακλάδωσης (tournament predictor)
  - Πολλές προβλέψεις για κάθε διακλάδωση (local ή global predictors) και έναν μηχανισμό επιλογής που διαλέγει ποια διάταξη πρόβλεψης να ενεργοποιήσει

Branch Target Buffer (BTB)



Correlating Predictors

```
if (aa==2)
   aa=0;
if (bb==2)
   bb=0;
if (aa!=bb) {
```

```
    ADDI R3,R1,-2     ;R1 <- aa
    BNEZ R3,L1        ;branch b1 (aa!=2)
    ADD R1,R0,R0      ;aa=0
L1: ADDI R3,R2,-2     ;R2 <- bb
    BNEZ R3,L2        ;branch b2 (bb!=2)
    ADD R2,R0,R0      ;bb=0
L2: SUB R3,R1,R2      ;R3=aa-bb
    BEQZ R3,L3        ;branch b3 (aa==bb)
```

- Εάν οι διακλαδώσεις b1 και b2 είναι NT (not taken) τότε η διακλάδωση b3 είναι T (taken)
  - Αυτήν τη συμπεριφορά μπορούν να την προβλέψουν μηχανισμοί που συσχετίζουν την συμπεριφορά της διακλάδωσης b3 με την συμπεριφορά των b1 και b2

Εξαιρέσεις (exceptions)

- Πιθανές αιτίες εξαιρέσεων
  - Αριθμητική υπερχείλιση
  - Χρήση μιας μη ορισμένης εντολής
  - Κλήση μιας υπηρεσίας του λειτουργικού συστήματος
  - Αίτηση συσκευής εισόδου/εξόδου
  - Εσφαλμένη λειτουργία του υλικού
- Το υλικό και το λογισμικό πρέπει να συνεργαστούν για να χειριστούν τις εξαιρέσεις

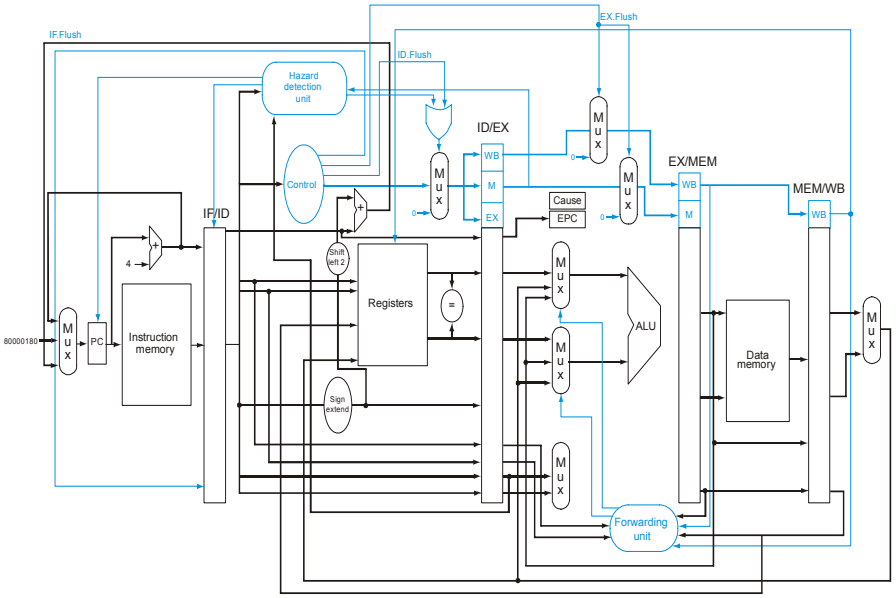
Χειρισμός εξαιρέσεων

- Η εντολή add οδηγεί σε υπερχείλιση
  - add $1, $2, $1
- Τι ενέργειες πρέπει να γίνουν από το υλικό;
  - Να σταματήσει η εκτέλεση της προβληματικής εντολής
  - Να ολοκληρωθεί η εκτέλεση των προηγούμενων εντολών
  - Να εκκενωθούν οι εντολές που ακολουθούν
  - Να αποθηκευτεί η αιτία της εξαίρεσης
    - Cause Register
  - Να αποθηκευτεί η διεύθυνση της προβληματικής εντολής
    - Exception Program Counter (EPC)
  - Να μεταφερθεί ο έλεγχος σε μια προκαθορισμένη διεύθυνση
    - Ρουτίνα χειρισμού εξαιρέσεων (exception service routine)

Χειρισμός εξαιρέσεων

- Τι ενέργειες πρέπει να γίνουν από το λογισμικό (λειτουργικό σύστημα);
  - Για μη ορισμένη εντολή, αστοχία υλικού, ή αριθμητική υπερχείλιση
    - το λειτουργικό σύστημα «σκοτώνει» (kills) το πρόγραμμα και επιστρέφει μια ένδειξη
  - Για αίτηση συσκευής εισόδου/εξόδου ή κλήση εξυπηρέτησης του λειτουργικού συστήματος
    - το λειτουργικό σύστημα αποθηκεύει την κατάσταση του προγράμματος
    - πραγματοποιεί την κατάλληλη εργασία
    - επαναφέρει αργότερα το πρόγραμμα για να συνεχίσει την εκτέλεση

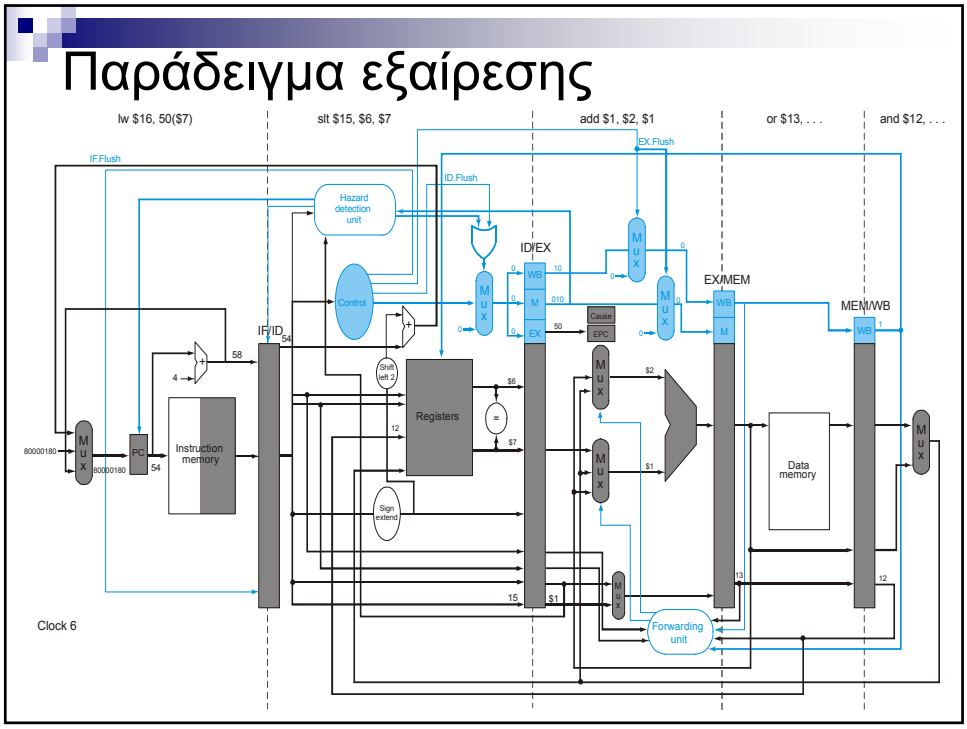
Διοχέτευση με εξαιρέσεις



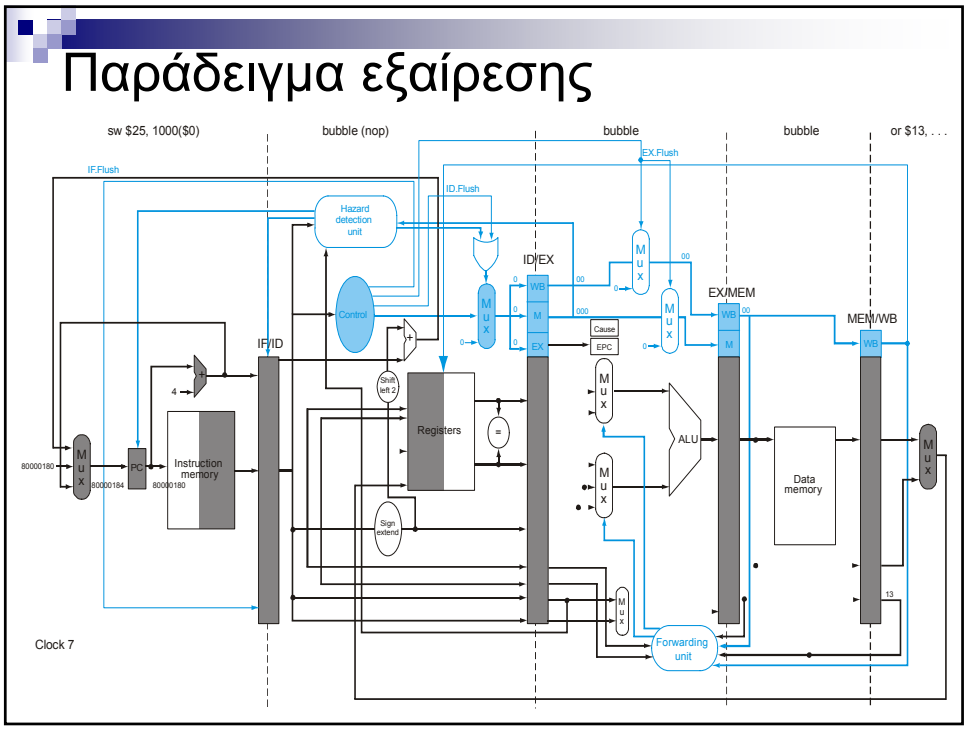
Παράδειγμα εξαίρεσης

- Ακολουθία εντολών
  - 40_h sub $11,$2,$4
  - 44_h and $12,$2,$5
  - 48_h or $13,$2,$6
  - $4C_h$ add $1,$2,$1
  - 50_h slt $15,$6,$7
  - 54_h lw $16,50($7)
- Τι συμβαίνει στη διοχέτευση αν συμβεί μια εξαίρεση υπερχείλισης στην εντολή add
- Υποθέστε ότι οι εντολές που θα κληθούν σε μια εξαίρεση είναι:
  - 80000180_h sw $25,1000($0)
  - 80000184_h sw $26,1004($0)

Παράδειγμα εξαίρεσης



Παράδειγμα εξαίρεσης



Προηγμένη διοχέτευση

- Παραλληλία επιπέδου εντολής (Instruction Level Parallelism, ILP)
- Μέθοδοι αύξησης της παραλληλίας επιπέδου εντολής
  - Αύξηση του βάθους της διοχέτευσης
  - Πολλαπλή εκκίνηση (multiple issue)

Πολλαπλή εκκίνηση

- Ιδέα: Επανάληψη των εσωτερικών πόρων ώστε να είναι δυνατή η εκκίνηση πολλαπλών εντολών σε κάθε στάδιο
- Μειονέκτημα: επιπλέον εργασία για να διατηρηθούν απασχολημένοι όλοι οι πόροι
- Η εκκίνηση πολλών εντολών ανά στάδιο οδηγεί σε CPI < 1
  - Χρήση του IPC (instructions per clock)
- Παράδειγμα: επεξεργαστής 4 δρόμων πολλαπλής εκκίνησης στα 6 GHz
  - Μέγιστος ρυθμός 24 δισεκατομμυρίων εντολών ανά δευτερόλεπτο
  - CPI 0,25 ή IPC 4 στην ιδανική περίπτωση
  - Σε διοχέτευση 5 σταδίων: 20 εντολές σε εκτέλεση κάθε στιγμή
  - Περιορισμοί στην πολλαπλή εκκίνηση:
    - Τύποι εντολών που δεν μπορούν να εκτελεστούν ταυτόχρονα
    - Εξαρτήσεις δεδομένων μεταξύ των εντολών

Πολλαπλή εκκίνηση

- Ανάλογα με τον καταμερισμό εργασιών μεταξύ υλικού και μεταγλωττιστή:
  - Στατική πολλαπλή εκκίνηση (static multiple issue)
    - πολλές αποφάσεις λαμβάνονται από το μεταγλωττιστή πριν από την εκτέλεση
  - Δυναμική πολλαπλή εκκίνηση (dynamic multiple issue)
    - πολλές αποφάσεις λαμβάνονται από τον επεξεργαστή κατά τη διάρκεια της εκτέλεσης

Διοχέτευση πολλαπλής εκκίνησης

- Συγκέντρωση εντολών σε υποδοχές εκκίνησης (issue slots)
  - Πόσες και ποιες εντολές μπορούν να ξεκινήσουν σε ένα δεδομένο κύκλο ρολογιού;
- Αντιμετώπιση κινδύνων δεδομένων και ελέγχου
  - Πώς αντιμετωπίζονται οι συνέπειες των κινδύνων δεδομένων και ελέγχου;

Εικασία (speculation)

- Εικάζει το αποτέλεσμα μιας εντολής, ώστε να επιτραπεί η εκκίνηση άλλων εντολών που μπορεί να εξαρτώνται από αυτήν
  - Μεταγλωττιστής: χρησιμοποιεί την εικασία για να αναδιατάξει εντολές
  - Υλικό: αναδιατάσσει τις εντολές κατά το χρόνο εκτέλεσης

Εικασία (speculation)

- Τι γίνεται στην περίπτωση που η εικασία είναι λανθασμένη;
  - Μεταγλωττιστής: προσθέτει επιπλέον εντολές που ελέγχουν την ορθότητα της εικασίας και ρουτίνες επιδιόρθωσης
  - Υλικό: τοποθετεί τα αποτελέσματα σε προσωρινές μνήμες μέχρι να διαπιστώσει ότι δεν είναι εικασίες
- Τι γίνεται στην περίπτωση που η εικασία εισάγει εξαιρέσεις που κανονικά δεν έπρεπε να συμβούν;

Στατική πολλαπλή εκκίνηση

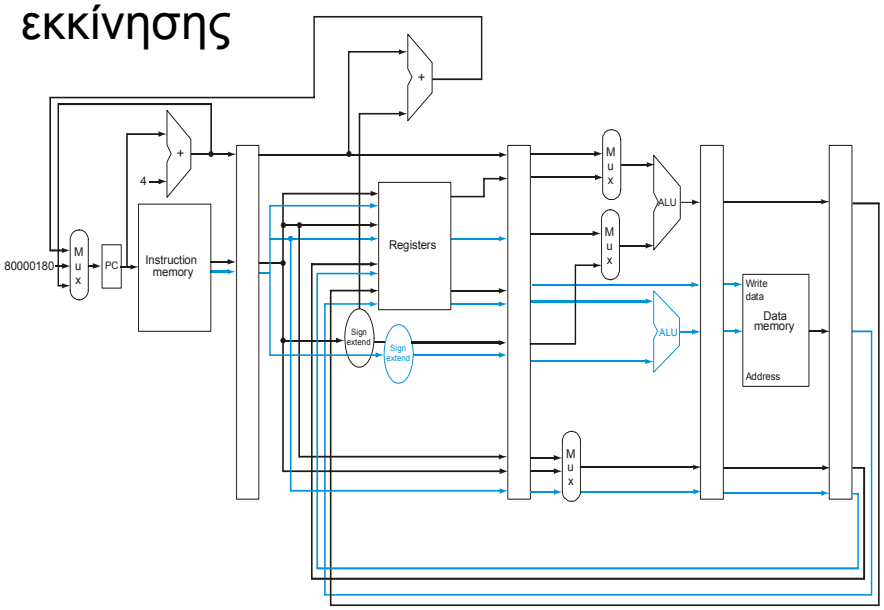
- Πακέτο εκκίνησης (issue packet): σύνολο εντολών που ξεκινούν σε έναν κύκλο ρολογιού
  - Συνήθως το μίγμα των εντολών ενός πακέτου εκκίνησης είναι περιορισμένο
    - θεωρούν το πακέτο εκκίνησης ως μια μοναδική εντολή που με διαφορετικές λειτουργίες σε συγκεκριμένα προκαθορισμένα πεδία.
  - Πολύ Μεγάλη Λέξη Εντολής (VLIW: Very Long Instruction Word):
    - θεωρούν το πακέτο εκκίνησης ως μια μεγάλη εντολή με πολλές λειτουργίες
  - Itanium (2000), Itanium 2 (2002):
    - αρχιτεκτονική Intel IA-64, EPIC (Explicitly Parallel Instruction Computer)

Διοχέτευση στατικής διπλής εκκίνησης

- Εντολή 1: Εντολή ALU ή διακλάδωση
- Εντολή 2: Εντολή φόρτωσης ή αποθήκευσης
- Προσκόμιση και αποκωδικοποίηση εντολών 64-bit

Τύπος εντολής	Στάδια διοχέτευσης							
Εντολή ALU ή διακλάδωσης	IF	ID	EX	MEM	WB			
Εντολή φόρτωσης ή αποθήκευσης	IF	ID	EX	MEM	WB			
Εντολή ALU ή διακλάδωσης		IF	ID	EX	MEM	WB		
Εντολή φόρτωσης ή αποθήκευσης		IF	ID	EX	MEM	WB		
Εντολή ALU ή διακλάδωσης			IF	ID	EX	MEM	WB	
Εντολή φόρτωσης ή αποθήκευσης			IF	ID	EX	MEM	WB	
Εντολή ALU ή διακλάδωσης				IF	ID	EX	MEM	WB
Εντολή φόρτωσης ή αποθήκευσης				IF	ID	EX	MEM	WB

Στατική διαδρομή δεδομένων διπλής εκκίνησης



Χρονοπρογραμματισμός κώδικα πολλαπλής εκκίνησης

```
Loop:   lw   $t0,0($s1)
        addu $t0,$t0,$s2
        sw   $t0,0($s1)
        addi $s1,$s1,-4
        bne  $s1,$zero,Loop
```

- Αναδιατάξτε τις εντολές ώστε να αποφύγετε όσο το δυνατό περισσότερες καθυστερήσεις
- Υποθέστε ότι οι κίνδυνοι ελέγχου διαχειρίζονται από το υλικό

Χρονοπρογραμματισμός κώδικα πολλαπλής εκκίνησης

```
Loop:   lw   $t0,0($s1)
        addi $s1,$s1,-4
        addu $t0,$t0,$s2
        sw   $t0,4($s1)
        bne  $s1,$zero,Loop
```

	Εντολή ALU ή διακλάδωσης	Εντολή μεταφοράς δεδομένων	Κύκλος ρολογιού
`Loop:`		`lw $t0,0($s1)`	1
	`addi $s1,$s1,-4`		2
	`addu $t0,$t0,$s2`		3
	`bne $s1,$zero,Loop`	`sw $t0,4($s1)`	4

Ξετύλιγμα βρόχου (loop unrolling)

```
/* πριν */
for (i=0; i<4; i++)
  a[i] = b[i] * c[i];

/* μετά */
for (i=0; i<2; i++) {
  a[i*2] = b[i*2] * c[i*2];
  a[i*2+1] = b[i*2+1] *
  c[i*2+1];}

/* ή ξετύλιγμα 4 φορές */
i = 0;
a[i] = b[i] * c[i];
a[i+1] = b[i+1] * c[i+1];
a[i+2] = b[i+2] * c[i+2];
a[i+3] = b[i+3] * c[i+3];
```

Ξετύλιγμα βρόχου

```
Loop: lw   $t0,0($s1)
      addu $t0,$t0,$s2
      sw   $t0,0($s1)
      addi $s1,$s1,-4
      bne $s1,$zero,Loop
```

```
Loop: lw   $t0,0($s1)
      addu $t0,$t0,$s2
      sw   $t0,0($s1)
      addi $s1,$s1,-16
      lw   $t0,12($s1)
      addu $t0,$t0,$s2
      sw   $t0,12($s1)
      lw   $t0,8($s1)
      addu $t0,$t0,$s2
      sw   $t0,8($s1)
      lw   $t0,4($s1)
      addu $t0,$t0,$s2
      sw   $t0,4($s1)
      bne s1,$zero,Loop
```


Μετονομασία καταχωρητών

```
Loop: lw   $t0,0($s1)
      addu $t0,$t0,$s2
      sw   $t0,0($s1)
      addi $s1,$s1,-16
      lw   $t0,12($s1)
      addu $t0,$t0,$s2
      sw   $t0,12($s1)
      lw   $t0,8($s1)
      addu $t0,$t0,$s2
      sw   $t0,8($s1)
      lw   $t0,4($s1)
      addu $t0,$t0,$s2
      sw   $t0,4($s1)
      bne  $s1,$zero,Loop
```

```
Loop: lw   $t0,0($s1)
      addu $t0,$t0,$s2
      sw   $t0,0($s1)
      addi $s1,$s1,-16
      lw   $t1,12($s1)
      addu $t1,$t1,$s2
      sw   $t1,12($s1)
      lw   $t2,8($s1)
      addu $t2,$t2,$s2
      sw   $t2,8($s1)
      lw   $t3,4($s1)
      addu $t3,$t3,$s2
      sw   $t3,4($s1)
      bne  $s1,$zero,Loop
```

Χρονοπρογραμματισμός ξετυλιγμένου κώδικα πολλαπλής εκκίνησης

	Εντολή ALU ή διακλάδωσης	Εντολή μεταφοράς δεδομένων	Κύκλος ρολογιού
`Loop:`	`addi $s1,$s1,-16`	`lw $t0,0($s1)`	1
		`lw $t1,12($s1)`	2
	`addu $t0,$t0,$s2`	`lw $t2,8($s1)`	3
	`addu $t1,$t1,$s2`	`lw $t3,4($s1)`	4
	`addu $t2,$t2,$s2`	`sw $t0,16($s1)`	5
	`addu $t3,$t3,$s2`	`sw $t1,12($s1)`	6
		`sw $t2,8($s1)`	7
	`bne $s1,$zero,Loop`	`sw $t3,4($s1)`	8