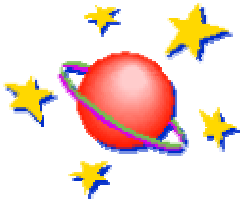
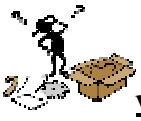


Ενότητα 5η : *Η μέθοδος Tomasulo*

Σκοπός **Ο σκοπός της ενότητας αυτής είναι να παρουσιάσει την τεχνική του Tomasulo, που αποτελεί μία άλλη προσέγγιση του δυναμικού προγραμματισμού, καθώς επίσης και να συγκρίνει τη μέθοδο Tomasulo με αυτή του πίνακα αποτελεσμάτων.**

Προσδοκώμενα Αποτελέσματα **Όταν θα έχετε μελετήσει την ενότητα, θα είστε σε θέση:**

<u>να εφαρμόζετε την τεχνική του Tomasulo,</u>



<u>να αναφέρετε παραδείγματα της τεχνικής Tomasulo,</u>

<u>να συγκρίνετε την τεχνική Tomasulo με την τεχνική του πίνακα αποτελεσμάτων.</u>

τεχνική Tomasulo, σταθμός κράτησης, μετονομασία καταχωρητών, πεδίο ετικέτα

Η τεχνική TOMASULO

Η τεχνική Tomasulo αποτελεί μία άλλη προσέγγιση του δυναμικού προγραμματισμού. Επινοήθηκε από τον Robert Tomasulo από όπου και πήρε το όνομά της. Αυτή η τεχνική, όπως και η τεχνική του πίνακα αποτελεσμάτων, επιτρέπει την εκτέλεση εντολών παρουσία κινδύνων.

Για πρώτη φορά, χρησιμοποιήθηκε για την μονάδα κινητής υποδιαστολής του IBM 360/91 με στόχο την υψηλή απόδοσή της. Ο υπολογιστής IBM 360/91 ολοκληρώθηκε περίπου τρία χρόνια μετά τον CDC 6600. Η τεχνική Tomasulo συνδυάζει κάποια βασικά χαρακτηριστικά των τεχνικών του πίνακα αποτελεσμάτων και της μετονομασίας καταχωρητών.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 1

Θυμάστε τι είναι η μετονομασία καταχωρητή; Να αναφέρετε τι επιτυγχάνεται με τη χρήσης αυτής της τεχνικής, καθώς και τον τρόπο με τον οποίο υλοποιείται αυτή η τεχνική. Για περισσότερες λεπτομέρειες καλό θα ήταν να ανατρέξετε στη 2η ενότητα: «Είδη εξαρτήσεων, εξαρτήσεις δεδομένων και ονομάτων».

ΑΠΑΝΤΗΣΗ ΔΡΑΣΤΗΡΙΟΤΗΤΑΣ 1

Η μέθοδος της μετονομασίας καταχωρητή εξαλείφει δυναμικά τις εξαρτήσεις ονομάτων με σκοπό να αποφευχθούν οι WAR και WAW κίνδυνοι. Αυτό επιτυγχάνεται αντικαθιστώντας τα ονόματα των καταχωρητών με άλλα, από ένα μεγαλύτερο σύνολο εικονικών καταχωρητών.
Η μετονομασία καταχωρητή μπορεί να υλοποιηθεί με δύο τρόπους :

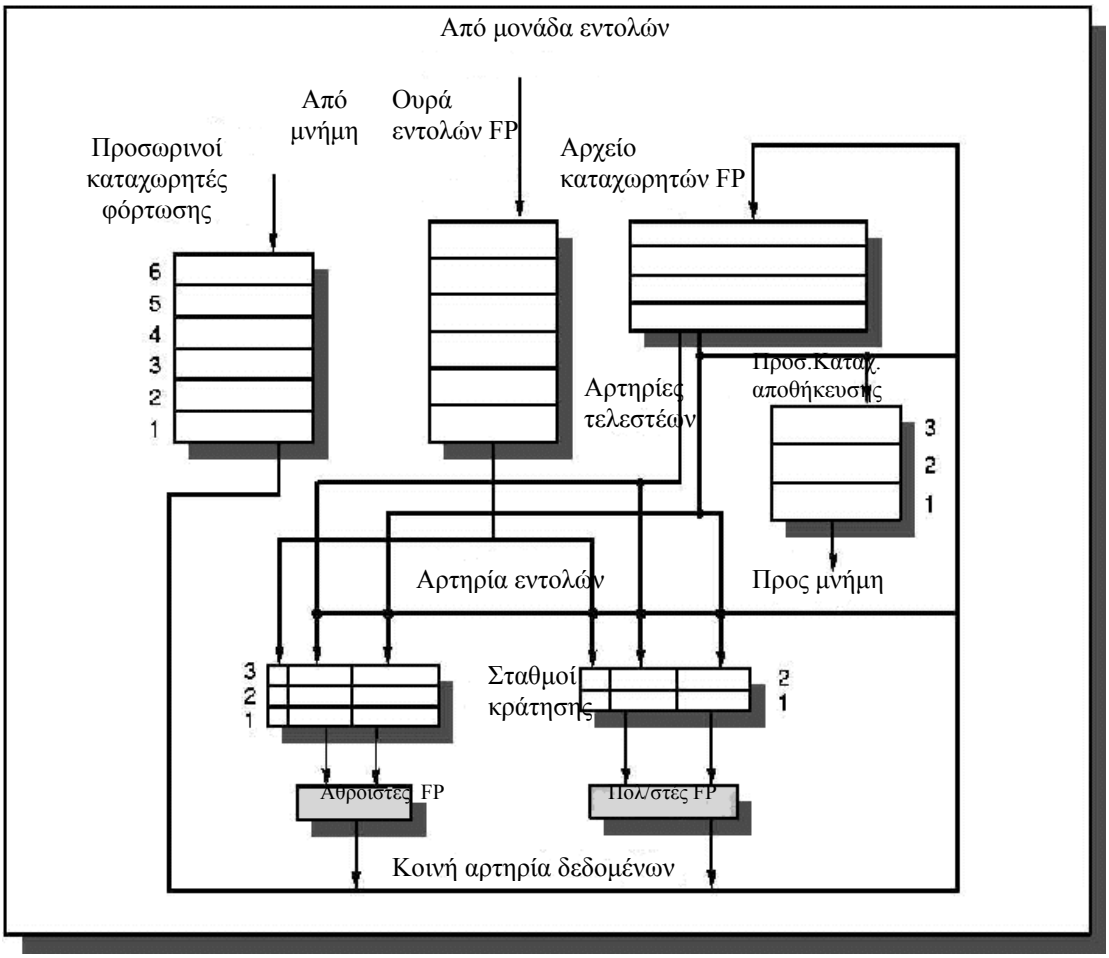
- **Στατικά, από έναν μεταγλωττιστή.**
- **Δυναμικά με χρήση του υλικού.**

Γενικά υπάρχουν οι σωληνωμένες λειτουργικές μονάδες και οι πολλαπλές λειτουργικές μονάδες. Η διαφορά τους είναι ότι η σωληνωμένη λειτουργική μονάδα ξεκινάει το πολύ μία λειτουργία σε κάθε κύκλο ρολογιού. Επειδή δεν υπάρχουν βασικές διαφορές θα υποθέσουμε ότι έχουμε πολλαπλές λειτουργικές μονάδες κατά την περιγραφή του αλγορίθμου του Tomasulo.

Στον αλγόριθμο Tomasulo η μετονομασία των καταχωρητών πραγματοποιείται με την βοήθεια των σταθμών κράτησης. Οι σταθμοί κράτησης

ανακαλούν και καταχωρούν προσωρινά έναν τελεστέο μιας εντολής, μόλις βέβαια γίνει διαθέσιμος. Με αυτό τον τρόπο δε χρειάζεται να επαναληφθεί η ανάγνωση ενός συγκεκριμένου τελεστέου από έναν καταχωρητή, κατά τη διάρκεια εκτέλεσης του προγράμματος.

Οι εντολές που βρίσκονται σε αναμονή προσδιορίζουν τον σταθμό κράτησης από τον οποίο θα πάρουν τις εισόδους τους. Επιπλέον, όταν υπάρχουν διαδοχικές εγγραφές σε έναν καταχωρητή, τότε το περιεχόμενο του καταχωρητή ενημερώνεται σύμφωνα με την τελευταία εγγραφή.
Έτσι λοιπόν, κατά τη διευθέτηση των εντολών οι καταχωρητές, οι οποίοι περιέχουν τους τελεστέους που είναι σε αναμονή, μετονομάζονται στα ονόματα που βρίσκονται στο σταθμό κράτησης. Αυτή η διαδικασία καλείται ***μετονομασία καταχωρητών.***
Επομένως, αφού μπορούν να υπάρξουν περισσότεροι σταθμοί κράτησης από ότι πραγματικοί καταχωρητές, η τεχνική αυτή έχει την δυνατότητα εξάλειψης κινδύνων τους οποίους δεν μπορεί να αποφύγει ο μεταγλωττιστής.



Σχήμα 2.5.1 – Η βασική δομή μιας μονάδας κινητής υποδιαστολής του DLX επεξεργαστή βασισμένη στον αλγόριθμο του Tomasulo.

Όπως φαίνεται και στο σχήμα 2.5.1, με τη χρήση της τεχνικής Tomasulo:

- Οι εντολές κινητής υποδιαστολής όταν διευθετούνται, στέλνονται από τη μονάδα εντολών στην ουρά εντολών κινητής υποδιαστολής.
- Οι σταθμοί κράτησης περιέχουν τις διευθετημένες εντολές που περιμένουν να εκτελεστούν σε μία λειτουργική μονάδα. Επίσης περιέχουν τους τρέχοντες τελεστέους των παραπάνω εντολών ή την προέλευση των τελεστέων, καθώς και πληροφορίες για έλεγχο και αποφυγή κινδύνων κατά την έναρξη εκτέλεσης της εντολής.
- Οι προσωρινοί καταχωρητές φόρτωσης και αποθήκευσης κρατούν δεδομένα ή διευθύνσεις που αντίστοιχα προέρχονται ή κατευθύνονται στη μνήμη και θα χρησιμεύσουν σε εντολές που βρίσκονται σε αναμονή.
- Το αρχείο καταχωρητών κινητής υποδιαστολής συνδέεται με τις λειτουργικές μονάδες μέσω ενός ζεύγους αρτηριών ενώ με τους προσωρινούς καταχωρητές αποθήκευσης μέσω μιας αρτηρίας.
- Τα αποτελέσματα των λειτουργικών μονάδων κινητής υποδιαστολής ή του προσωρινού καταχωρητή φόρτωσης, τοποθετούνται στην κοινή αρτηρία δεδομένων, η οποία συνδέεται με το αρχείο καταχωρητών κινητής υποδιαστολής καθώς επίσης και με τους σταθμούς κράτησης και τους προσωρινούς καταχωρητές αποθήκευσης.

Κάθε εντολή κατά την εκτέλεσή της ακολουθεί τρία βήματα **:**

1. Διευθέτηση – Εάν πρόκειται για λειτουργία κινητής υποδιαστολής η εντολή διευθετείται, με την προϋπόθεση ότι υπάρχει ελεύθερος σταθμός κράτησης και οι τελεστέοι στέλνονται στον σταθμό κράτησης, αν αυτοί βρίσκονται σε καταχωρητές. Εάν πρόκειται για λειτουργία φόρτωσης ή αποθήκευσης η εντολή διευθετείται, εφόσον υπάρχει διαθέσιμος προσωρινός καταχωρητής. Αν δεν υπάρχει ελεύθερος σταθμός κράτησης ή προσωρινός καταχωρητής, τότε υπάρχει κατασκευαστικός κίνδυνος και η εντολή καθυστερεί μέχρι να απελευθερωθεί κάποιος από τους δύο. Σε αυτό το βήμα πραγματοποιείται και η μετονομασία των καταχωρητών που αναφέρθηκε προηγουμένως.
2. Εκτέλεση - Εάν ένας ή περισσότεροι τελεστέοι δεν είναι διαθέσιμοι, γίνεται έλεγχος της κοινής αρτηρίας δεδομένων μέχρι να υπολογιστούν οι τελεστέοι. Όταν ένας τελεστέος γίνει διαθέσιμος, τότε τοποθετείται στον αντίστοιχο σταθμό κράτησης. Όταν και οι δύο τελεστέοι είναι διαθέσιμοι, εκτελείται η λειτουργία. Αυτό το βήμα ελέγχει και για κινδύνους RAW.
3. Εγγραφή αποτελέσματος - Όταν το αποτέλεσμα είναι διαθέσιμο γράφεται στην κοινή αρτηρία δεδομένων και από εκεί στους καταχωρητές και σε οποιουσδήποτε σταθμούς κράτησης που το περιμένουν.

ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2

Θυμάστε την τεχνική του πίνακα αποτελεσμάτων; Για περισσότερες λεπτομέρειες καλό θα ήταν να ανατρέξετε στην 4η ενότητα: «Δυναμικός προγραμματισμός – Η τεχνική του πίνακα αποτελεσμάτων SCOREBOARDING».

Υπάρχουν τρεις σημαντικές διαφορές μεταξύ των βημάτων της τεχνικής του Tomasulo και αυτών της τεχνικής του πίνακα αποτελεσμάτων. Συγκεκριμένα για την τεχνική Tomasulo :

- Δεν υπάρχει έλεγχος για WAW και WAR κινδύνους. Αυτοί εξαλείφονται όταν, κατά την διάρκεια της διευθέτησης, μετονομάζονται οι καταχωρητές τελεστέων.

- Η κοινή αρτηρία δεδομένων χρησιμοποιείται για να μεταδώσει αποτελέσματα παρά για να εξυπηρετήσει τους καταχωρητές.

- Η φόρτωση και η αποθήκευση από και προς τη μνήμη θεωρούνται σαν βασικές λειτουργικές μονάδες.

Οι προσωρινοί καταχωρητές φόρτωσης και αποθήκευσης, οι σταθμοί κράτησης καθώς και το αρχείο καταχωρητών χρησιμοποιούν δομές δεδομένων χρήσιμες για τον έλεγχο και την αποφυγή κινδύνων. Όλες οι παραπάνω μονάδες, εκτός από τον προσωρινό καταχωρητή φόρτωσης περιέχουν ένα **πεδίο ετικέτα** για κάθε καταχώρηση. Αυτές οι ετικέτες είναι χαρακτηριστικά ονόματα ενός εκτεταμένου συνόλου εικονικών καταχωρητών που χρησιμοποιούνται στην μετονομασία.
Το πεδίο ετικέτα περιγράφει ποιος σταθμός κράτησης περιέχει την εντολή η οποία θα δώσει ένα αποτέλεσμα, που προορίζεται για πηγαίος τελεστέος.

Στον αλγόριθμο του Tomasulo οι ετικέτες αναφέρονται στον προσωρινό καταχωρητή ή στην μονάδα που θα παράγει κάποιο αποτέλεσμα. Τα ονόματα καταχωρητών διαγράφονται όταν μία εντολή διευθετείται σε έναν σταθμό κράτησης.

Κάθε σταθμός κράτησης έχει τα εξής πεδία :

O_p : Η λειτουργία που εκτελείται πάνω στους πηγαίους τελεστέους S1 και S2.

Q_j,Q_k : Οι σταθμοί κράτησης που θα δώσουν τον αντίστοιχο πηγαίο τελεστέο. Το μηδέν δείχνει ότι ο πηγαίος τελεστέος είναι ήδη διαθέσιμος στο πεδίο V_j ή στο V_k , ή ότι δεν χρειάζεται.

V_j,V_k : Η τιμή των πηγαίων τελεστέων. Προσέξτε ότι για κάθε τελεστέο μόνο ένα από τα V ή Q πεδία ισχύει.

Busy : Δείχνει ότι ο συγκεκριμένος σταθμός κράτησης και η λειτουργική μονάδα που τον συνοδεύει είναι απασχολημένοι.

Το αρχείο καταχωρητών και οι προσωρινοί καταχωρητές αποθήκευσης έχουν ένα πεδίο, Q_i :

Qi : Ο αριθμός του σταθμού κράτησης που περιέχει τη λειτουργία της οποίας το αποτέλεσμα πρέπει να αποθηκευτεί σ' αυτό τον καταχωρητή ή στη μνήμη. Εάν ο Qi δεν έχει τιμή, τότε καμία τρέχουσα ενεργή εντολή δεν υπολογίζει το αποτέλεσμα που προορίζεται για αυτόν τον καταχωρητή ή τον προσωρινό καταχωρητή. Για έναν καταχωρητή αυτό σημαίνει ότι η τιμή δίνεται απλά από το περιεχόμενο του καταχωρητή.

Για όλους τους προσωρινούς καταχωρητές φόρτωσης και αποθήκευσης απαιτείται ένα ενεργό Qi πεδίο που να δείχνει πότε ένας προσωρινός καταχωρητής είναι διαθέσιμος. Όταν το Qi δεν είναι ενεργό τότε δεν έχει τιμή. Οι προσωρινοί καταχωρητές αποθήκευσης έχουν επίσης ένα πεδίο V όπου κρατείται η τιμή που πρόκειται να αποθηκευτεί.

Παράδειγμα

Έστω ότι έχουμε την παρακάτω ακολουθία εντολών :

```
LD      F6,34(R2)
LD      F2, 45(R3)
MULTD   F0, F2, F4
SUBD    F8, F6, F2
DIVD    F10, F0, F6
ADDD    F6, F8, F2
```

Απάντηση:

Έχουμε εξετάσει το σχήμα του πίνακα αποτελεσμάτων για ένα στιγμιότυπο αυτής της ακολουθίας εντολών, όταν μόνο η πρώτη εντολή φόρτωσης έχει γράψει το αποτέλεσμά της.

Εντολή	Κατάσταση εντολής		
	Διευθέτηση	**Εκτέλεση**	**Εγγραφή αποτελέσματος**
LD F6,34(R2)	✓	✓	✓
LD F2, 45(R3)	✓	✓	
MULTD F0, F2, F4	✓		
SUBD F8, F6, F2	✓		
DIVD F10, F0, F6	✓		
ADDD F6, F8, F2	✓		

Ο πίνακας κατάστασης εντολών δίνεται για καλύτερη κατανόηση του αλγορίθμου, παρότι δεν είναι μέρος του υλικού. Όλες οι εντολές έχουν διευθετηθεί, αλλά μόνο η πρώτη εντολή φόρτωσης έχει ολοκληρωθεί και έχει γράψει το αποτέλεσμα της στην

κοινή αρτηρία δεδομένων. Η κατάσταση κάθε λειτουργίας που έχει διευθετηθεί κρατείται σε ένα σταθμό κράτησης.

Σταθμοί κράτησης						
Name	**Busy**	**Op**	**Vj**	**Vk**	**Qj**	**Qk**
Add1	Yes	SUB	Mem[34+Regs[R2]]			Load2
Add2	Yes	ADD			Add1	Load2
Add3	No					
Mult1	Yes	MULT		Regs[F4]	Load2	
Mult2	Yes	DIV		Mem[34+Regs[R2]]	Mult1	

Οι αριθμοί που επισυνάπτονται στις εντολές Add, Mult και Load αντιπροσωπεύουν τις ετικέτες για τους παραπάνω σταθμούς κράτησης. Έτσι, για παράδειγμα στην εντολή Add1, το ένα, είναι η ετικέτα για το αποτέλεσμα από την πρώτη μονάδα πρόσθεσης. Τα πεδία V_j και V_k δείχνουν την τιμή ενός τελεστέου. Οι προσωρινοί καταχωρητές φόρτωσης και αποθήκευσης δεν φαίνονται στο σχήμα. Ο μοναδικός ενεργός προσωρινός καταχωρητής φόρτωσης είναι ο No2 και ενεργεί για λογαριασμό της δεύτερης εντολής στην ακολουθία εντολών, φορτώνοντας από τη μνήμη τη διεύθυνση R3+45. Ένας τελεστέος προσδιορίζεται είτε από ένα πεδίο Q ή από ένα πεδίο V κάθε χρονική στιγμή.

Κατάσταση καταχωρητή									
Field	**FO**	**F2**	**F4**	**F6**	**F8**	**F10**	**F12**	**...**	**F30**
Qi	Mult1	Load2		Add2	Add1	Mult2			

Για κάθε προσωρινό καταχωρητή φόρτωσης ή αποθήκευσης υπάρχει ένα ενεργό πεδίο Q_i που δείχνει πότε ένας προσωρινός καταχωρητής είναι διαθέσιμος. Στην περίπτωση που το Q_i δεν είναι ενεργό τότε δεν έχει τιμή.

Παρατηρούμε δύο σημαντικές διαφορές μεταξύ των πινάκων της τεχνικής Tomasulo και της τεχνικής του πίνακα αποτελεσμάτων, για το συγκεκριμένο παράδειγμα. Συγκεκριμένα για την τεχνική Tomasulo**:**

- Η τιμή ενός τελεστέου αποθηκεύεται στον σταθμό κράτησης σε ένα από τα V πεδία μόλις αυτό γίνει διαθέσιμο. Η τιμή δε διαβάζεται από το αρχείο καταχωρητών ούτε από έναν σταθμό κράτησης, αφού η εντολή έχει διευθετηθεί.

- Η εντολή ADDD, η οποία στον πίνακα αποτελεσμάτων παρουσίασε εμπλοκή λόγω ενός WAR κινδύνου κατά το στάδιο WB, έχει διευθετηθεί και μπορεί να ολοκληρωθεί πριν εισαχθεί η εντολή DIVD.

Παραδείγματα της τεχνικής του Tomasulo

Παράδειγμα

Υποθέστε πως για τις λειτουργικές μονάδες κινητής υποδιαστολής έχουμε τις επόμενες καθυστερήσεις μετρημένες σε κύκλους ρολογιού. Η πρόσθεση καθυστερεί 2 κύκλους ρολογιού, ο πολλαπλασιασμός 10 κύκλους ρολογιού και η διαίρεση 40. Για ένα στιγμιότυπο του παρακάτω κώδικα να δείξετε πώς γίνονται οι πίνακες καταστάσεων όταν η εντολή MULTD είναι έτοιμη να γράψει το αποτέλεσμά της.

```
LD      F6,34(R2)
LD      F2, 45(R3)
MULTD   F0, F2, F4
SUBD    F8, F6, F2
DIVD    F10, F0, F6
ADDD    F6, F8, F2
```

Απάντηση:

Τμήμα κατάστασης εντολής

Εντολή	Κατάσταση εντολής Διευθέτηση	Εκτέλεση	Εγγραφή αποτελέσματος
LD F6,34(R2)	✓	✓	✓
LD F2, 45(R3)	✓	✓	✓
MULTD F0, F2, F4	✓	✓	
SUBD F8, F6, F2	✓	✓	✓
DIVD F10, F0, F6	✓		
ADDD F6, F8, F2	✓	✓	✓

Αντίθετα με το αντίστοιχο παράδειγμα του πίνακα αποτελεσμάτων, η εντολή ADDD έχει ολοκληρωθεί, αφού οι τελεστέοι της εντολής DIVD έχουν αντιγραφεί, ξεπερνώντας έτσι τον WAR κίνδυνο. Παρατηρήστε ότι ακόμα και αν καθυστερούσε η φόρτωση του F6, η πρόσθεση στον F6 θα μπορούσε να εκτελεστεί χωρίς να ενεργοποιηθεί WAW κίνδυνος. Οι εντολές MULTD και DIVD δεν έχουν γράψει το αποτέλεσμά τους. Αυτή η περίπτωση διαφέρει από την αντίστοιχη του πίνακα αποτελεσμάτων, μια και η εξάλειψη των WAR κινδύνων επέτρεψε στην εντολή ADDD να τελειώσει ακριβώς μετά την εντολή SUBD με την οποία είχε εξάρτηση.

Τμήμα σταθμών κράτησης

Σταθμοί κράτησης						
Name	**Busy**	**Op**	**Vj**	**Vk**	**Qj**	**Qk**
Add1	No					
Add2	No					
Add3	No					
Mult1	Yes	MULT	Mem[45+Regs[R3]]	Regs[F4]		
Mult2	Yes	DIV		Mem[34+Regs[R2]]	Mult1	

Τμήμα κατάστασης καταχωρητή

Κατάσταση καταχωρητή									
Field	**FO**	**F2**	**F4**	**F6**	**F8**	**F10**	**F12**	**...**	**F30**
Qi	Mult1					Mult2			

Παράδειγμα

Στην παρακάτω ακολουθία εντολών του αρχικού παραδείγματος έχουμε διευθετήσει και την εντολή DIVD και την εντολή ADDD, παρόλο που υπάρχει ένας WAR κίνδυνος λόγω της παρουσίας του καταχωρητή F6. Παρακάτω φαίνεται πως εξαλείφεται ο κίνδυνος.

LD F6,34(R2)
LD F2, 45(R3)
MULTD F0, F2, F4
SUBD F8, F6, F2
DIVD F10, F0, F6
ADDD F6, F8, F2

Απάντηση:

Τμήμα κατάστασης εντολής

Κατάσταση εντολής			
Εντολή	**Διευθέτηση**	**Εκτέλεση**	**Εγγραφή αποτελέσματος**
LD F6,34(R2)	✓	✓	✓
LD F2, 45(R3)	✓	✓	
MULTD F0, F2, F4	✓		
SUBD F8, F6, F2	✓		
DIVD F10, F0, F6	✓		
ADDD F6, F8, F2	✓		

Τμήμα σταθμών κράτησης

Σταθμοί κράτησης						
Name	**Busy**	**Op**	**Vj**	**Vk**	**Qj**	**Qk**
Add1	Yes	SUB	Mem[34+Regs[R2]]			Load2
Add2	Yes	ADD			Add1	Load2
Add3	No					
Mult1	Yes	MULT		Regs[F4]	Load2	
Mult2	Yes	DIV		Mem[34+Regs[R 2]]	Mult1	

Αν η εντολή που δίνει την τιμή για την εντολή DIVD έχει ολοκληρωθεί, τότε ο V_k θα αποθηκεύσει το αποτέλεσμα, επιτρέποντας στην εντολή DIVD να εκτελεστεί ανεξάρτητα από την εντολή ADDD. Αυτή είναι και η περίπτωση που φαίνεται στο αρχικό παράδειγμα.
Από την άλλη, αν η εντολή LD δεν είχε ολοκληρωθεί, η Q_k θα έδειχνε στο Load1 και η εντολή DIVD θα ήταν ανεξάρτητη από την εντολή ADDD. Έτσι, η εντολή ADDD μπορεί να διευθετηθεί και να ξεκινήσει την εκτέλεσή της. Οποιεσδήποτε χρήσεις του αποτελέσματος της εντολής MULTD θα έδειχναν στο σταθμό κράτησης επιτρέποντας στην εντολή ADDD να ολοκληρώσει την εκτέλεσή της και να αποθηκεύσει την τιμή της στους καταχωρητές χωρίς να επηρεάζει την εντολή DIVD.

Τμήμα κατάστασης καταχωρητή

Κατάσταση καταχωρητή									
Field	**FO**	**F2**	**F4**	**F6**	**F8**	**F10**	**F12**	**...**	**F30**
Qi	Mult1	Load2		Add2	Add1	Mult2			

Παράδειγμα

Το παρακάτω σχήμα δίνει τα βήματα που πρέπει να περάσει κάθε εντολή.

Κατάσταση εντολής	**Wait until**	**Action or bookkeeping**

Διευθέτηση	Station or buffer empty	if (Register['S1'].$Q_i \neq 0$)
		{RS[r].$Q_j \leftarrow$Register[S1].Q_i}
		else { RS[r].$V_j \leftarrow$S1; RS[r].$Q_j \leftarrow 0$};
		if (Register['S2'].$Q_i \neq 0$)
		{RS[r].$Q_k \leftarrow$Register[S2].Q_i};
		else { RS[r].$V_k \leftarrow$S2 ; RS[r].$Q_k \leftarrow 0$};
		RS[r].Busy ← yes;
		Register ['D'].Q_i=r ;

Εκτέλεση	(RS[r].Q_j=0) and (RS[r].Q_k=0)	None- operands are in V_j and V_k
Εγγραφή αποτελέσματος	Execution completed at r and CDB availiable	∀x (if (Register[x].Q_i = r) { Fx←result; Register[x].$Q_i \leftarrow 0$}); ∀x (if (RS[x].Q_j = r) { RS[x].V_j←result; RS[x].$Q_j \leftarrow 0$}); ∀x (if (RS[x].Q_k = r) { RS[x].V_k←result; RS[x].$Q_k \leftarrow 0$}); ∀x (if (Store[x].Q_i = r) { Store[x].V←result; Store[x].$Q_i \leftarrow 0$}); RS[r].Busy←No

Για την διευθετούμενη εντολή, D είναι ο προορισμός, S1 και S2 οι πηγαίοι αριθμοί καταχωρητών και r ο σταθμός κράτησης ή ο προσωρινός καταχωρητής με τον οποίο ο D συσχετίζεται. RS είναι η δομή δεδομένων του σταθμού κράτησης. Η τιμή που επιστρέφεται από έναν σταθμό κράτησης ή από τη μονάδα φόρτωσης ονομάζεται result. Register είναι η δομή δεδομένων για τον καταχωρητή, ενώ Store είναι η δομή δεδομένων για τον προσωρινό καταχωρητή αποθήκευσης. Το Ri σημαίνει το όνομα του καταχωρητή Ri και όχι τα περιεχόμενά του. Όταν διευθετείται μία εντολή, ο καταχωρητής προορισμού έχει το Q_i πεδίο του ρυθμισμένο στον αριθμό του προσωρινού καταχωρητή ή του σταθμού κράτησης στον οποίο διευθετείται η εντολή. Αν οι τελεστέοι είναι διαθέσιμοι στους καταχωρητές, αυτοί αποθηκεύονται στα V πεδία. Διαφορετικά, τα Q πεδία ρυθμίζονται για να δηλώσουν το σταθμό κράτησης που θα δώσει τις τιμές που είναι αναγκαίες ως πηγαίοι τελεστέοι. Η εντολή παραμένει στο σταθμό κράτησης μέχρι και οι δύο τελεστέοι της να είναι διαθέσιμοι, γεγονός που υποδηλώνεται από το μηδέν στα Q πεδία. Τα Q πεδία παίρνουν την τιμή μηδέν είτε όταν η συγκεκριμένη εντολή διευθετείται, είτε όταν μία εντολή από την οποία εξαρτάται ολοκληρώνεται και γράφει το αποτέλεσμά της. Όταν μία εντολή έχει τελειώσει την εκτέλεσή της και η κοινή αρτηρία δεδομένων είναι διαθέσιμη, μπορεί να γράψει το αποτέλεσμά της. Όλοι οι προσωρινοί καταχωρητές, οι καταχωρητές και

σταθμοί κράτησης που συμμετέχουν στη διαδικασία ολοκλήρωσης, ενημερώνουν τις τιμές τους από την κοινή αρτηρία δεδομένων και μαρκάρουν τα πεδία Q για να δηλώσουν ότι οι τιμές έχουν ληφθεί. Έτσι, η κοινή αρτηρία δεδομένων μπορεί να μεταδώσει το αποτέλεσμα σε πολλούς προορισμούς σε ένα μόνο κύκλο ρολογιού και αν οι εν αναμονή εντολές έχουν τους τελεστέους τους μπορούν όλες να αρχίσουν την εκτέλεση στον επόμενο κύκλο ρολογιού. Για απλότητα, υποθέτουμε ότι όλες οι ενέργειες αρχικοποίησης γίνονται σε ένα κύκλο.

Παράδειγμα

Για να καταλάβουμε τη δυναμική της εξάλειψης WAR και WAW κινδύνων μέσω της δυναμικής μετονομασίας καταχωρητών, πρέπει να δούμε ένα βρόχο. Θεωρήστε την παρακάτω ακολουθία εντολών για τον πολλαπλασιασμό των στοιχείων ενός πίνακα με μια βαθμωτή ποσότητα του F2.

```
Loop: LD      F0, 0(R1)
      MULTD   F4, F0, F2
      SD      0(R1), F4
      SUB     R1, R1, #8
      BNEZ    R1, loop     ; διακλάδωση εάν R1≠0
```

Απάντηση:

Αν προβλέψουμε ότι οι διακλαδώσεις ακολουθούνται, η χρησιμοποίηση σταθμών κράτησης θα επιτρέψει σε πολλαπλές εκτελέσεις του βρόχου να προχωρήσουν αμέσως. Αυτό γίνεται χωρίς να αναπτύξουμε το βρόχο, μια και αναπτύσσεται δυναμικά από το υλικό. Ο αλγόριθμος Tomasulo υποστηρίζει την επικαλυπτόμενη εκτέλεση πολλαπλών επαναλήψεων του βρόχου με τη χρήση ενός μικρού αριθμού καταχωρητών.
Ας υποθέσουμε ότι έχουμε διευθετήσει όλες τις εντολές σε δύο επιτυχημένες επαναλήψεις του βρόχου, αλλά καμία από τις κινητής υποδιαστολής φορτώσεις/αποθηκεύσεις ή λειτουργίες δεν έχει ολοκληρωθεί. Η κατάσταση των εντολών, οι σταθμοί κράτησης, η κατάσταση των καταχωρητών και των προσωρινών καταχωρητών φόρτωσης και αποθήκευσης φαίνονται στους παρακάτω πίνακες.

Εντολή	Κατάσταση Εντολής Από επανάληψη	Διευθέτηση	Εκτέλεση	Εγγραφή αποτελέσματος
LD F0, 0(R1)	1	✓	✓	
MULTD F4, F0, F2	1	✓		
SD 0(R1), F4	1	✓		
LD F0, 0(R1)	2	✓	✓	
MULTD F4, F0, F2	2	✓		
SD 0(R1), F4	2	✓		

Σταθμοί Κράτησης						
Name	**Busy**	**Op**	**Vj**	**Vk**	**Qj**	**Qk**
Add1	No					
Add2	No					
Add3	No					
Mult1	Yes	MULT		(F2)	Load1	
Mult2	Yes	MULT		(F2)	Load2	

Κατάσταση Καταχωρητή									
Field	**FO**	**F2**	**F4**	**F6**	**F8**	**F10**	**F12**	**...**	**F30**
Qi	Load2		Mult2						
Busy	yes	no	Yes	No	no	No			

Προσωρινοί καταχωρητές Αποθήκευσης			
Field	Store1	Store2	Store3
Qi	Mult1	Mult2	
Busy	Yes	Yes	No
Address	(R1)	(R1)-8	

Προσωρινοί καταχωρητές Φόρτωσης			
Field	Load1	Load2	Load3
Address	(R1)	(R1)-8	
Busy	Yes	Yes	No

Αφού το σύστημα φτάσει σ' αυτή την κατάσταση δύο επαναλήψεις του βρόχου θα μπορούσαν να κρατηθούν σε ένα CPI πολύ κοντά στο 1, δεδομένου ότι οι πολλαπλασιασμοί θα μπορούσαν να ολοκληρωθούν σε τέσσερις κύκλους ρολογιού. Η εντολή φόρτωσης από την τελευταία επανάληψη του βρόχου μπορούσε εύκολα να ολοκληρωθεί πριν από την αποθήκευση της πρώτης επανάληψης, παρόλο που η κανονική σειριακή εξέλιξη είναι διαφορετική. Η φόρτωση και η αποθήκευση μπορούν να γίνουν με ασφάλεια σε διαφορετική σειρά, δεδομένου ότι έχουν πρόσβαση σε διαφορετικές διευθύνσεις. Αυτό ελέγχεται εξετάζοντας τις διευθύνσεις στον προσωρινό καταχωρητή αποθήκευσης όταν διευθετείται μία φόρτωση. Αν η διεύθυνση φόρτωσης συμπίπτει με τη διεύθυνση του προσωρινού καταχωρητή αποθήκευσης, πρέπει να σταματήσουμε και να περιμένουμε έως ότου ο προσωρινός καταχωρητής αποθήκευσης να πάρει τιμή. Μπορούμε τότε να έχουμε πρόσβαση σε αυτή ή να πάρουμε τιμή από τη μνήμη.

Πλεονεκτήματα και μειονεκτήματα της τεχνικής Tomasulo- Σύγκριση των δύο τεχνικών.

Η τεχνική Tomasulo έχει δύο βασικά **πλεονεκτήματα:**

- Τα λογικά κυκλώματα ανίχνευσης κινδύνων που είναι κατανεμημένα. Αυτό υλοποιείται με τους κατανεμημένους σταθμούς κράτησης και την χρήση της κοινής αρτηρίας δεδομένων. Αν πολλαπλές εντολές περιμένουν ένα αποτέλεσμα και κάθε εντολή έχει ήδη τον άλλο τελεστέο, τότε οι εντολές μπορούν να αποδεσμευτούν ταυτόχρονα από τη μετάδοση στην κοινή αρτηρία δεδομένων. Στον πίνακα αποτελεσμάτων οι εν αναμονή εντολές πρέπει να διαβάσουν τα αποτελέσματά τους από τους καταχωρητές, όταν οι αρτηρίες καταχωρητών είναι ελεύθερες.

- Την εξάλειψη των καθυστερήσεων λόγω των WAR και WAW κινδύνων. Οι κίνδυνοι εξαλείφονται με τη χρήση της μετονομασίας των καταχωρητών που χρησιμοποιούν τους σταθμούς κράτησης και με τη διαδικασία αποθήκευσης των τελεστέων στους σταθμούς κράτησης μόλις αυτοί γίνουν διαθέσιμοι.

Η τεχνική Tomasulo παρουσιάζει ορισμένα **μειονεκτήματα:**

- Απαιτεί μία μεγάλη ποσότητα υλικού. Συγκεκριμένα, υπάρχουν πολλές συσχετιζόμενες αποθηκεύσεις που πρέπει να τρέξουν σε μεγάλη ταχύτητα, καθώς και περίπλοκη λογική ελέγχου.

- Περιορίζει το κέρδος της απόδοσης λόγω της μονής αρτηρίας δεδομένων. Ενώ μπορούν να προστεθούν και άλλες κοινές αρτηρίες δεδομένων, κάθε μία από αυτές πρέπει να αλληλεπιδράσει με όλο το υλικό της σωλήνωσης, συμπεριλαμβανομένων και των σταθμών κράτησης.

Η τεχνική Tomasulo επιλέγεται στην περίπτωση που ο σχεδιαστής είναι αναγκασμένος να σωληνώσει μία αρχιτεκτονική για την οποία είναι δύσκολο να προγραμματίσει κώδικα ή εμφανίζει έλλειψη καταχωρητών.

Σύγκριση των τεχνικών του πίνακα αποτελεσμάτων και του Tomasulo:

Πίνακας Αποτελεσμάτων	*Tomasulo*
1.Η αντιμετώπιση των κινδύνων δεδομένων WAR γίνεται κατά τη διευθέτηση, ενώ των κινδύνων WAW γίνεται πριν την εγγραφή των αποτελεσμάτων.	**1.**Οι WAR και WAW κίνδυνοι εξαλείφονται με τη βοήθεια της μετονομασίας καταχωρητών μέσω των σταθμών κράτησης.
2.Η ανίχνευση των κινδύνων και ο έλεγχος της εκτέλεσης των εντολών γίνονται στον πίνακα.	**2.**Η ανίχνευση κινδύνων και ο έλεγχος της εκτέλεσης των εντολών είναι κατανεμημένα. Οι σταθμοί κράτησης σε κάθε λειτουργική μονάδα ελέγχουν πότε μία εντολή μπορεί να ξεκινήσει την εκτέλεσή της σε αυτή τη μονάδα.
3.Τα αποτελέσματα γράφονται στους καταχωρητές όπου περιμένουν μέχρι οι λειτουργικές μονάδες να τα χρειαστούν.	**3.**Τα αποτελέσματα μεταδίδονται απευθείας στις λειτουργικές μονάδες από τους σταθμούς κράτησης με τη βοήθεια της κοινής αρτηρίας δεδομένων, μόλις είναι διαθέσιμα, χωρίς να χρειάζεται η επικοινωνία με τους καταχωρητές.

Ανακεφαλαιώνοντας λοιπόν…

Η τεχνική Tomasulo αποτελεί μία άλλη προσέγγιση του δυναμικού προγραμματισμού. Αυτή η τεχνική, όπως και η τεχνική του πίνακα αποτελεσμάτων, επιτρέπει την εκτέλεση εντολών παρουσία κινδύνων.

Η μέθοδος της μετονομασίας καταχωρητών εξαλείφει δυναμικά τις εξαρτήσεις ονομάτων με σκοπό να αποφευχθούν οι WAR και WAW κίνδυνοι. Αυτό επιτυγχάνεται αντικαθιστώντας τα ονόματα των καταχωρητών με άλλα, από ένα μεγαλύτερο σύνολο εικονικών καταχωρητών.

Οι σταθμοί κράτησης ανακαλούν και καταχωρούν προσωρινά έναν τελεστέο μιας εντολής, μόλις βέβαια γίνει διαθέσιμος, εξαλείφοντας έτσι την ανάγκη να διαβάσουμε τον τελεστέο από έναν καταχωρητή.

Κάθε εντολή κατά την εκτέλεση της ακολουθεί τρία βήματα, που αντικαθιστούν τα ID, EX και WB στάδια στη βασική DLX σωλήνωση: Διευθέτηση, Εκτέλεση και Εγγραφή αποτελέσματος.

Σε μία μονάδα κινητής υποδιαστολής του DLX επεξεργαστή βασισμένη στον αλγόριθμο του Tomasulo, διακρίνουμε τους σταθμούς κράτησης, τους προσωρινούς καταχωρητές φόρτωσης και αποθήκευσης, τους καταχωρητές κινητής υποδιαστολής και την κοινή αρτηρία δεδομένων.