Αρχιτεκτονική Υπολογιστών - Εργασία

Δεληγιαννάκης Χαράλαμπος
4383

Ιανουάριος 2024

Εισαγωγή

Κώδικας

Το ΑΕΜ ειναι 4383, οπότε:

- Η αρχική τιμή του καταχωρητή **$9** είναι ίση 53.
- Το όρισμα **imm** είναι ίσο με 8.

Επομένως, ο τελικός κώδικας είναι ο εξής:

```
Loop:  lw $2, 400($1)       # Load word from memory into register $2
lw $3, 800($1)       #Load word from memory into register $3
xor $5, $5, $2       # XOR operation between $5 and $2, result stored in $5
sw $5, 600($4)       # Store word from $5 into memory
addi $1, $1, 4       # Add immediate value 4 to $1
add $6, $1, $2       # Add values in registers $1 and $2, result stored in $6
addi $9, $9, -8       # Add immediate value -8 to $9
bne $9, $0, Loop       # Branch not equal, jump to Loop if $9 is not zero
```

1 Πρώτο ζητούμενο

Διάγραμμα χρονισμού

Τα stalls (εντολές nop που βάζει αυτόματα compiler) συμβολίζονται με @

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
lw $2, 400($1)	IF	ID	RR	EX	M	WB																			
lw $3, 800($1)		IF	ID	RR	EX	M	WB																		
xor $5, $5, $2			IF	ID	RR	@	EX	M	WB																
sw $5, 600($4)				IF	ID	@	RR	@	@	EX	M	WB													
addi $1, $1, 4					IF	@	ID	@	@	RR	EX	M	WB												
add $6, $1, $2						@	IF	@	@	ID	RR	@	@	EX	M	WB									
addi $9, $9, -8						@		@	@	IF	ID	@	@	RR	EX	M	WB								
bne $9, $0, Loop						@		@	@		IF	@	@	ID	RR	@	@	EX	M	WB					
lw $2, 400($1)						@		@	@			@	@			@	@	@	@	IF	ID	RR	EX	MB	WB

Figure 1: Διάγραμμα χρονισμού

Πιθανοί κίνδυνοι και αντιμετώπιση

Οι κίνδυνοι, που με τη χρήση των stalls (@) στο παραπάνω διάγραμμα αποτράπηκαν, είναι κίνδυνοι δεδομένων - data hazards που υπάρχουν σε συγκεκριμένες εντολές. Για να λυθούν, πρέπει η συγκεκριμένη εντολή να περιμένει κάποια προηγούμενη εντολή να ολοκληρώσει την ανάγνωση/εγγραφή δεδομένων της). Αυτό γίνεται με την καθυστέρηση (stall) του αγωγού

(pipeline). Το κάθε stall σημαίνει και μία εντολή `nop`, που προστίθεται αυτόματα από τον compiler.

Συγκεκριμένα:

```
Loop:  lw $2, 400($1)
lw $3, 800($1)
xor $5, $5, $2      # Data hazard από την 1η εντολή
sw $5, 600($4)      # Data hazard από την 3η εντολή
addi $1, $1, 4
add $6, $1, $2      # Data hazard από την 5η εντολή
addi $9, $9, -8
bne $9, $0, Loop      # Data hazard από την 7η εντολή
```

Δηλαδή:

- `Loop:  lw $2, 400($1)`
  `[εντολή 2]`
  `xor $5, $5, $2`
  Stall στον κύκλο 6:
  Ο καταχωρητής $2 για την εντολή `xor` (η ανάγνωση του οποίου γίνεται στο στάδιο RR) γίνεται διαθέσιμος στο τέλος του κύκλου 6 (στάδιο WB) από την εντολή `lw`.

- `xor $5, $5, $2`
  `sw $5, 600($4)`
  Stalls στους κύκλους 8,9:
  Ο καταχωρητής $5 για την εντολή `sw` (η ανάγνωση του οποίου γίνεται στο στάδιο RR) γίνεται διαθέσιμος στο τέλος του κύκλου 9 (στάδιο WB) από την εντολή `xor`.

- `addi $1, $1, 4`
  `add $6, $1, $2`
  Stalls στους κύκλους 12,13:
  Ο καταχωρητής $1 για την εντολή `add` (η ανάγνωση του οποίου γίνεται στο στάδιο RR) γίνεται διαθέσιμος στο τέλος του κύκλου 13 (στάδιο WB) από την εντολή `addi`.

- `addi $9, $9, -8`
  `bne $9, $0, Loop`
  Stalls στους κύκλους 16,17:
  Ο καταχωρητής $9 για την εντολή `bne` (η ανάγνωση του οποίου γίνεται στο στάδιο

RR) γίνεται διαθέσιμος στο τέλος του κύκλου 17 (στάδιο WB) από την εντολή `addi`.

Τέλος, υπήρχε και ένας κίνδυνος ελέγχου (control hazard), κάθε φορά που εκτελούνταν η πρώτη εντολή μιας επανάληψης (εκτός από την πρώτη επανάληψη). Γενικά, τα branches γίνονται resolve στο στάδιο M, οπότε η πρώτη εντολή μιας νέας επανάληψης, πρέπει να περιμένει το στάδιο M της προηγούμενης για να ξεκινήσει, γιατί σε αυτήν λαμβάνεται η απόφαση για το αν συνεχίσει η επανάληψη ή όχι. Επομένως, βάζοντας stalls, περιμένει μέχρι να καθοριστεί το αποτέλεσμα της διακλάδωσης πριν προσκομίσει την επόμενη εντολή.

Συγκεκριμένα:

- `bne $9, $0, Loop`
  `lw $2, 400($1)`
  Stalls στους κύκλους 18,19:
  Στο τέλος του κύκλου 19 (τέλος σταδίου M), θα έχει γίνει resolve το branch, και θα είναι γνωστό αν η επανάληψη συνεχίσει ή όχι.

2 Δεύτερο ζητούμενο

Απαιτούμενοι κύκλοι για την εκτέλεση του κώδικα

Η πορεία τιμών του καταχωρητή $9 (μειώνεται κατά οχτώ σύμφωνα με την εντολή `bne $9, $0`) είναι η εξής: 53, 45, ..., 13, 5, -3, -11, ...
Επίσης η εντολή `bne $9, $0` καθορίζει πότε θα τερματιστεί η επανάληψη. Συγκεκριμένα, όταν η τιμή του καταχωρητή $9 γίνει ίση με την τιμή του καταχωρητή $0, δηλαδή ίση με το μηδέν, τότε η επανάληψη θα τερματίσει και θα γίνει jump στην επόμενη εντολή. Αλλιώς, η επανάληψη συνεχίζει και γίνεται jump στην πρώτη εντολή της Loop (`lw $2, 400($1)`).
Έτσι, το συμπέρασμα που βγάζουμε είναι ότι εφόσον η τιμή του καταχωρητή $9 δεν γίνεται ποτέ 0, **τότε η επανάληψη δεν τερματίζει ποτέ (ατέρμων βρόγχος).**

Ειδική περίπτωση τερματισμού

Τα AEM (xyzw) για τα οποία ο αριθμός $50 + w$ είναι πολλαπλάσιο του αριθμού z θα είχαν καταχωρητή $9 που θα έφτανε στο 0, οπότε θα τερμάτιζε η επανάληψη τους.

Απόδειξη:
Αν $50 + w$ είναι πολλαπλάσιο του αριθμού z, τότε: $50 + w = \lambda \cdot z$, με $\lambda \in \mathbb{N}$.
Θα αποδείξουμε ότι υπάρχει $\kappa \in \mathbb{N}$ τέτοιο ώστε $50 + w - \kappa \cdot z = 0$, δηλαδή μετά από έναν αριθμό επαναλήψεων, η τιμή του καταχωρητή θα γίνει 0 και, επομένως, θα τερματίσει:

$$50 + w - \kappa \cdot z = 0 \Rightarrow \lambda \cdot z - \kappa \cdot z = 0 \Rightarrow z \cdot (\lambda - \kappa) = 0.$$

Δεδομένου ότι $z \neq 0$ (σύμφωνα με την εκφώνηση), πρέπει να ισχύει $\lambda - \kappa = 0 \Rightarrow \lambda = \kappa$. Επομένως, η τιμή του καταχωρητή \$9 θα γίνει 0 μετά από λ επαναλήψεις και, συνεπώς, η επανάληψη θα τερματίσει.

Απαιτούμενοι κύκλοι αν η επανάληψη τερμάτιζε

Έστω ότι η επανάληψη θα τερμάτιζε μετά από λ φορές, δηλαδή θεωρούμε λ τον αριθμό επαναλήψεων. Τότε, σύμφωνα με το διάγραμμα χρονισμού, οι απαιτούμενοι κύκλοι για την εκτέλεση του κώδικα είναι:

Για τις πρώτες $\lambda - 1$ επαναλήψεις έχουμε $(\lambda - 1) \cdot 19cc$
Για την τελευταία επανάληψη (την λ_η) έχουμε 20cc
Άρα, οι απαιτούμενοι κύκλοι είναι $(\lambda - 1) \cdot 19 + 20$ κύκλοι.

Για παράδειγμα, αν γινόταν στρογγυλοποίηση, ο κώδικας θα τερμάτιζε μετά από $53/8 = 6$ επαναλήψεις.
Άρα για $\lambda = 6$ οι απαιτούμενοι κύκλοι θα ήταν $5 * 19 + 20 = 115$cc

Λύση προβλήματος

Η λύση του προβλήματος του ατέρμονος βρόγχος είναι απλή. Δεδομένου ότι για κάποια ΑΕΜ, η τιμή του καταχωρητή \$9 δεν γίνεται ποτέ ίση με το 0, η εντολή `bne $9, $0, Loop` μπορεί να αντικατασταθεί από την εντολή `bgt $9, $0, Loop` η αλλιώς `bgtz $9, Loop`. Η πρώτη εντολή ελέγχει αν η τιμή του πρώτου καταχωρητή (\$9) είναι μεγαλύτερη από την τιμή του δεύτερου καταχωρητή (\$0) και αν ισχυέι, συνεχίζει την Loop. Η δεύτερη εντολή έχει την ίδια λογική, αλλά ελέγχει απευθείας αν η τιμή του πρώτου ορίσματος (\$9) είναι μεγαλύτερη του 0.