Αρχιτεκτονική Υπολογιστών

Γιώργος Δημητρίου

Πανεπιστήμιο Θεσσαλίας - Τμήμα Πληροφορικής

Ενότητα 7η:

Στατική Δρομολόγηση Εντολών (Επεξεργαστές VLIW)

Εκμετάλλευση ILP

- Περιορισμοί στη δυναμική δρομολόγηση εντολών:
  - Μέγεθος παραθύρου εντολών
  - Αριθμός φυσικών καταχωρητών
  - Αποτυχία στην πρόβλεψη διακλαδώσεων
  - Εξαρτήσεις στη μνήμη
- Στη στατική δρομολόγηση εντολών ο μεταγλωττιστής προσφέρει:
  - Απεριόριστο παράθυρο εντολών
  - Μετασχηματισμούς που μειώνουν την επίδραση των υπόλοιπων περιορισμών

Αρχιτεκτονικές VLIW & Στατική Δρομολόγηση Εντολών

- Κωδικοποίηση πολλών επιμέρους εντολών σε μία Πολύ Μεγάλη Λέξη Εντολής (Very Long Instruction Word)
  - Επεξεργαστές πολλαπλών μονάδων εκτέλεσης
- Παράλληλες εντολές
  - Στατική δρομολόγηση εντολών από το μεταγλωττιστή
  - Έλλειψη ελέγχου εξαρτήσεων μεταξύ εντολών της ίδιας λέξης

Οργάνωση Αρχιτεκτονικών VLIW

- Πολλές μονάδες εκτέλεσης
  - 5 ή περισσότερες παράλληλες εκτελέσεις, για μικρότερο εύρος προτιμάται η υπερβαθμωτή αρχιτεκτονική με δυναμική δρομολόγηση
- Οι επιμέρους εντολές εκτελούνται όπως έρχονται από τη μνήμη εντολών
  - Συνήθως οι επιμέρους εντολές εκτελούνται στις ίδιες κάθε φορά μονάδες εκτέλεσης

Εντολές VLIW

- Παράδειγμα οργάνωσης επιμέρους εντολών:
  - οι δύο πρώτες εντολές είναι εντολές προσπέλασης μνήμης
  - οι δύο επόμενες εντολές είναι κινητής υποδιαστολής
  - η τελευταία εντολή είναι σταθερής υποδιαστολής
- Αν δεν μπορούμε να συμπληρώσουμε τις επιμέρους εντολές, αυτές μένουν κενές!

Παραγωγή Κώδικα Υψηλού ILP

- Συνήθως κώδικας υψηλού ILP προκύπτει από κατάλληλη δρομολόγηση βρόχων
  - Ανίχνευση παράλληλων βρόχων
  - Βελτιστοποιητικοί μετασχηματισμοί βρόχων που αρχικά δεν είναι παράλληλοι
- Δρομολόγηση εντολών βρόχου μετά από
  - Ξεδίπλωμα ενός αριθμού επαναλήψεων
  - Συμβολικό ξεδίπλωμα
- Δρομολόγηση ενός δρόμου

Δρομολόγηση Σώματος Βρόχου

- Έστω ο βρόχος:

```
for (i=1000; i>0; i=i-1)
      x[i] = x[i] + s;
```

- με αντίστοιχο κώδικα MIPS:

```
Loop: ldc1   F0,0(R1)    ; F0 = array element
      add.d  F4,F0,F2    ; add scalar in F2
      sdc1   F4,0(R1)    ; store result
      daddiu R1,R1,-8    ; decrement pointer
      bne    R1,R2,Loop  ; branch if R1!=R2
```

Παράδειγμα

- Έστω:
  - FP → FP = 3cc
  - FP → Store = 2cc
  - Load → FP = 1cc
  - Load → Store = 0cc
  - Διακλαδώσεις στη φάση ID με καθυστέρηση
- Χωρίς δρομολόγηση:

```
ldc1   F0,0(R1)    ; t=1
add.d  F4,F0,F2    ; t=3
sdc1   F4,0(R1)    ; t=6
daddiu R1,R1,-8    ; t=7
bne    R1,R2,Loop  ; t=9
ldc1   F0,0(R1)    ; t=11
```

Με δρομολόγηση:

```
ldc1   F0,0(R1)    ; t=1
daddiu R1,R1,-8    ; t=2
add.d  F4,F0,F2    ; t=3
bne    R1,R2,Loop  ; t=5
sdc1   F4,8(R1)    ; t=6
ldc1   F0,0(R1)    ; t=7
```

Ξεδίπλωμα Βρόχων

- Χωρίς δρομολόγηση:

```
ldc1   F0,0(R1)    ; t=1
add.d  F4,F0,F2    ; t=3
sdc1   F4,0(R1)    ; t=6
ldc1   F0,-8(R1)   ; t=7
add.d  F4,F0,F2    ; t=9
sdc1   F4,-8(R1)   ; t=12
ldc1   F0,-16(R1)  ; t=13
add.d  F4,F0,F2    ; t=15
sdc1   F4,-16(R1)  ; t=18
ldc1   F0,-24(R1)  ; t=19
add.d  F4,F0,F2    ; t=21
sdc1   F4,-24(R1)  ; t=24
daddiu R1,R1,-32   ; t=25
bne    R1,R2,Loop  ; t=27
ldc1   F0,0(R1)    ; t=29
```

Με δρομολόγηση:

```
ldc1   F0,0(R1)    ; t=1
ldc1   F6,-8(R1)   ; t=2
ldc1   F10,-16(R1) ; t=3
ldc1   F14,-24(R1) ; t=4
add.d  F4,F0,F2    ; t=5
add.d  F8,F6,F2    ; t=6
add.d  F12,F10,F2  ; t=7
add.d  F16,F14,F2  ; t=8
sdc1   F4,0(R1)    ; t=9
sdc1   F8,-8(R1)   ; t=10
daddiu R1,R1,-32   ; t=11
sdc1   F12,16(R1)  ; t=12
bne    R1,R2,Loop  ; t=13
sdc1   F16,8(R1)   ; t=14
ldc1   F0,0(R1)    ; t=15
```

Υπερβαθμωτή Δρομολόγηση

- Εκτέλεση μιας εντολής σταθερής και μιας εντολής κινητής υποδιαστολής ανά cc:

```
ldc1    F0,0(R1)                                  ; t=1
ldc1    F6,-8(R1)                                 ; t=2
ldc1    F10,-16(R1)       add.d   F4,F0,F2        ; t=3
ldc1    F14,-24(R1)       add.d   F8,F6,F2        ; t=4
ldc1    F18,-32(R1)       add.d   F12,F10,F2      ; t=5
sdc1    F4,0(R1)          add.d   F16,F14,F2      ; t=6
sdc1    F8,-8(R1)         add.d   F20,F18,F2      ; t=7
sdc1    F12,-16(R1)                               ; t=8
daddiu  R1,R1,-40                                 ; t=9
sdc1    F16,16(R1)                                ; t=10
bne     R1,R2,Loop                                ; t=11
sdc1    F20,8(R1)                                 ; t=12
ldc1    F0,0(R1)                                  ; t=13
```

Δρομολόγηση VLIW

- Με βάση την προαναφερθείσα οργάνωση των επιμέρους εντολών:
  (Διακλαδώσεις χωρίς καθυστέρηση)

ldc1 F0,0(R1)	ldc1 F6,-8(R1)			
ldc1 F10,-16(R1)	ldc1 F14,-24(R1)			
ldc1 F18,-32(R1)	ldc1 F22,-40(R1)	add.d F4,F0,F2	add.d F8,F6,F2	
ldc1 F26,-48(R1)		add.d F12,F10,F2	add.d F16,F14,F2	
		add.d F20,F18,F2	add.d F24,F22,F2	
sdc1 F4,0(R1)	sdc1 F8,-8(R1)	add.d F28,F26,F2		
sdc1 F12,-16(R1)	sdc1 F16,-24(R1)			daddiu R1,R1,-56
sdc1 F20,24(R1)	sdc1 F24,16(R1)			
sdc1 F28,8(R1)				bne R1,R2,Loop
ldc1 F0,0(R1)	ldc1 F6,-8(R1)			

Προβλήματα Αρχιτεκτονικών VLIW

- Μεγάλος κώδικας
  - Πολλαπλό ξεδίπλωμα βρόχων
  - Κενές επιμέρους εντολές
- Συγχρονισμένη εκτέλεση επιμέρους εντολών
  - Πάγωμα όλων των επιμέρους εντολών
- Μη φορητότητα κώδικα
- Ανάγκη εύρεσης υψηλού ILP

Υβριδικές Αρχιτεκτονικές

- Προσθήκη ελέγχου εξαρτήσεων και δρομολόγηση για εκτέλεση εκτός σειράς
  - όχι για επιμέρους εντολές, η δρομολόγηση των οποίων παραμένει στην ευθύνη του μεταγλωττιστή
- Κατάλληλη κωδικοποίηση για φορητότητα κώδικα