SPM → L.3

Hazard
- structural: apare • de la proiectare dacă structura nu e complet ppl.
  • când o instr. (sau două) au nevoie de aceeași unitate funcțională;
- de dte.: apare când o instr. are nevoie de un operand care încă nu a fost calculat

Ex: ADD R_1, R_2, R_3 ⇒ $R_1 + R_2 \rightarrow R_3$ (i_1)
ADD R_3, R_4, R_5 ⇒ $R_3 + R_4 \rightarrow R_5$ (i_2)

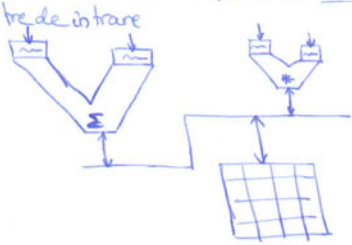
IF DEC OF EX WB
i_1
i_2 → nu are cum să fie aici ptr. că R_3 nu a fost calculat ⇒ hazard de dte.

hazardul de dte. poate fi evitat la compilare ptr. că compilatorul rearanjează instr. (ex: pune o altă instr. între i_1 și i_2) sau pune NOP.

Determinarea hazardului (dependențe între dte.):
→ dinamic: ptr. det. paralelismului la niv. de buclă;
→ static: se scanează codul sursă.

∃ dep. de dte. ce nu pot fi det. static / dinamic → suntadr. de load.
Met. de det. a dep. de dte.: < Tomasulo
Scoreboard (display)

• Tomasulo: - atribuirea elem. fizice de execuție a 2 seturi de regis-
→ sumatoare, multiplicatoare



- fiecărui reg. i se adaugă încă 2 câmpuri:
  - busy – reg. e fol. ca și destinație ptr. operație dar nu a fost încă evaluat;
  - tal – reg. ce identifică unitatea ce va scrie în registru

Reg. X unit. fct. sunt legate printr-un Common Data Bus → sumarul a terminat de facut un calcul, iar rez. lui poate fi fol. de multiplicator fără a mai fi nevoie să îl încărcăm în registru.

- Scoreboard: - se păstrează info de stare despre instr. exec, regiștrii X unit. fcționale ⇒ când o instr. e trimisă spre execuție în fct. reg. și a celorlalte unit. fct. este înscrisă în scoreboard;
  - un instantaneu Up scoreboard spune dac o instr. poate fi emisă în execuție;
  - Scoreboard are 3 tabele:
    - instruction status: spune dc. o instr. a fost emisă spre exec;
      - dc. da, în tabel vedem starea instrucțiunii;
      - după ce instr. a fost extr. și dec. scoreboard încearcă să o emită la exec. la unit. fct. ⇒ 2 condiții:
        - u fct. să fie liberă;
        - să nu o altă instr. cu același reg. dest;
      - dc. aceste cond. nu sunt îndep. se întârzie execuția;
    - function unit status: arată starea u fct. - dc. instr. nu a fost încă exec, tab. identifică reg. destinație X starea reg. surse;
    - destination register status: tabl. ce indică reg. dest ce nu au fost încă înscrise ⇒ ptr. a. reg. se identifică u fct. care le va înscrie.

Hazard de comenzi ⇒ se referă la comenzile condiționale (ce det. salt) { condiționat; necondiționat (call, jump) → nu apare hazard de comenzi. }

La salt condiționat, pt. a ști pe ce ramură o ia th. să evaluez cond. de salt → dc. am greșit ramura th. golit din nou ppl!

La Pentium s-a introdus un Branch Prediction → acuratețe ≥ 99%.

Predicția salturilor: → statică: decizia de a face salt e luată înainte de execuția programului;
→ dinamică: UP ia decizia de salt în funcție de comportamentul la salt al acelei instr. în trecut; la pr. instr. o ia aleator sau o ia mereu true;

Variante de a face predicția dinamică:
păstrăm într-un bloc ce decizie s-a făcut înainte;
sau asocierea unui contor de n biți:
→ Counter Based Branch Prediction
(e mai eficient pt că J o plajă mai mare de valori).

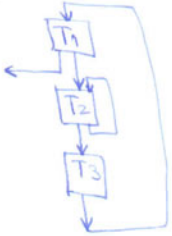
După pr. exec. a instr. de salt contorului i se atribuie o val. T în caz că s-a făcut salt sau (T-1) dc. nu s-a făcut salt. La o nouă execuție se face salt, dc. val. contorului e ≥ T și se increm. T, sau nu se face salt, dc. val. contor. < T și se decrem. contorul. Nu se decrem. sub 0 și nu se increm. peste $2^n - 1$ (?).

O altă var. este Glade Branching → rearanjarea codului ca să se introducă instr. între instr. de salt și unu. instr. de după salt. Când nu se pot introduce astfel de instr. se introduce NOP.

Altă var. → Multiple Prefaching → UP extrage ambele trasee, iar după luarea deciziei traseul greșit este ignorat. Astfel se asigură un flux constant de instr. și se reduc întârzierile date de golirea / umplerea ppl.

ppl. < liniare → ce am discutat până acum
dinamice

Ppb. dinamice:



⇒ ca să știm cum se plimbă instr. în o tabelă de rezervări:

	t_1	t_2	t_3	t_4	t_5
T1	*				*
T2		*	*		
T3				*	

Th. să aflăm cadența cu care th. introduse instr. în ppb

Cadență posibilă: ~~0~~ 1 2 3 4 5 ~

Cadența 0 este interzisă

1 nu este posibilă

2 nu este bună ptr. că se suprap. cu 5

3 e ok

4 nu ptr. că se suprap. cu 1

Cadența se găsește astfel: primul nr. întreg care nu are multiple interzise

Vector de coliziune de lung. egală cu cea mai lungă instr.:

0	1	2	3	4	5
1	1	1	0	1	1

→ vector de coliziune.

Pe baza vect. de coliziune ⇒ latența minimă $= \frac{1+1+1+1+1}{2} = 2,5$

dar se ia o dată 2 o dată 3.

↓

Î în curs un exemplu detaliat (pb. de examen!!!)

Tema: Parametri de performanță ai rețelelor de interconectare. (2h)

→ 5 ÷ 7 pg. în RO

→ prezentare orală peste 2 săpt.