Desempenho Cache

↓ mr: associatividade, algoritmo
↓ penalidade: multinível

$T_{exec\ CPU} = (ciclos_{CPU} + ciclos_{paradas}) \cdot T_{CPU}$

↳ incluindo hazards e pipeline

↳ falta em cache

$ciclos_{paradas} = ciclos_{paradas}^{(leitura)} + ciclos_{paradas}^{(escrita)}$

→ $ciclos_{paradas}^{(leitura)} = \frac{leituras}{programa} \cdot mr_{(leitura)} \cdot penalidade_{(leitura)}$

↳ miss rate

→ Escrita: $ciclos_{paradas}^{(escrita)} = \frac{escritas}{programa} \cdot mr_{(escrita)} \cdot \frac{penalidade_{(escrita)}}{(*)}$

(*)
- write throught: escrever na cache, escrever na memória
  - ↳ penalidade: acesso e escrita na memória principal
- write back: só vai copiar para a memória quando for substituído
  - ↳ penalidade: tempo para copiar na memória principal
- write allocate: deu write miss (**) e escreve nessa posição na cache
  - ↳ penalidade:

(**) quando você vai escrever na posição da cache e as tags não são iguais.

↳
- write allocate: escrevo nessa posição na cache, destruindo o anterior
  - ↳ Obs: vai evitar a escrita na memória (+ lento)
- write-no-allocate: escrevo nessa posição na memória.

↳ write buffer

ou seja: $ciclos_{paradas} = \frac{acessos\ memória}{programa} \cdot mr \cdot penalidade$

$= \frac{instruções}{programa} \cdot \frac{mr}{instrução} \cdot penalidade$

ex:

Instruções
- ↳ mr: 2%

Dados
- ↳ mr: 4%

- CPI = 2
- penalidade = 100 ciclos
- % load e store = 36%

1) $I\ ciclos_{miss} = I \cdot 2\% \cdot 100 = 2I$

$D\ ciclos_{miss} = (\underbrace{I \cdot 36\%}_{peso\ das\ load/store}) \cdot \overbrace{4\%}^{mr} \cdot 100 = 1{,}44I$

2) $ciclos_{paradas} = 3{,}44I$

3) $\div I \rightarrow CPI = 3{,}44_{paradas}$

4) Tot CPI = CPI base + CPI paradas = 2 + 3,44 = 5,44 ciclos

→ AMAT: average memory access time
↳ considera os hits e misses

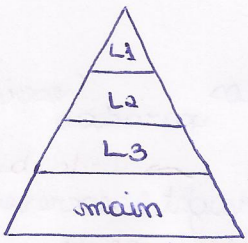
Obs: usado para comparação de designs diferentes.

$$AMAT = T_{hit} + mr \cdot penalidade$$

Ex:

$T = 1\,ns$
penalidade = 20 ciclos
mr = 0,05 = 5%
acesso = 1 ciclo

$AMAT = 1 + 0,05 \cdot 20$
$AMAT = 2$ ciclos



Multilevel Caches

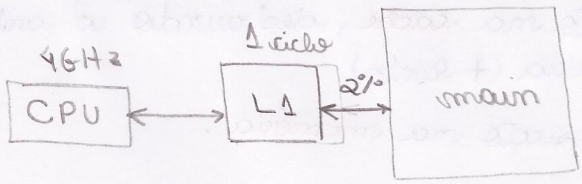
→ Motivo:
↳ retardar a busca na memória principal, adicionando caches maiores (penalidade menor que a memória) em níveis inferiores.

Ex: $CPI_{base} = 1$
clock = 4GHz
$T = 0,25\,ns$

penalidade (main) = 100 ns
I mr = 2%
100/0,25 → 400 ciclos

L2 → acesso = 5 ns
miss rate global = 0,5%

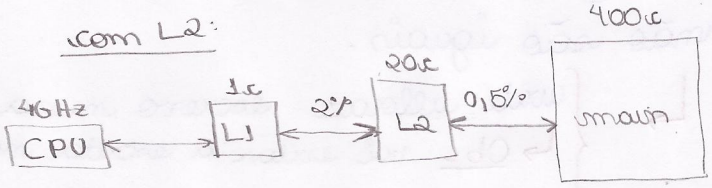
usó L1:



Tot CPI = CPI base + CPI paradas
= 1 + ciclos paradas / I
= 9 ciclos

→ ciclos paradas = I . mr . penalidade
= I . 0,02 . 400 = 8I

com L2:



Tot CPI = CPI base + CPI paradas
= 1 + ciclos paradas / I
= 3,4 ciclos

→ ciclo paradas = I . (0,02 . 20 + 0,005 . 400)
= I . 2,4

↕

I [(2% − 0,5%) . 20 + 0,5% . 400]