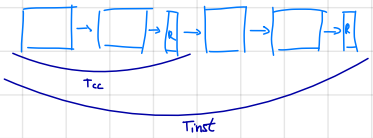
Tempo de execução → T_{EXEC} = # clock cycles · T_{cc} = CPI · #I / f = CPI · #I · T_{cc}

→ + # ciclos de penalização

→ o período do relógio

T_{inst} = nº de estágios · T_{cc}

T_{inst} → tempo que demora para executar uma instrução



Período do relógio → $T_{cc} = \frac{1}{f}$ → frequência (10^9 = 1GHz)

1 GHz → $1 \cdot 10^9$ Hz

(milis.) ms → 10^{-3}

(micro s.) µs → 10^{-6}

(nano s.) ns → 10^{-9}

(pico s.) ps → 10^{-12}

clock cycles = CPI · #I

MIPS (milhões de instruções por segundo - métrica enganadora) = $\frac{\#I}{T_{EXEC} \cdot 10^6}$

Hit rate = # hits / # acessos

Miss rate = 1 - hit rate (0 a 1)

—— // ——

$CPI = CPI_{CPU} + CPI_{Mem}$ (ciclos por instrução)

$T_{EXEC} = \#I \cdot (CPI_{CPU} + CPI_{Mem}) / f$

$CPI_{Mem} = CPI_{Mem\_I} + CPI_{Mem\_D}$ → data (acesso a ler os dados)

↳ ler na memória

$CPI_{Mem} = (mr_I + \%Mem \cdot mr_D) \cdot mp$ → varia de 0 a 1

- mr_I: miss rate
- $\%Mem$: % de instruções que acedem à memória ($\frac{\#I\ Mem}{\#I\ Total}$)
- mr_D: miss rate
- mp: miss penalty (custo em ciclos); mp_T = x seg. ⇒ $mp = mp_T \cdot f$ = y ciclos de relógio

→ cache infinita ⇒ mr=0

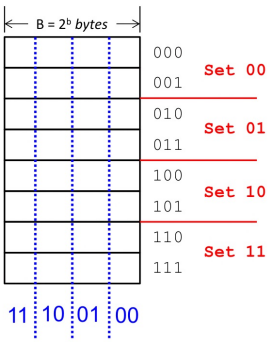
→ sem cache ⇒ mr=1

Fator		Prefixo	Símbolo
10^{-24}	= 0,000 000 000 000 000 000 000 001	yocto	y
10^{-21}	= 0,000 000 000 000 000 000 001	zepto	z
10^{-18}	= 0,000 000 000 000 000 001	atto	a
10^{-15}	= 0,000 000 000 000 001	femto	f
10^{-12}	= 0,000 000 000 001	pico	p
10^{-9}	= 0,000 000 001	nano	n
10^{-6}	= 0,000 001	micro	µ
10^{-3}	= 0,001	mili	m
10^{-2}	= 0,01	centi	c
10^{-1}	= 0,1	deci	d
10^{0}	= 1		
10^{1}	= 10	deca	da
10^{2}	= 100	hecto	h
10^{3}	= 1 000	kilo	k
10^{6}	= 1 000 000	mega	M
10^{9}	= 1 000 000 000	giga	G
10^{12}	= 1 000 000 000 000	tera	T
10^{15}	= 1 000 000 000 000 000	peta	P
10^{18}	= 1 000 000 000 000 000 000	exa	E
10^{21}	= 1 000 000 000 000 000 000 000	zetta	Z
10^{24}	= 1 000 000 000 000 000 000 000 000	yotta	Y

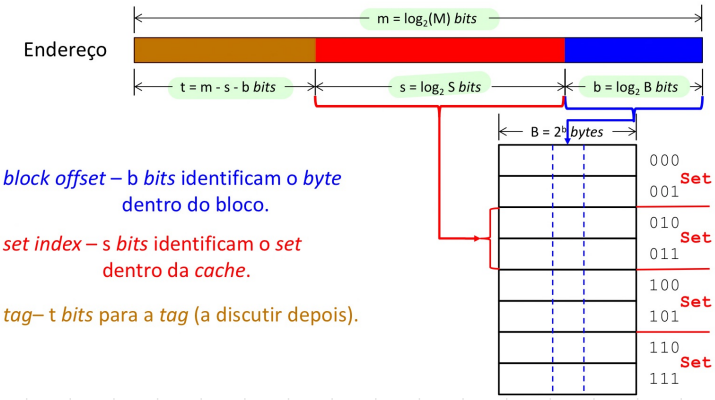
- A memória (central) tem M *bytes*.
  O endereço tem m = $\log_2$ (M) *bits*
- A *cache* organiza-se em linhas (blocos) de B *bytes*.
  b=$\log_2$ (B) *bits* identificam um *byte*.
- As linhas encontram-se agrupadas em S *sets*.
  s=$\log_2$ (S) *bits* identificam um *set*.
- Cada *set* contém E linhas.
- A organização da *cache* é caracterizada por (S, E, B, m)

Número total de linhas = S * E
Capacidade da *cache* = S * E * B

Cache (4, 2, 4, m (?))



- Cache (S, E, B, m)



block offset – b *bits* identificam o *byte* dentro do bloco.

set index – s *bits* identificam o *set* dentro da *cache*.

tag– t *bits* para a *tag* (a discutir depois).