

---

# **Chapter 1: Computer Abstractions and Technology**

## **1.6: Performance**

**Arquitetura e Organização  
de Computadores**

---

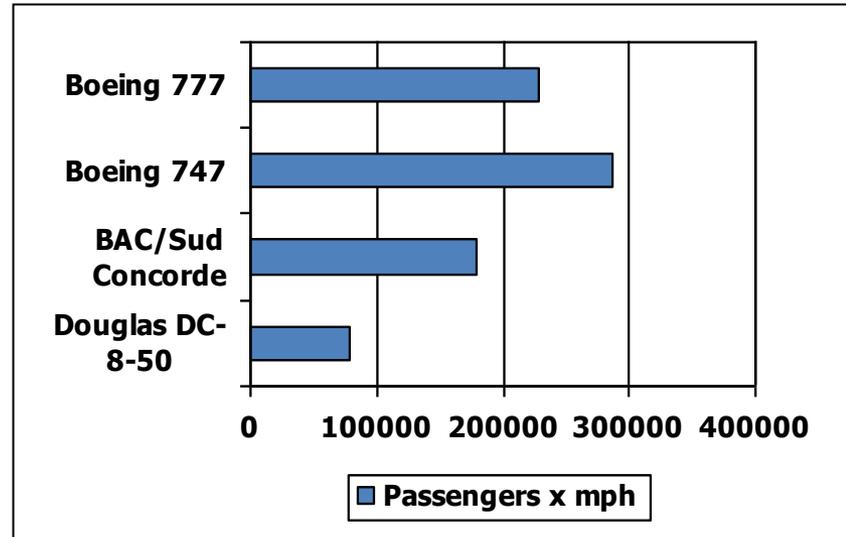
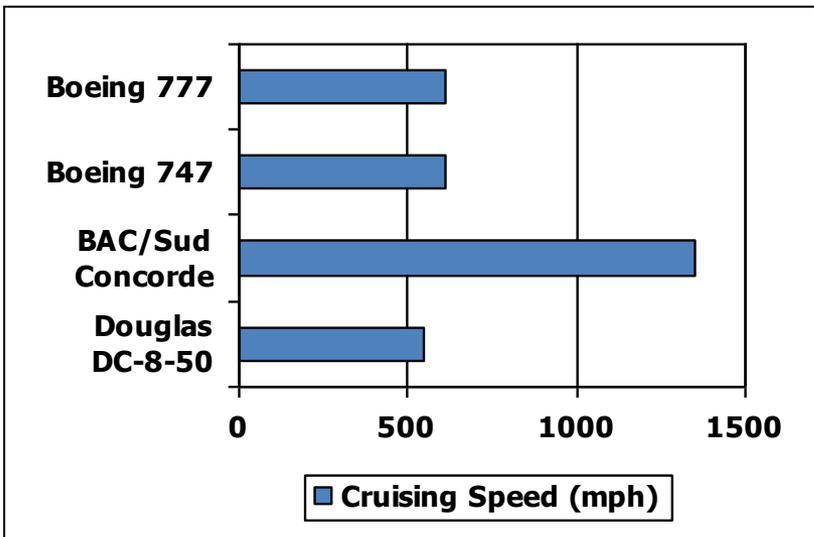
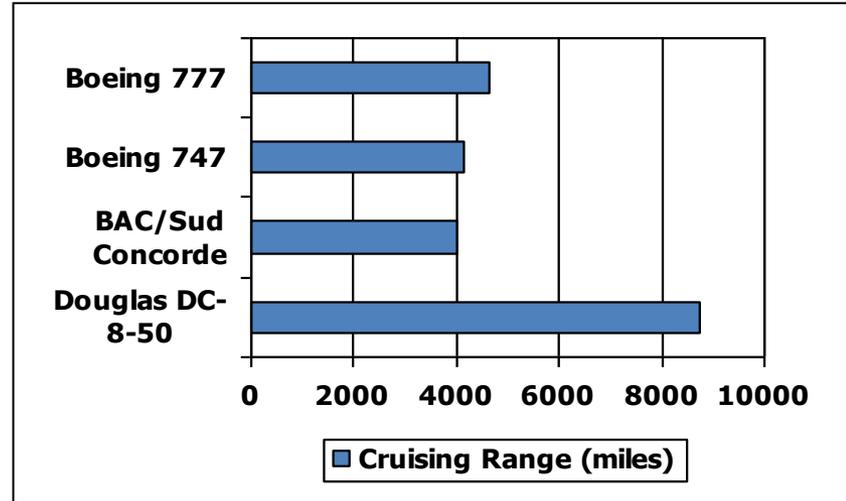
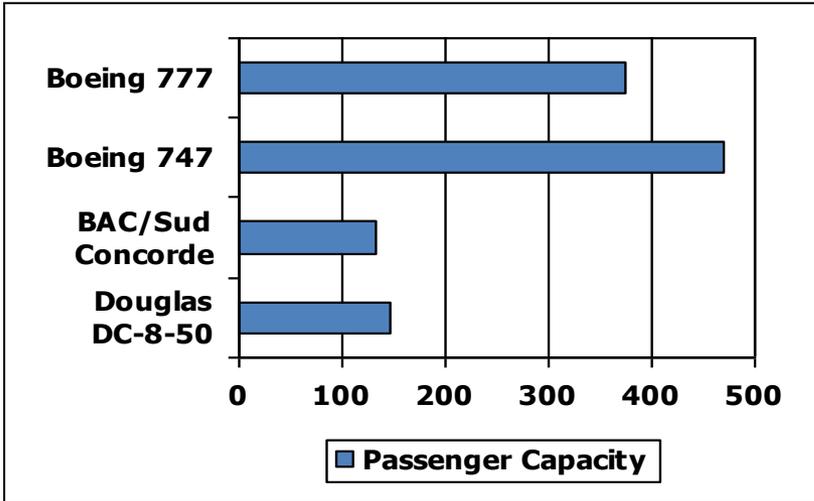
# Performance dos Computadores

---

- O que é performance/desempenho? Qual a **definição**?
  - Diferentes maneiras de avaliar
- Escolhida a maneira, como medir? Quais as **métricas**?
  - Ponto de vista do usuário
  - Ponto de vista do projetista
- Como as métricas estão **relacionadas**?
- É possível resumir tudo em uma **equação**?

# Definição de Performance/Desempenho

- Que avião tem a melhor performance?



# Definição de Performance/Desempenho

---

- Que avião tem a melhor performance?

<b>Airplane</b>	<b>Passenger capacity</b>	<b>Cruising range (miles)</b>	<b>Cruising speed (m.p.h.)</b>	<b>Passenger throughput (passengers × m.p.h.)</b>
Boeing 737	240	3000	564	135,360
BAC/Sud Concorde	132	4000	1350	178,200
Boeing 777-200LR	301	9395	554	166,761
Airbus A380-800	853	8477	587	500,711

# Definição de Performance/Desempenho

---

- **Tempo de Resposta (tempo de execução, latência):**
  - Quem termina o trabalho primeiro? Quanto tempo demora?
  - É o tempo total exigido para o computador completar uma tarefa, incluindo acessos ao disco, acessos à memória, atividades de I/O, overhead do sistema operacional, tempo de execução de CPU, tempo em espera, etc.
  - Tempo entre o início e o término de uma tarefa.



# Definição de Performance/Desempenho

---

- **Throughput (largura de banda):**
  - Quantidade total de trabalho (tarefas) realizado em um determinado tempo
  - Número de tarefas completadas por unidade de tempo
    - **tarefas/transações/... por hora**



# Definição de Performance/Desempenho

---

- Desktops, mobile, embarcado, etc.:
  - Tempo de Resposta ou Throughput?
- Servidores:
  - Tempo de Resposta ou Throughput?



# Definição de Performance/Desempenho

---

- **1º Pense e Responda:** as seguintes alterações em um computador aumentam o throughput, diminuem o tempo de resposta ou ambos?
  - A) Substituir o processador por uma versão mais rápida.
  - B) Acrescentar processadores adicionais (iguais ao já existente) em um computador que utiliza múltiplos processadores para tarefas separadas.

# Definição de Performance/Desempenho

---

- **1º Pense e Responda:** as seguintes alterações em um computador aumentam o throughput, diminuem o tempo de resposta ou ambos?
  - A) Substituir o processador por uma versão mais rápida.
    - Diminui o tempo de resposta
    - Aumenta o throughput
  - B) Acrescentar processadores adicionais (iguais ao já existente) em um computador que utiliza múltiplos processadores para tarefas separadas.
    - Aumenta o throughput
    - Diminui o tempo de resposta (em alguns casos)

# Definição de Performance/Desempenho

---

- Vamos considerar, agora no início da disciplina, apenas a discussão sobre o **Tempo de Resposta** (ou **Tempo de Execução**).
- Definimos então a PERFORMANCE (ou desempenho) de um determinado computador “X” como **o inverso do tempo de execução**:

$$\text{Performance}_X = \frac{1}{\text{Execution time}_X}$$

- **2º Pense e Responda:** calcule a performance dos computadores A e B na realização de diversas tarefas. Indique qual computador tem melhor performance em cada tarefa.

# Definição de Performance/Desempenho

---

- Em geral estamos sempre interessados na **performance relativa**: quantas “vezes” o computador A é melhor do que o computador B. Cuidado: confusão gramatical!
  - “O computador A é ***n*** vezes mais rápido do que o B.”

$$\begin{aligned} \text{Performance}_X &> \text{Performance}_Y \\ \frac{1}{\text{Execution time}_X} &> \frac{1}{\text{Execution time}_Y} \\ \text{Execution time}_Y &> \text{Execution time}_X \end{aligned}$$

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = n$$

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution time}_Y}{\text{Execution time}_X} = n$$

- **3º Pense e Responda**: calcule a performance relativa entre os computadores A e B.

# Definição de Performance/Desempenho

---

- **Tempo de Resposta (tempo de execução, latência):**
  - É o tempo total que o computador leva para executar uma tarefa, incluindo todos os aspectos:
    - tempo de execução da CPU
    - acessos ao disco
    - acessos à memória
    - demais atividades de I/O
    - overhead do sistema operacional
    - tempo em espera
    - etc...
- Dá para medir isso de verdade? Compile e execute o programa `“sum_full.c”` no próximo exercício!

# Definição de Performance/Desempenho

- **4º Pense e Responda:** meça o tempo de resposta para realizar a soma de N números aleatórios de ponto flutuante, e anote o resultado. O que você conclui?

```
[yanyh@fornax ~]$ time ./sum 1000000
```

```
=====
```

```
Sum 1000000 numbers
```

```
-----
```

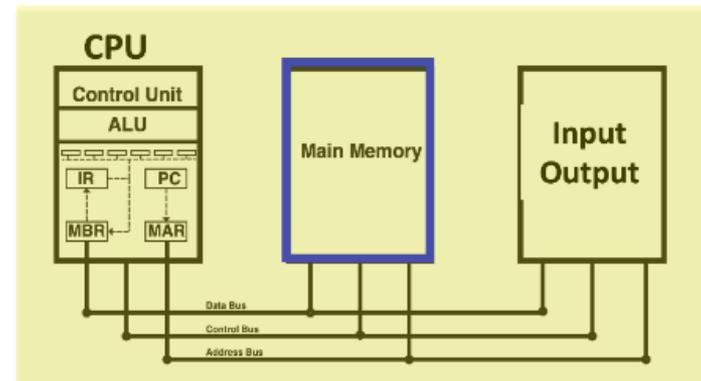
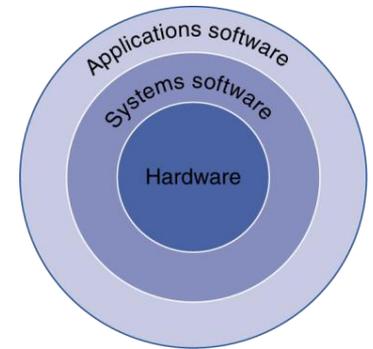
Performance:	Runtime (ms)	MFLOPS
Sum:	24.999857	800.004578

```
-----
```

```
real    0m0.200s
```

```
user    0m0.179s
```

```
sys     0m0.020s
```



# Definição de Performance/Desempenho

---

- Qual o principal problema de usar o Tempo de Resposta (tempo de execução, latência):
  - É o tempo total que o computador leva para executar uma tarefa, incluindo todos os aspectos:
    - tempo de execução da CPU
    - acessos ao disco
    - acessos à memória
    - demais atividades de I/O
    - overhead do sistema operacional
    - tempo em espera
    - etc...
- **Mistura tudo! Como distinguir entre o tempo decorrido e o tempo que o computador realmente está trabalhando em nossa tarefa?**

# Definição de Performance/Desempenho

---

- **Tempo de CPU (Tempo de Execução de CPU):**
  - É o tempo real que a CPU gasta realizando uma tarefa específica, sem incluir o tempo de I/O, o tempo de espera, etc.
  - É dividido em 2 componentes:
    - **Tempo de CPU de Usuário:**
      - O tempo de CPU gasto no programa propriamente dito.
    - **Tempo de CPU de Sistema:**
      - O tempo de CPU gasto no sistema operacional, realizando tarefas em favor do programa.
- Dá para medir isso de verdade? Volte ao programa “`sum_full.c`” no próximo exercício!

# Entendendo o comando “time”

- **“Real”**: tempo de resposta, do relógio – o tempo gasto entre o início e o fim da tarefa. Compreende todo o tempo gasto incluindo fatias de tempo utilizados por outros processos, I/O, acessos à disco, etc.
- **“User”**: tempo de CPU gasto no código do modo de usuário **dentro do processo** (fora do kernel). Este é o tempo real de CPU usado na execução do processo (programa). Outros processos e o tempo que o processo passa bloqueado não contam para este valor (não leva em conta os outros aspectos que influenciam no tempo de resposta, tais como outros processos, I/O, acessos à disco, etc.).
- **“Sys”**: tempo de CPU gasto **dentro do kernel** em benefício do processo. Compreende o tempo que a CPU gastou executando chamadas do sistema dentro do kernel (fora do processo), em oposição ao código da biblioteca, que ainda está em execução no espaço do usuário. Também não leva em conta os outros aspectos que influenciam no tempo de resposta.

```
[yanyh@fornax ~]$ time ./sum 10000000
=====
Sum 10000000 numbers
-----
Performance:           Runtime (ms)      MFLOPS
-----
Sum:                   24.999857      800.004578

real    0m0.200s
user    0m0.179s
sys     0m0.020s
```

# Entendendo o comando “time”

- Timer: usado para calcular o tempo decorrido na função de soma:

```
elapsed = read_timer();  
REAL result = sum(N, X, a);  
elapsed = (read_timer() - elapsed);
```

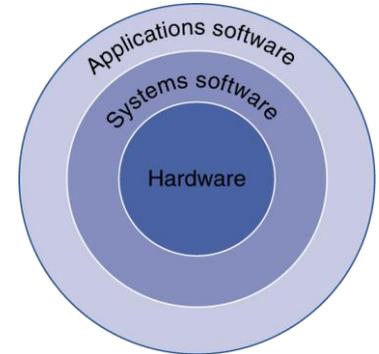
```
[yanyh@fornax ~]$ time ./sum 10000000
```

```
=====
```

Sum 10000000 numbers

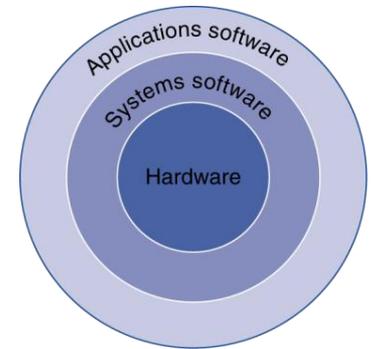
Performance:	Runtime (ms)	MFLOPS
Sum:	24.999857	800.004578

```
real    0m0.200s  
user    0m0.179s  
sys     0m0.020s
```



# Definição de Performance/Desempenho

- **5º Pense e Responda:** meça o tempo de resposta para realizar a soma de N números aleatórios de ponto flutuante, e anote o resultado. O que você conclui?



```
[yanyh@fornax ~]$ time ./sum 1000000
```

```
=====
```

```
Sum 10000000 numbers
```

```
-----
```

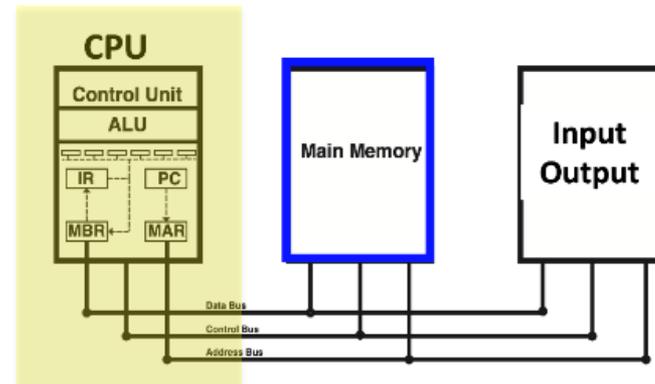
Performance:	Runtime (ms)	MFLOPS
Sum:	24.999857	800.004578

```
-----
```

```
real    0m0.200s
```

```
user    0m0.179s
```

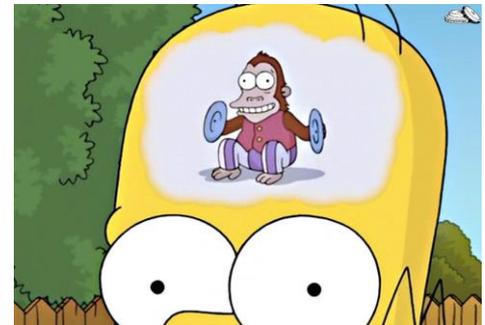
```
sys     0m0.020s
```



# Definição de Performance/Desempenho

---

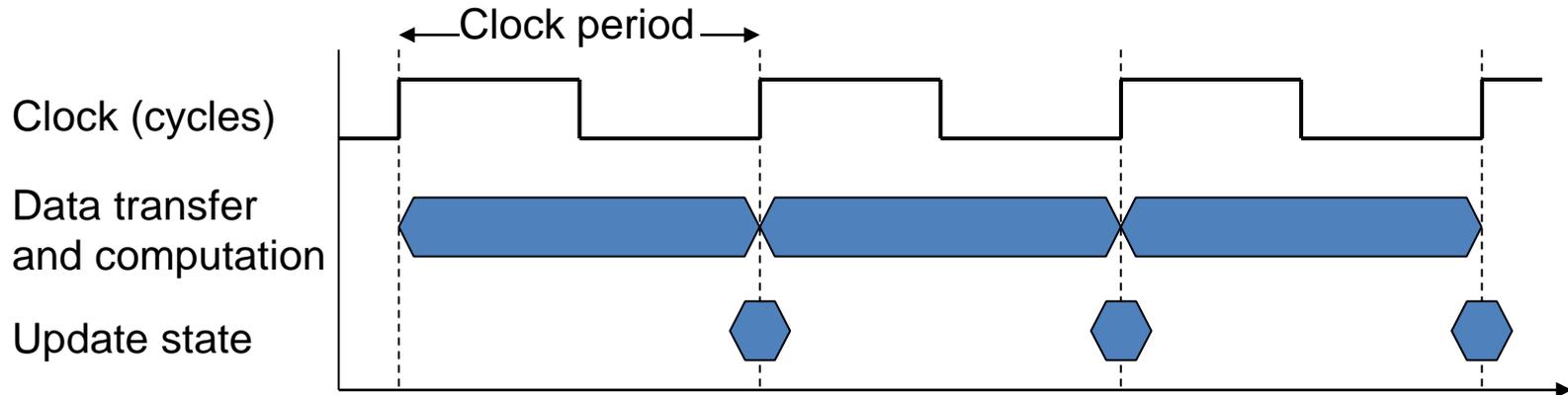
- Até agora discutimos a performance mais do ponto de vista do **usuário** do computador, a rapidez para executar tarefas (do início ao fim; quantidade):
  - Tempo de Resposta
  - Tempo de CPU (usuário/sistema)
  - Throughput
- **Engenheiros/projetistas** precisam de outro modo de medir a performance. Eles estão mais interessados na **velocidade com que o hardware pode realizar suas funções básicas!** Para isso precisamos de outras maneiras de medir a performance:
  - Ciclo de Clock
  - Período de Clock
  - Taxa de Clock



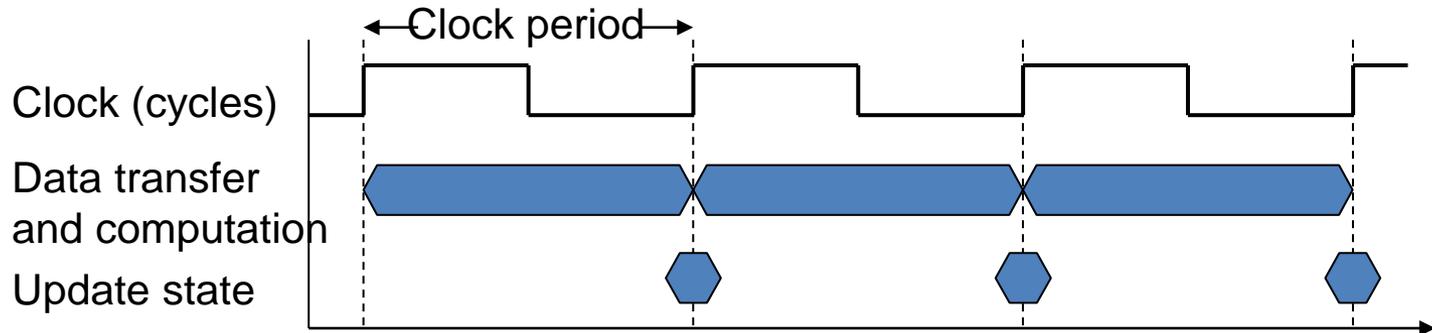
# Definição de Performance/Desempenho

- **O que é o clock?**

- É um “mecanismo” embutido que governa a operação dos hardwares digitais, alternando entre estados de baixa e alta voltagem (os famosos estados binários 0 e 1).



# Definição de Performance/Desempenho



- **Ciclo de Clock:** intervalos discretos de tempo entre as ocorrências das “batidas” do clock (normalmente do clock do processador), que trabalha a uma velocidade constante.
- **Período de Clock:** a duração (normalmente em alguma unidade de segundos) de cada ciclo de clock, o tempo para um ciclo de clock completo.
  - Ex.: 250 ps (picosegundos)
- **Taxa de Clock:** o inverso do Período de Clock, quantos ciclos de clock ocorrem em uma unidade de tempo.
  - Ex.: 4 GHz (gigahertz)

# Definição de Performance/Desempenho

- 6º Pense e Responda: qual a Taxa de Clock de um computador que tem Período de Clock de 250 ps?

PREFIXO	SÍMBOLO	POTÊNCIA	MULTIPLICADOR
DECA	da	$10^1$	10
HECTO	h	$10^2$	100
QUILO	k	$10^3$	1000
MEGA	M	$10^6$	1000000
GIGA	G	$10^9$	1000000000
TERA	T	$10^{12}$	1000000000000
PETA	P	$10^{15}$	1000000000000000
EXA	E	$10^{18}$	1000000000000000000
ZETA	Z	$10^{21}$	1000000000000000000000
IOTA	Y	$10^{24}$	1000000000000000000000000
DECI	d	$10^{-1}$	0,1
CENTI	c	$10^{-2}$	0,01
MILI	m	$10^{-3}$	0,001
MICRO	$\mu$	$10^{-6}$	0,000001
NANO	n	$10^{-9}$	0,000000001
PICO	p	$10^{-12}$	0,000000000001
FEMTO	f	$10^{-15}$	0,000000000000001
ATO	a	$10^{-18}$	0,000000000000000001
ZEPTO	z	$10^{-21}$	0,000000000000000000001
IOCTO	y	$10^{-24}$	0,000000000000000000000001

# Cuidado com as Unidades!

$10^{-3}$ s	<b>ms</b>	<b>millisecond</b>
$10^{-6}$ s	<b><math>\mu</math>s</b>	<b>microsecond</b>
$10^{-9}$ s	<b>ns</b>	<b>nanosecond</b>
$10^{-12}$ s	<b>ps</b>	<b>picosecond</b>

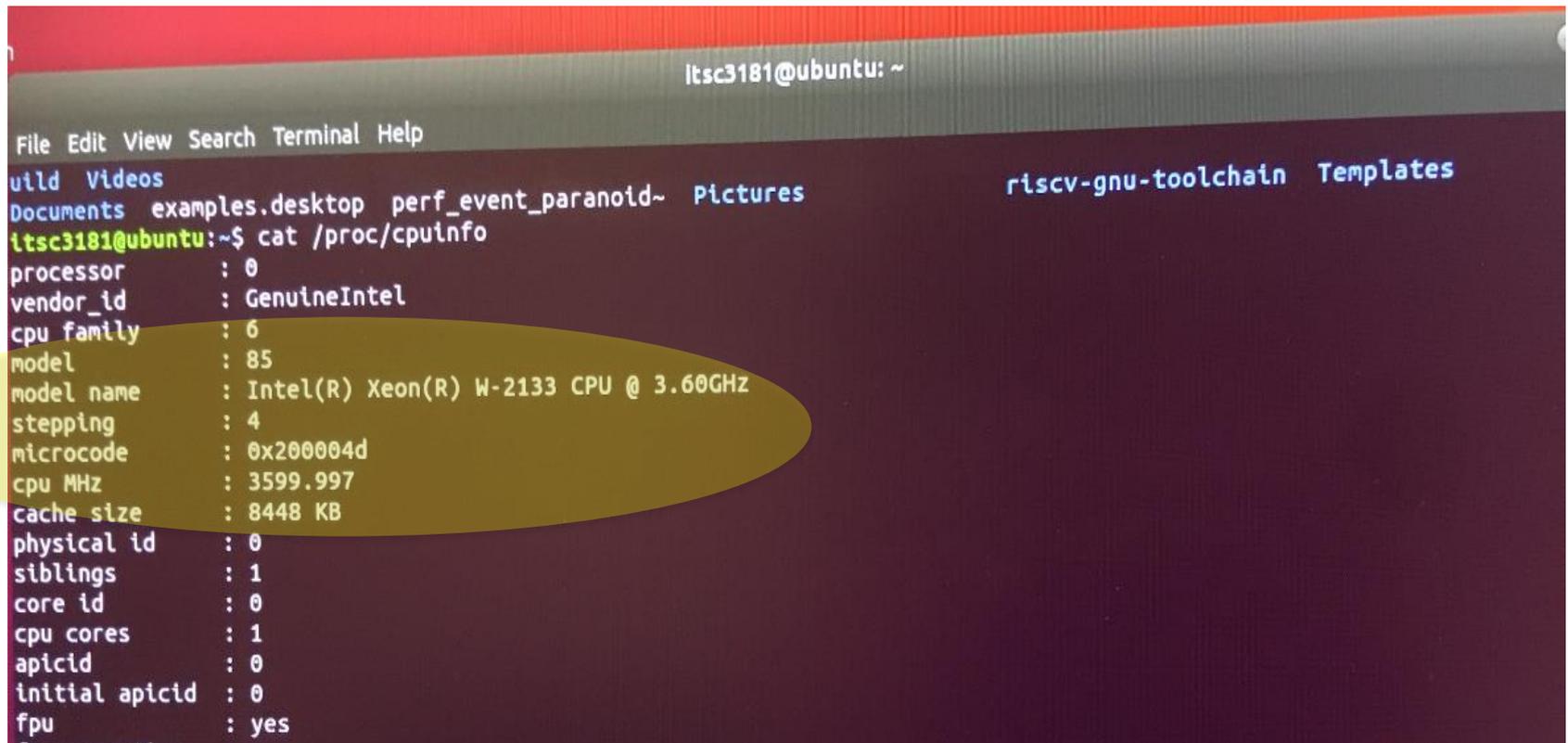
$10^3$ Hz	<b>kHz</b>	<b>kilohertz</b>
$10^6$ Hz	<b>MHz</b>	<b>megahertz</b>
$10^9$ Hz	<b>GHz</b>	<b>gigahertz</b>

## Prefixes for multiples of bits (bit) or bytes (B)

Decimal				Binary			
Value		SI		Value		IEC	
1000	$10^3$	k	kilo	1024	$2^{10}$	Ki	kibi
1000 <sup>2</sup>	$10^6$	M	mega	1024 <sup>2</sup>	$2^{20}$	Mi	mebi
1000 <sup>3</sup>	$10^9$	G	giga	1024 <sup>3</sup>	$2^{30}$	Gi	gibi
1000 <sup>4</sup>	$10^{12}$	T	tera	1024 <sup>4</sup>	$2^{40}$	Ti	tebi
1000 <sup>5</sup>	$10^{15}$	P	peta	1024 <sup>5</sup>	$2^{50}$	Pi	pebi
1000 <sup>6</sup>	$10^{18}$	E	exa	1024 <sup>6</sup>	$2^{60}$	Ei	exbi
1000 <sup>7</sup>	$10^{21}$	Z	zetta	1024 <sup>7</sup>	$2^{70}$	Zi	zebi
1000 <sup>8</sup>	$10^{24}$	Y	yotta	1024 <sup>8</sup>	$2^{80}$	Yi	yobi

# Definição de Performance/Desempenho

- **7º Pense e Responda:** descubra a **Taxa de Clock** de seu processador e calcule o **Período de Clock**.



```
ltsc3181@ubuntu: ~  
File Edit View Search Terminal Help  
uild Videos  
Documents examples.desktop perf_event Paranoid Pictures riscv-gnu-toolchain Templates  
ltsc3181@ubuntu:~$ cat /proc/cpuinfo  
processor      : 0  
vendor_id     : GenuineIntel  
cpu family    : 6  
model         : 85  
model name    : Intel(R) Xeon(R) W-2133 CPU @ 3.60GHz  
stepping      : 4  
microcode    : 0x200004d  
cpu MHz       : 3599.997  
cache size   : 8448 KB  
physical id   : 0  
siblings      : 1  
core id      : 0  
cpu cores    : 1  
apicid       : 0  
initial apicid : 0  
fpu          : yes
```

# Definição de Performance/Desempenho

---

- PARE! Certifique-se que você entendeu onde estamos:
  - Métricas de interesse do USUÁRIO:
    - Tempo de Resposta
    - Tempo de CPU (usuário/sistema)
    - Throughput
  - Métricas de interesse do PROJETISTA:
    - Ciclos de Clock
    - Períodos de Clock
    - Taxa de Clock
- Dá para relacionar os interesses do USUÁRIO com os interesses do PROJETISTA? **Por que seria interessante descobrir a relação entre essas métricas?**

# Definição de Performance/Desempenho

---

- Dá para relacionar os interesses do USUÁRIO com os interesses do PROJETISTA? **Por que seria interessante descobrir a relação entre essas métricas?**
  - Se for possível achar uma **relação entre as métricas do PROJETISTA com as métricas do USUÁRIO**, poderíamos **determinar o efeito de uma mudança no projeto de hardware sobre o desempenho experimentado pelo usuário!**

- Equação simples:

$$\text{CPU execution time for a program} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$$

$$\text{CPU execution time for a program} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

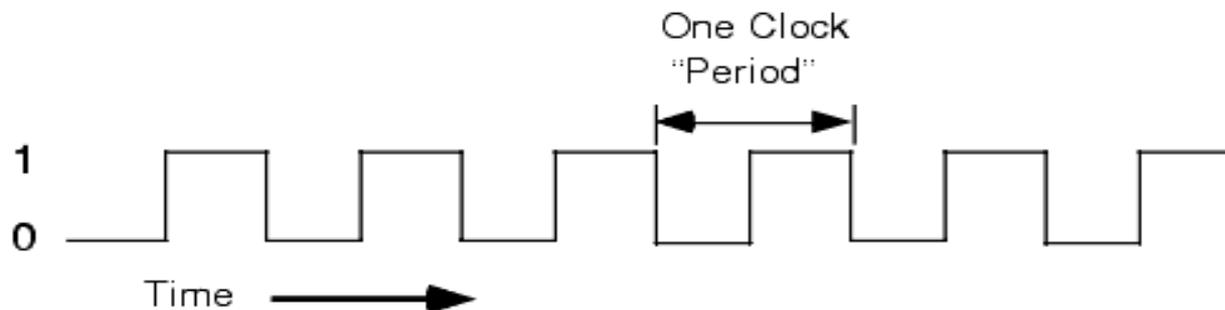
- **8º Pense e Responda:** de acordo com as equações simples acima, como o projetista pode melhorar o desempenho (tempo de CPU) de um computador?

# Tempo de CPU

- É possível melhorar:
  - Reduzindo a quantidade de ciclos de clock necessários para rodar um programa
  - Aumentando a taxa de clock (diminuindo o período de clock)
- Existe um BALANÇO entre a quantidade de ciclos de clock e o período de clock: aumentar um pode diminuir o outro e vice-versa.

$$\text{CPU Time}(s) = \# \text{ CPU Clock Cycles} \times \text{Clock Cycle Time}(s)$$

$$= \frac{\# \text{ CPU Clock Cycles}}{\text{Clock Rate (Hz)}}$$



# Melhorando a Performance

- **9º Pense e Responda:** um determinado programa executa em 10 segundos no computador A, que tem um clock de 2 GHz. Um engenheiro quer construir o computador B, que executará esse mesmo programa em 6 segundos. O engenheiro calculou que é possível aumentar a taxa de clock, mas esse aumento afetará o restante do projeto da CPU, fazendo com que o computador B precise executar esse programa com uma quantidade de ciclos de clock 20% maior do que a quantidade de ciclos do computador A. Qual taxa de clock o computador B deve ter?

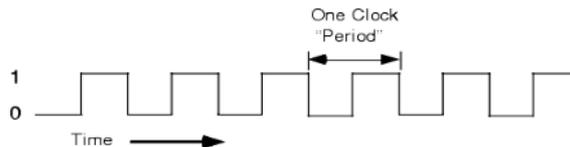
$$\text{CPU execution time for a program} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}} \times \text{Clock cycle time}$$

$$\text{CPU execution time for a program} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

# Outra Forma de Cálculo: usar o CPI

- A CPU executa um programa instrução por instrução
- Até agora as equações não levaram em conta o número de instruções necessárias para executar um programa
- **Ciclos de Clock por Instrução (CPI):** número médio de ciclos de clock por instrução para um programa
- Diferentes instruções podem exigir diferentes quantidades de tempo, portanto o CPI é uma média de todas as instruções executadas no programa

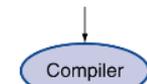
$$\text{CPU clock cycles} = \text{Instructions for a program} \times \text{Average clock cycles per instruction}$$



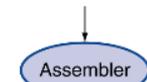
# Outra Forma de Cálculo: usar o CPI

- Diferentes instruções podem exigir diferentes quantidades de tempo, portanto o CPI é uma média de todas as instruções executadas no programa
- A quantidade de instruções para um programa é determinada pelos seguintes fatores:
  - O próprio programa
  - A arquitetura do conjunto de instruções (ISA)
  - O compilador
- **10º Pense e Responda:** dado o que você aprendeu até aqui, por que o CPI seria importante?

```
swap(int v[], int k)
{int temp;
  temp = v[k];
  v[k] = v[k+1];
  v[k+1] = temp;
}
```



```
swap:
  muli $2, $5,4
  add $2, $4,$2
  lw $15, 0($2)
  lw $16, 4($2)
  sw $16, 0($2)
  sw $15, 4($2)
  jr $31
```



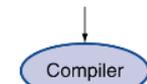
```
000000001010000100000000000011000
000000000000110000001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
0000001111100000000000000001000
```

# Outra Forma de Cálculo: usar o CPI

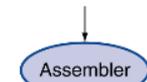
- O CPI nos **permite comparar duas implementações diferentes da mesma arquitetura do conjunto de instruções**, já que o número de instruções executadas para um programa será, obviamente, o mesmo.

- **11<sup>o</sup> Pense e Responda:** dois computadores, A e B, executam a mesma arquitetura de conjunto de instruções. O computador A tem um tempo de ciclo de clock de 250 ps e um CPI de 2,0 para um determinado programa, e o computador B tem um tempo de ciclo de clock de 500 ps e um CPI de 1,2 para o mesmo programa. Qual computador é mais rápido para esse programa e por quanto?

```
swap(int v[], int k)
{int temp;
  temp = v[k];
  v[k] = v[k+1];
  v[k+1] = temp;
}
```



```
swap:
  muli $2, $5,4
  add $2, $4,$2
  lw $15, 0($2)
  lw $16, 4($2)
  sw $16, 0($2)
  sw $15, 4($2)
  jr $31
```



```
000000001010000100000000000011000
00000000000110000001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
0000001111100000000000000001000
```

# A Equação Clássica da Performance da CPU

---

- Podemos reescrever as equações anteriores em termos do **contador de instruções** (número de instruções realizadas por um programa), do **CPI** e do **tempo de ciclo de clock**:

$$\text{CPU time} = \text{Instruction count} \times \text{CPI} \times \text{Clock cycle time}$$

$$\text{CPU time} = \frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}}$$

- Pode ser utilizada para comparar duas implementações diferentes ou para alternativas de projeto!

# CPI em mais detalhes

- **12º Pense e Responda:** Um projetista de compilador está tentando decidir entre duas sequências de código que exigem as seguintes contagens de instruções:

Code sequence	Instruction counts for each instruction class		
	A	B	C
1	2	1	2
2	4	1	1

- Os projetistas de hardware forneceram os seguintes fatos:

	CPI for each instruction class		
	A	B	C
CPI	1	2	3

- Pergunta-se:
  - Qual sequência de código executa mais instruções?
  - Qual sequência de código será mais rápida?
  - Qual é o CPI para cada sequência?

$$\text{CPU clock cycles} = \sum_{i=1}^n (\text{CPI}_i \times C_i)$$

$$\text{CPI} = \frac{\text{CPU clock cycles}}{\text{Instruction count}}$$

# Componentes da Performance x Unidades

Components of performance	Units of measure
CPU execution time for a program	Seconds for the program
Instruction count	Instructions executed for the program
Clock cycles per instruction (CPI)	Average number of clock cycles per instruction
Clock cycle time	Seconds per clock cycle

$$\text{Time} = \text{Seconds/Program} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

- Como determinar esses fatores na equação?
  - Tempo de execução: medindo
  - Tempo de ciclo de clock: documentação
  - Contagem de instruções e CPI: difícil!
    - Ferramentas de software / hardware

# Componentes do Desempenho do Programa

Hardware or software component	Affects what?	How?
Algorithm	Instruction count, CPI	The algorithm determines the number of source program instructions executed and hence the number of processor instructions executed. The algorithm may also affect the CPI, by favoring slower or faster instructions. For example, if the algorithm uses more divides, it will tend to have a higher CPI.
Programming language	Instruction count, CPI	The programming language certainly affects the instruction count, since statements in the language are translated to processor instructions, which determine instruction count. The language may also affect the CPI because of its features; for example, a language with heavy support for data abstraction (e.g., Java) will require indirect calls, which will use higher CPI instructions.
Compiler	Instruction count, CPI	The efficiency of the compiler affects both the instruction count and average cycles per instruction, since the compiler determines the translation of the source language instructions into computer instructions. The compiler's role can be very complex and affect the CPI in varied ways.
Instruction set architecture	Instruction count, clock rate, CPI	The instruction set architecture affects all three aspects of CPU performance, since it affects the instructions needed for a function, the cost in cycles of each instruction, and the overall clock rate of the processor.

# Atenção

---

- A contagem do número de instruções:
  - Depende da arquitetura do conjunto de instruções
  - Não depende da implementação da arquitetura do conjunto de instruções
- O CPI:
  - Depende de diversas coisas no projeto do computador:
    - Estrutura do processador
    - Sistema de memória
    - Mistura de tipos de instruções executados em uma aplicação (e, portanto, da própria aplicação)
    - Implementações diferentes da mesma arquitetura do conjunto de instruções

# Dica

---

- Nos PSETs (e na prova), as questões relativas à performance dos computadores, em geral, apresentarão cenários de dois casos (dois computadores, por exemplo) com alguns parâmetros conhecidos e outros não. Você terá que resolver para os parâmetros desconhecidos.

# Acabou?

---

**Ainda não...**

- Lab 2 (só terminar)**
- PSET 1 (lágrimas de sangue)**
- Lab 3 (próxima aula)**
- PSET 2 (lágrimas de sangue)**
- .....**