Processamento Paralelo

AULA 2

Conceitos Básicos 2

Professor: Luiz Augusto Laranjeira
luiz.laranjeira@gmail.com

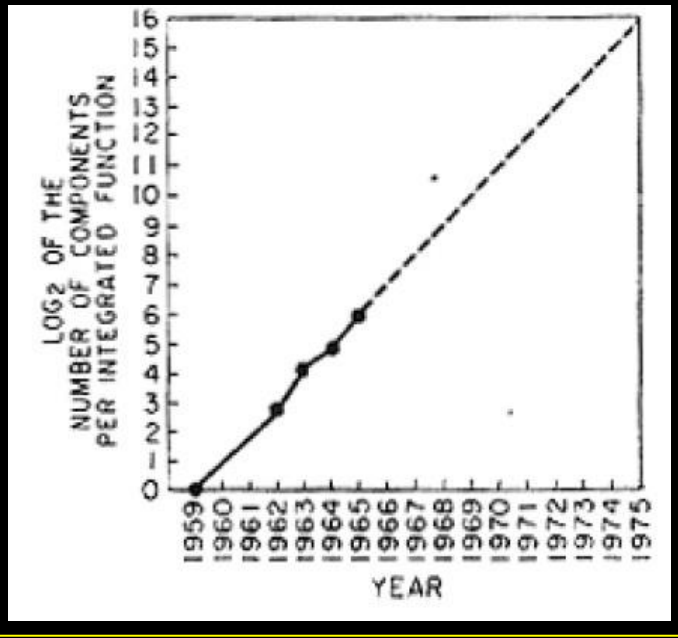
Material originalmente produzido pelo Prof. Jairo Panetta (ITA) e adaptado para a FGA pelo Prof. Laranjeira

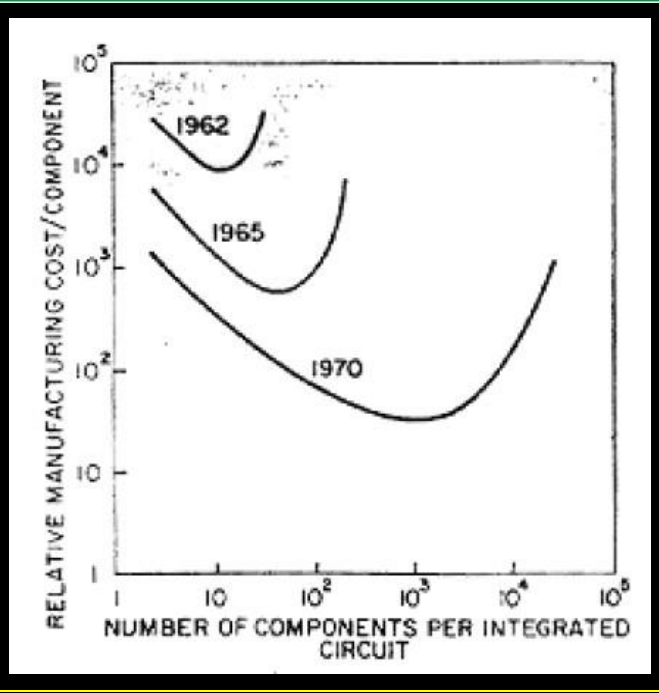
Agenda

- Definição de Paralelismo
- Níveis de Paralelismo
- Métricas de Desempenho Paralelo
- Lei de Amdahl
- Necessidade e Utilidade de Paralelismo
- Lei de Moore
- Memory Wall, Power Wall
- Cray no IME

Porque Paralelismo?
Há Motivos Recentes

“Lei" de Moore





Primeira forma (1965):

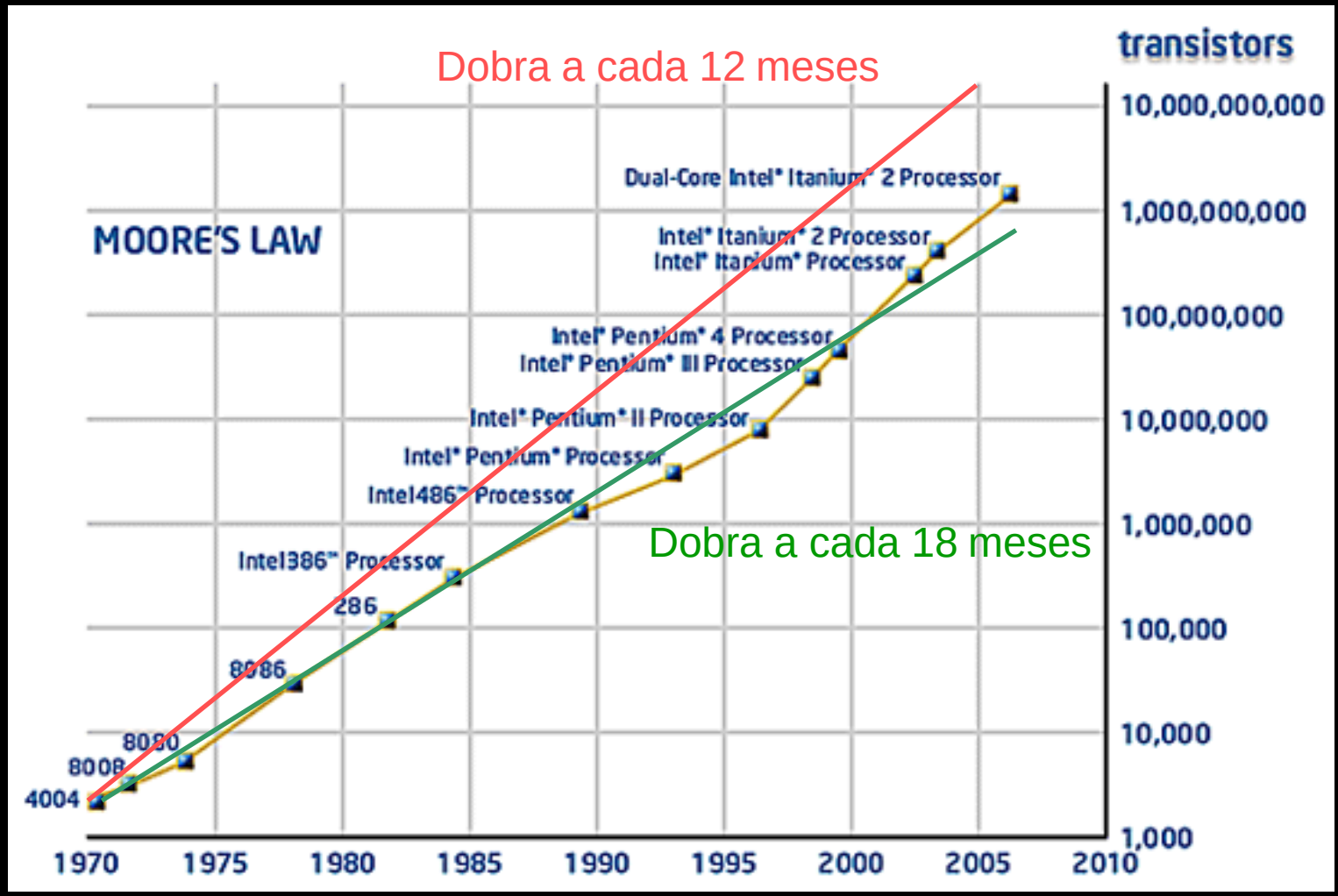
- Nº de componentes p/ circ. integrado minimizando o custo por componente dobra a cada ano
- Observação (não “lei”) no trabalho original
  - Extrapolação baseada em 5 pontos
  - Reconsiderar após 10 anos

Segunda forma (1975):

- Dobra a cada dois anos
- Moore nunca disse 18 meses
- Profecia auto realizável

G. Moore: “Cramming more components onto integrated circuits”, Eletronics, 1965

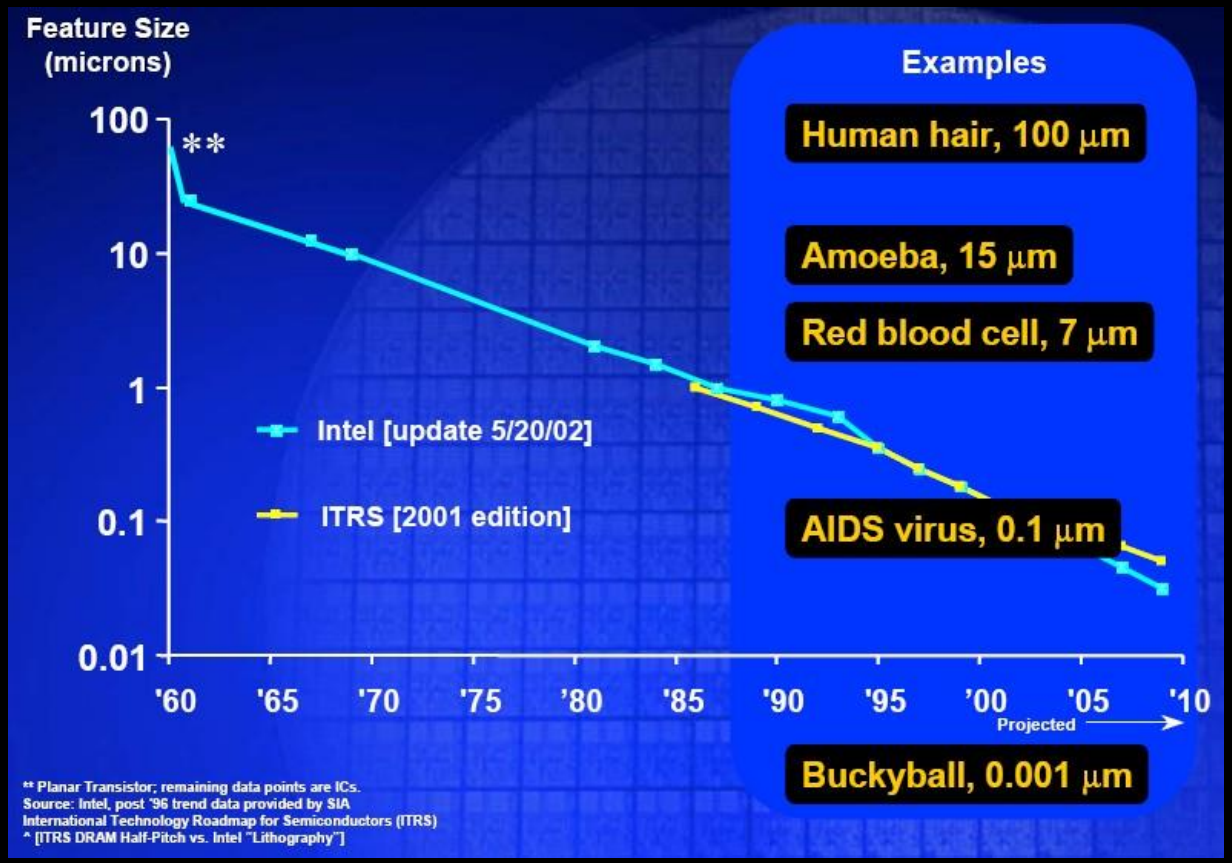
Verificação da Lei de Moore



http://www.intel.com/technology/mooreslaw

Força Motriz da Lei de Moore

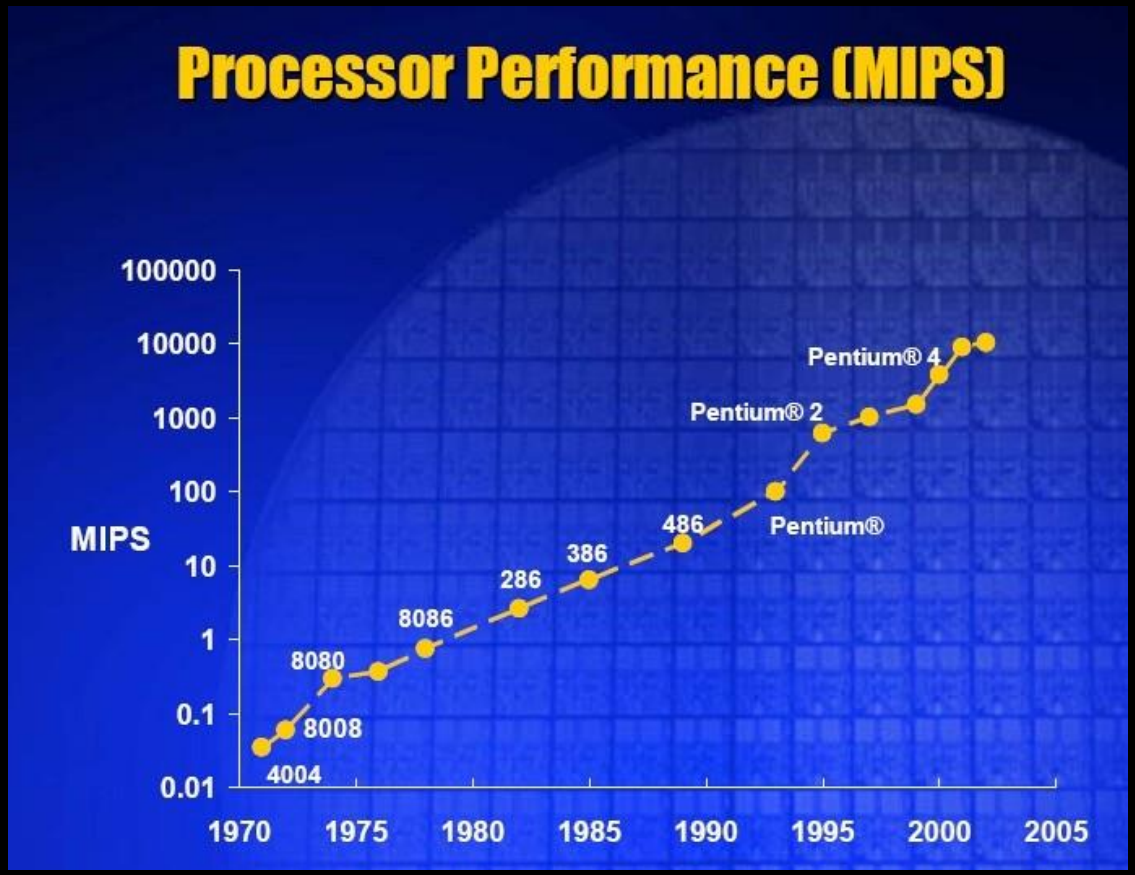
Tecnologia de Litografia Constantemente Reduz Tamanho de Componentes em CI



Gordon Moore, palestra convidada, International Solid-State Circuits Conference (ISSCC) 2003

Até Recentemente

Canalizar aumento no número de componentes para aumentar a freqüência de operação gera CPUs progressivamente mais rápidas



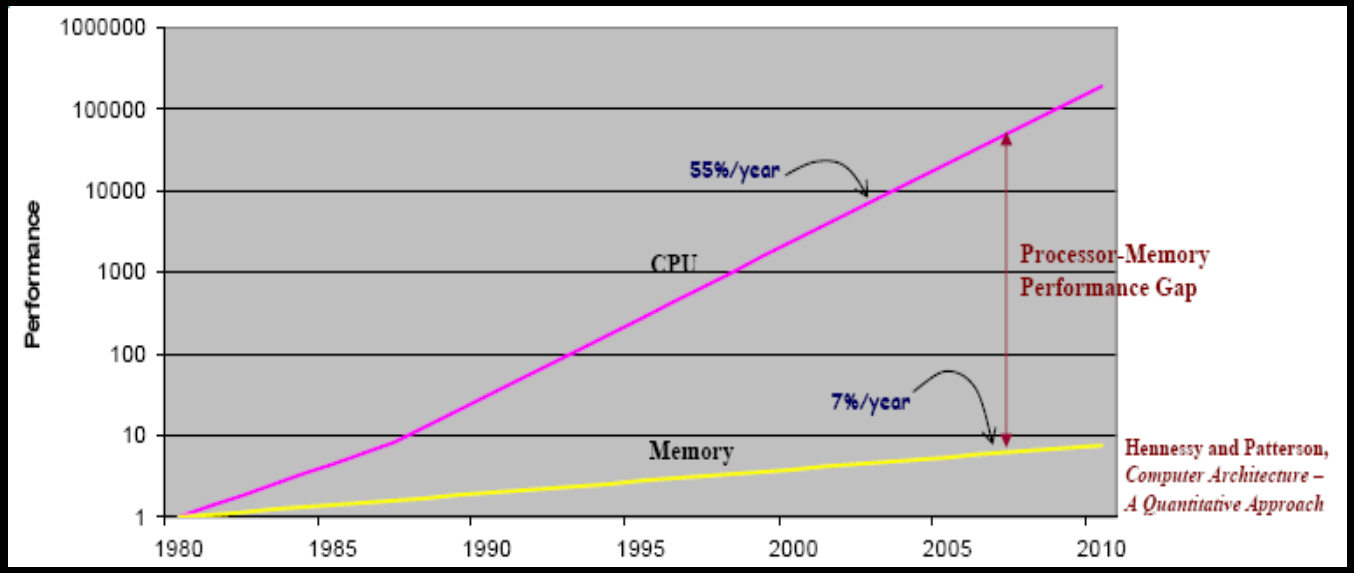
G. Moore, palestra convidada ISSCC 2003

Mas o ganho anual em velocidade de um processador é bem menor que o ganho anual no número de componentes, pois há barreiras

Agenda

- Definição de Paralelismo
- Níveis de Paralelismo
- Métricas de Desempenho Paralelo
- Lei de Amdahl
- Necessidade e Utilidade de Paralelismo
- Lei de Moore
- Memory Wall, Power Wall
- Cray no IME

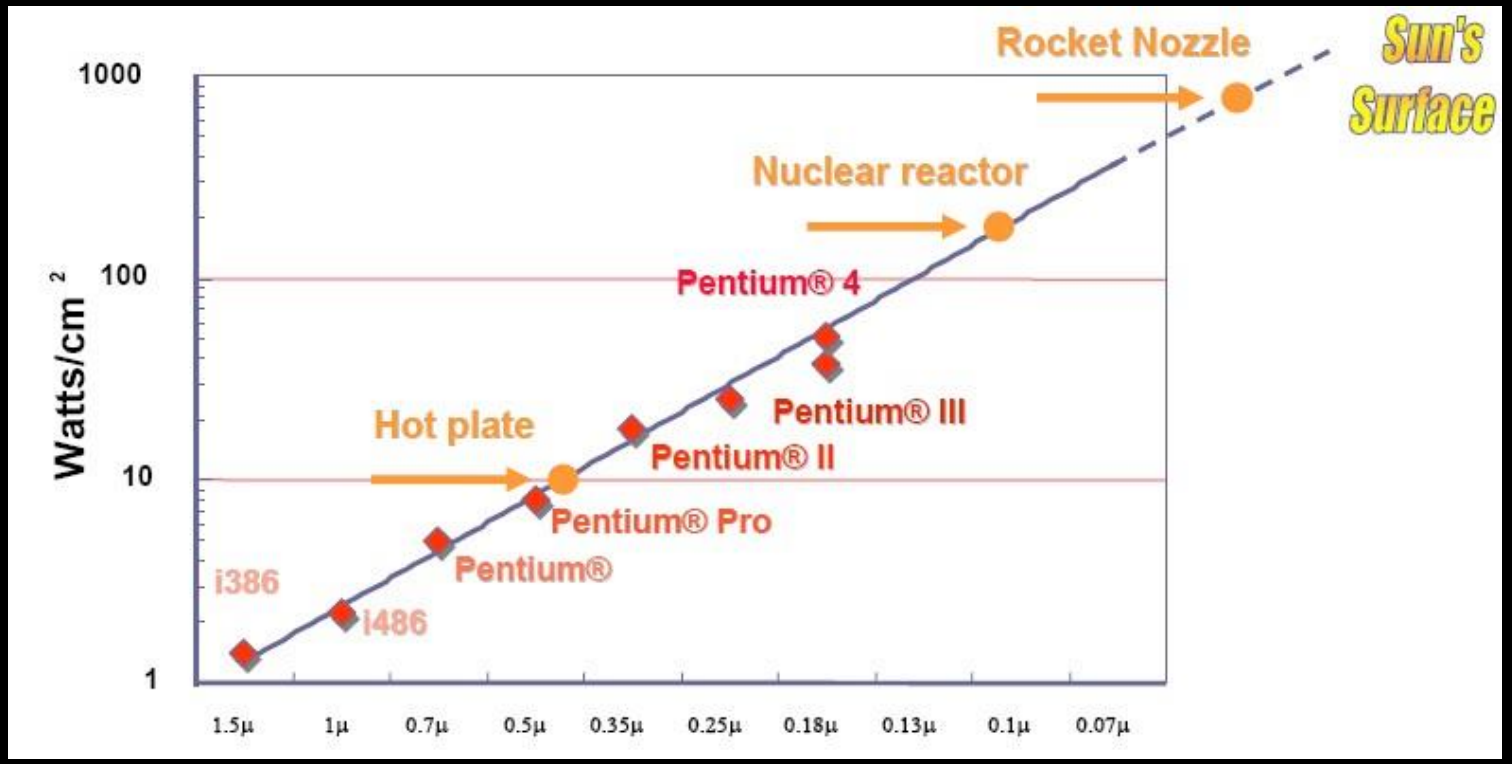
Barreira: Memory Wall



- A velocidade de acesso à memória escala mais lentamente que a velocidade da CPU, ao longo dos anos
- Acesso à memória torna-se o gargalo da eficiência
- Largura de banda (*bandwidth*) vem sendo acomodada (economia)
- Latência (*latency*) é a questão crucial

J. L. Gaudiot, palestra convidada SBAC-PAD 2006

Barreira: Power Wall



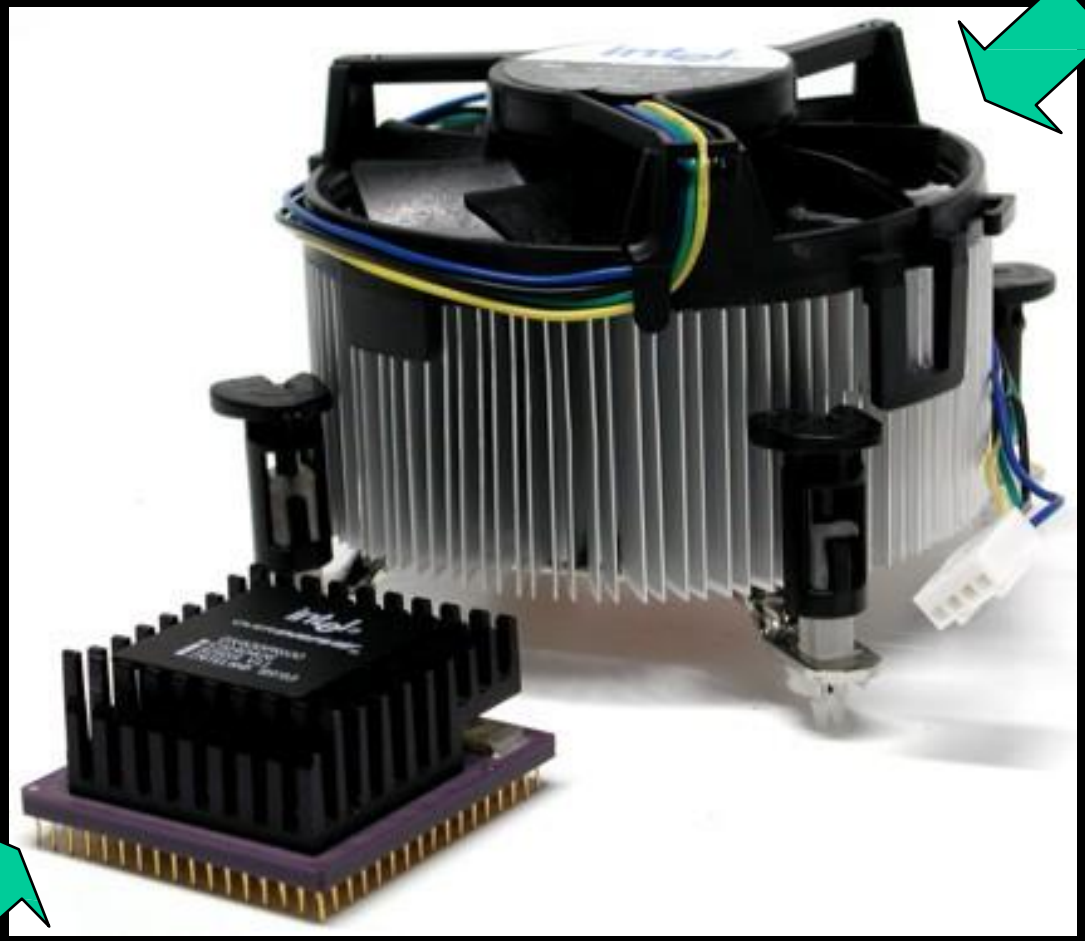
A dissipação (de calor) atingiu níveis intoleráveis

Figura: Fred Pollack, palestra convidada IEEE MICRO 1999

S. Borkar, "Design Challenges of Technology Scaling", IEEE Micro, July 1999

Dissipação de Potência

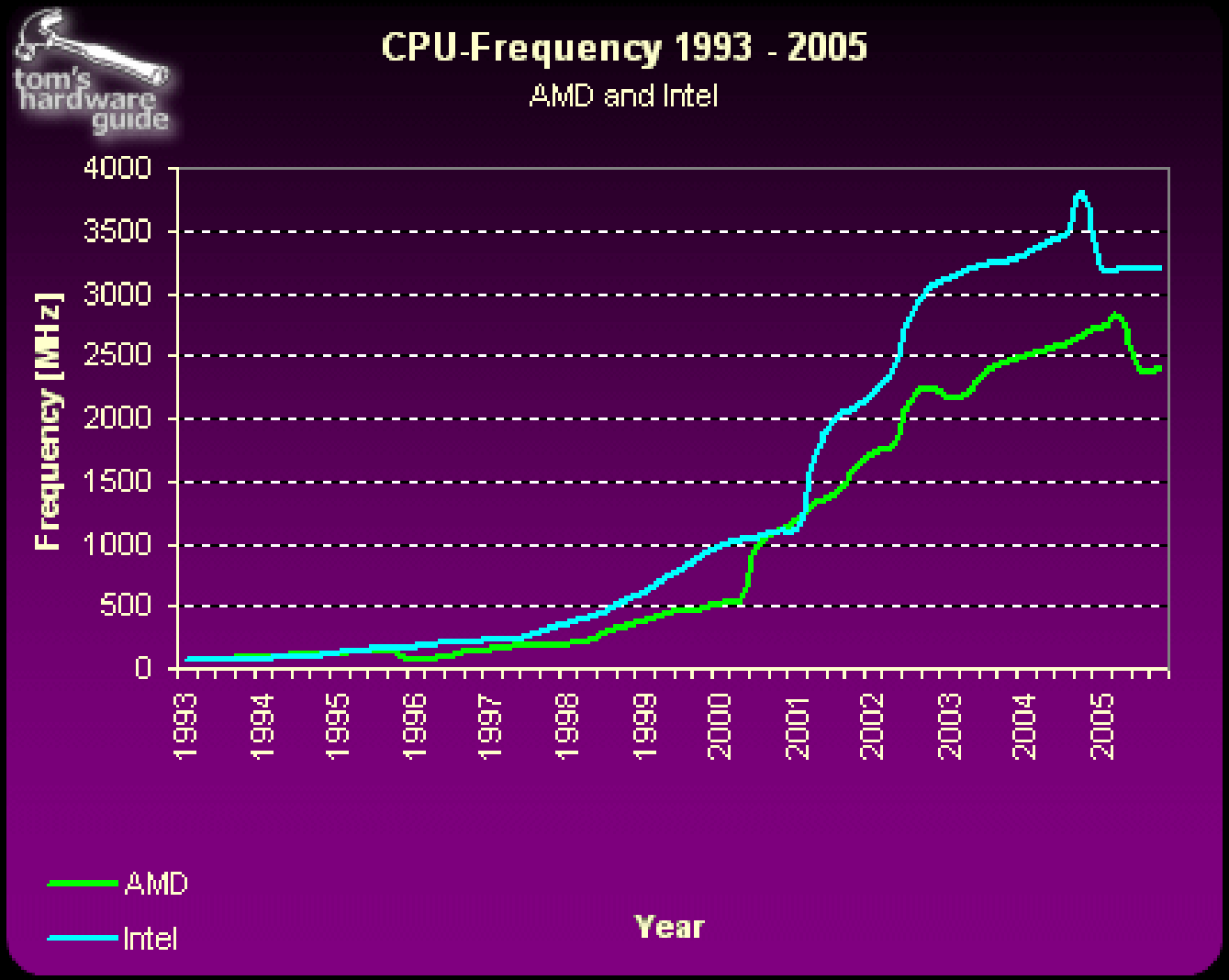
Pentium e Dissipador, 2005



Pentium e Dissipador, 1995

www.tomshardware.com/2005/11/21/the_mother_of_all_cpu_charts_2005

Dissipação de Potência Impede Novo Aumento de Frequência



www.tomshardware.com/2005/11/21/the_mother_of_all_cpu_charts_2005

Em Síntese

- Ida à memória limita aumento da velocidade de programas, pois a freqüência de acesso não escala proporcionalmente à frequência da CPU e a latência menos ainda (*memory wall*)
- Dissipação térmica atinge nível absurdo (*power wall*)
- E há outras barreiras...

Paralelismo: Novos Motivos

Como usar maior número de componentes para gerar máquinas mais rápidas?

Aumentar a freqüência não é mais possível.

Tendência clara:
Múltiplas CPUs de menor freqüência no mesmo chip

Paralelismo: Novos Motivos

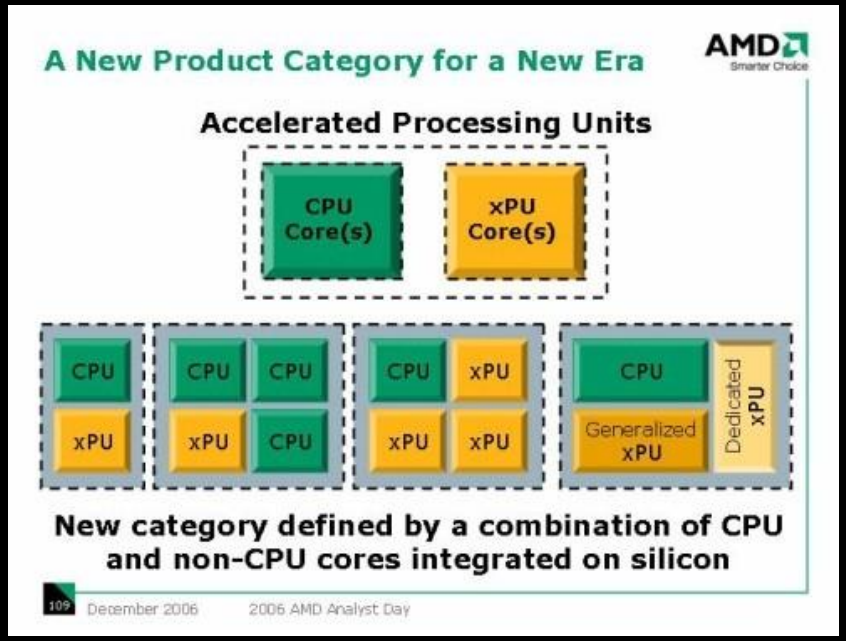
Power and Heat: Intel Embraces Multicore

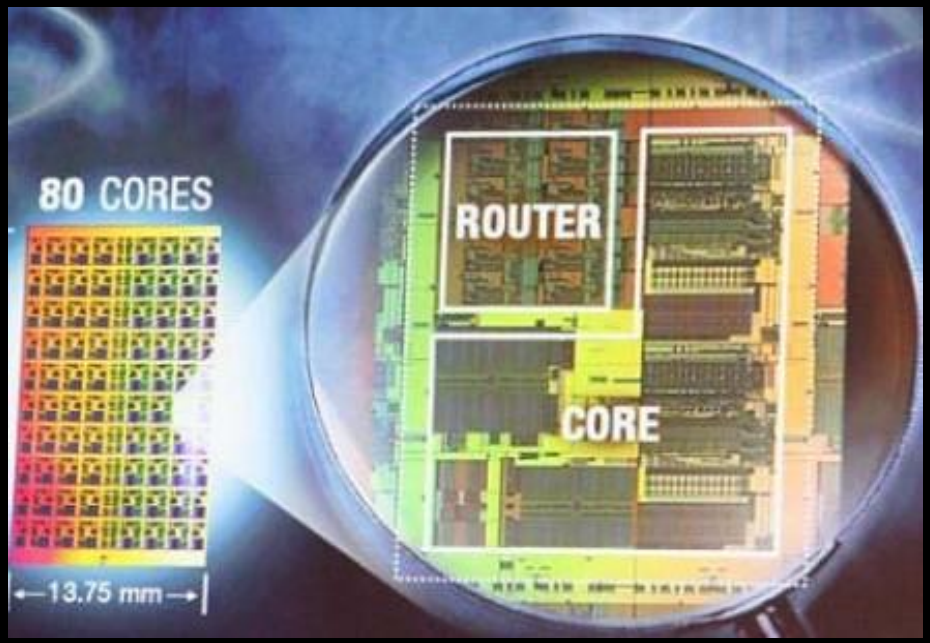
May 17, 2004 … Intel, the world's largest chip maker, publicly acknowledged that it had hit a "thermal wall" on its microprocessor line. As a result, the company is changing its product strategy and disbanding one of its most advanced design groups. Intel also said that it would abandon two advanced chip development projects …

Now, Intel is embarked on a course already adopted by some of its major rivals: obtaining more computing power by stamping multiple processors on a single chip rather than straining to increase the speed of a single processor … Intel's decision to change course and embrace a "dual core" processor structure shows the challenge of overcoming the effects of heat generated by the constant on-off movement of tiny switches in modern computers … some analysts and former Intel designers said that *Intel was coming to terms with escalating heat problems so severe they threatened to cause its chips to fracture at extreme temperatures*…

New York Times, May 17, 2004

Por enquanto, poucas CPUs por chip.





Forte tendência de muitas CPUs no mesmo chip, idênticas ou não, com uma ou múltiplas threads

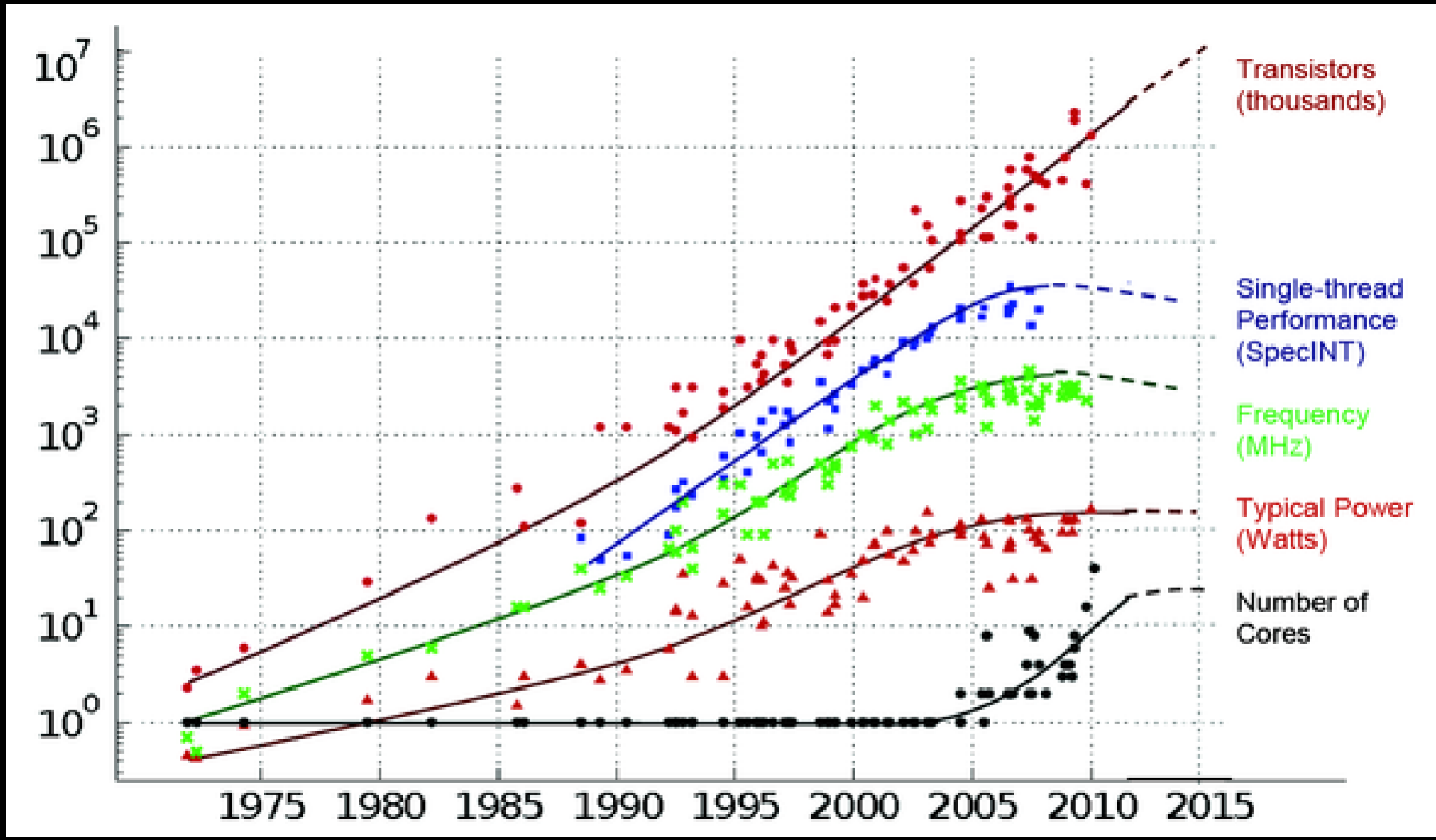
Fonte: INTEL eAMD

Porque Paralelismo: Conclusão

Paralelismo (uso simultâneo de múltiplas CPUs em um programa) é uma tecnologia disponível que reduz a distância entre as necessidades dos usuários e a velocidade de uma única CPU

Mas também é a forma viável de converter o aumento no número de componentes da Lei de Moore em aumento da velocidade de processamento

Frequência x Núcleos



Karl Rupp, 40 Years of Microprocessor Trend Data

Power Boost

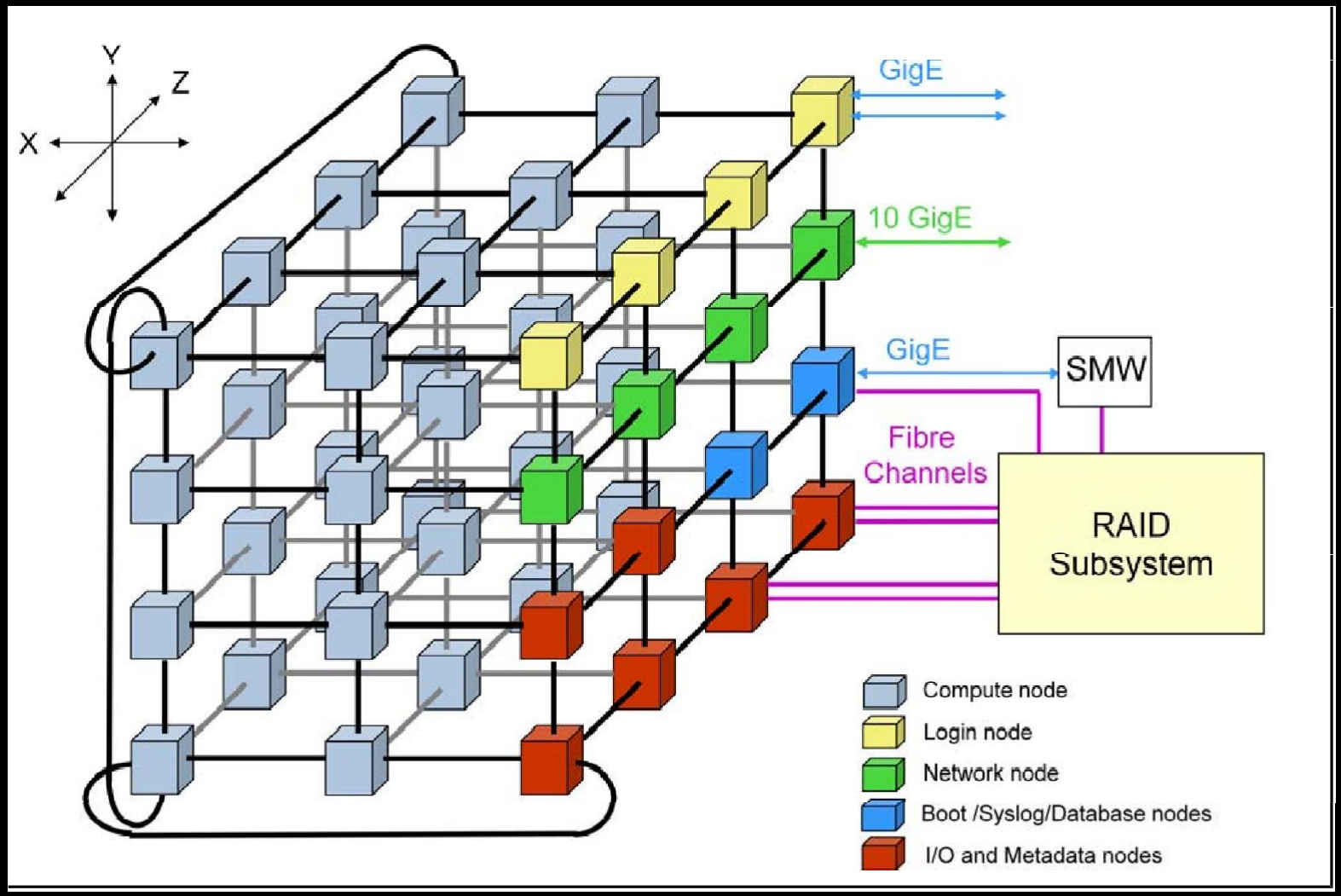
- Há recentes lançamentos de chips “multicore” com frequências similares às empregadas em 2004.
  - AMD Opteron 6276 do IME @ 2300MHz
  - Intel Xeon E5-1650 da Petrobras @ 3200 MHz
- Porque os novos chips não derretem?
- As frequências indicadas são as máximas atingíveis operacionalmente
- Sensores de energia e temperatura permitem desligar trechos do chip e reduzir ou aumentar a frequência de operação dinamicamente (“power boost”)
  - Por exemplo, utiliza frequência máxima quando apenas um núcleo está em operação, porém a frequência é reduzida quando múltiplos núcleos operam simultaneamente
  - Impacta ”speed-up” fortemente

Agenda

- Definição de Paralelismo
- Níveis de Paralelismo
- Métricas de Desempenho Paralelo
- Lei de Amdahl
- Necessidade e Utilidade de Paralelismo
- Lei de Moore
- Memory Wall, Power Wall
- Cray no IME

Cray XK6 Nano do IME

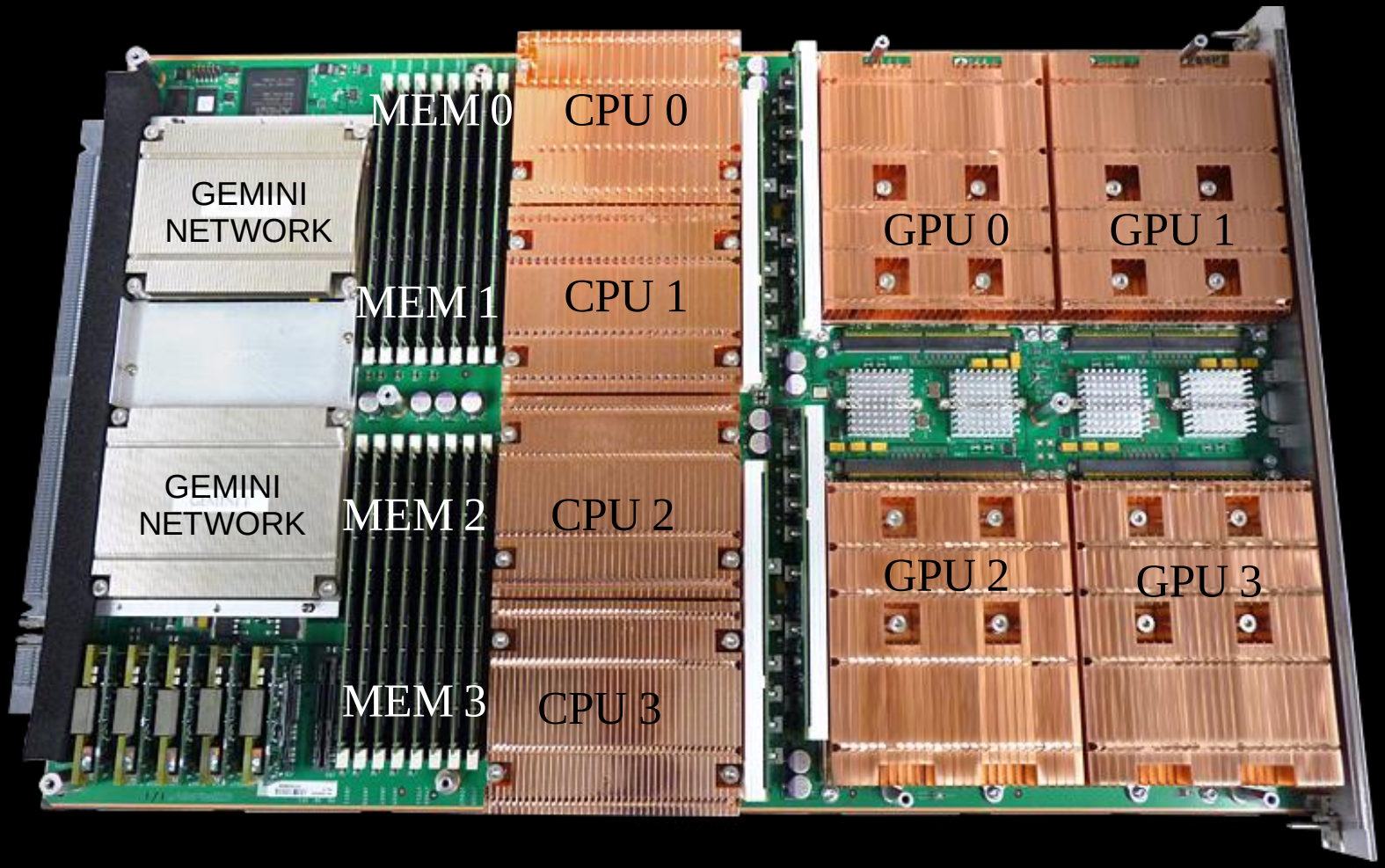
Cray XK6 Nano @ IME



Cray XK6 Nano @ IME

- 40 Nós Computacionais, cada um com:
  - CPU AMD Opteron 6276 com 16 núcleos
  - GPU NVIDIA Tesla K20
  - 32GB Memória Central
- 4 Nós de Login
- Rede de Comunicações Gemini (10 Gbit/s Ethernet)
  - 4 nós de rede
  - Rede conecta todos os nós
- Sistema de Arquivos Paralelo com 60TB
  - Lustre, visível de todos os nós computacionais

Computer Blade: 4 Nós Computacionais



Uso do Cray

- Contas e Acesso
  - Vide pessoal do IME
- Sistema Operacional Linux
  - Completo nos *nós de login*
  - Reduzido nos *nós computacionais*
- Ações:
  - *Nós computacionais* servem para executar programas
    - Não são acessíveis
  - *Nós de login* não devem executar programas. Servem para:
    - Trabalhar no sistema de arquivos
    - Utilizar as ferramentas do “bash shell” (editor, compilador, etc)
    - Enviar programas para execução nos “compute nodes”

Compilação de Programas

- O sistema de software da Cray utiliza os comandos *cc* e *ftn* para compilar programas
  - O sistema de software da Cray mapeia cc e ftn nos compiladores escolhidos e carregados pelo usuário

- O software *module* gerencia a carga das ferramentas de software
  - Ajusta todas as variáveis de ambiente (incluindo trajetórias) necessárias para o correto funcionamento das ferramentas

- Alguns comandos do module (vide *man module*):
  - *module list* lista os módulos atualmente carregados
  - *module avail* lista todos os módulos disponíveis
  - *module load* carrega um módulo
  - *module rm* remove um módulo
  - *module swap* troca os módulos selecionados
    - module swap PrgEnv-cray PrgEnv-gnu

Execução de Programas

- Para executar um programa é necessário utilizar duas ferramentas:
  - *Batch scheduler*: impõe politica de uso da máquina ao controlar filas de tarefas
  - *Job dispatcher*: controla o uso dos ”compute nodes”, escolhendo quais nós utilizar e disparando copias do programa nos nós escolhidos
- Utilizo um único script para acessar as duas ferramentas
- O *batch scheduler* é comandado por um script PBS (Moab)
- No interior do script PBS, solicito o uso dos nós computacionais pelo *comando aprun*
  - Interface do Application Level Placement Scheduler (ALPS), que gerencia o uso dos nós computacionais

Exemplo

- Arquivo script qsub.sh para submeter programa Poly.exe:

```
#!/bin/bash
#PBS -l mppwidth=1
#PBS -l mppdepth=1
#PBS -l nppnppn=1
#PBS -N P1
#PBS -j oe
#PBS -o Poly_1.out
#PBS -q workq

ulimit -s unlimited
aprun -b -n 1 -d 1 -N 1 /home/panetta/Poly.exe
```

PBS

UNIX no login node

- Para submeter o script ao PBS, use o comando "qsub qsub.sh" nos nós de login

- Para acompanhar a execução, use o comando PBS "qstat", que lista os jobs em execução e seu estado

Tópicos Abordados

- Definição de Paralelismo
- Níveis de Paralelismo
- Métricas de Desempenho Paralelo
- Lei de Amdahl
- Necessidade e Utilidade de Paralelismo
- Lei de Moore
- Memory Wall, Power Wall
- Cray no IME

Exercício

Jogo da Vida

Jogo da Vida

- O Jogo da Vida, criado por John H. Conway, utiliza um autômato celular para simular gerações sucessivas de uma sociedade de organismos vivos.

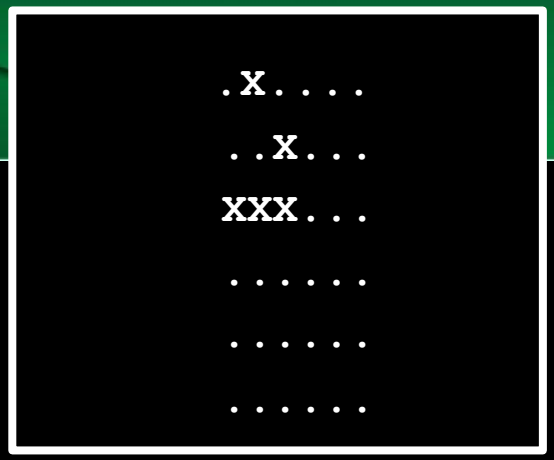
- É composto por:
  - um tabuleiro bi-dimensional de células, individualmente classificadas como vivas ou mortas
  - regras que estabelecem o próximo estado de cada célula

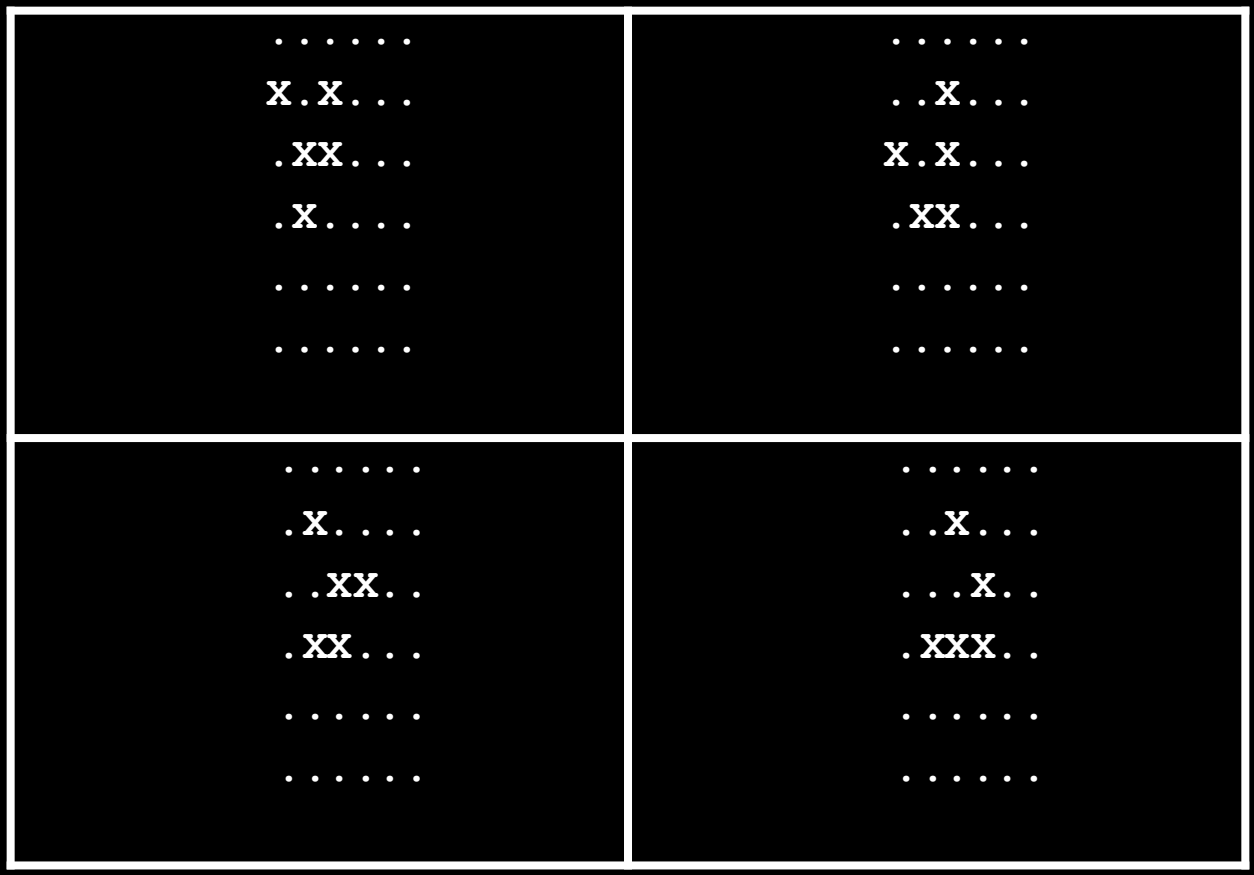
- Sociedade evolui de uma geração para a próxima aplicando simultaneamente as regras a todas as células do tabuleiro

Jogo da Vida

- Cada célula tem exatamente oito células vizinhas

- As regras de evolução são:
  - Células vivas com menos de 2 vizinhas vivas morrem por abandono;
  - Células vivas com mais de 3 vizinhas vivas morrem de superpopulação;
  - Células mortas com exatamente 3 vizinhas vivas tornam-se vivas;
  - As demais células mantêm seu estado anterior.





Exercício

- Forneço:
  - Fonte sequencial em C
    - Dois programas principais, um para verificar a correção e outro para medir tempos de execução
  - Método para medir o tempo de execução (“wall clock”)
  - Makefile e script de execução no Cray
- Solicito:
  - Compile e execute o Jogo da Vida no Cray
  - Meça o tempo de execução de trechos do segundo programa principal para diversos tamanhos do tabuleiro
  - Obtenha a complexidade assintótica do Jogo da Vida e contraste com os tempos de execução obtidos

Detalhes do Script PBS

PBS controla filas ”batch”, impondo política de uso da maquina

#!/bin/bash
#PBS -l mppwidth=Numero total de processos
#PBS -l mppdepth=Núcleos por processo
#PBS -l nppnppn=Processos por nó
#PBS -N Nome do job (aparece em qstat)
#PBS -j oe Colapsa stdout e stderr em um arquivo
#PBS -o Nome do arquivo de saída
#PBS -q Nome da fila batch (workq, gpu_queue)

Detalhes do Comando "aprun"

APRUN dispara executáveis nos nós computacionais

aprun -b -n <*int*> -d <*int*> -N <*int*> <*executável*> <*args executável*>

- -b: não copia o executável para os nós
  - Reduz o tempo de inicialização quando o executável está no sistema de arquivos visível aos nós
- -n: numero de copias do executável
- -N: numero de executáveis por nó
- -d: numero de núcleos por executável