ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES II

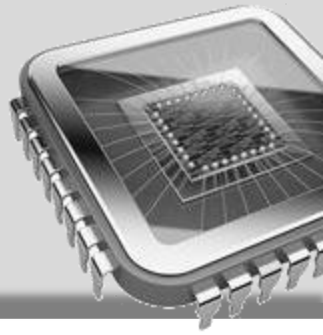
AULA 06: PROGRAMAÇÃO EM MÁQUINAS PARALELAS

Prof. Max Santana Rolemberg Farias
max.santana@univasf.edu.br
Colegiado de Engenharia de Computação

GESE
Grupo de Engenharia de Sistemas Embarcados

CEComp
Engenharia da Computação
UNIVASF

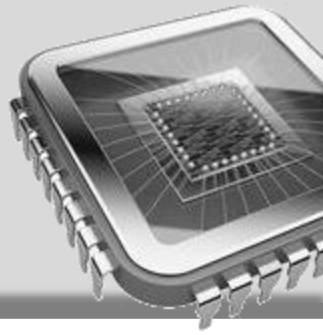
PROGRAMAÇÃO PARALELA



- Há vários modelos de programação paralela em uso atualmente:
  - Memória compartilhada
  - Threads
  - Passagem de mensagens
  - Dados paralelos
  - Híbridos

- Modelos de programação paralela existem como uma **abstração acima da arquitetura de hardware e de memória**.

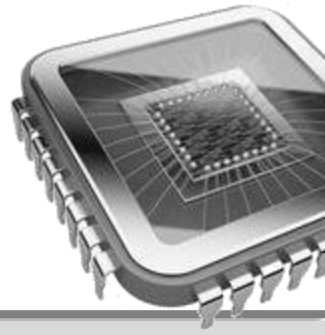
PROGRAMAÇÃO PARALELA



- Apesar de não ser evidente, os modelos não são específicos para um determinado tipo de arquitetura.
  - Pode ser implementado em qualquer hardware

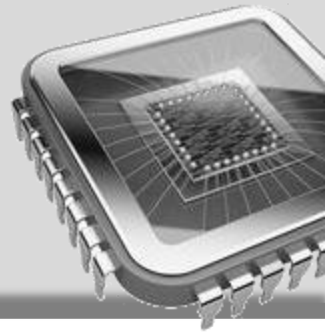
- Não existe o melhor modelo. No entanto há melhores implementações de alguns modelos sobre outros.

Qual modelo utilizar?



MODELO DE MEMÓRIA COMPARTILHADA

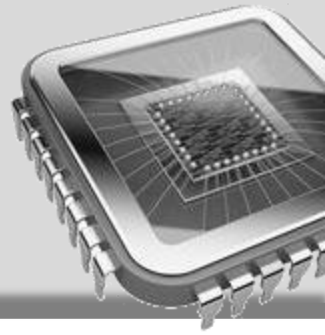
MODELO DE MEMÓRIA COMPARTILHADA



- As tarefas compartilham um espaço de endereço comum.
  - As leituras e escritas podem ser simultaneamente.

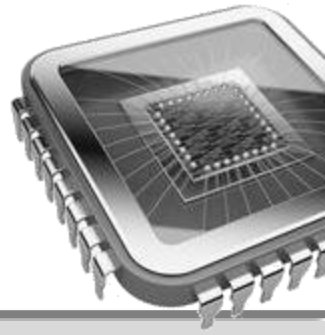
- Geralmente são utilizados mecanismos do tipo semáforos para controlar o acesso à memória compartilhada.
  - Uma vantagem para o programador.
  - Não existe a necessidade de comunicação explicita de dados entre as tarefas.
  - Dificuldade de entender e gerenciar a localidade dos dados.

MODELO DE MEMÓRIA COMPARTILHADA



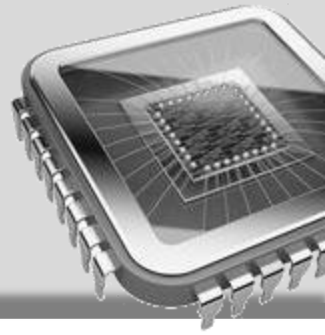
- Precisam manter os dados locais no processador.
  - Para minimizar os acessos à memória que ocorrem quando múltiplos processadores utilizam o mesmo dado.

- Os SMPs (multicore) representam uma implementação comum de uma arquitetura de memória compartilhada.



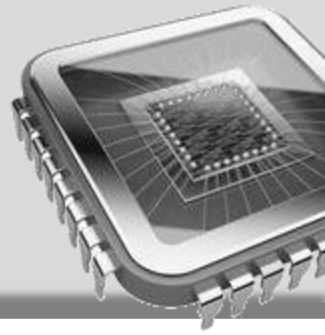
MODELO DE THREADS

MODELO DE THREADS



- No modelo de threads, cada processo (tarefa) pode ter múltiplos caminhos de execução concorrente.
  - O programa principal é escalonado para ser executado pelo SO.
  - O programa principal é executado sequencialmente, mas cria tarefas (threads) que podem ser escalonadas e executadas simultaneamente pelo SO.
  - Cada thread tem dados locais e compartilha recursos como o programa principal.
    - Isso diminui o overhead associado com a replicação dos recursos entre vários processos.
    - Cada thread tem uma visão global de memória, para compartilhar o espaço de memória com o programa principal.

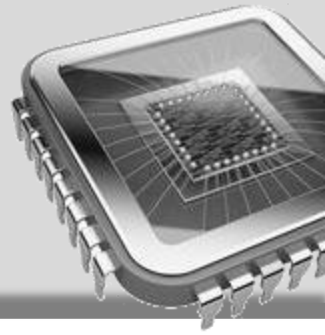
MODELO DE THREADS



- As threads se comunicam umas com as outras por meio da memória global.
  - Isso requer sincronização para garantir a consistência dos dados durante a execução das threads.

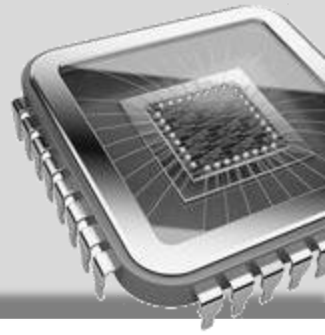
- As threads podem ser criadas ou destruídas, mas o programa principal continua ativo.
  - Para fornecer os recursos compartilhados necessários até que a tarefas seja concluídas

MODELO DE THREADS



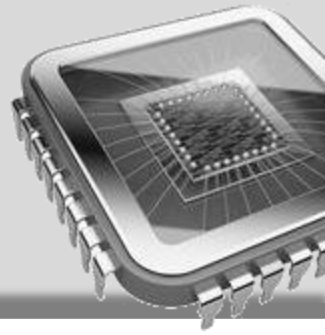
- Threads são comumente associadas com arquiteturas de memória compartilhada e sistemas operacionais.
  - Uma biblioteca de sub-rotinas que são chamadas a partir do código fonte paralelo
  - Um conjunto de diretivas de compilador no código paralelo ou serial.
  - O programador é responsável por determinar todo o paralelismo

MODELO DE THREADS

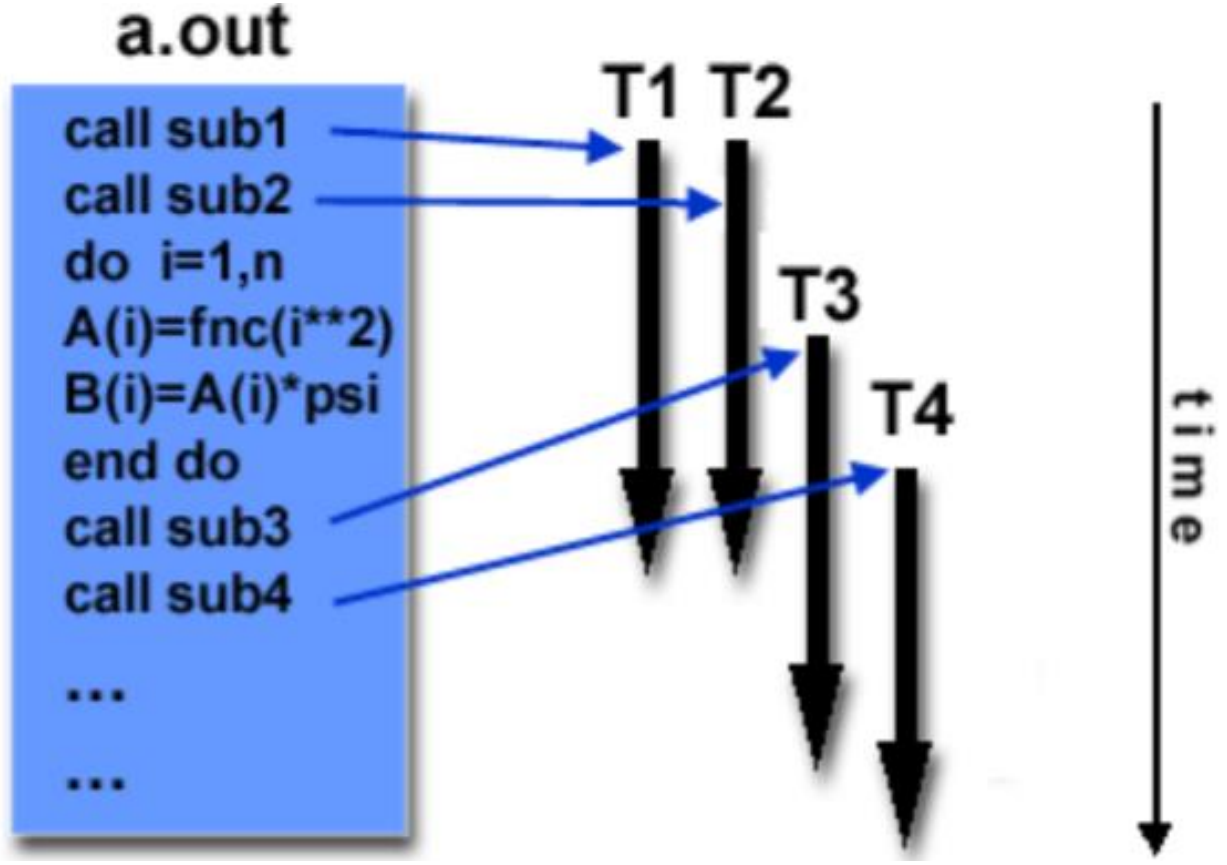


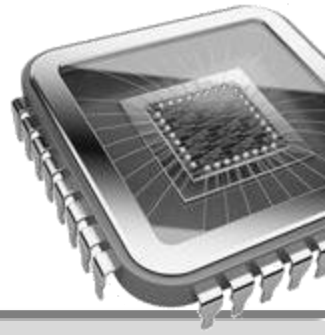
- **OpenMP**
  - Padrão definido e apoiado pelos maiores fabricantes de hardware e software do mundo.
  - API OpenMP Fortan (28/10/1957)
  - API OpenMP C/C++ (Final de 1998)
  - Portável e multiplataforma (Unix e Windows)
  - É fácil e simples de usar
  - A Microsoft tem sua própria implementação de threads
    - Não relacionada com o padrão Unix POSIX ou OpenMP

MODELO DE THREADS



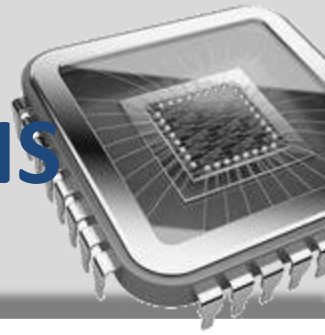
- Posix Threads
  - Especificado pelo padrão IEEE POSIX 1003.1c (1995)
  - Somente linguagem C
  - Conhecido como Pthreads
  - A maioria dos fabricantes de hardware já fornecem pthreads
  - Paralelismo explícito
    - Reque muita atenção do programador para detalhes de codificação


a.out
call sub1
call sub2
do i=1,n
A(i)=fnc(i**2)
B(i)=A(i)*psi
end do
call sub3
call sub4
...
...
T1
T2
T3
T4
time



MODELO DE PASSAGEM DE MENSAGENS

MODELO DE PASSAGEM DE MENSAGENS

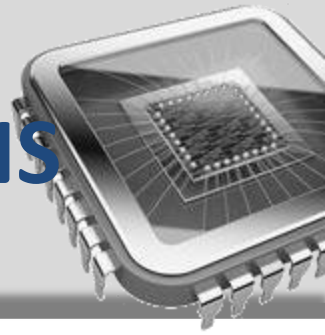


- Conjunto de tarefas que utilizam a memória local durante a computação.
  - Várias tarefas podem residir na mesma máquina.

- As tarefas trocam dados através de comunicação, enviando e recebendo mensagens

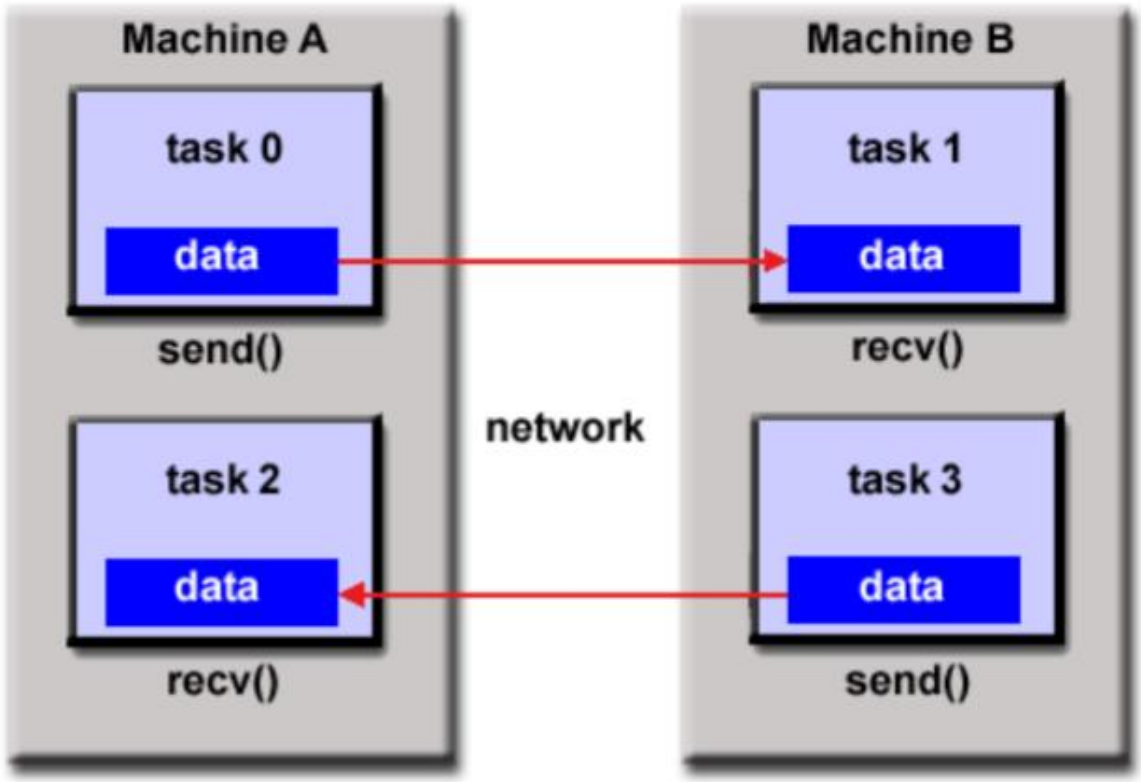
- A transferência de dados em geral requer operações de cooperação a serem executadas por cada processo
  - Operação de envio deve ter uma operação de recepção correspondente.

MODELO DE PASSAGEM DE MENSAGENS

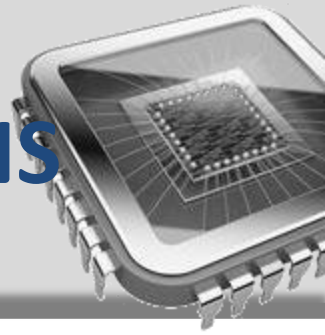


- O programador é responável por determinar todo o paralelismo.
- Várias bibliotecas disponíveis desde a década de 1980
  - Muitas diferenças. Dificuldade de desenvolver aplicações portáveis
- MPI
  - Padrão da indústria para ambientes de passagem de mensagens.

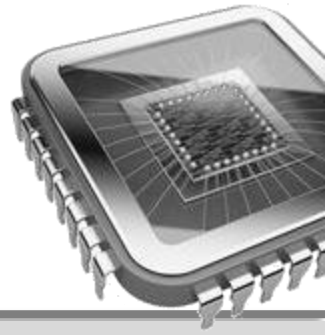
MODELO DE PASSAGEM DE MENSAGENS



MODELO DE PASSAGEM DE MENSAGENS

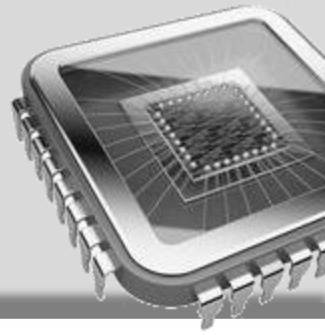


- Para arquitetura de memória compartilhada, as implementações do MPI geralmente não utilizam redes para a comunicação entre as tarefas.
  - Ao invés disso é utilizada a memória compartilhada (cópia de memória) por razões de desempenho.



MODELO DE PARALELO DE DADOS

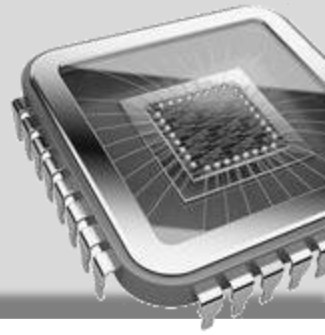
MODELO PARALELO DE DADOS



- A maioria das tarefas paralelas envolve a execução de operações em um conjunto de dados.
  - Os dados são geralmente organizados em uma estrutura, como uma matriz ou um cubo.

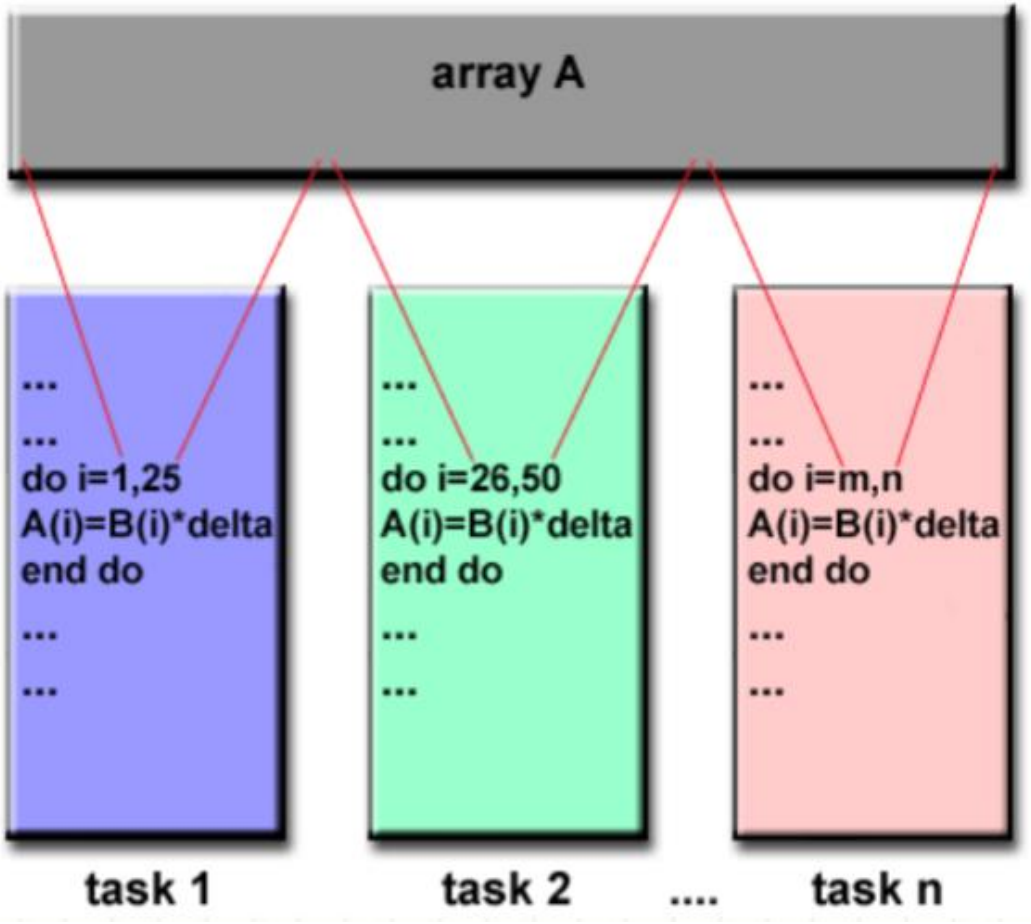
- Um conjunto de tarefas trabalham coletivamente na mesma estrutura de dados.
  - No entanto, cada tarefa opera em uma partição diferente da mesma estrutura de dados.

MODELO PARALELO DE DADOS

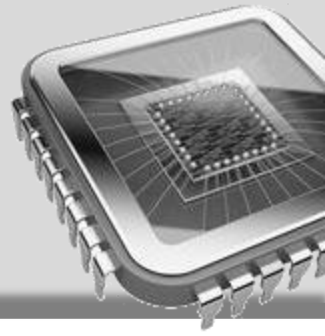


- Tarefas executam a mesma operação em sua partição de trabalho.
  - Adicionar valores a cada elemento de um array
    - Em arquiteturas de memória compartilhada, todas as tarefas podem ter acesso à estrutura de dados armazenada na memória global.
    - Em arquitetura de memória distribuída, a estrutura de dados é dividida e distribuída na memória local de cada processador.

MODELO PARALELO DE DADOS

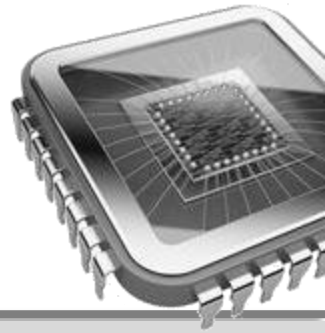


MODELO PARALELO DE DADOS IMPLEMENTAÇÃO



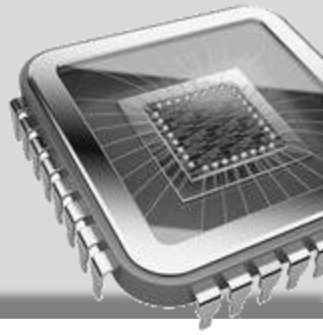
- Escreve-se um programa definindo a organização paralela dos dados

- A organização pode ser definida por uma biblioteca de dados paralelos, ou por diretivas do compilador.



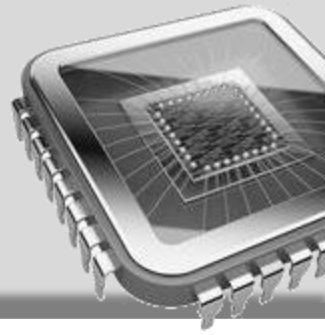
OUTROS MODELOS

OUTROS MODELOS



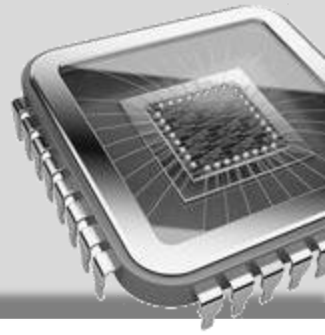
- Além dos modelos já mencionados há outros modelos de programação paralela. Três dos mais comuns são descritos:
  - Híbrido
  - SPMD
  - MPMD

OUTROS MODELOS
HÍBRIDO



- Combinação de dois ou mais modelos de programação paralela
  - Combinação do modelo de passagem de mensagens (MPI) com o modelo de threads (POSIX Threads) ou o modelo de memória compartilhada (OpenMP)
  - Combinação de dados paralelos com passagem de mensagens em arquitetura de memória distribuída (utiliza-se passagem de mensagens para transmitir dados entre as tarefas, de forma transparente para o programador).

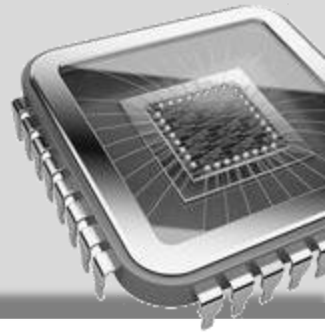
OUTROS MODELOS
SINGLE PROGRAM MULTIPLE DATA (SPMD)



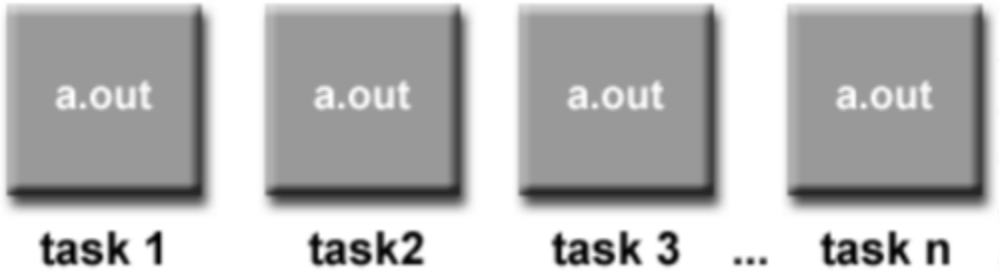
- Modelo de programação de alto nível que pode ser construído a partir da combinação dos modelos de programação paralela previamente mencionados.

- O processamento é colaborativo entre todas as máquinas.
- O código executado em todas é o mesmo (single program)
- Os dados (multiple data) são trocados de modo coordenado entre os processos

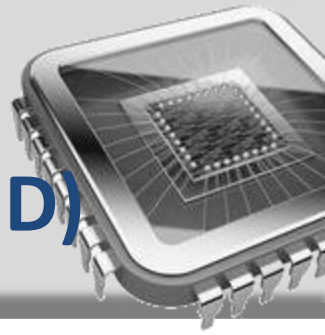
OUTROS MODELOS
SINGLE PROGRAM MULTIPLE DATA (SPMD)



- Programsa SPMD geralmente possuem a lógica necessária em seu programa para permitir que diferentes tarefas se ramifiquem ou condicionalmente executem apenas as partes do programa que se destinam a executar.
  - Todas as tarefas podem utilizar diferentes dados.



OUTROS MODELOS
MULTIPLE PROGRAM MULTIPLE DATA (MPMD)



- Modelo de programação de alto nível que pode ser construído a partir da comunicação dos modelos de programação paralela.
  - Um único processo chamado mestre controla todas as tarefas que serão dadas para os outros processos.
  - Todos podem executar o mesmo programa ou diferentes programas.