9. Multiprocessadores

- **Processamento paralelo ➔ um programa sendo executado por múltiplos processadores simultaneamente.**

- **Questões básicas para projeto de sistemas multiprocessados:**
  - **Compartilhamento de dados;**
  - **Coordenação entre os processadores;**
  - **Quantidade de processadores;**

- **Compartilhamento de dados**
  - **Processadores com um único espaço de endereçamento**
    - **Processadores de memória compartilhada.**
      - **UMA – uniform memory access ou SMP – symmetric multiprocessors**
      - **NUMA – nonuniform memory access**

- **Processadores de memória privada.**

  - **Troca de mensagens para comunicação entre processos ➔ send e receive**

- **Novo modelo de paralelismo ➔ clusters ➔ computadores ligados por uma rede.**

Categoria	Tipo		Número de processadores
Modelo de Comunicação	Troca de Mensagens		8-256
	Endereçamento compartilhado	NUMA	8-256
		UMA	2-64
Conexão Física	Rede		8-256
	Bus		2-32

- **Programação de multiprocessadores**

  - **Speedup**

  Exemplo

  Suponha que queiramos achar o speedup linear com 100 processadores. Que fração da computação original pode ser seqüencial ?

Solução

Tempo de execução após melhora = (tempo de execução afetado pela melhora / quantidade de melhora) + tempo de execução não afetado

Tempo de execução após melhora / 100 = (tempo de execução afetado pela melhora / 100) + tempo de execução não afetado

Tempo de execução não afetado = (Tempo de execução após melhora / 100) - (tempo de execução afetado pela melhora / 100)

Exemplo

Suponha que queremos fazer duas somas: uma de duas variáveis escalares e uma de duas matrizes 1000 X 1000. Qual o speedup para 100 processadores ?

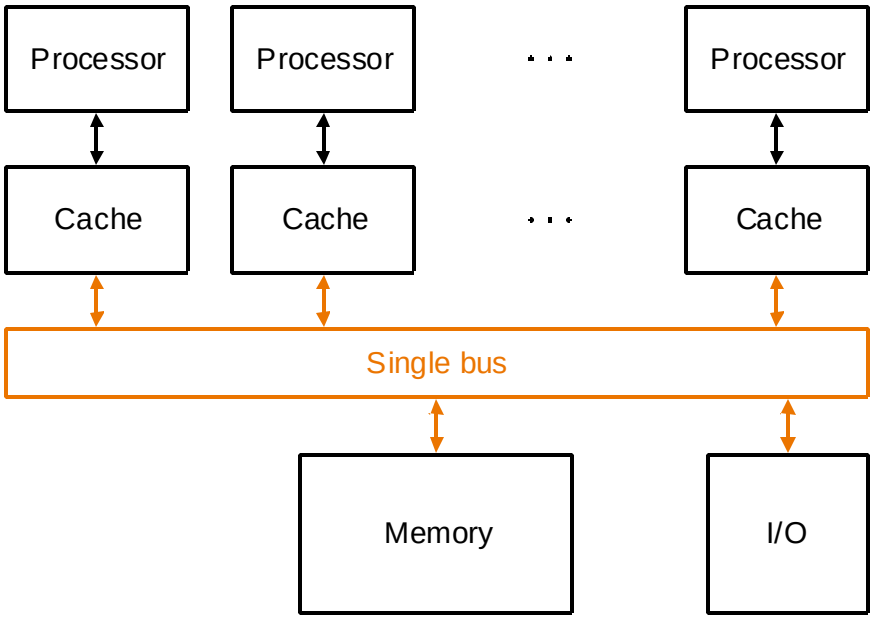
Solução

Se assumirmos que a performance é função do tempo de adição *t,* então existe 1 adição não beneficiada pelo paralelismo e 1.000.000 que são ➔ tempo antes = 1.000.001

Tempo de execução depois melhora = (tempo de execução afetado pela melhora / quantidade de melhora) + tempo de execução não afetado

Tempo de execução depois melhora = (1.000.000/100)*t* + 1*t* = 1.001*t*
Speedup = 1.000.001/1.001 = 999

- **Multiprocessadores conectados por um barramento simples**



- **Computadores com múltiplos processadores conectados em um único barramento backplane**

Nome	# max. de proc	Nome do proc.	Clock rate MHz	Max. mem/sist MB	Bandwidth max/sist MB/seg
Compaq Proliant 5000	4	Pentium Pro	200	2.048	540
Digital AlphaServer 8400	12	Alpha 21164	440	28.672	2.150
HP 9000 K460	4	PA-8000	180	4.096	960
IBM RS/6000 R40	8	PowerPC 604	112	2.048	1.800
SGI Power Challenge	36	MIPS R10000	195	16.384	1.200
Sun Enterprise 6000	30	UltraSPARC 1	167	30.720	2.600

Exemplo – Programa paralelo em single bus

Suponha que queiramos somar 100.000 números em um computador com múltiplos processadores ligados em um único barramento. Vamos assumir que temos 10 processadores.

Solução

Pn ➔ Processador n

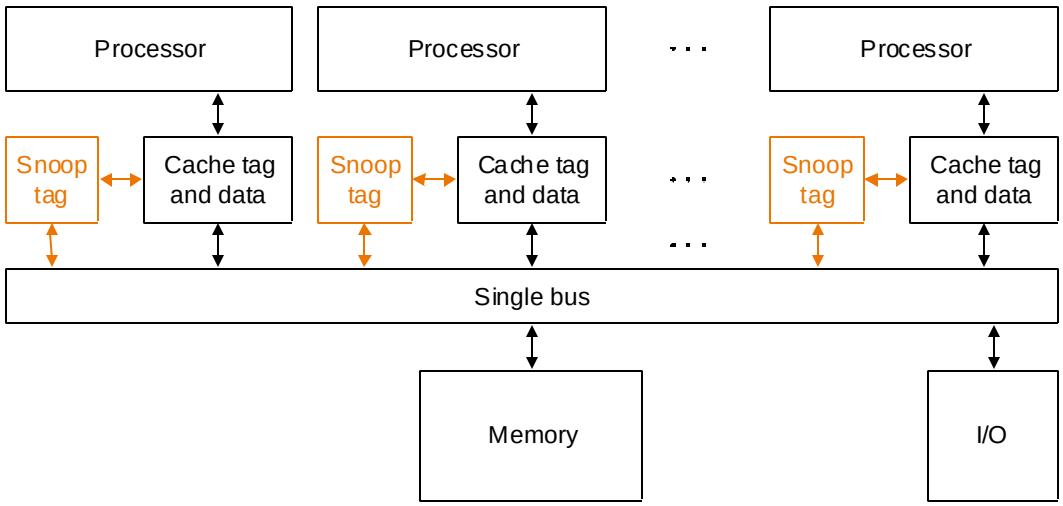
1. **Somar ¨áreas de números" em cada processador**

```
sum[Pn] = 0;
for (i = 10000*Pn; i < 10000*(pn+1); i = i + 1;)
        sum[Pn] = sum[pn] + A[i] ; /* soma as áreas assinaladas*/
```

2. **Somar as diversas sub-somas ➔ metade dos processadores somam pares de somas parciais, um quarto somam pares das novas somas parciais ,etc., etc., etc., etc..**

```
half = 10;
repeat
        synch(); /* primitiva de sincronização ➔ espera a soma
                    parcial ser feita */
        if (half%2 != 0 && Pn ==0)
                sum[0] = sum[0] + sum[half –1];
        half = half/2;
        if (Pn < half) sum[Pn] = sum[Pn] + sum[Pn+halh];
until (half ==1);
```

- **Coerência de cache em multiprocessadores ➔ snooping**



- **snoop ➔ controlador que monitora o bus para determinar se existe ou não uma cópia de um bloco compartilhado.**

- **Manutenção da coerência ➔ problema só na escrita.**

- **Processador tem que ter acesso exclusivo na escrita.**

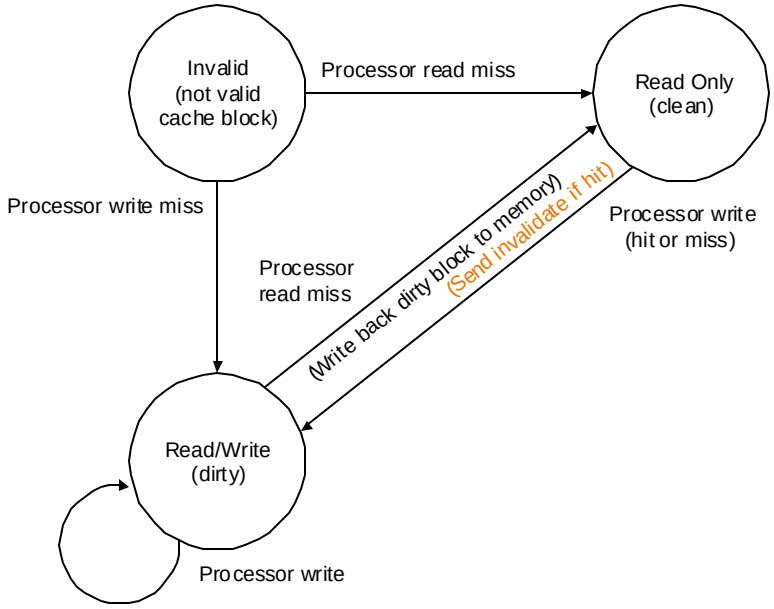
- **Todos os processadores tem que ter a cópia mais recente após uma escrita.**

- **Protocolo snooping:**

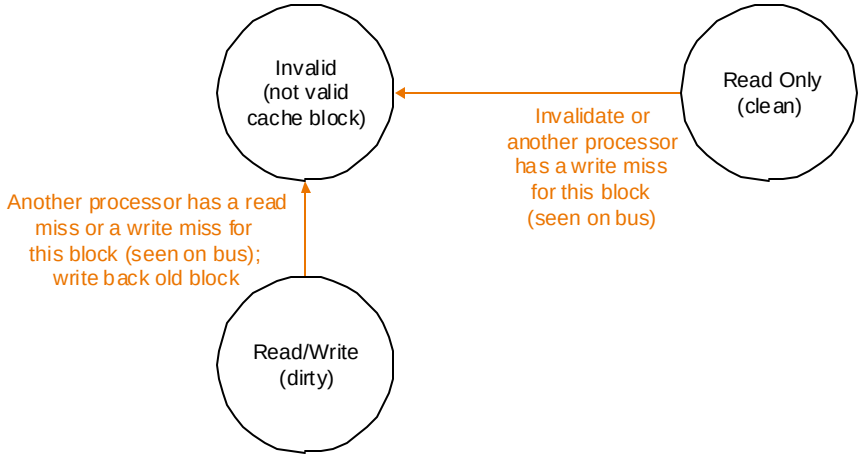
  - **write-invalidate ➔ uma escrita faz com que todas as cópias em outras caches tornem-se inválidas até a atualização. O processador que irá escrever manda um sinal de inválido no bus e todas as caches checkam para ver se tem uma cópia, se sim, tornam o bloco que contem a palavra inválido.**

- **write-update (write-broadcast) ➔ o processador que escreve propaga o novo dado no bus e todas as caches com cópia são atualizadas.**

- **Exemplo de protocolo de coerência de cache**

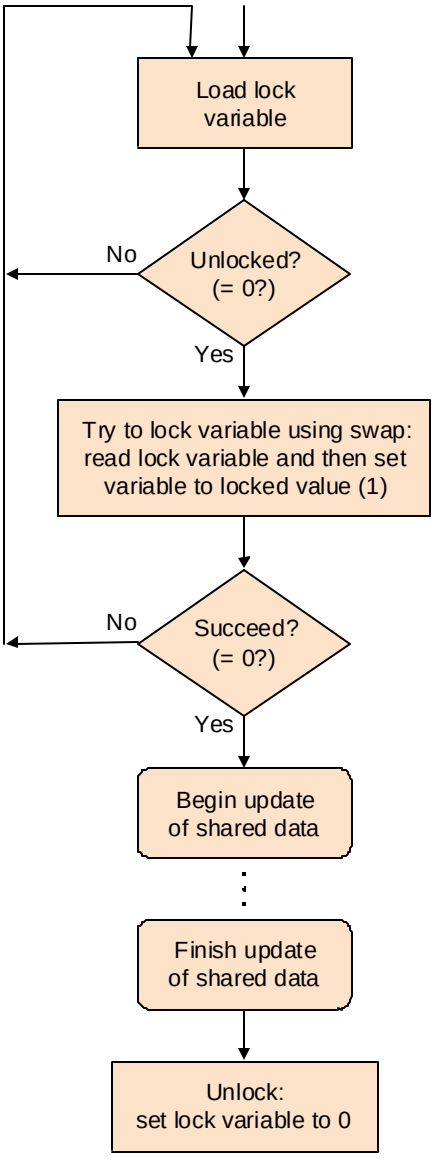


a. Cache state transitions using signals from the processor

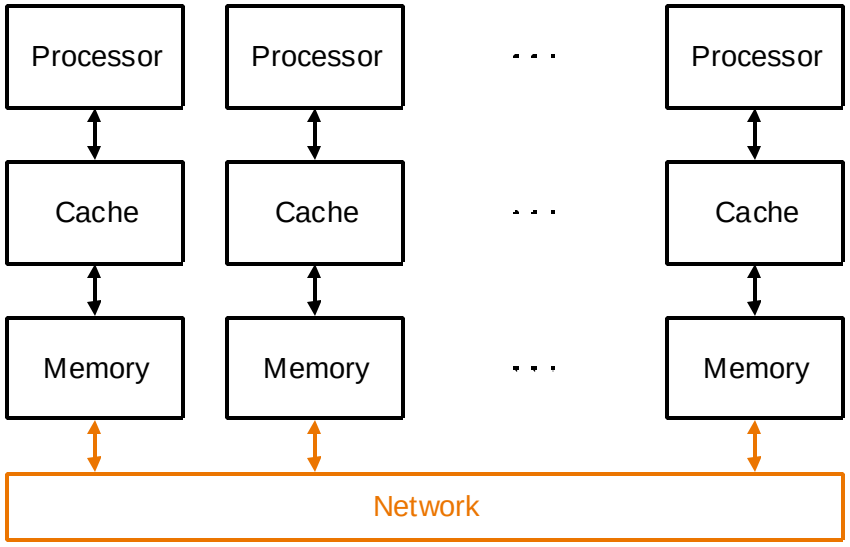


b. Cache state transitions using signals from the bus

- **Sincronização usando coerência ➔ lock variables (semáforos)**



- **Multiprocessadores conectados por uma rede**



Exemplo – Programa paralelo troca de mensagens

Suponha que queiramos somar 100.000 números em um computador com múltiplos processadores ligados em um único barramento. Vamos assumir que temos 100 processadores.

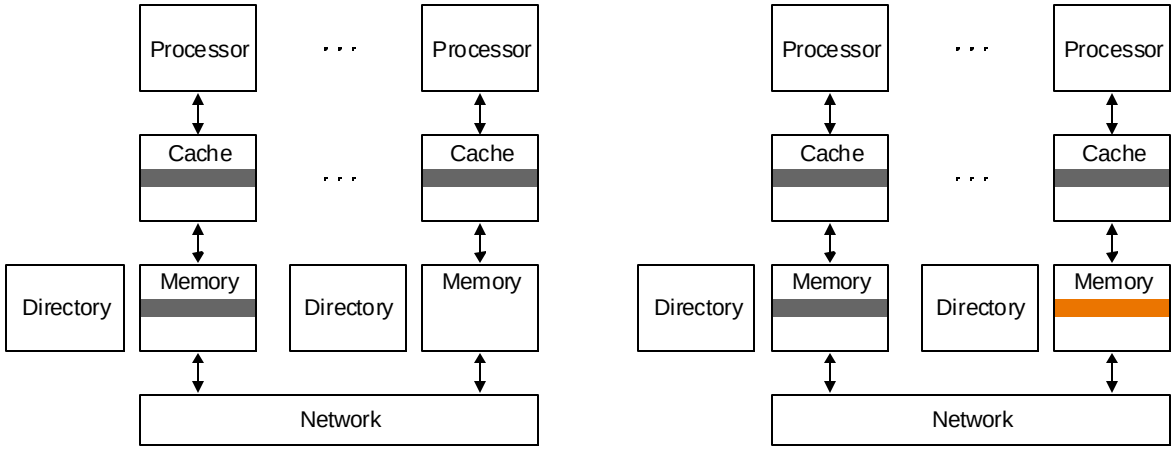
Solução

```
sum = 0;
for (i = 0; i < 1000; i = i + 1;)
        sum = sum + A[i] ; /* soma arrays locais*/
```

Somar as diversas sub-somas ➔ send (x,y) onde x = processador Pn e y o valor.

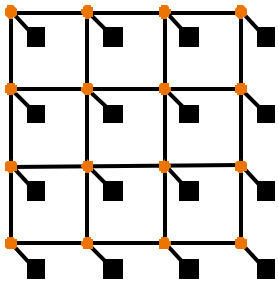
```
half = 100;
repeat
        half = (half+1)/2;
        if (Pn >= half && Pn < limit) send (pn – half, sum)
        if (Pn < limit/2 -1) sum = sum + receive();
        limit = half;
until (half ==1);
```

- **Single address space in a large-scale parallel processor**

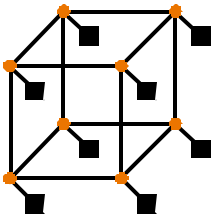


- **Clusters ➔ clusters de N máquinas tem N memórias independentes e N cópias do SO.**

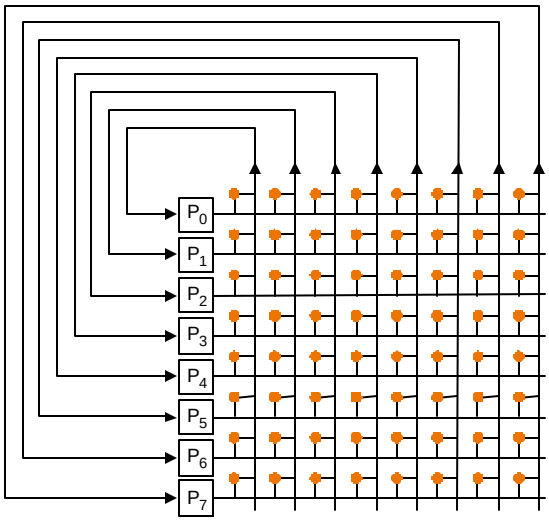
- **Topologias de Redes**



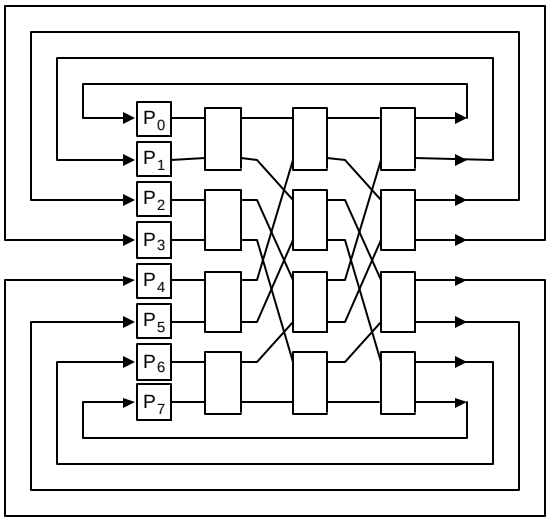
a. 2D grid or mesh of 16 nodes



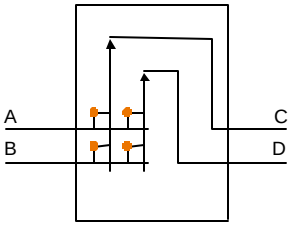
b. n-cube tree of 8 nodes (8 = 2^3 so n = 3)



a. Crossbar



b. Omega network



c. Omega network switch box