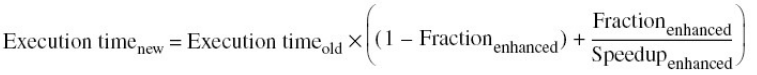
**ENERGIA: Pdinâmica = ½ x Carga capacitiva do transistor x Voltagem2 x Frequência das transições**

**Energydynamic = Capacitive load \* Voltagem2  Edinâmica = ½ x Carga capacitiva x Voltagem2**

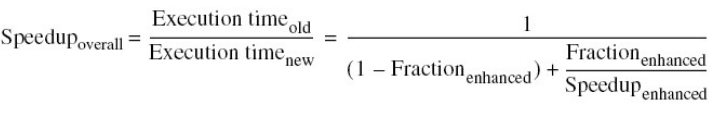
**CUSTOS:**

**sendo Wafer Yield aproximadamente = 1:**

**(Novo! N = 13,5) (Velho! = 4)**



**LEI DE AMDAHL:**



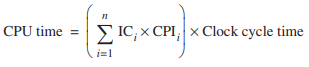
Ganho de velocidade:

**Sendo: Fração melhorada: sempre menor ou igual a 1. Ganho de velocidade melhorado: sempre maior ou igual a 1.**

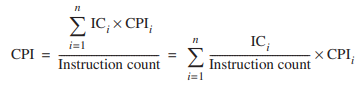
**DESEMPENHO:**

CPU time = CPU clock x Cycles for a program x Clock cycle time

CPU time = Instruction count x Cycles per instruction x Clock cycle time

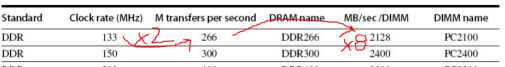




**Dependability:**

MTTF = tempo médio até a falha. FIT = Taxa de falhas = UNIDADE: 1/109 MTTR = tempo médio até o reparo

MTBF = Tempo médio entre falhas = MTTR + MTTF

**Memória:**

****

**Tempo de CPU** = ( ciclos de clock de execução da CPU + ciclos de clock de stall de memória ) x tempo de ciclo de clock

**Ciclos de clock da CPU** = Instruções para um programa x Média dos ciclos de clock por instrução (CPI)

**Ciclos de clock de stall de memória** = ciclos de stall de leitura + ciclos de stall de escrita

**Ciclos de stall da memória** = IC x (Acesso à memória/ Instrução ) x Taxa de falhas x Penalidade de falha

**Tempo de CPU** = IC x tempo de ciclo de clock x [ CPI + (Acesso à memória/ Instrução ) x Taxa de falhas x Penalidade de falha)]

**Tempo médio de acesso à memória** = Tempo de acerto + taxa de falha x penalidade de falha

**Tempo médio de acesso** = % acerto x tempo de acerto + %falhas x penalidade de falha

Em caches multi-nível (L1 e L2): **TMA** = Tempo acertoL1 + Taxa de falha L1 x ( Tempo acerto L2 + Taxa de falha L2 x Penalidade de falha L2)

**10 otimizações:** Caches pequenas e simples, Previsão de via, Acesso à cache em pipeline, Caches sem bloqueio, Caches multibanco, Palavra crítica primeiro e reinício antecipado, Write buffer merge para escrita, Otimizações do compilador, Pré-busca do hardware, Pré-busca controlada pelo compilador. **Memória:** Largura de banda : maior preocupação dos multiprocessadores e da E/S. A latência da memória principal é a preocupação da cache. **Otimizações Memória:** Múltiplos acessos à mesma linha, Transferências de dados tanto na subida quanto na descida do clock, Barramentos maiores, Múltiplos bancos em cada DRAM. **Máquinas virtuais:** Método de emulação que oferece uma interface de software-padrão, Dá suporte ao isolamento e segurança, Permite o compartilhamento de um computador entre vários usuários não relacionados, Em uma VM vários SOs compartilham os recursos de hardware. O VMM possui um nível de privilégio mais alto do que a VM convidada (guest).O software convidado não é capaz de mudar a alocação de recursos reais do sistema diretamente.

**Memória virtual:** Técnica que usa a memória principal como uma “cache” para armazenamento secundário. Implementa a tradução do espaço de endereçamento de um programa para os endereços físicos. Mantém os processos no seu próprio espaço de memória, Protege certos aspectos do status da CPU, Somente o SO pode atualizar a tabela de página.O endereço é desmembrado em número de página virtual e um offset de página. Sistemas de memória virtual usam o write-back. **Tabela de páginas:** Cada entrada contém o número da página física para esta página virtual se a página estiver na memória, Bit validade: indica se a página está na memória principal.

**TLB** (Translation-Lookaside Buffer): uma cache que monitora os mapeamentos de endereços recentemente usados para evitar um acesso à tabela de páginas

Random: burst = tam mem / bits de ECC; largura de banda = taxa de clock\*2(pos e negedge)\*8 (MB/s/DIMM) Tempo Ciclos = Tempo de acerto/Tempo de ciclo (pertence Naturais)