***Arquitetura e Organização de Computadores***

***Processamento em Batch:* é o processamento em lotes, as informações são coletadas ou recebidas, armazenadas e submetidas a processamento posterior. No nosso dia-a-dia muitos processamentos são realizados em Batch e não nos damos conta disso como, por exemplo: Leituras de consumo de água, luz, cartões de crédito e débito para o comerciante. Explicando: Na leitura de consumo de água, por exemplo, a distribuidora de água não fica sabendo qual foi o seu consumo automaticamente após a leitura, e sim somente após a chegada do funcionário na sede da empresa onde todos os dados serão processados.**

Existem dois tipos de cachê: níveis um e dois

* A **Cache Nível Um (L1)**é integrada diretamente ao núcleo do processador, por isso também chamada de **cache interna**. A sua capacidade pode ir até aos 128 Kbytes, divididos em duas partes (uma para dados e outra para instruções).
* A **Cache Nível Dois (L2)** pode ser construída na própria motherboard ou no processador (mais recentemente). Quando é externa, a sua capacidade depende do chipset presente na motherboard. Quando é interna, a capacidade varia de 128 Kbytes a 2Mbytes.

A **unidade de controle** é considerada o dispositivo mais complexo do processador, pois ele possui a lógica necessária para realizar a movimentação dos dados e de instruções de e para o processador. Isso é feito através dos sinais de controle, que são emitidos em instantes de tempo determinados, de acordo com prévia programação. A área funcional de controle possui alguns dispositivos básicos. São considerados alguns desses dispositivos: Unidade de Controle – UC, Registrador de Instrução – RI, Registrador de Dados de Memória – RD.

As memórias cache do tipo **SRAM** apresentam **maior velocidade** de acesso do que uma memória principal do tipo **DRAM**, mas possuem **maior custo** por bit.

* **SRAM - Estática** - Mais rápida e Não precisa ter seus dados atualizados milésimos de segundo.
* **DRAM - Dinâmica** - Mais Lenta e seus dados precisam ser atualizados por milésimos de segundos, caso contrário, serão perdidos.

As SRAM ocupam maior espaço físico que as DRAM, o motivo disso é que as DRAM são mais densas que a SRAM, ou seja, é possível armazenar mais bits em um chip. Ex.: em um chip de X tamanho é possível armazenar mais dados com uma memória DRAM do que a SRAM. A SRAM ocupa maior espaço físico e é mais rápida e mais cara do que a DRAM;

Arquitetura RISC:

* ênfase no software;
* uma instrução toma um único ciclo do processador;
* complexidade fica para o programador;
* pipelining fácil.
* RISC utiliza basicamente os modos LOAD e STORE para referenciar a memória.
* LEMBRE-SE RISC - R de reduzido.

Arquitetura CISC:

* ênfase no hardware;
* uma instrução toma vários ciclos do computador;
* usa muita micro programação, a complexidade fica para o processador;
* pipelining difícil;
* LEMBRE-SE CISC - C de complexo.

**RISC:**

* **Instruções Simples, levando um ciclo;**
* **Instruções LOAD e STORE referenciam a memória;**
* **Utiliza muito pipeline;**
* **Instruções de formato fixo;**
* **Compilador complexo;**
* **Muitos registradores;**
* **Instruções executadas pelo hardware;**
* **Poucos modos de endereçamento.**

**CISC:**

* **Instruções Complexas;**
* **Qualquer instrução referencia a memória;**
* **Usa pouco pipeline;**
* **Instruções de vários formatos;**
* **Micro programa;**
* **Poucos registradores;**
* **Instruções interpretadas pelo micro programa.**
* **Muitos modos de endereçamento.**

O processador é dividido em **3 partes:**

* **Unidade de controle:** busca a instrução da memória e decodifica-a.
* **ALU / ULA**(ou unidade aritmética e lógica): realiza operações aritméticas e booleanas.
* **Registradores:** memória rápida para guardar informações de controle, resultados intermediários.

O **registrador** de uma CPU é a memoria RAM que armazena n bits.  
  
Os registradores estão no topo da hierarquia de memória, sendo o meio mais rápido e caro de se armazenar um dado. Lembrando que os registradores são tipicamente usados como um dispositivo de **armazenamento temporário**.

**Volatilidade** - registradores são memórias de semicondutores e, portanto, necessitam de energia elétrica para funcionarem. Assim, registradores são **memórias voláteis**.

A **unidade aritmética e lógica (UAL**) de um microprocessador é **SIM** encarregada de realizar operações matemáticas com os dados, pois a UAL é o dispositivo do processador que efetivamente executa as operações matemáticas com os dados.

Entretanto, ela **NÃO** pode receber de um a quatro valores de entrada, mas apenas de **um a dois**. A UAL é capaz de operar apenas sobre **dois binários**e não quatro.

O**sequenciador** é encarregado de sincronizar a execução das instruções ao ritmo de um relógio. Ele é, assim, o responsável pelo envio dos sinais de comando ao registrador **Program Counter,** registrador esse que contém o endereço da micro instrução que será processada na sequência.

**Um exemplo de decomposição do processamento de instruções:**

**(5 estágios de pipeline RISC)**

**- Buscar Instrução (FI - Fetch Instruction)**: lê a próxima instrução esperada em um bufer;

**- Decodificar instrução (DI):** determina o opcode e os especificadores dos operandos;

-**Calcular operandos (CO):**calcula o endereço efetivo de cada operando de origem;

**- Obter operandos (FO - Fetch Operands):** obter cada operando da memória;

**- Executar a instrução (EI):** efetuar a operação indicada e armazenar o resultado;

**- Escrever operando (WO - Write Operands):** armazenar o resultado na memória.

***Pipeline*** é uma técnica de hardware que permite que a CPU realize a busca de uma ou mais instruções além da próxima a ser executada. Estas instruções são colocadas em uma fila de memória dentro do processador (CPU) onde aguardam o momento de serem executadas: assim que uma instrução termina o primeiro estágio e parte para o segundo, a próxima instrução já ocupa o primeiro estágio.

Em resumo, é o processo pelo qual uma instrução de processamento é subdividida em etapas, uma vez que cada uma destas etapas é executada por uma porção especializada da CPU, podendo colocar mais de uma instrução em execução simultânea. Isto traz um uso mais racional da capacidade computacional com ganho substancial de velocidade. Entre os problemas enfrentados estão a dependência de instruções anteriores e desvios que dificultam o processo, bem como a diferença de complexidade de instruções que fazem com que as mesmas possam levar um tempo variável para execução.

A técnica de pipeline é utilizada para acelerar a velocidade de operação da CPU, uma vez que a próxima instrução a ser executada está normalmente armazenada nos registradores da CPU e não precisa ser buscada da memória principal que é muito mais lenta.

**Arquitetura de Von Neuman:**

* Apenas **um barramento** externo compartilhado para os endereços e os dados;
* Instruções e dados são armazenados na mesma memória.

**Arquitetura de Harvard**

* **Dois barramento**s externos que são independentes para dados e endereços.
* Memórias independentes para dados e endereços.

Há dois tipos de projeto de micro instruções:

* **Micro instruções Horizontais:**têm a vantagem de controlar **várias** **micro operações** em **paralelo**, ou seja, no mesmo ciclo de relógio.

**"Uma unidade de controle cujo projeto é embasado no controle por micro programação através de micro instrução horizontal tem a possibilidade de iniciar várias micro operações simultaneamente." (Verdade)**

* **Micro instruções Verticais**: possuem como vantagem a **redução do custo a UC**, devido à redução dos bits das micro instruções. A UC necessita de um **decodificador** para identificar quais linhas de controle serão ativadas.

**CPU** **– Modos de funcionamentos** - A CPU apresenta o modo núcleo e o modo usuário. Executando o **modo núcleo**, a CPU pode executar qualquer instrução de seu conjunto de instruções e usar cada atributo de seu hardware. É o caso típico do sistema operacional que tem acesso a todo hardware.

Quando o processador trabalha no **modo usuário**, uma aplicação só pode executar instruções não privilegiadas, tendo acesso a um número reduzido de instruções, enquanto no modo núcleo ou supervisor a aplicação pode ter acesso ao conjunto total de instruções do processador.

**A CPU é composta por várias partes distintas. A unidade de controle é responsável por buscar instruções na memória principal e determinar seu tipo. A unidade aritmética e lógica efetua operações como adição e AND (E) booleano para executar as instruções. Também contém uma pequena memória de alta velocidade usada para armazenar resultados temporários e para um certo controle de informações, sendo composta de uma quantidade de registradores.**

A arquitetura de **Neumann** utiliza apenas um barramento de instruções e dados. Já a de Harvard, utiliza um barramento individual para instruções e um para dados. ESTA É A PRINCIPAL DIFERENÇA. TENHA ISTO EM MENTE!

**O analisador léxico, ou scanner** como também é chamado, faz a varredura do programa fonte caractere por caractere e, traduz em uma sequência de símbolos léxicos ou tokens. É nessa fase que são reconhecidas as palavras reservadas, constantes, identificadores e outras palavras que pertencem a linguagem de programação. O analisador léxico executa outras tarefas como, por exemplo, o tratamento de espaços, eliminação de comentários, contagem do número de linhas que o programa possui e etc.

Importante diferenciar

* ***Linguagens Declarativas***, o programador fornece apenas o conjunto das tarefas a serem realizadas, não estando preocupado com os detalhes de como o executor da linguagem (interpretador, compilador ou a própria máquina real ou virtual de execução) realmente implementará essas tarefas. Em outras palavras, a linguagem enfatiza a declaração descritiva de um problema ao invés de sua decomposição em implementações algorítmicas, não necessitando, em geral, de tantas linhas de código para definir uma certa tarefa.
* ***Linguagem Procedural*** deve-se informar ao computador cada passo a ser executado. Pode-se afirmar que, em linguagens procedurais, o programador possui um maior controle do código, sendo capaz de estabelecer todo o fluxo de controle e execução de seu programa. Entretanto, para isso, ele deve ser bem qualificado e conhecer bem os recursos de implementação.

**O Multiprocessamento Simétrico -** É aquele em que o processador inteiro executa as tarefas no sistema operacional. Não tem um **relacionamento mestre-escravo** como no multiprocessamento assimétrico. Todos os processadores aqui se comunicam usando a memória **compartilhada**.

Os processadores começam a executar os processos da fila de lista comum. Cada processador também pode ter sua própria fila de processos particulares pronta para execução.

O multiprocessamento simétrico tem um **balanceamento de carga**adequado, uma melhor **tolerância a falhas**.

**É complexo** uma vez que a memória é compartilhada entre todos os processadores. No multiprocessamento simétrico, uma falha no processador resulta em uma capacidade de computação reduzida.

**O Multiprocessamento assimétrico -** Tem o relacionamento mestre-escravo entre os processadores. Existe um processador mestre que controla o processador escravo restante. O processador mestre atribui processos ao processador escravo ou eles podem ter alguma tarefa predefinida para executar.

**Principais diferenças entre o multiprocessamento simétrico e assimétrico**

1. O ponto mais distinto entre o multiprocessamento simétrico e assimétrico é que as tarefas no sistema operacional são tratadas apenas pelo processador mestre no multiprocessamento assimétrico. Por outro lado, todos os processadores no multiprocessamento simétrico executam as tarefas no sistema operacional.
2. No multiprocessamento simétrico, cada processador pode ter sua própria fila de processos privados pronta ou pode obter processos de uma fila de lista comum. Mas, no multiprocessamento assimétrico, o processador mestre atribui processos aos processadores escravos.
3. Todo o processador no multiprocessamento simétrico tem a mesma arquitetura. Mas a estrutura dos processadores multiprocessadores assimétricos pode ser diferente.
4. Os processadores no multiprocessamento simétrico se comunicam através da memória compartilhada. No entanto, os processadores de multiprocessamento assimétricos não precisam se comunicar, pois são controlados pelo processador principal.
5. No caso de o processador mestre falhar, um processador escravo se torna o processador mestre para continuar a execução. Porém, se um processador no multiprocessamento simétrico falhar, a capacidade de computação do sistema será reduzida.

**Conclusão:**

Os multiprocessadores aumentam a velocidade do sistema, pois é possível executar vários processos simultaneamente. O multiprocessamento assimétrico é simples, apenas um processador (mestre) pode acessar a estrutura de dados. Embora o multiprocessamento simétrico seja complexo, uma vez que a estrutura de dados é compartilhada entre todos os processadores e todo o processador precisa trabalhar em sincronização.

Uma interrupção é um sinal que o hardware manda para o processador quando precisa da sua atenção.

* **Interrupções Mascaráveis:** Esse tipo de interrupção pode estar em dois estados, mascarado e desmascarado; uma interrupção mascarada é ignorada pela unidade de controle enquanto permanecer mascarada. Todas as interrupções geradas por dispositivos de E/S são mascaráveis.
* **Interrupções não-mascaráveis:** Interrupções que não podem ser mascaradas, ou seja, nunca são ignoradas. Só alguns poucos eventos críticos geram interrupções não-mascaráveis.

**Paginação e Segmentação**

* **Segmentação** - programas são divididos em segmentos de tamanhos variados.
* **Paginação** - páginas lógicas de tamanho fixo.

**MODOS DE ENDEREÇAMENTO**

* **Imediato** - O valor do dado é indicado no campo operando da instrução. - Nenhum outro acesso à memória é feito além da busca da instrução. - Maior rapidez. - Tamanho dos operandos é limitado ao tamanho do campo de endereços.
* **Direto -** O valor binário contido no campo operando da instrução indica o endereço de memória onde se localiza o dado. - Campo de endereço contém o endereço efetivo do operando. - Só uma referência à memória é feita, além da busca da instrução.
* **Indireto -** - Campo de endereço referencia um apontador em memória, que referencia o operando. - Para uma palavra de tamanho N, um espaço de endereçamento de 2^N é acessível. - Duas referências à memória são feitas: mais lento.
* **Por registrador -**  - Similar ao endereçamento direto, contudo o campo de endereço refere-se a um registrador em vez de memória. - Como há número limitado de registradores, campo de endereço (por registrador) é pequeno. - Não há necessidade de referências à memória Þ maior rapidez.
* **Indexado -** Similar ao endereçamento indireto. - Operando é obtido por referência de um apontador mantido em registrador. - Maior espaço de endereçamento (2^N) que o modo registrador. - Uma referência de memória a menos que o modo indireto.
* **Deslocamento (Soma e Deslocamento) -** Tem características semelhantes ao modo indexado, visto que o endereço de acesso a uma célula de memória se obtém através da soma de dois valores, um inserido no campo apropriado da instrução.