Assessment – AT

Vitor Hugo Tato Coriolano

Professor: Felipe Fink Grael
Disciplina: Arquitetura de Computadores e Sistemas Operacionais
Turma: Noite

Rio de Janeiro – setembro de 2021

INTRODUÇÃO.

O trabalho apresentado está focado na apresentação final de todo o conhecimento adquirido na matéria Arquitetura de Computadores e Sistemas Operacionais.
A nós serão apresentadas questões as quais deveremos apresentar as devidas respostas pautadas em tudo o que foi ensinado em aula.
Ao final será apresentado um sólido conhecimento sobre todo os assuntos abordados ao longo do bloco, demonstrando os mesmos de forma clara, didática e objetiva.

QUESTÃO 01.

1. Explique o que diferencia os computadores de 3ª geração dos da 4ª geração.

1.1. Computadores de terceira geração – 1964 a 1975.

1.1.1. Os computadores da terceira geração funcionavam por circuitos integrados. Esses substituíram os transistores e já apresentavam uma dimensão menor e maior capacidade de processamento.
Foi nesse período que os chips foram criados e a utilização de computadores pessoais começou.

1.1.2. Algumas características relevantes a essa geração:
1.1.2.1. Surgem os circuitos integrados.
1.1.2.2. Diminuição do tamanho.
1.1.2.3. Maior capacidade de processamento.
1.1.2.4. Início da utilização dos computadores pessoais.

1.2. Computadores de quarta geração – 1975 até os dias atuais.

1.2.1. Com o desenvolvimento da tecnologia da informação, os computadores diminuem de tamanho, aumentam a velocidade e capacidade de processamento de dados. São incluídos os microprocessadores com gasto cada vez menor de energia.
Nesse período, mais precisamente a partir da década de 90, há uma grande expansão dos computadores pessoais.
Além disso, surgem os softwares integrados e a partir da virada do milênio, começam a surgir os computadores de mão. Ou seja, os smartphones, iPod, iPad e tablets, que incluem conexão móvel com navegação na web.

1.2.2. Algumas características relevantes a essa geração:
1.2.2.1. Surgem os softwares integrados
1.2.2.2. Processadores de Texto
1.2.2.3. Planilhas Eletrônicas
1.2.2.4. Gerenciadores de Banco de Dados
1.2.2.5. Gráficos
1.2.2.6. Gerenciadores de Comunicação

QUESTÃO 02.

1. Informe a quantidade exata em bytes dos seguintes valores (não use multiplicadores):

1.1. – 12 Gbyte = **12884901888** **– Bytes**
1.2. – 20 Kbytes = **20480** **– Bytes**
1.3. – 256 bits = **32** **– Bytes**

QUESTÃO 03.

1. Defina os padrões de barramento SATA e SCSI e cite suas diferenças.
    1.1. SATA:
        1.1.1. É uma interface de comunicação que permite a dispositivos como discos rígidos, SSDs, unidades de DVD / Blu-ray e afins serem conectados à placa-mãe de um desktop, servidor ou notebook seguindo dois princípios básicos: a de que a conexão seja feita de modo fácil e seguro (impedindo que o conector seja encaixado incorretamente, por exemplo) e de que proporcione boa velocidade de transferência de dados. O padrão surgiu por volta do ano 2000 para substituir a antiga interface PATA *(*Paralell ATA*),* também conhecida como *IDE*.
            1.1.1.1. Padrões SATA

                1.1.1.1.1. SATA 150 ou SATA I: foi a primeira versão desta tecnologia e o nome ficou conhecido pela sua taxa de transferência de 150 MB/s.

                1.1.1.1.2. SATA II: não aumentou a taxa de transferência, porém, foi adicionado recursos como NCQ (Native Command Queueing), técnicas para diminuir o movimento da cabeça de leitura que também é utilizada na tecnologia SCSI que será mostrada posteriormente.

                1.1.1.1.3. SATA 300 ou SATA/300: trabalhava com uma taxa de transmissão de 300 MB/s, o dobro da última versão da PATA.

                1.1.1.1.4. SATA 600: seguindo o nome, tem uma taxa de transmissão de 600 MB/s. Só lembrando que essas taxas mencionadas ao longo do artigo são taxas máximas, totalmente teórica. A previsão é de chegar a uma taxa de transferência bem maior no futuro.

1.2. SCSI

1.2.1. Sigla para Small Computer System Interface, SCSI é uma tecnologia desenvolvida para permitir a comunicação entre dispositivos computacionais de maneira rápida e confiável. Sua aplicação é mais comum em HDs (discos rígidos), embora outros tipos de equipamentos tenham sido lançados tirando proveito dessa tecnologia, como impressoras, scanners e unidades de fita (usadas para backup).
Trata-se de uma tecnologia antiga. Sua chegada ao mercado aconteceu oficialmente em 1986, mas seu desenvolvimento foi iniciado no final da década de 1970, tendo o pesquisador *Howard Shugart*, considerado como o criador do *floppy disk (disquete)*, como principal nome por trás do projeto. Pronunciado como "iscãzi", essa tecnologia se mostrou extremamente importante nos anos seguintes, especialmente porque os processadores passaram a ficar cada vez mais rápidos. Com o SCSI, os HDs e outros dispositivos puderam, de certa forma, acompanhar esse aumento de velocidade.

1.2.1.1. Padrões SCSI

1.2.1.1.1. SCSI-1: em 1981, este padrão surgiu para suprir a necessidade de criar algum meio que permitisse uma taxa de transferência de dados para discos rígidos mais elevada. Essa tecnologia ganhou especificações da ANSI (American National Standards Institute) e assim pode ser comercializada.
Em 1983, começaram a surgir os primeiros HDs que usavam esse padrão.
Otimistas em relação ao sucesso e seus benefícios que essa interface poderia trazer, pesquisadores começaram trabalhar em protocolos de comunicação com o objetivo de alcançar um alto desempenho na comunicação com os dispositivos SCSI, mas só no ano de 1986 que o SCSI se consolidou.

1.2.1.1.2. SCSI-2: no ano em que este padrão se consolidou 1986, estava em estudo a versão 2 que entre outras características, permitia o uso de drives de CD-ROM, um verdadeiro avanço naquela época.
O SCSI-2 chegou efetivamente ao mercado em 1988 e permaneceu como o tipo mais consumido, mesmo após o lançamento do SCSI-3, em 1993.
O padrão SCSI-2, além de ter acumulado as especificações do SCSI-1, ainda incluiu um novo recurso, chamado de Fast SCSI. Trata-se de um barramento adicional de 10 MHz (o SCSI-1 usava 5 MHz). Outro recurso, foi a implantação do Wide SCSI, que permitia uso de cabos de 16 ou 32 bits, ao invés dos 8 bits oferecidos pelo SCSI-1.
Foi nesse período que scanners e outros

periféricos começaram a usar o SCSI.

1.2.1.1.3. SCSI-3: Em 1995, o SCSI-3 passou a ser reconhecido, mas logo ganhou uma variação, que ficou conhecida como Ultra-SCSI, que funcionava à velocidade de 20 Mbps. Um ano depois, o SCSI-3 passou a ter especificações P1394, da IEEE (Institute of Electrical and Electronics Engineers) e ficou compatível com protocolos de fibra óptica oferecendo suporte a comandos e algoritmos de drives de CD-ROM.
No ano de 1997, o SCSI-3 ganhou algumas especificações. A mais importante delas o funcionamento em 40 MHz, passando a se chamar Ultra-2 SCSI. Em 1999, essa velocidade aumentou aumenta para 80 MHz e então, surgiu o Ultra-3 SCSI.

1.2.1.1.4. É de suma importância deixar claro que o SCSI ainda conta com outras variações sendo uma delas a SAS – a Serial Attached SCSI, que pode atingir velocidade de até 6 Gb/s (gigabits por segundo) e suporta a conexão de até 128 dispositivos. Isso é possível, entre outros motivos, porque essa variação utiliza um esquema de transmissão serial de dados (nas versões mostradas anteriormente, a transmissão é feita de maneira paralela) combinado com frequências maiores.

1.3. Em resumo quando nos limitamos a comparar as interfaces (SATA e SCSI), logo concluímos que ambas se diferem na questão de versatilidade, capacidade e desempenho.
O padrão SATA é destinado às mais diversas finalidades, podendo ser utilizado para desempenhar qualquer tipo de tarefa, mesmo que fora do ideal.
A interface SCSI foi desenvolvida para atender às necessidades de grandes infraestruturas de TI, correspondendo ao alto desempenho exigido por esses processos.

QUESTÃO 04.

1. Classifique as memórias de acordo com sua velocidade. Depois explique qual a função da RAM e qual a função do HD.

1.1. A memória de um computador nada mais é que um circuito eletrônico ou um meio magnético, com capacidade de armazenagem de dados, os quais são imprescindíveis ao processamento: dados de entrada, como: programas, sistemas operacionais, arquivos, softwares de aplicação e instruções gerais para um bom funcionamento do computador. Existem dois grandes grupos de memória:
Memória Interna e a Memória externa.

1.1.1. A memória interna está diretamente ligada aos componentes da CPU, como por exemplo as memórias; principal (RAM); de leitura (ROM) e a cache.

1.1.1.1. Memória principal – é também é conhecida como memória central, é uma memória de rápido acesso e que armazena os dados e informações como programas, dados de entrada e saída, dados do sistema operacional, entre outros. E isto devido a três tipos de registros, sendo dois deles associados a operações de leitura e gravação e o outro aos endereços.

1.1.1.2. Memória de Leitura – com sua característica não-volátil, ou seja, estes circuitos conservam os dados que estavam registrados nela mesmo que a máquina esteja desligada. Por outro lado, a máquina trabalhando em modo de operação normal não permite registrar nada nelas, apenas possível ler o que está registrado.
Como por exemplo a memória ROM – read only memory ou memória apenas para leitura.
Este tipo de memória já vem instalada de fábrica na placa-mãe e traz gravadas as informações básicas para o funcionamento da máquina, ativando os dispositivos necessários para a inicialização das tarefas.

1.1.1.3. Memória Cache – Esta memória está localizada na placa-mãe e é formada por vários circuitos integrados RAM do tipo SRAM (Static RAM) e é muito mais rápida que a RAM convencional, porque não usa o método capacitivo de armazenamento e sim dispositivos de dois estados como os Flips-Flops (circuito eletrônico que pode assumir um de dois estados, determinados por uma ou duas entradas).
Por ocuparem mais espaço de armazenamento, tornam-se mais caros, motivo pelo qual são utilizados em tamanhos reduzidos, em relação à memória do tipo DRAM.
Sua capacidade varia de 8 Mb a 1024 Kb, sendo o mais utilizados é 256 Kb, 512 Kb.
Para a identificação rápida dos dados,a memória cache

utiliza-se de um dispositivo, localizado geralmente ao lado de seu banco de chips, chamado de SRAM TAG, que é onde estão localizados os caracteres de identificação rápida dos arquivos.

1.1.2. A Memória externa é aquela que se vale de meios externos à CPU, como HDs, fitas, disquetes, discos zip, CD-Rom's, entre outros, para armazenar informações, as quais não são possíveis serem gravados na memória principal, pelo fato da mesma ser volátil. Esses meios ficam ligados indiretamente à CPU e podem ser rapidamente acessados eles operam com velocidade menor do que a RAM, porém, têm uma capacidade de armazenamento infinitamente maior, não são voláteis e podem ser desconectados fisicamente do computador e transportados para outro local, sem que ocorram prejuízo às informações armazenadas

.

1.2. A memória RAM tem a função de ser responsável pelo armazenamento de informações necessárias para a execução de aplicativos em uso e para o funcionamento do próprio sistema operacional. Ela inclusive, facilita o trabalho do processador que pode acessar os dados essenciais mais rapidamente.

1.3. Um disco rígido (HD) é um hardware usado para armazenar conteúdo digital e dados em computadores. Cada computador tem um disco rígido interno, mas você também pode obter discos rígidos externos que podem ser usados para expandir o armazenamento de um computador. Os HDs são hardwares não voláteis, ou seja, as informações gravadas neles não se perdem ao desligarmos a máquina na qual eles estejam conectados.

QUESTÃO 05.

1. O que é o Gateway e quem pode servir de gateway?

1.1. Um Gateway, ou porta de ligação, é uma máquina intermediária geralmente destinada a interligar redes, separar domínios de colisão, ou mesmo traduzir protocolos. Exemplos de gateway podem ser os roteadores e firewalls, já que ambos servem de intermediários entre o utilizador e a rede.
Um proxy também pode ser interpretado como um gateway (embora em outro nível, aquele da camada em que opera), já que serve de intermediário também.
Cabe igualmente ao gateway traduzir e adaptar os pacotes originários da rede local para que estes possam atingir o destinatário, mas também traduzir as respostas e devolvê-las ao par local da comunicação. Assim, é frequente a utilização de protocolos de tradução de endereços, como o NAT – que é uma das implementações de gateway mais simples.
Outros exemplos de gateway (além do proxy citado acima) pode ser um PC com duas (ou mais) placas de rede ou um dispositivo dedicado, ambos sendo utilizados para unir duas redes.

QUESTÃO 06.

1. Quais as funções exercidas pelo Sistema Operacional para o gerenciamento de memória? Explique.

    1.1. O gerenciamento de memória é a tarefa desempenhada pela parte do Sistema Operacional que controla o uso da memória. Sendo função dessa parte conhecer quais regiões da memória estão em uso e quais não estão sendo usadas, alocar memória para processos quando eles necessitarem e desalocá-la quando os processos terminarem de ser executados, gerenciar o swapping entre a memória principal e o disco, quando a memória principal não for grande o suficiente para comportar todos os processos

QUESTÃO 07.

1. Explique a diferença entre exceção e interrupção.

    1.1. A diferença entre exceção e interrupção é dada pelo tipo de evento ocorrido.

        1.1.1. “Interrupção – A interrupção é o mecanismo que torna possível a implementação da concorrência nos computadores, sendo o fundamento básico dos sistemas multi programados. É em função desse mecanismo que o sistema operacional sincroniza a execução de todas as suas rotinas e dos programas dos usuários, além de controlar os dispositivos.”[1]
        Uma interrupção é sempre gerada por algum evento externo ao programa, e, nesse caso, independe da instrução que está sendo executada. Um exemplo de interrupção ocorre quando um dispositivo de entrada e saída sinaliza ao processador que uma operação de entrada saída está completa. Neste caso, o processador deve interromper o programa que está sendo executado para tratar o termino da operação de entrada saída.
        Ao final da execução de cada instrução, a unidade de controle verifica a ocorrência de algum tipo de interrupção. Nesse caso, o programa em execução é interrompido e o controle desviado para uma rotina responsável por tratar o evento ocorrido, chamada rotina de tratamento de interrupção. Como existem vários tipos de interrupção, podem haver diferentes rotinas de tratamento de interrupção.
        Para que o programa interrompido possa retomar sua execução exatamente do ponto onde foi interrompido, é necessário que, no momento da interrupção, sejam guardadas informações a respeito da execução do programa.
        Estas informações são basicamente o conteúdo dos registradores que deverão ter seus valores restaurados quando a execução do programa interrompido for reiniciada.

1.1.2. “Exceção – Difere-se da interrupção por meio do motivo pelo qual é gerada, sendo decorrente de eventos previsíveis, um de cada vez. É o resultado direto da execução de uma instrução do próprio programa. Sempre que ocorre uma exceção, o fluxo do programa é desviado para uma rotina de tratamento de exceção, a qual pode ser escrita pelo próprio programador.” [1]
As exceções são parecidas com as interrupções, a principal diferença é o motivo pelo qual o evento é gerado. A exceção é o resultado direto de uma instrução dentro do próprio programa, como a divisão por zero ou a ocorrência de um erro de estouro de memória em uma operação aritmética (overflow). As exceções são portanto geradas por eventos síncronos dentro do próprio programa e portanto tais eventos são previsíveis e podem ser tratados pelo programador que pode incluir tratamento de exceção no programa para que o mesmo não seja interrompido.
O tratamento de exceção é similar ao tratamento de interrupções. Para cada exceção gerada durante a execução de um programa a ação é interromper a execução do programa e invocar uma rotina para o tratamento da exceção. Existe uma rotina de tratamento de exceção, para a qual o fluxo de execução é desviado quando a exceção é lançada durante a execução do programa e o programa é interrompido.

1 – Retirado da etapa 8 da referida matéria descrita no moodle.

QUESTÃO 08.

1. Cite e explique os Estados de Processos.
   1.1. Em sistemas Multitarefas o processo não é executado o tempo todo pelo processador, com isso temos os “Estados de Processos”.
      1.1.1. Pronto – O processo está pronto e esperando para ser executado pela CPU.
         1.1.1.1. Espera – O processo está esperando algum evento externo ou por algum recurso para poder prosseguir seu processamento.
            1.1.1.1.1. Bloqueado – O processo está esperando por algum recurso do sistema que não se encontra disponível.
      1.1.2. Em execução – O processo está sendo executado pela CPU.

   1.2. A mudança de estado do processo.
      1.2.1. Estas mudanças de estado podem ocorrer de forma voluntária, ou seja, ocorrem por conta do próprio processo ou involuntária, geralmente causadas pelo próprio Sistema operacional.
      Estas mudanças podem ser caracterizadas de quatro formas.
         1.2.1.1. Pronto ➔ Execução = Quando um processo é criado, é colocado em uma lista de processos no estado pronto. Então é escolhido pelo sistema para ser executado.
         1.2.1.2. Execução ➔ Espera = O processo passa para espera quando aguarda a conclusão de um evento solicitado.

1.2.1.3. Espera ➔ Pronto = O processo passa para pronto quando a operação solicitada é atendida ou o recurso esperado é concedido.
1.2.1.4. Execução ➔ Pronto = O processo passa de execução para pronto por eventos gerados pelo sistema.

QUESTÃO 09.

1. Cite e explique os modos principais de implementação de Threads.

1.1. Thread é um pequeno programa que trabalha como um subsistema, sendo uma forma de um processo se autodividir em duas ou mais tarefas.
É o termo em inglês para linha ou encadeamento de uma execução.
Essas tarefas múltiplas podem ser executadas simultaneamente para rodar mais rápido do que um programa em um único bloco ou praticamente juntas, mas que são tão rápidas que parecem estar trabalhando em conjunto ao mesmo tempo.
As diversas threads que existem em um programa podem trocar dados e informações entre si e compartilhar os mesmos recursos do sistema, incluindo o mesmo espaço de memória. Assim, um usuário pode utilizar uma funcionalidade do sistema enquanto outras partes da execução estão trabalhando e realizando outras operações.
A grosso modo podemos definir que é como se houvesse um usuário virtual trabalhando no mesmo computador e ao mesmo tempo que o usuário físico logado.
Devido à maneira rápida que a mudança de uma thread e outra acontece, aparentemente é como se elas estivessem sendo executadas paralelamente de maneira simultânea em hardwares equipados com apenas uma CPU.
Esses sistemas são chamados de monothread. Já para os hardwares que possuem mais de uma CPU, as threads são realmente feitas concorrencialmente e recebem o nome de multithread.
As threads possuem vantagens e desvantagens (assim como tudo), ao dividir um programa em vários processos. Uma das vantagens é que isso facilita o desenvolvimento, visto que torna possível elaborar e criar o programa em módulos, experimentando-os isoladamente no lugar de escrever em um único bloco de código. Outro benefício das threads é que elas não deixam o processo parado, pois enquanto um deles está aguardando um determinado dispositivo de entrada ou saída, ou ainda outro recurso do sistema, outra thread pode estar trabalhando.
No entanto, uma das desvantagens é que com várias threads o trabalho fica mais complexo, justamente por causa da interação que ocorre entre elas.
Uma thread pode autorresponder-se sem que seja preciso duplicar um processo inteiro, economizando recursos como memória, processamento e aproveitando dispositivos de entrada e saida, variáveis e outros meios. Também pode por conta própria abandonar a CPU por não ver a necessidade de continuar com o processamento proposto pela própria CPU ou pelo usuário. Isso é realizado por meio do método thread-yield.

Os sistemas operacionais executam de maneiras diferentes os processos e threads.
No caso do Windows, ele trabalha com maior facilidade para gerenciar programas com apenas um processo e diversas threads do que quando gerencia vários processos e poucas threads. Isso acontece porque no sistema da Microsoft a demora para criar um processo e alterná-los é muito grande.
Enquanto isso, o Linux e os demais sistemas baseados no Unix podem criar novos processos de maneira muito mais rápida. No entanto, ao serem alterados, os programas podem apresentar o mesmo desempenho tanto no Linux quanto no Windows.

QUESTÃO 10.

1. Cite e explique as quatro situações necessárias para que ocorra um Deadlock.

    1.1. O deadlock ocorre quando cada processo de um conjunto de processos está esperando por um evento que apenas outro processo do mesmo conjunto pode causar. Em muitos casos, o evento esperado é a liberação de um recurso qualquer, isto é, cada membro do conjunto está esperando pela liberação de um recurso que apenas outro membro do conjunto pode liberar, com isso temos quatro condições para que ocorra um deadlock.

        1.1.1. Exclusão Mútua: cada recurso ou está associado a exatamente um processo ou está disponível.

        1.1.2. Posse e espera: um processo que já possui algum recurso pode requisitar outros e aguardar por sua liberação.

        1.1.3. Não existe preempção: recursos dados a um processo não podem ser tomados de volta, precisam ser liberados pelo processo.

        1.1.4. Espera Circular: deve haver uma cadeia circular de dois ou mais processos, cada um dos quais aguardando um recurso em posse do próximo membro da cadeia.

QUESTÃO 11.

1. Explique o problema do Barbeiro e sua solução.

    1.1. Enunciado: O problema do barbeiro dorminhoco é um problema clássico de comunicação Inter processos e sincronização entre múltiplos processos.
    O problema é análogo a manter o barbeiro ocupado enquanto há clientes, e descansando quando não há nenhum, sendo que fazendo isso de uma maneira ordenada.
    O barbeiro e seus clientes correspondem aos processos mencionados acima.

    1.2. O problema: Na barbearia há um barbeiro, uma cadeira de barbeiro e n cadeiras para eventuais clientes esperarem a vez.
    Quando não há clientes, o barbeiro senta-se na cadeira de barbeiro e cai no sono.
    Quando chega um cliente, ele (o cliente) precisa acordar o barbeiro.
    Se outros clientes chegarem enquanto o barbeiro estiver cortando o cabelo de um cliente, eles se sentarão (se houver cadeiras vazias) ou sairão da barbearia (se todas as cadeiras estiverem ocupadas).
    O problema é programar o barbeiro e os clientes sem cair em condições de disputa.

    1.3. A solução: Usando três **semáforos**: **clientes**, que conta os clientes à espera de atendimento, exceto o cliente que sendo atendido, ou seja, ele não está esperando.
    barbeiros, o número de barbeiros (0 ou 1) que estão ociosos à espera de clientes, e mutex, que é usado para exclusão mútua.
    Precisamos ainda de uma variável, **esperando**, que também conta os clientes à espera de atendimento.
    É essencialmente uma cópia de clientes. A razão de se ter esperando é que não há uma maneira de ler o valor atual do semáforo.
    Nessa solução, um cliente que entra na barbearia deve contar o número de clientes à espera de atendimento. Se este for menor que o número de cadeiras, ele ficará; do contrário, ele sairá.
    Nesta solução, quando chega de manhã para trabalhar, o barbeiro executa o método barbeiro, que o leva a bloquear sobre o semáforo clientes, que inicialmente está em 0.
    O barbeiro então vai dormir, e permanece dormindo até que o primeiro cliente apareça.
    Quando chega, o cliente executa cliente e inicia obtendo mutex para entrar em uma região crítica. Se um outro cliente chega logo em seguida, o segundo nada pode fazer até que o primeiro libere o mutex.
    O cliente então verifica se o número de clientes à espera é menor que o número de cadeiras. Se não for, ele liberará o mutex e sairá sem cortar o cabelo. Se houver uma cadeira disponível, o cliente incrementará a variável inteira esperando.
    Ele faz então um up no semáforo clientes, que acorda o barbeiro.
    Nesse ponto, o cliente e o barbeiro estão ambos acordados. Quando

o cliente libera
mutex, o barbeiro o pega, faz alguma limpeza e começa a cortar o cabelo.
Quando termina o corte de cabelo, o cliente sai do procedimento e deixa a barbearia.
Diferente de nossos exemplos anteriores, não há um laço para o cliente porque para cada um deles terá apenas um corte de cabelo.
O barbeiro, contudo, contém um laço para tentar obter o próximo cliente.
Se houver algum outro cliente, um novo corte de cabelo será iniciado.
Do contrário, o barbeiro cairá no sono.

QUESTÃO 12.

1. Enumere os componentes principais de uma CPU.

    1.1. Unidade de controle: busca a instrução da memória e decodifica-a

    1.2. ALU (ou unidade aritmética e lógica): realiza operações aritméticas e booleanas.

    1.3. Registradores: memória rápida para guardar informações de controle, resultados intermediários.

QUESTÃO 13.

1. Explique o Princípio da Localidade.

    1.4. Princípio da Localidade é dividido em:

        1.4.1. Localidade temporal: ao acessar uma palavra na memória principal é muito provável que o processador volte a acessar essa mesma palavra novamente durante a execução dos programas(loops).

        1.4.2. Localidade espacial: ao acessar uma palavra na memória principal é provável que em seguida o processador tente acessar uma palavra de memória subjacente à acessada previamente.

QUESTÃO 14.

1. Explique a diferença entre comunicações seriais e paralelas.

    1.1. Comunicação serial: consiste na transmissão de dados em série, ou seja, um bit de cada vez e depois de 8 pulsos de clock monta-se o primeiro byte. Se ocorrer algum erro durante a transmissão, apenas o bit faltante é retransmitido e não todo o Byte (conjunto de 8 bits).
    A comunicação serial é considerada simples, pois com apenas um canal de envio e um de recepção é possível fazer a troca de dados entre dois dispositivos.
    O USB (Universal Serial Bus) é um grande exemplo disso utilizando apenas quatro vias na composição do seu cabo, onde dois são conectores de energia de 5V (positivo e negativo), utilizado nos carregadores de celular, e os outros dois para enviar e receber dados.
    Alguns exemplos de transmissões em série:
    Porta Serial (COM), USB, SATA(SerialATA), FireWire, entre outros.

    1.2. Comunicação paralela: consiste em um conjunto de vias enviando simultaneamente um padrão de 8 Bits ou de múltiplos de 8. Ou seja, ele envia vários bits ao mesmo tempo de maneira sincronizada.
    Para cada bit enviado é preciso um fio para a transmissão, isso faz com que o cabo/conector fique muito grande em caso de se querer enviar vários bits, como é o caso de uma transmissão a 32 bits que utilizaria 32 fios/vias no cabo só para os dados.
    Na Comunicação Paralela as vias usadas para envio de dados também são utilizadas para recebimento, porém isso não pode acontecer simultaneamente.
    Um conjunto de dados não pode ser enviado ao mesmo tempo que estamos recebendo outro.
    No caso do serial, citado anteriormente, o envio e recebimento de dados utilizam vias diferentes (um só envia e outro só recebe).
    Alguns exemplos de transmissões em paralelo:
    Porta LPT1 (porta paralela), SCSI, IDE, PCI(não a express), AGP, entre outras.

QUESTÃO 15.

1. Explique o escalonamento de processos chamado de Escalonamento Round_Robin.

    1.1. É o tipo de escalonamento preemptivo mais simples e consiste em repartir uniformemente o tempo da CPU entre todos os processos prontos para a execução. Os processos são organizados numa fila circular, alocando-se a cada um uma fatia de tempo da CPU, igual a um número inteiro de quanta. Caso um processo não termine dentro de sua fatia de tempo, ele é colocado no fim da fila e uma nova fatia de tempo é alocada para o processo no começo da fila.
    O escalonamento circular é muito simples, mas pode trazer problemas se os tempos de execução são muito discrepantes entre si. Quando existirem muitas tarefas ativas e de longa duração no sistema, as tarefas curtas terão o seu tempo de resposta degradado, pois as tarefas longas reciclarão continuamente na fila circular, compartilhando de maneira equitativa a CPU com tarefas curtas.

REFERÊNCIAS BIBLIOGRÁFICAS

Todo o texto teve como base e referências o roteiro de aprendizagem e explicações em sala de aula com o auxílio de pesquisas na Web e com o material bibliográfico descrito abaixo.

- Prof. Dr. Theo Ungerer, Multithreaded-von-Neumann-Architekturen, Vieweg+Teubner Verlag, pp 317-345, 1993 doi:10.1007/978-3-322-94688-1_7 ISBN 978-3-519-02128-5 Online ISBN 978-3-322-94688-1 (em alemão)

- Wolfgang Händler, On classification schemes for computer systems in the Post-Von-Neumann-Era, Springer Berlin Heidelberg , pp 439-452, 1975 doi:10.1007/3-540-07141-5_246 ISSN 0302-9743ISBN 978-3-540-07141-9 Online ISBN 978-3-540-37424-4 (em inglês)

- Hélène Collavizza, Dominique Borrione, Specifying the Micro-program Parallelism for Microprocessors of the Von Neumann style, Springer London, pp 153-170, 1991 ISSN 1431-1682 doi:10.1007/978-1-4471-3544-9_9 ISBN 978-3-540-19659-4 Online ISBN 978-1-4471-3544-9 (em inglês)

- Robert A. Iannucci, A Dataflow/von Neumann Hybrid , Springer US, pp 49-91, 1990 ISSN 0893-3405 doi:10.1007/978-1-4613-1543-8_3 ISBN 978-1-4612-8827-5 Online ISBN 978-1-4613-1543-8 (em inglês)

- P. Hines, Can a Quantum Computer Run the von Neumann Architecture ? , Springer Berlin Heidelberg, pp 941-982, 2011, ISSN 0075-8450 doi:10.1007/978-3-642-12821-9_14 ISBN 978-3-642-12820-2Online ISBN 978-3-642-12821-9 (em inglês)

- MONTEIRO, Mário A. Introdução à organização de computadores. Rio de Janeiro: LTC, 2007.

- Tanenbaum, Andrew S., Sistemas Operacionais Modernos - 4ª Ed. 2016 Editora – Pearson