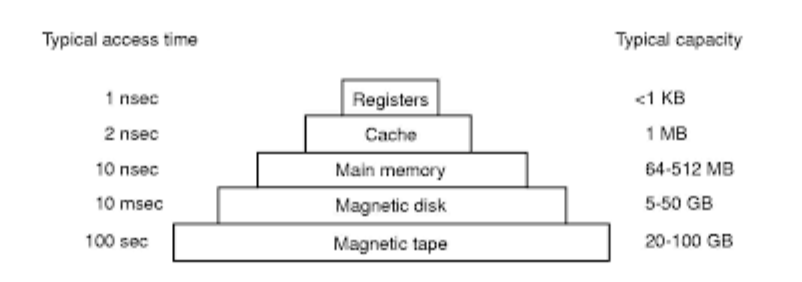
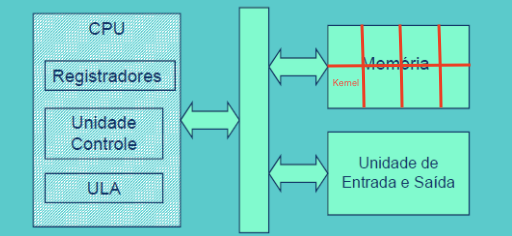
Robson Novato Lobao - 20.1.4018

Questão 1 -

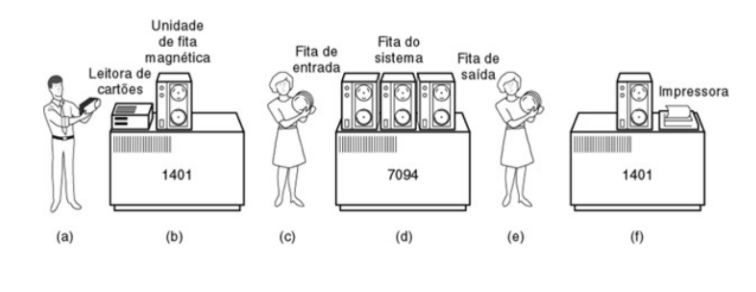
Dividindo a pergunta 1 em partes, os 27 mil anos citados em sala é uma forma de ilustrar o quanto o acesso a memória é mais rápido que o acesso a disco, tendo em vista que se fosse para considerar que demoramos 1 dia para fazer acesso a memória, levaríamos cerca de 27 mil anos para acessar o disco, é uma forma de demonstrar o quanto o acesso a memória é mais rápido e como a disco é uma "eternidade", paralelo entre o tempo de 1nsec e 100sec dos reais tempos de acesso.



Este acesso associado às interrupções são importantes para os sistemas operacionais modernos pois, tendo em vista uma arquitetura von Neumann, a gente divide a memória em 8 partes sendo uma o kernel, e as outras 7 são programas em execução (processo).



O kernel, toda vez que tiver entrada ou saída, como o processador vai demorar 27 mil anos, ele vai pegar o estado (conteúdo dos registradores) armazenar na memória e evitar os 27 mil anos, como ele executa um processo só de cada vez ao invés do programa ficar em loop esperando ser executado é feito uma interrupção pelo kernel para não ficar em espera ocupada. Por que fazemos isso?



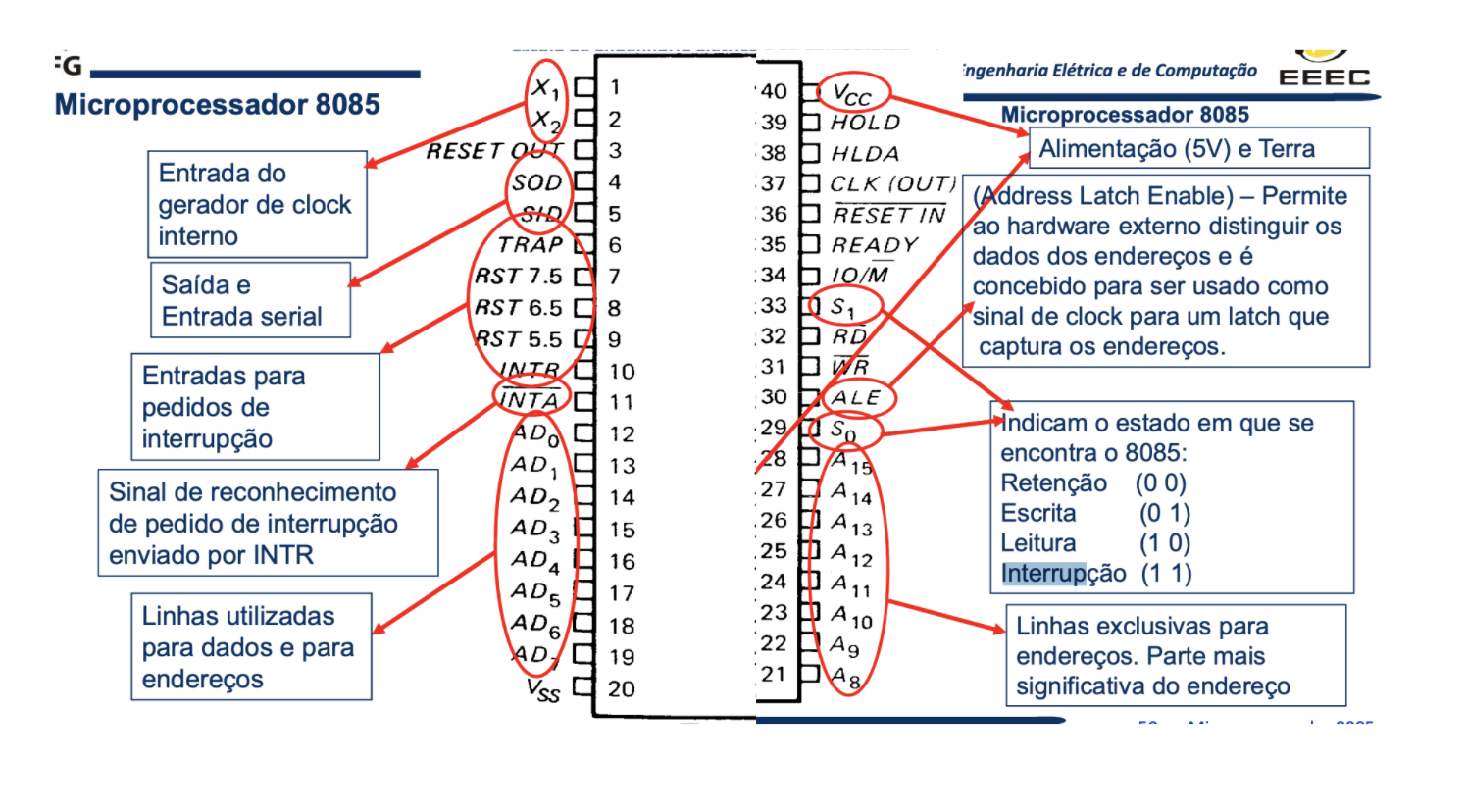
Relaciona-se a imagem, fazemos isso pois a entrada e saída é custosa, e para a cpu não ficar ociosa já deixamos outro processo executando, mas, por que não mandar a fita direto pra saída? pelos 27 mil anos, a cpu gerencia os recursos e processos da máquina para que os processos sejam executados extraindo o máximo possível dos recursos. Os processos são programas em execução e as threads são onde esses processos rodam, podendo acontecer vários "ao mesmo tempo" já que a memória dividida guarda o estado dos registros para serem facilmente re-acessados, parecendo ser simultâneo, enquanto na verdade só é muito rápido.

Vídeo explicando:

https://youtu.be/JIR6yeh\_BfE

Questão 2-

Quando acionado a tecla que vai nos direcionar para a interrupção no pino RST7, primeiro vai ser acionado e o INTR vai mostrar a vontade do usuário de causar a interrupção.



A partir desse momento o controlador termina a atividade que estava em progresso e armazena o estado corrente do processador, para restaurar, no retorno, a situação no momento à interrupção, depois desvia o controle de execução para Rotina de Serviço de Interrupção, as informações são armazenadas em uma pilha, primeiro da o PUSH no PC que é o program Counter, ou seja armazena na pilha o endereço da instrução seguinte ao da instrução CALL; O conteúdo do PC é alterado para conter o endereço da subrotina e o Stack é iniciado na posição alta de memória para evitar que o programa seja destruído pelas informações da pilha.

Em resumo, ativamos o RST7, ele mostra a intenção de causar uma interrupção, o processador se encarrega de chamar o PC para voltar para a atividade que já estava sendo executada, cria-se uma pilha pra registrar o estado do processo que era o principal, ditando o tamanho pelo StackPointer que dá o push nas informações, é chamado o comando RET e assim acaba a rotina de interrupção.

Link do vídeo:

https://youtu.be/hP8DcB1KlWs