IMPLANTAÇÃO DE UM SISTEMA DE TEMPO COMPARTILHADO EM UM MINICOMPUTADOR

Edil Severiano Tavares Fernandes

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIA (M.Sc.).

Aprovada por:

Prof. Pierre-Jean Lavelle (COBRA)

Prof. Célio Cardoso Guimarães (UNICAMP)

Prof. Nelson Maculan Filho (COPPE-UFRJ)

RIO DE JANEIRO

ESTADO DO RIO DE JANEIRO - BRASIL

DEZEMBRO DE 1976

D E D I C A T Ó R I A

À

Marialice

A G R A D E C I M E N T O S

Ao Professor Pierre-Jean Lavelle pelas idéias e pela paciente orientação ministrada.

Ao Professor Célio Cardoso Guimarães pelas valiosas sugestões.

Ao Professor Nelson Maculan Filho pelo estímulo e colaboração.

Aos colegas do Programa de Engenharia de Sistemas e Computação pelo apoio que recebi, em especial ao Professor Gerhard Schwarz.

À Angela pela dedicação no serviço datilográfico e ao Sr. Muniz da Silveira na parte de revisão.

À COPPE pelos recursos oferecidos durante a execução deste trabalho.

R E S U M O

Neste trabalho estudamos como implementar um sistema de Tempo Compartilhado num minicomputador dotado de disco.

O sistema é do tipo conversacional e é provido de um mecanismo simulador de memória virtual.

Sua construção foi orientada para as aplicações encontradas nas Universidades. Os programas dos usuários são interpretados. Esse procedimento visa evitar que erros cometidos por um usuário venham - durante a execução - descontinuar o funcionamento do sistema, acarretando com isso na perda de todas as tarefas que estejam em execução.

O trabalho ficou dividido em 3 etapas. Na primeira parte, fazemos um levantamento do que precisamos para que nosso objetivo seja alcançado. Na segunda, fazemos breve discussão sobre memória virtual. Finalmente na terceira etapa descrevemos a estrutura do sistema, como o usuário interage com o sistema e a implementação no minicomputador utilizado.

ABSTRACT

We study the implementation of a time-sharing system on a minicomputer with disk facilities. The system presented is conversational with virtual memory simulation.

Its construction was oriented towards University applications. User's programs are interpreted. This avoids discontinuation of the system's operation due to run-time errors in the user's program, with consequent loss of all jobs being executed.

This work is divided into three main parts. The first part studies the needs of the system. The second part discusses briefly the characteristics of virtual memory. Finally, the third part describes the system's structure, how the user interacts with the system, and the actual implementation of the minicomputer.

I N D I C E

Páginas

Páginas

Páginas

I N T R O D U Ç Ã O

Entre os projetos que orientaram as atividades acadêmicas do Programa de Engenharia de Sistemas e Computação da COPPE/UFRJ, nos dois últimos anos, havia um, cujo objetivo primordial era o da construção de um "Laboratório de Ensino da Computação". A idéia era a de fornecer ao minicomputador da C.I.I. , MITRA-15, disponível no Laboratório do Programa "Software" que permitisse:

- Usar conversacionalmente linguagens de programação do tipo Algol 60;
- A presença de diversos usuários explorando os recursos do sistema, em tempo compartilhado ("TIME-SHARING");
- A criação de um editor de textos;
- Colocar um interpretador do computador "MIX" (descrito por Knuth) à disposição dos interessados;
- Etc.

Inúmeras são as vantagens oferecidas por esse projeto. Entre outras destacamos:

a) Crescimento dos recursos disponíveis às pessoas em fase de aprendizado ou de trei

namento em assuntos relacionados com a área de computação;

b) Incremento na pesquisa de sistemas em tempo real, multiprogramado etc., por parte dos integrantes do Programa com o consequente aparecimento de diversos temas de tese, etc.

O presente trabalho representa uma pequena parcela desse projeto. Nele estudamos como implementar um sistema de "TIME-SHARING" - (dedicado a supervisão de tarefas interpretadas) - em um minicomputador dotado de disco.

CAPÍTULO 1

OBJETIVOS DO SISTEMA

1.1. O QUE DESEJAMOS

Desejamos construir um sistema que permita a exploração dos recursos da máquina por diversas pessoas "simultaneamente".

Como em nosso minicomputador existe apenas uma U.C.P. (Unidade Central de Processamento), essa simultaneidade é aparente. Ela poderá ocorrer de fato, no decurso de operações de E/S (Entrada e Saída) enquanto a U.C.P. (ao mesmo tempo) executa outros tipos de instruções.

O fato de dispormos de apenas uma U.C.P. e desejarmos que diversas pessoas sejam atendidas, uma das soluções é de que elas usem o sistema em tempo compartilhado.

Nesses sistemas, fornecemos a cada usuário pequenos intervalos de tempo de U.C.P., para que eles explorem os recursos. Uma vez esgotado o prazo, novo usuário é atendido e assim sucessivamente. Um mesmo usuário só receberá mais tempo quando todos os outros já foram contemplados.

É bastante comum que determinados usuários, em algumas ocasiões, não participem do rateio de tempo de U.C.P.. Isso acontece quando eles necessitam de recursos que não envolvam

dedicação exclusiva da U.C.P., como ocorre por exemplo na transmissão de caracteres durante as operações de E/S.

Para um melhor controle, as tarefas serão organizadas em filas de espera. Teremos assim diversas filas. As tarefas dos usuários que estão em uma fila, estão aguardando o mesmo recurso. À medida que uma dessas necessidades já foi satisfeita (ou apareceu outra) , mudamos a tarefa de fila. Esse mecanismo está melhor explicado a seguir.

1.2. ESTADOS ASSUMIDOS PELA TAREFA DE UM USUÁRIO

Conforme veremos em (2.1) cada usuário possui o seu terminal. Até o momento da chegada de um usuário ao terminal, ele estará na fila dos "fora de uso".

Quando o usuário deprime as teclas de controle, pedindo recursos necessários para que sua tarefa seja executada, ele irá para a fila dos que estão "aguardando trecho de memória". Após ter sido providenciado espaço para a execução do seu programa, ele novamente mudará de fila. Desta vez irá para a fila dos que estão "aguardando tempo de U.C.P." para que seu programa seja processado. Nessa fila permanecerá compartilhando da U.C.P. até que a ocorrência de um evento provoque uma nova mudança.

Os eventos possíveis são:

- Necessidade de uma E/S, nesse caso o usuário irá para a fila dos que estão "aguardando E/S";
- Necessidade de espaço na memória, usuário irá para a fila dos que estão "aguardando memória" (ausência de uma página);
- Usuário que "conversar com o sistema", tarefa irá para a fila dos que estão "aguardando diálogo";
- Encerramento na execução da tarefa, usuário irá para a fila dos que estão "fora de uso";
- Erro no programa do usuário, obrigando o encerramento, usuário irá para a fila dos que estão "fora de uso";
- Esgotamento do tempo fornecido, usuário continuará na mesma fila, sendo atendido após todos os outros.

CAPÍTULO 2

REQUISITOS DO SISTEMA

2.1. TERMINAIS

Sob o ponto de vista do sistema, há uma associação biunívoca entre um terminal e o usuário correspondente. Por isso é que no presente trabalho usamos os termos: usuário, terminal, tarefa de um usuário como significando - a tarefa do usuário que está em um determinado terminal. Por exemplo em (1.2) quando falamos: "usuário "i" irá para a fila dos que estão "aguardando E/S" queremos dizer com isso:

> "a tarefa do usuário que está no terminal "i", irá para a fila dos usuários cujas tarefas estão "aguardando E/S"".

A função desempenhada por um terminal é dupla:

- periférico responsável pelas operações de E/S de um usuário;
- periférico responsável pela interação "usuário x sistema".

Essa dupla função contribuiu acentuadamente para a grande popularidade dos sistemas de tempo compartilhado .

Pois:

- considerando que cada usuário tem o seu periférico para fazer E/S, o sistema vai liberando os registros (as linhas de impressão) tão logo eles estejam prontos, não esperando (como é feito em outros sistemas) que sejam impressos somente quando todo o arquivo estiver gerado e ainda houver um periférico (impressora) disponível no sistema.

Como o usuário vai recebendo os resultados mais rapidamente, pode inclusive tomar medidas corretivas, se for o caso.

- a segunda função de um terminal-orgão de comando - permite que o sistema seja conversacional. Isso é muito útil. A qualquer momento o usuário pode solicitar por exemplo que sejam impressos os conteúdos dos registros ("Trace"), ou que determinada posição de memória deve ser alterada ou ainda que seu programa pare quando passar por uma determinada posição ("Break point") etc. Essas facilidades permitem a construção de poderosas técnicas de depuração de programas, atingindo perfeitamente nossos objetivos no ensino e trei-

namento em áreas da computação.

2.2. LIMITAÇÕES NO NÚMERO DE USUÁRIOS

O número máximo de pessoas que podem explorar o sistema em um determinado instante, depende antes de mais nada da quantidade de terminais ligados à máquina.

Surge então uma pergunta: quantos terminais podem fazer parte da configuração de um "Sistema de Tempo Compartilhado"?.

Se soubermos responder corretamente essa questão então grandes serão as possibilidade de termos um sistema bem balanceado, satisfazendo nossas expectativas.

No dimensionamento dessa quantidade, deveremos levar em consideração dentre outros os seguintes fatores:

a) Características das aplicações - informações do tipo - quanto tempo de U.C.P. é consumido entre duas operações de E/S; qual a média e a variância dos intervalos entre duas operações E/S; idem para duas instruções que referenciem posições que não estejam presentes na memória principal - são vantajosas, pois o fato de

um usuário pedir uma E/S ou referenciar uma posição ausente, impede que ele utilize tempo de U.C.P., favorecendo dessa forma os usuários restantes, que não precisarão dividir o tempo com mais uma pessoa . Em sistemas do tipo conversacional, podemos ter um número maior de terminais, pois as aplicações apresentam uma percentagem razoável de operações de E/S, sobrando dessa maneira bastante tempo de U.C.P. para (possivelmente) os raros usuários que estão (em um certo instante) na fila dos que estão "aguardando tempo de U.C.P.". Como esses raros usuários provavelmente nem chegarão a esgotar sua fatia de tempo, devido à necessidade de uma operação de E/S, o sistema sempre terá alguma folga.

b) Velocidade da U.C.P. - quanto mais veloz for a U.C.P. um número maior de terminais poderá ter um melhor atendimento se as características de suas aplicações forem uniformes.

c) Limitações impostas por "hardware" - a arquitetura da máquina, pode fixar o número máximo de terminais permitidos.

d) "OVERHEAD" do sistema - quanto tempo de U.C.P. é consumido para iniciar uma E/S ;

para salvar e restaurar os registros quando da mudança de usuários, etc.

e) Velocidade de transmissão do terminal - quanto mais lenta for a transmissão, mais terminais poderão ser ligados.

2.3. NECESSIDADE DE MEMÓRIA VIRTUAL

Normalmente o tamanho da memória num minicomputador é reduzido. Como queremos que diversas tarefas estejam presentes na memória (tantas quantas forem os terminais), necessitaremos de implantar um sistema com memória virtual. Foi por isto que restringimos a implementação do sistema a apenas minicomputadores que fossem dotados de dispositivos de armazenamento com acesso aleatório. Assim, conforme veremos no capítulo 3, parte do programa de um usuário fica na memória principal e o restante na memória auxiliar.

2.4. NECESSIDADE DE INTERPRETADORES

Conforme vimos na introdução, um dos objetivos do projeto era o de munir o laboratório do Programa com técnicas que facilitassem o ensino da Computação. Espera-se então que os usuários serão, em geral, ou pessoas sem muita experiência, ou

ainda pessoas que usarão o sistema ou para testar algoritmos ou ainda para executar programas que rodarão apenas uma vez.

Este é o tipo dominante de aplicações encontradas nas Universidade. Para esses tipos de usuários e/ou aplicações, é mais eficiente interpretar seus programas ao invés de gerar código de máquina. Por que perder tempo de máquina gerando código para um programa que rodará apenas uma vez? Por que gerar código de máquina se é grande a probabilidade de ocorrência de um erro cujo nível de severidade não vai permitir posteriormente a execução do programa?

Há ainda uma consideração que deve ser feita. É possível (e muito frequente) que programas-fonte que passaram sem ocorrências pela fase de tradução, quando em tempo de execução, por erro de lógica invadam áreas não permitidas. Nas máquinas dotadas de dispositivos de proteção de memória um incidente desses, simplesmente, faria com que o programa infrator fosse descontinuado, ficando assim a memória resguardada dessas tentativas de destruição.

Todavia, nem todos os minicomputadores estão munidos desses dispositivos. Então, seria muito comum nessas máquinas, quando da execução de um desses programas, que o supervisor fosse destruído, acarretando com isso perda dos outros programas que estivessem no sistema.

Esses foram os argumentos preponderantes na escolha de interpretadores em nosso sistema.

2.5. NECESSIDADE DE INTERRUPÇÕES

Em nosso sistema são executados dois tipos de tarefas. Existem as tarefas cuja finalidade é a de fazer a manutenção do sistema, como por exemplo: as rotinas de E/S, as rotinas que trocam usuários de filas de espera, a rotina de distribuição de tempo de U.C.P., etc. O outro tipo de tarefas constitui-se daquelas trazidas ao sistema pelos usuários. As tarefas de manutenção devem estar em um nível mais alto de prioridade. O sistema deve optar pela mais prioritária na hora de escolher quem receberá o controle da U.C.P..

Dependendo do nível de prioridade da tarefa em curso, há a necessidade em algumas ocasiões de interrompê-la . Por exemplo:

- para evitar que um usuário monopolize a máquina, é necessário que sua tarefa seja interrompida, quando do esgotamento do tempo recebido, para que o sistema processe a troca de usuários;
- no término de uma operação de E/S, a tarefa do usuário corrente precisa ser paralisada, de modo que o sistema recoloque a tarefa que estava aguardando a E/S - na fila dos que estão aguardando U.C.P.. Após a troca de fila, o sistema devolverá o

controle à tarefa paralisada.

Como podemos observar pelos dois exemplos acima, interromper uma tarefa nada mais é do que desviar o controle da máquina para um outro programa. Precisamos garantir, quando desse desvio, que o contexto (ou seja os conteúdos dos registros e dos indicadores da máquina) relativo ao programa interrompido seja preservado, de maneira que possa ser reiniciado do ponto onde ocorreu a interrupção.

Para que esse reinício seja possível, basta que os registros e indicadores da máquina sejam restaurados com aqueles antigos valores que foram preservados.

Uma vez vista a necessidade de interrupções, veremos a seguir como isto pode ser alcançado.

2.6. COMO INTERROMPER AS TAREFAS

As máquinas atuais normalmente já são dotadas de sistemas de interrupções. Nesses sistemas, as interrupções são organizadas em níveis prioritários. Quando a máquina é sensibilizada pelo aparecimento de uma interrupção de nível mais alto que o atual, a troca de contexto (2.5) é feita automaticamente (por"hard ware"). O programa associado ao nível mais atual é suspenso, sendo ativado o programa que está relacionado com a interrupção ora chegada.

Após a "resolução" (ou seja a execução do

programa correspondente) dessa interrupção, haverá novamente uma preservação de contexto, e o programa de nível mais alto (que estiver ativado, é claro) assumirá o controle da máquina.

Nas máquinas desprovidas de sistemas de interrupções, Wilkes [2] sugere que sejam acrescentados ao repertório das instruções de todas as tarefas, ramificações condicionadas ao conteúdo de alguns registros. Essas instruções seriam acrescentadas pelo próprio programador, em posições adequadas de seu programa. Essas posições devem estar situadas em trechos frequentemente executados de forma que o teste, digamos do término de uma operação de E/S, seja feito regularmente.

O fato de deixar ao programador a decisão de informar onde pode seu programa ser interrompido não é muito seguro.

Nos interpretadores isto pode ser facilmente obtido, bastando, para tanto, colocar em posições estratégicas, digamos antes da interpretação da próxima instrução, o teste e a respectiva ramificação condicional.

Nos programas que não são interpretados, sugerimos que os tradutores sejam alterados de forma que pudessem fazer a geração dos códigos adicionais em pontos executados regularmente, como por exemplo, no interior de blocos lógicos. Garantindo dessa forma que regularmente o teste fosse executado, interrompendo caso necessário o presente programa.

2.7. MÓDULOS DO SISTEMA

A idéia de modularmos o sistema tinha como objetivos:

- a atribuição de níveis de prioridade na execução dos tipos de tarefas (2.5);
- minimizar as dificuldades na implementação do sistema, pois o problema ficaria dividido em partes menores.

Assim, deveríamos ter um módulo responsável pela gerência da memória, outro pela interpretação dos programas e finalmente um módulo superior, que além de controlar os dois anteriores, controlaria as operações de E/S dos usuários, forneceria intervalos de tempo de U.C.P. para eles, controlaria o fluxo de informações entre a memória principal e a auxiliar, interagia com os usuários fornecendo-lhes informações, distribuindo entre eles os recursos do sistema etc.

Desde o início do projeto, verificamos que um dos pontos críticos na implementação do sistema, era a comunicação (passagem de parâmetros) que precisava ser estabelecida entre os módulos.

As palavras onde ficarão os parâmetros devem ser acessíveis a todos os módulos. Cada "bit" dessas palavras (às vezes grupos de "bits") encerra as informações que um módulo deseja transmitir ao outro. Por exemplo, existe um "bit"- posicionado pelo supervisor - que informa ao interpretador se deve ou não

continuar atendendo ao mesmo usuário. Há um outro - que é manipulado pelo interpretador - cujo objetivo é o de informar ao supervisor que a mudança de usuário já pode ser procedida etc.

De uma forma geral existem "bits" que são posicionados por apenas um dos módulos, dirigidos para outro módulo, como também existem outros "bits" que podem ser manipulados por todos os outros módulos.

CAPÍTULO 3

MEMÓRIA VIRTUAL

3.0. CONCEITO DE MEMÓRIA VIRTUAL

Conforme vimos em (2.3), a limitação no tamanho da memória impedia a presença integral de todos os programas que estivessem explorando o sistema.

A saída para esse impasse é a simulação(por "software") da técnica da memória virtual. Essa técnica consiste em conservar na memória principal, apenas parte (ou partes) do programa que está atualmente usando a U.C.P.. Quanto aos outros programas, poderão estar parcialmente na memória ou não.

Para implementar memória virtual, os fabricantes podem empregar ou a filosofia da segmentação ou a filosofia da paginação. Contudo existem sistemas que combinam as duas filosofias na implantação da técnica.

3.1. A SEGMENTAÇÃO

Na segmentação, os programas são divididos em unidades logicamente independentes denominados de segmentos ou módulos. Um conjunto de dados e um subprograma são alguns exem -

plos de segmentos. O tamanho do segmento é variável. Quanto maior for o tamanho do conjunto de dados (ou do subprograma) maior será o tamanho do segmento correspondente.

Em tempo de execução de um programa o sistema só precisa garantir a presença na memória principal, dos segmentos envolvidos.

Graças à sua concepção, a filosofia da segmentação permite:

- que não haja desperdício de trechos de memória;
- recolocação dinâmica de memória;
- modularidade de programas;
- que estruturas de dados cresçam dinamicamente;
- que os módulos sejam protegidos dos acessos não autorizados;
- que os segmentos sejam compartilhados por diversos programadores.

Para que essas vantagens sejam possíveis, a segmentação conta com o apoio de um poderoso mecanismo de formação de endereços. Esse mecanismo é constituído pela integração do "software" com o "hardware". Organick |[14]| e Daley |[8]| descrevem as estruturas de dois sistemas segmentados.

3.2. A PAGINAÇÃO

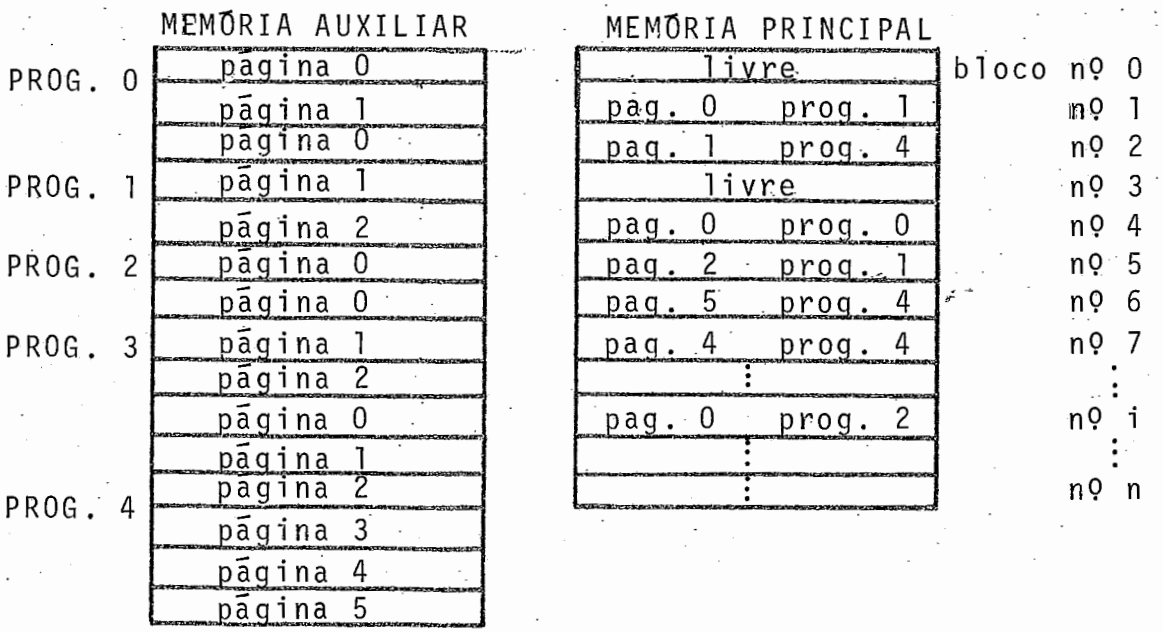
Na paginação, a memória é dividida em blocos de tamanho constante. Os programas também são divididos em trechos (cujo tamanho coincide com o do bloco) denominados de páginas. Assim, se por exemplo a página for constituída de 100 palavras (ou 100 linhas), então um programa com 325 palavras seria dividido em 4 páginas. Na primeira página - de número zero - estariam as 100 primeiras palavras, na segunda as 100 subsequentes, na terceira mais 100 e finalmente na quarta as 25 restantes, ficando fora de uso portanto 75 palavras. Cada uma das páginas pode conter dados ou instruções. Como o que limita uma página é somente o tamanho, unidades logicamente independentes poderão estar na mesma página. Isso é desvantajoso pois durante a execução de um bloco lógico, todo o restante da página também estará (indevidamente) ocupando a memória principal.

Outra desvantagem apresentada pelo tamanho fixo de uma página, é aquela em que um bloco lógico ultrapassa a fronteira de 2 páginas. Quando isso acontece, ou o sistema mantém as 2 páginas na memória, ou fica trocando-as. As duas possibilidades são indesejáveis. Por exemplo, supondo que o tamanho da página é igual a 1000 palavras, e que o bloco lógico em questão é constituído por um conjunto de instruções que ocupam apenas 20 posições de memória situadas em duas páginas distintas, e que tal bloco deverá ser executado inúmeras vezes então se o sistema mantiver as 2 páginas presentes na memória principal estará havendo um desper-

dício de 99% de memória. Por outro lado, se o sistema alternar a presença das duas páginas, então para cada execução do bloco lógico - ou seja para cada "loop" - haverá a necessidade de copiar a segunda página na memória auxiliar e depois ler para o bloco copiado a primeira página. Procedimento recíproco deverá ser adotado quando for necessário executar as instruções da segunda página. Considerando que o bloco será executado inúmeras vezes, podemos prever os efeitos causados pelo tempo necessário para que as páginas sejam trocadas entre os dois níveis hierárquicos de armazenamento.

A paginação não permite o crescimento dinâmico das estruturas de dados nem que trechos de um programa sejam compartilhados por outros.

A seguinte figura mostra um possível instantâneo dos dois níveis de armazenamento.



Podemos observar na figura que:

- é possível que nem todas as páginas de um programa estejam na memória principal;
- as páginas de um programa podem estar em blocos não adjacentes;
- é possível que um programa não tenha nenhuma página na memória.

Como na segmentação, cada programa em execução possui seu mapa (ou tabela) de páginas conforme mostra Denning [3]. Esse mapa é formado pelo sistema operacional quando do carregamento do programa. Cada mapa terá tantas entradas quantas forem as páginas. Essas entradas informam em qual dos dois níveis de armazenamento está a página e qual o seu endereço (ou em que setor do disco encontra-se a página).

Na paginação os endereços são constituídos por pares, que indicam o número da página e qual a posição dentro da página (deslocamento) que está sendo referenciada.

Esses endereços podem ser formados quando da tradução do programa.

Quando da execução dessa instrução, o mecanismo responsável pelo cálculo de endereço físico, usaria o número da página para saber qual das entradas do mapa deve ser olhada. Ao examinar a entrada o mecanismo geraria uma interrupção se a página estivesse ausente. Caso contrário, o endereço de bloco (associada à página) seria adicionado ao deslocamento (o segundo número do par) formando o endereço físico.

3.3. SEGMENTAÇÃO OU PAGINAÇÃO

Ao confrontarmos estas duas filosofias, logicamente somos levados a optar pela segmentação em virtude de melhor aproveitamento de memória que ela faz.

Todavia as linguagens de programação que seriam usadas no sistema (a curto prazo) não apresentavam as características das linguagens estruturadas.

Como dividir em segmentos um programa escrito em uma linguagem não estruturada?

Essa questão apesar de não ser trivial é passível de resolução em diferentes níveis. Supondo que temos o código-fonte de um programa, então os segmentos seriam o que as sub-rotinas são para o "FORTRAN" ou os parágrafos de "COBOL" ou as "PROCEDURES" do "ALGOL" etc. Por outro lado se possuirmos somente o código da máquina, a solução seria formar segmentos à medida que um laço ("LOOP") fosse fechado ou seja à medida que encontrassemos ramificações para posições anteriores (desde que não houvesse incompatibilidade lógica).

É claro que para fazermos isso, acarretaria na construção de novos tradutores, novos editores e de novos carregadores.

A paginação apesar de limitar bastante os recursos de programação foi a filosofia que empregamos na simulação da memória virtual.

Os fatores preponderantes nessa escolha fo -

ram:

- a filosofia da paginação é mais simples de ser implementada;
- as linguagens não sendo estruturadas obrigavam-nos a construir tradutores especializados.

3.4. O TAMANHO DA PÁGINA

Quanto maior for o tamanho da página, menor a probabilidade de que posições ausentes sejam referenciadas. Por outro lado, quando da execução de blocos lógicos há um aumento na percentagem da má utilização da memória, como já vimos em (3.2).

Nem sempre a última página de cada programa está completamente preenchida. Em média cada programa ocupa apenas 50% da capacidade da última página, ficando a outra metade (desocupada) impedida de ser utilizada pelo sistema. Quanto maior o tamanho da página, maior será o desperdício dessa última página.

Se reduzirmos o tamanho da página estaremos aumentando o tamanho do mapa de cada programa, perdendo portanto espaço de memória.

É muito vantajoso que o tamanho da página seja uma potência de base de numeração que a máquina emprega, pois o mecanismo que calcula os endereços fica muito simples.

Em nossa simulação de memória virtual, o que pesou na escolha do tamanho da página foram as características da máquina.

Considerando que no "MITRA-15":

- só é permitido endereçar diretamente até 128 palavras;
- o disco é dividido em setores de 128 palavras.

Optamos então por páginas constituídas de 128 palavras.

Assim, eliminamos a necessidade de que os registros sofressem o processo de blocagem e desblocagem nas operações de leitura e gravação.

CAPÍTULO 4

ESTRUTURA DO SISTEMA

4.0. INTERAÇÃO USUÁRIO/SISTEMA

4.0.0. O CARACTERE "BREAK"

Os terminais que receberão os usuários estão ligados à máquina por intermédio de linhas "Half-Duplex". Nessas linhas uma nova transmissão só poderá ser processada se a anterior já foi concluída.

Conforme vimos (2.1), os terminais além de processarem as operações de E/S, devem estar sempre preparados para a recepção de um caractere de controle (enviado pelo terminal) no caso de que o usuário queira interagir com o processamento de sua tarefa. Como a aparição desse caractere pode ocorrer a qualquer momento, a linha sempre estará ocupada, aguardando uma possível transmissão. O fato de estar a linha ocupada impede que o programa execute operações de E/S. Uma possível solução (muito deselegante), seria a de orientar os usuários, para que regularmente deprimam a tecla de "Retorno do Carro" (que é equivalente a um fim de transmissão) para que a linha ficasse desocupada, permitindo dessa forma que as operações de E/S fossem executadas.

Geralmente os terminais são munidos com a chave "BREAK" (Wilkes |[2]|).

Em nosso minicomputador, ao comandarmos uma recepção do caractere "BREAK", a linha ficará ocupada também. Mas, se posteriormente comandarmos uma operação de E/S, a recepção do "BREAK" é suspensa, vigorando somente quando terminar a operação de E/S.

4.0.1. OS COMANDOS

Sempre que um usuário desejar interagir com o sistema, deverá deprimir a tecla que enviará o caractere "BREAK", antes de transmitir o comando propriamente dito.

Os comandos possíveis são os seguintes:

?INTERPRETE nome - o usuário deseja que sua tarefa seja executada por interpretadores (conforme especificado pelo "nome").

?DESCONTINUE - indica que o usuário deseja descontinuar a execução de sua tarefa.

?MODIFIQUE p v - indica que a posição "p" de sua tarefa deve receber o valor "v"

?DUMP i s - essa mensagem informa que o trecho de memória compreendido entre o limite inferior "i" e o limite superior "s" deve ser impresso.

?TRACE i s - informa que os "registros do interpretador" devem ser

impressos após a execução de qualquer instrução que esteja situada no trecho de memória compreendido entre o limite inferior "i" e o limite superior "s".

?PARE p - informa que após a execução da instrução localizada na posição "p" deverá haver uma paralisação no processamento da tarefa. A execução da tarefa prosseguirá se um novo comando de "PARE p" ou um comando de "SIGA" for transmitido pelo usuário.

?SIGA - serve para informar que o precessamento deve prosseguir.

Outros comandos poderão ser implantados de acordo com as necessidades que forem surgindo com a utilização do sistema.

4.0.2. A TAREFA DE UM USUÁRIO SOB O PONTO DE VISTA DO SISTEMA

Com o objetivo de termos uma melhor visão do sistema, descreveremos em seguida as providências tomadas pelo sistema, para que a tarefa de um usuário seja executada.

Inicialmente, o terminal que servirá de base para a execução da tarefa de um usuário deverá estar fora de uso.

Como a chegada de um usuário pode ocorrer a qualquer instante, o supervisor sempre estará aguardando a transmissão de um caractere "BREAK".

Quando o usuário chega ao terminal, e envia

o caractere "BREAK", o supervisor é reativado. Ao verificar que o "BREAK" foi transmitido por um usuário que estava fora do sistema, o supervisor após comandar uma operação de entrada, mudará o usuário para a lista dos que estão aguardando E/S (para o supervisor (5.1.3)).

Quando a operação de entrada estiver concluída, o supervisor testará se a mensagem transmitida é igual ao comando "INTERPRETE nome". Se diferente então ele imprimirá uma mensagem de "COMANDO INVÁLIDO" no terminal do usuário, e comandará uma nova operação de entrada. Se a mensagem for igual, então o supervisor passará o usuário para a fila dos que estão aguardando trecho de memória.

No momento que o trecho de memória para o usuário estiver disponível, o supervisor ao verificar que a entrada do diretório (4.1), relativa ao usuário, está vazia então destinará o bloco para abrigar a página-contexto (4.2). Para isso, a entrada receberá o número inicial de acessos e o apontador para o bloco (5.1.4), e o subprograma que fornece valores iniciais às páginas-contexto será chamado.

Esses valores iniciais informarão em que posição da memória encontra-se o "BUFFER" do usuário (4.3), qual o número máximo de páginas que o programa poderá referenciar, qual o estado da tarefa, que o estado do interpretador é o de início de "LOADER" etc.

Quando o subprograma terminar, o supervisor colocará a tarefa do usuário no final da fila dos que estão aguardando U.C.P.. Essa fila é circular. Quando chegar a vez do usuário

em questão, o supervisor fornecerá uma fatia de tempo para ele e após ativar o módulo interpretador ficará desativado.

O módulo interpretador ao ser ativado, após a restauração dos registros com os valores contidos na página-contexto do usuário, irá examinar qual o valor da variável "estado do interpretador". Ao verificar que é necessário carregar o programa do usuário nas suas páginas (tarefa do subprograma chamado "LOADER") ativará o supervisor, informando que é preciso ler o código objeto via terminal.

O supervisor, ao ser ativado, irá isolar e analisar os parâmetros transmitidos pelo interpretador. Ao verificar o pedido do interpretador, mudará o usuário de lista, comandará uma operação de entrada e continuará atendendo os demais usuários.

Sendo encerrada a operação de entrada, o usuário irá para o final da lista dos que estão aguardando U.C.P.. O interpretador ao ser ativado para esse usuário, após as restaurações de praxe, investigará em que fase do "LOADER" encontra-se a tarefa. O código transmitido será testado; se válido, então esse código será carregado na página do usuário.

Como essa página não está presente na memória, o interpretador após modificar o valor da variável "estado do interpretador" e atualizar o valor da variável "página ausente" , ativará o supervisor para que forneça a página requerida. O supervisor após isolar e analisar os parâmetros, colocará o usuário na lista dos que estão aguardando trecho de memória.

Na ocasião em que o bloco de memória esti -

ver disponível, o supervisor após atualizar a entrada do mapa de memória (essa entrada é especificada pela variável "página ausente") com o número inicial de acessos e com o número do bloco onde está localizada a página, colocará a tarefa do usuário no início da fila dos que estão aguardando tempo de U.C.P..

O interpretador ao ser ativado colocará nessa página o código transmitido anteriormente pelo terminal, e ficará pedindo entradas e armazendo-as nessa página, até que todas as instruções da tarefa tenham sido transmitidas. O que delimita a última entrada é um registro especial, que informará qual a posição da primeira instrução a executar. Ao receber esse registro (que contém uma identificação de fim de programa) o interpretador atualizará a variável "I.A.R.", atualizará a variável "estado da tarefa" com o valor correspondente a "pronto para a execução" e ativará o supervisor.

Este ficará aguardando que o usuário transmita ou uma mensagem pedindo depuração (por exemplo "DUMP", "TRACE", "PARE p", etc.) ou um comando de "SIGA". Enquanto a mensagem não for concluída, a tarefa permanecerá na lista dos que estão aguardando o término de E/S, e o supervisor estará atendendo os outros usuários.

Em função da mensagem transmitida, o supervisor poderá imprimir a memória do usuário (caso de "DUMP"), ou associará valores às variáveis que demarcam os pontos onde o "TRACE" ou o "BREAKPOINT" ("PARE p") devem estar em operação.

Após a transmissão da mensagem "SIGA", a tarefa do usuário irá para o final da fila dos que estão aguardando

tempo de U.C.P..

A tarefa do usuário permanecerá nessa fila , recebendo fatias de tempo até que seja encerrada. Poderá haver mudanças de fila se uma operação de E/S for solicitada ou se o usuário transmitir um caractere de "BREAK" ou ainda se uma página ausente for referenciada.

4.1. O DIRETÓRIO

O objetivo do diretório é o de informar ao sistema onde pode ser localizada a página-contexto de cada usuário (4.2).

Para cada terminal há uma entrada no diretório. Essas entradas podem conter 3 tipos de informações.

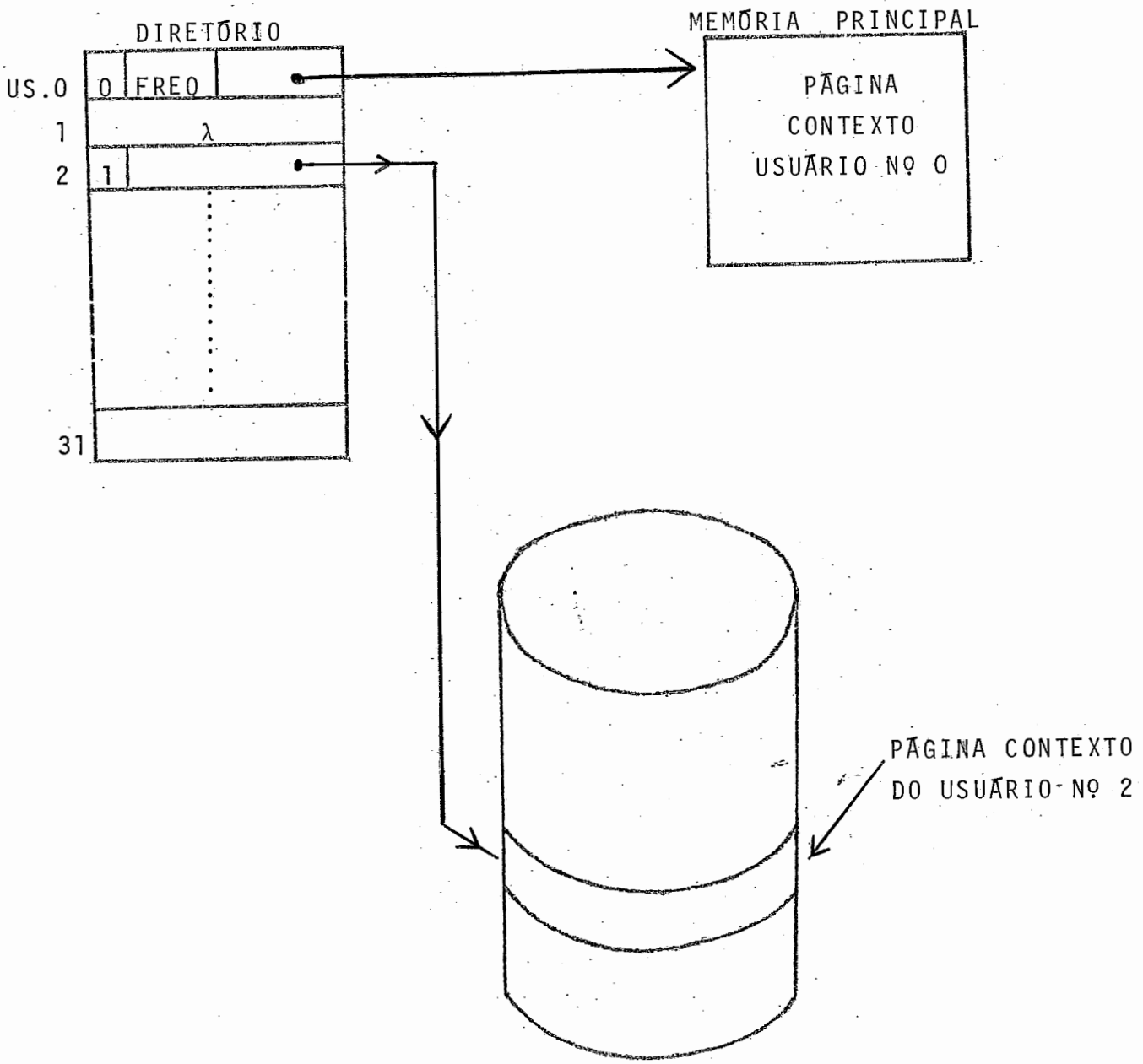
Estando o terminal fora de uso, então ela conterá λ. Quando o terminal está associado a um usuário então teremos duas possibilidades. A primeira é a de que esta entrada está apontando para o bloco de memória onde a página-contexto do usuário pode ser encontrada; nesse caso o conteúdo da entrada é um número não negativo. Porém, se a página-contexto estiver ausente da memória principal então o conteúdo será um número negativo cujo valor absoluto é o número do setor do disco onde a página está armazenada.

No caso de estar a página presente, apenas

parte da entrada é que aponta para o bloco de memória, o restante informa a frequência de acessos que foram feitos à página (5.2.).

O acesso a cada uma das entradas é feito pelo modo direto.

A figura seguinte mostra um esquema do diretório.



4.2. A PÁGINA-CONTEXTO

A página-contexto é a peça fundamental de cada tarefa. Nela estão preservadas as informações indispensáveis à execução da tarefa.

Essas informações interessam (ou seja são manipuladas) tanto aos módulos do sistema como ao usuário.

A seguir listamos algumas das informações que ela armazena:

- o número máximo de páginas que a tarefa pode solicitar;
- o número de páginas que a tarefa já referenciou;
- o mapa de páginas da tarefa;
- o endereço físico do "BUFFER" de E/S da tarefa (4.3.);
- os registros e indicadores que o interpretador deverá carregar quando a tarefa receber mais fatia de tempo de U.C.P.;
- quanto tempo de U.C.P. o programa já consumiu, quantas operações de E/S já foram processadas;
- qual o limite em que as rotinas de "TRACE" e de "DUMP" devem estar em operação;
- qual a última palavra da memória do usuário que foi processada pela rotina de "DUMP";

- em que posição a presente tarefa deve ser paralisada ("BREAKPOINT");
- qual a página ausente que o usuário está necessitando para que a sua tarefa prossiga;
- qual a hora do início da tarefa;
- qual o estado do programa sob o ponto de vista do supervisor;
- qual o estado do programa sob o ponto de vista do interpretador;
- etc.

Essas informações são comuns a todas as tarefas independentes do tipo de interpretador que elas estão empregando. Cada interpretador possui características que podem exigir informações adicionais na página-contexto.

É interessante ressaltar que o conteúdo de cada uma das entradas do mapa de páginas, assume os mesmos valores como se fossem entradas do diretório (4.1.). A única diferença é que o diretório aponta para a página-contexto, enquanto esta aponta para os trechos de memória relativos ao programa do usuário.

4.3. OS "BUFFERS"

Considerando que o terminal é o veículo responsável pelas operações de E/S (2.1), reservamos para cada tarefa um espaço de memória com a finalidade de servir de "BUFFER" de E/S. Dessa maneira, se uma tarefa desejar fazer uma operação de entrada , então os dados serão transmitidos do terminal para essa região . Se o requerido for uma operação de saída, então a transmissão será feita do "BUFFER" para o terminal.

Os interpretadores são informados do endereço do "BUFFER"-associado ao usuário corrente - através de uma das posições contidas na página-contexto (4.2).

Possuidores dessa informação, os interpretadores poderão manipular o conteúdo do "BUFFER", fornecendo o que a tarefa do usuário desejar.

O supervisor - como responsável pelas operações de E/S - também manipula o conteúdo dos "BUFFERS" conforme veremos (5.1.).

Em algumas ocasiões somente o supervisor terá acesso ao "BUFFER". Para isso basta que o terminal esteja sendo usado na sua segunda modalidade - orgão de comando - conforme vimos (2.1.).

Devido ao reduzido número de terminais disponíveis em nosso laboratório (apenas 4 conforme veremos (5.0.) optamos por "BUFFERS" estáticos, ou seja, independente da presença ou não de um usuário no sistema, sempre há um espaço de 37 pala -

vras reservado para cada um deles (nossos terminais estão capacitados para ler ou imprimir até 72 caracteres por linha, os dois outros servem para o retorno do carro e para o espacejamento do formulário).

É claro que essa opção não é permanente. Quando o número de terminais (adicionais) adquiridos for suficiente para justificar a outra alternativa, após algumas modificações, estaremos com "BUFFERS" dinamicamente relocáveis no sistema.

Nesse novo esquema, os "BUFFERS" estarão contidos nos blocos de memória, recebendo portanto o mesmo tratamento dispensado às páginas (trocas constantes entre os dois níveis de armazenamento).

As vantagens decorrentes de "BUFFERS" dinâmicos são:

- só precisam ser mantidos na memória os "BUFFERS" dos programas que estão processando uma operação de E/S; quanto aos outros poderão estar ou não;
- o "BUFFER" pode ser alocado em qualquer bloco disponível de memória.

4.4. BLOCO DE CONTROLE

Para que as operações de E/S sejam executadas no "MITRA-15" é necessário fornecer à sub-rotina alguns parâmetros. Esses parâmetros informarão o número de caracteres que deve

rão ser transmitidos, qual o periférico que participará da transmissão, qual o endereço do "BUFFER", qual a operação desejada (se entrada ou saída), qual o nível de interrupção que deverá ser ativado no final da transmissão, qual o tipo de transmissão (se binária ou alfanumérica), qual o setor (no caso de E/S no disco) etc.

A rotina responsável pela operação por sua vez também fornece valores a esses parâmetros, informando qual o estado da operação em cada instante, se a operação foi bem sucedida, etc.

Com o objetivo de controlar o estado de cada um dos terminais, criamos uma estrutura. Essa estrutura é constituída por duas filas. A primeira fila é formada pelos blocos que controlam as operações de E/S. A segunda é composta pelos blocos que reconhecem a transmissão de caractere "BREAK" (4.0.0).

A cada instante, pela simples inspeção de um caractere de cada bloco (exatamente aquele que informa qual o estado da operação), o supervisor pode tomar a decisão adequada.

4.5. FILAS DE ESPERA

Conforme (1.2) o sistema agrupa os usuários que apresentam o mesmo tipo de necessidade em filas de espera. Fisicamente existe apenas uma tabela. Essa tabela deve conter tantas entradas quantos forem os terminais. Cada entrada é composta por

3 campos. Um dos campos contém o código relativo à fila de espera associada ao usuário. Os outros 2 campos ligam a entrada (ou seja o nó) à estrutura em que o usuário está. A primeira ligação aponta o nó antecessor e a segunda para a entrada sucessora. Todas as estruturas são circulares. Para cada uma delas existe um apontador que indica qual o nó cabeça da fita.

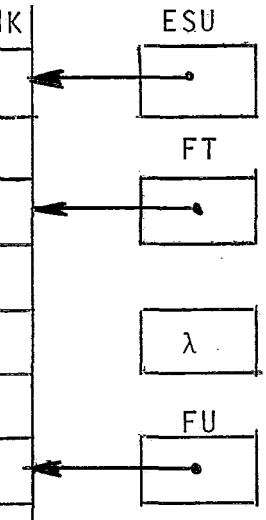
Com essa estrutura, o sistema tem acesso direto a qualquer nó, podendo trocar os nós entre as listas sem ter que percorrer a estrutura partindo do nó-cabeça. Outra vantagem do acesso direto, é que o sistema pode determinar imediatamente qual o estado (qual a fila) da tarefa de um usuário.

Considerando que as estruturas são semelhantes, as rotinas de remoção e inserção são as mesmas para qualquer uma das filas de espera.

Os algoritmos utilizados (5.1.8) para inserção e remoção são os descritos por Knuth [4]. O algoritmo que roda uma lista também é muito simples, para tanto basta fazer com que o conteúdo do apontador receba seu sucessor.

A figura a seguir exibe a estrutura encontrada no sistema. Para facilitar, colocamos no campo de informação os símbolos "FU", "AM", "ESU", "ESS", "ET" para representar "fora de uso", "aguardando memória", "E/S para o usuário", "E/S para o supervisor" e "aguardando tempo de U.C.P.,respectivamente". O campo "LLINK" aponta para o nó antecessor e o "RLINK" para o sucessor.

USUÁRIO	NO	LLINK	INFORMAÇÃO	RLINK
	0	4	ESU	4
	1	5	FT	2
	2	1	FT	5
	3	6	FU	6
	4	0	ESS	0
	5	2	FT	1
	6	3	FU	3



4.6. TEMPO DE U.C.P.

Nos sistemas dotados de memória virtual, o "OVERHEAD" é bastante ampliado pela constante troca de trechos dos programas, entre os dois níveis hierárquicos de armazenamento(memória principal e auxiliar), conforme Denning [3].

Se executarmos um mesmo programa em dois sistemas sendo um deles com memória virtual e o outro não, o tempo de U.C.P. gasto é o mesmo, porém o de resposta não é, pois no segundo, como o programa estava integralmente na memória - não havendo portanto trocas dos trechos do programa - receberemos a resposta mais rapidamente.

Conforme vimos (1.1), o sistema fornece intervalos de tempo de U. C. P. a cada uma das tarefas que estão

na fila de espera, para que elas progridam.

Essa fatia de tempo deve ser superior ao "OVERHEAD" necessário para atualizar os "registros do interpretador" com o contexto do programa em questão.

Por outro lado, esse tempo deve ser limitado superiormente, para que os usuários tenham a impressão de que a máquina está dedicada integralmente às suas tarefas.

Em nosso sistema as fatias de tempo, fornecidas aos usuários, são iguais a 30 milissegundos. Escolhemos esse valor porque coincide com o tempo necessário para que uma operação de E/S em disco seja executada.

A escolha desse valor não é definitiva. No módulo supervisor, criamos algumas variáveis cujos conteúdos indicam qual o número de usuários em cada fila. Essas variáveis além de dirigirem o fluxo do sistema (por exemplo se não há usuários aguardando U.C.P. o sistema ficará desativado até que a ocorrência de um evento (1.2) o reative), prestam-se também para fornecer-nos informações que possibilitem um melhor ajuste no sistema. Assim, poderemos saber:

- se o sistema está bem balanceado com a fatia de tempo de U.C.P. fornecida a cada uma das tarefas;
- se o número de blocos que constituem a memória física, pode ser reduzido sem afetar o rendimento do sistema, ou então se esse número precisa ser ampliado para que o atendimento aos usuários melhore;

- se o sistema não ficará sobrecarregado se colocarmos mais terminais (2.2.);
- etc.

CAPÍTULO 5

IMPLEMENTAÇÃO DO SISTEMA

5.0. CONFIGURAÇÃO DO EQUIPAMENTO E LINGUAGEM EMPREGADA

A configuração do minicomputador "MITRA-15", empregado na implementação do nosso sistema é:

- 16 K palavras (de 16 "bits") de memória R. A.M.;
- 2 relógios de 12,8 kHz
- 32 níveis de interrupção;
- 4 terminais;
- 1 unidade com 2 discos de cabeça móvel com capacidade de armazenar até 10 M caracteres:
- 1 leitora de cartões;
- 1 "teletype" para controlar o sistema.

As instruções são executadas geralmente em 3 μs. O tempo médio para uma operação de E/S num setor de disco (256 caracteres) é igual a 30 ms.

Na implementação, empregamos a linguagem de programação "ASSEMBLER II" descrita no "Manuel de Référence" [5]. Essa escolha teve como objetivo possibilitar a exploração dos re-

cursos da máquina ao máximo.

Conforme vimos (2.7) o sistema ficou dividido em 3 módulos. Associamos a cada módulo um nível de interrupção. O módulo supervisor ficou associado ao nível mais prioritário, enquanto que o módulo de gerência ficou com o nível intermediário, cabendo finalmente aos interpretadores o nível de prioridade mais inferior do sistema. Com isso, se durante a interpretação terminar uma operação de E/S, a máquina automaticamente - após a preservação dos registros e indicadores - passará o controle para o nível associado ao módulo supervisor. O mesmo poderá ocorrer se for o módulo de gerência que estiver controlando a máquina (esse mecanismo de preservação dos registros/indicadores está descritos no "Manuel de Référence" |[5]|).

Esses 3 módulos são executados sob a supervisão do "MONITOR DE TEMPO REAL EM DISCO ESTENDIDO" (ou seja o "M.T.R.D.E." cujo funcionamento está descrito em MONITEURS |[6]|).

Sob o ponto de vista do M.T.R.D.E., nossos módulos são 3 programas independentes. Usamos as rotinas embutidas nesse monitor para estabelecer a ligação entre os 3 módulos. Dessa forma, ao invés de escrevermos rotinas que ativam ou desativam um nível, fazemos apenas uma chamada ao supervisor (CSV) para que isso seja feito.

As rotinas utilizadas para a execução das operações de E/S nos terminais ou no disco foram as do M.T.R.D.E. Sempre que havia a necessidade de uma operação de E/S, simplesmente carregávamos o endereço efetivo do bloco de controle no acumulador e depois o supervisor era chamado para que a operação fosse execu-

tada.

As rotinas de conversão de binário para decimal e vice-versa (apesar de muito simples) também não foram reescritas para que não houvesse duplicação de código.

Outro ponto relevante é que o M.T.R.D.E. poderá ficar executando programas que estejam associados aos demais níveis de prioridade.

5.1. O MÓDULO SUPERIOR

5.1.0. FUNCIONAMENTO DO SUPERVISOR

É o supervisor o responsável por todo o sistema, conforme vimos (2.7). Para descrever seu funcionamento, vamos supor que o nível a ele associado esteja desativado. A ocorrência de um evento (1.2) provoca uma mudança nos registros da máquina (5.0). Assim o registro I.A.R. (o registro da máquina que aponta para a próxima instrução que deverá ser executada) que antes estava apontando para uma instrução de outro módulo (ou melhor para a instrução de um programa cujo nível de interrupção é inferior ao nível do supervisor), estará agora apontando para uma instrução do supervisor.

Como o supervisor pode ser ativado por qualquer um dos eventos (ou ainda pela ocorrência de diversos eventos

simultaneamente), há a necessidade de interrogar qual (ou quais) foi a ocorrência. Essa resposta é obtida pela pesquisa de algumas variáveis especiais.

Acreditamos que a ordem em que essas perguntas serão feitas é importante. Elas devem seguir um critério hierárquico (sempre que possível), com o objetivo de que as tarefas sejam atendidas mais rapidamente. Sendo assim, adotamos a seguinte prioridade.

5.1.1. PARALISAÇÃO DO RELÓGIO

O primeiro passo consiste em impedir a movimentação do relógio. O contador do relógio contém a quantidade de tempo de U.C.P. que resta ao usuário corrente. Essa paralisação é feita para evitar que seja deduzido do contador o intervalo de tempo gasto pelo supervisor na identificação e a consequente resolução dos eventos que provocaram a presente ativação. Dessa maneira, fica assegurado que a fatia de tempo de U.C.P. destinada a um usuário não sofrerá alterações. Posteriormente, se ficar evidenciado que o interpretador não foi um dos responsáveis pelo aparecimento da interrupção, então a tarefa do usuário receberá novamente o controle da máquina, durante o período que o relógio assinalava quando da ativação.

5.1.2. TÉRMINO DE UMA OPERAÇÃO DE E/S

O segundo passo tem como objetivo verificar se as operações de E/S que estavam sendo processadas pelos terminais contribuíram para o aparecimento da interrupção. Esse teste é realizado por intermédio do exame das variáveis nos blocos de controle dos terminais que estão fazendo E/S (4.4). Todos os usuários que encerraram uma dessas operações, ou serão colocados no final da fila dos que estão aguardando tempo de U.C.P. ou então continuarão na mesma fila. A mudança de fila está condicionada ao tipo de informação armazenado no nó do usuário. Se a operação E/S foi pedida pela tarefa então haverá a troca. Se a operação foi solicitada pelas rotinas responsáveis pelo diálogo usuário x supervisor poderá não haver a troca.

5.1.3. RECEPÇÃO DE UM CARACTERE "BREAK"

O terceiro passo interroga o bloco de controle - responsável pelo reconhecimento do caractere "BREAK" (4.4) - de cada usuário. Todos os usuários que tentaram estabelecer contacto com o supervisor, transmitindo o caractere "BREAK", serão atendidos exceto se ele for o usuário que no momento está com o controle da máquina (usuário corrente). Nesse caso o seu bloco

de controle é saltado, sendo examinado somente na próxima vez em que o sistema for despertado (ativado).

O atendimento dispensado a um usuário que deprimiu a tecla "BREAK" é muito simples. Para tal, o supervisor comanda uma operação de entrada (leitura) ao terminal associado ao usuário, e quando esta se processar ele irá analisá-la.

Esse comando de leitura ao terminal não deve ser confundido com o pedido de entrada feito pelo programa do usuário. Para que tal confusão fosse evitada, colocamos o usuário na fila dos que estão aguardando E/S, mas inserimos, adicionalmente, uma marca no campo de informação do nó (4.5) de maneira que o sistema possa fazer a distinção. Toda vez que uma operação de E/S termina esse campo é investigado. Se a marca estiver ausente, então o usuário irá para o final da fila dos que estão aguardando U.C.P. (5.1.2) uma vez que o sistema já executou a operação requerida pela tarefa. No caso inverso então a mensagem será comparada com o repertório existente no supervisor. Se for uma mensagem válida então o supervisor fará o que foi solicitado (desde que for consistente). Se válida (ou inconsistente) então - após notificar o usuário da irregularidade - o supervisor comandará uma nova operação de entrada, para que o usuário possa fazer outra tentativa.

5.1.4. TÉRMINO DE UMA OPERAÇÃO NO DISCO

A estrutura do sistema aceita dois tipos

de páginas: aquelas que estão sendo referenciadas pela primeira vez - denominadas de páginas do tipo I - que podem ser identificadas pelo valor λ na entrada do diretório ou na entrada do mapa de páginas, conforme já vimos (4.1) e (4.2); as páginas de outra espécie são aquelas que já contém alguma informação - denominadas de páginas do tipo II - e são identificadas por um valor diferente de λ.

O quarto passo verifica se foi o término de alguma operação de E/S feita no disco que concorreu para que o supervisor fosse despertado. Dessa forma, inicialmente o bloco de controle associado ao disco é consultado. O fluxo será dirigido para o próximo passo se a operação do disco ainda não foi concluída.

Existem duas alternativas se a operação já foi executada:

Caso de uma entrada - então já está na memória a página solicitada pela tarefa do usuário-cabeça da lista dos que estão aguardando memória. Nesse caso após atualizar a entrada do diretório ou do mapa de páginas (com o número inicial de acessos e com o número do bloco de memória que recebeu a página do tipo II), o supervisor colocará o usuário no início da fila dos que estão aguardando tempo de U.C.P. e posicionará uma variável para que indique que o mecanismo responsável pelas trocas de páginas encontra-se no estado operacional. Depois, o controle passará para o próximo passo.

Caso de uma saída - então um bloco de memória já está livre. Esse trecho será destinado ao armazenamento de

uma página da tarefa do usuário-cabeça da lista dos que estão a guardando trecho de memória. Se a página solicitada pela tarefa for do tipo I, então o supervisor agirá da mesma forma como se houvesse terminado uma operação de entrada. Porém se a página solicitada for do tipo II, então o supervisor comandará a leitura da página ausente (que está no disco) para cima do bloco liberado , prosseguindo depois para o quinto passo.

5.1.5. ATENDIMENTO AO MÓDULO CORRENTE

No quinto passo o supervisor verficará se o módulo corrente o ativou.

Existe uma variável (chamada módulo corrente) cujo conteúdo indica se o módulo corrente é o interpretador , a gerência ou se há módulo ativo.

Ao ser examinada, se a variável informarque não há nenhum módulo ativo, o controle será desviado para o próximo passo.

Quando o módulo corrente for o da gerência os parâmetros de comunicação entre os módulos (2.7) serão isolados e analisados. Dependendo do resultado o supervisor poderá ser desativado - caso a gerência ainda não tenha concluído o que foi solicitado - ou não.

- Se a gerência já devolveu à lista de espaço disponível os blocos liberados (que estavam associados a um

programa que terminou) então o supervisor incrementará o contador de blocos disponíveis, atualizará a variável módulo corrente e ramificará para o sexto passo.

- Se a gerência já providenciou um bloco , o usuário corrente será atendido da mesma forma como se houvesse terminado a cópia de uma página em disco conforme já explanamos (5.1.4). A variável módulo corrente será atualizada (indicando nenhum módulo ativo) sendo o controle dirigido para o próximo passo.

- Se o bloco providenciado pela gerência estiver ocupado (5.2) o supervisor antes de destiná-lo a um usuário precisará copiá-lo no disco. Sendo assim, um comando de saída é iniciado, a entrada do mapa (ou do diretório) é atualizada (agora indicando que a página está ausente), a variável módulo corrente receberá o valor correspondente a nenhum módulo ativo e finalmente o controle do supervisor passará para a execução do próximo passo.

Na hipótese de que o módulo corrente seja o interpretador, os parâmetros também serão isolados e analisados.

- Se os parâmetros informem que o interpretador nada transmitiu, o contador do relógio (que havia sido congelado (5.1.1)) será examinado. No caso de haver sido esgotado o tempo fornecido ao usuário, o supervisor atualizará um dos "bits" dos parâmetros (para que o interpretador providencie - logo que possível - a mudança do usuário), fornecerá mais uma fatia de tempo para o usuário e ficará desativado. É claro que essa segunda fatia para um mesmo usuário não será consumida integralmen-

te, ela servirá para que o interpretador acabe de interpretar a instrução e depois preserve os registros da máquina na página-contexto do usuário.

- Se os parâmetros isolados indicarem que o interpretador quer algo, o supervisor o atenderá. Dessa forma se foi a tarefa que encerrou, o supervisor ativará o módulo de gerência para que seja processada a devolução dos blocos de memória ora liberados e ficará desativado até que a ocorrência de um evento o reative. Mas se foi a necessidade de uma operação de E/S, o supervisor após iniciá-la mudará o usuário de fila e ramificará para o próximo passo.

- Quando os parâmetros indicarem que foi detectada a ausência de uma página, o usuário irá para a fila dos que estão aguardando trecho de memória, passando o controle para o próximo passo.

- Se o interpretador avisa que já salvou as variáveis do usuário na página-contexto, o supervisor após atualizar a contabilidade do usuário irá executar o sexto passo.

5.1.6. ATENDIMENTO AOS USUÁRIOS QUE AGUARDAM TRECHO DE MEMÓRIA

O sexto passo tem como objetivo fornecer blocos de memória aos usuários que precisam da presença de uma das suas páginas. O número de usuários que podem ser atendidos cada vez que esse passo é executado, depende do número de blocos li

vres que estão atualmente na lista de espaço disponível (mantida pelo módulo de gerência), do tipo da página (5.1.4) que o usuário está querendo, do estado do disco (operacional ou em opera - ção).

Como só é possível processar uma E/S em disco se a anterior já estiver encerrada, não adianta ter diversos blocos liberados, se as páginas requeridas forem páginas do tipo II.

A diretriz que seguimos foi a de ir atendendo os usuários da fila enquanto existirem blocos disponíveis e o estado do disco for operacional. Quando aparecer o primeiro pedido de uma página do tipo II, o estado do disco é trocado para "em operação".

Nesse novo estado somente os pedidos de páginas do tipo I serão atendidos. Quando do esgotamento dos blocos (ou do percurso à lista) o controle do sistema será desviado para o sétimo passo do supervisor.

Se a lista de espaço disponível estiver vazia e o estado do disco for operacional, haverá a eleição da página que será removida para o disco. Se o disco já estiver em operação então o controle será dirigido para o próximo passo. Porém se ele estiver livre, o módulo de gerência será ativado para que escolha qual das páginas deverá ser removida. Para isso após fazer o módulo corrente ser o correspondente ao de gerência, o supervisor ficará desativado até que ocorra um evento.

5.1.7. ATENDIMENTO AOS USUÁRIOS AGUARDANDO TEMPO DE U.C.P.

No sétimo passo, o supervisor verifica se há usuários aguardando tempo de U.C.P.. Se a lista estiver vazia então o supervisor ficará desativado até que ocorra um outro evento. Caso contrário, o usuário cabeça da lista será contemplado com uma fatia de tempo. Para isso, o supervisor após fazer com que a variável "usuário corrente" receba o número do usuário, a variável módulo será atualizada com o valor relativo ao módulo interpretador, o relógio receberá o total de tempo de U.C.P. que será destinado ao usuário, os parâmetros receberão o valor do apontador da página-contexto do usuário, e depois de ativar o interpretador, o supervisor ficará desativado até que o aparecimento de uma interrupção o reative.

5.1.8. ROTINAS AUXILIARES DO SUPERVISOR

Na execução desses passos, o supervisor conta com uma série de subprogramas, que a seguir descreveremos.

Toda vez que for necessária a desativação do supervisor, o controle passa para o subprograma adequado. Esse subprograma, além de desativar o supervisor, ativará o módulo apontado pela variável módulo corrente (se houver algum). Se o módulo corrente for o interpretador, então adicionalmente o contador de

tempo será atualizado (com o valor da fatia) e o relógio entrará em operação.

A rotina responsável pelo início de uma operação de entrada, antes de comandar a operação limpa o "BUFFER" do usuário.

Já a rotina de saída varre o "BUFFER" (da direita para a esquerda) até encontrar o primeiro caractere diferente de branco, inserindo aí os dois caracteres que controlarão o retorno do carro e o espacejamento da linha de impressão. Após a impressão de uma linha, o "BUFFER" também recebe caracteres brancos.

As duas rotinas (responsáveis pelas operações de E/S), após comandar o início da transmissão, chamarão o subprograma que processará a troca de fila.

Os subprogramas que manipulam as filas (4.5), são constituídos pela combinação adequada dos 3 algoritmos básicos; inserir, remover e rodar. Empregando apenas esses algoritmos conseguimos fazer inserções à direita ou à esquerda de um nó, podemos percorrer a estrutura, retificar os nós, etc.

5.2. O MÓDULO GERÊNCIA DE MEMÓRIA

É no módulo gerência de memória que os blocos são reservados. Logo após o carregamento do módulo supervisor na memória, ocorre o do módulo de gerência. O supervisor ativa o

módulo de gerência para que os valores iniciais sejam atribuídos (para que a "inicialização" seja processada) às variáveis. Nessa fase os blocos - tantos quantos o supervisor especificar - serão incluídos na lista de espaço disponível.

Com o objetivo de economizar espaço os blocos são referenciados - em todos os módulos - não pelos seus endereços físicos mas sim por um número que pode variar de 0 a 255. Então, quando da inicialização, o supervisor pode informar que a memória deve ser constituída de até 256 blocos (de 128 palavras).

Esses blocos serão reservados em trechos de memória adjacentes. Além do número de blocos, o supervisor informa à gerência onde deve iniciar o primeiro deles. Com esses dados, a gestão de memória inclui os blocos na lista de espaço disponível e depois ativa supervisor.

Este passa para o módulo de gerência o endereço absoluto do início do primeiro bloco. Esse endereço - que denominamos de "endereço base" - será empregado por todos os módulos para o cálculo do endereço físico de qualquer posição. Por exemplo, se uma tarefa quer fazer acesso à linha "ℓ" (deslocamento dentro da página) da página "p", então o mecanismo que calcula os endereços executa os seguintes passos:

- O número do bloco associado à página "p" é determinado (isto é possível, graças ao exame da entrada índice "p" do mapa das páginas da tarefa em questão). Seja "b" tal número. (A página estará ausente se a dupla inequação não for satisfeita $0 \leq b \leq 255$);

- O número "b" é multiplicado por 256 (no "MITRA-15" o endereçamento é feito ao nível de caracteres). Ao invés da multiplicação, usamos uma operação de "SHIFT" por ser mais rápida;

- O resultado da operação acima é adicionado ao valor "ℓ" e ao endereço base, formando o endereço físico desejado.

O módulo de gestão, quando ativado, poderá seguir 3 possíveis caminhos. Para cada um desses caminhos foi construído um subprograma.

Para retirar ou devolver blocos da lista de espaço disponível, usamos os algoritmos descritos por Knuth |[4]|.

Para a retirada: If AVAIL = λ Then LRU;

Otherwise X $\leftarrow$ AVAIL, AVAIL $\leftarrow$ LINK(AVAIL).

Para a devolução: LINK (X) $\leftarrow$ AVAIL, AVAIL $\leftarrow$ X.

No caso da retirada, se a lista de espaço disponível estiver vazia, o terceiro subprograma será chamado (LRU). O supervisor nesse caso será notificado de que o bloco número "b" precisa ser copiado em disco antes de abrigar uma nova página.

É da competência do terceiro subprograma eleger qual das páginas será removida para o disco.

Essa escolha é baseada na estratégia L.R.U. ("Least Recently Used") descrita por Watson |[7]|. Quanto mais recentemente uma página foi referenciada, menor a probabilidade de ser ela removida. Conforme declara Shaw |[1]| é bem razoável que as páginas que não foram usadas por períodos relativamente longos não

serão referenciadas num futuro imediato.

A página que apresentar o menor número no contador de referências ((4.1) e (4.2)) será a escolhida. Uma página-contexto somente poderá ser removida se a sua tabela de páginas já estiver vazia.

Conforme veremos (5.3) o contador de acessos é incrementado (pelo interpretador) sempre que a tarefa faz acesso a uma página. Sempre que o número de acessos a uma página for igual a 128, todos os contadores de referências serão divididos por 2 ("SHIFT" de uma posição à direita). Como regularmente haverá "OVERFLOW" nos contadores, as páginas não referenciadas há algum tempo apresentarão o número de acessos cada vez menor. É por isso que o supervisor coloca um número inicial de acessos nos contadores das páginas logo após o carregamento. Escolhemos para o nosso sistema o valor inicial 64 por ser a média entre os valores extremos.

5.3. O MÓDULO INTERPRETADOR

Este módulo é composto por um conjunto de rotinas que serão utilizadas pelos interpretadores.

Dessa forma, temos o mecanismo que calcula os endereços conforme foi descrito (5.2).

Toda vez que solicitado, esse subprogra-

ma volta para o local de onde foi chamado ou ativa o supervisor pedindo uma página ausente. Antes de ativar o supervisor, esse programa chama a rotina responsável pela preservação da página-contexto do usuário.

Se a página estiver presente, ele incrementa o contador de referência. Quando o contador atingir o valor máximo (que é igual a 128), então todos os contadores serão divididos por 2, conforme vimos (5.2.).

As rotinas de interpretação variam segundo os interpretadores. Todavia se a instrução a interpretar for uma operação de E/S ou um "HALT" (fim de execução do programa), independente do interpretador, elas sempre chamarão a rotina de preservação de contexto, informando (através dos parâmetros) qual foi a ocorrência que motivou a ativação do supervisor, e finalmente, o supervisor será ativado.

Uma outra característica que independe do interpretador empregado, é a rotina responsável pela execução da próxima instrução. Essa rotina, antes de buscar a próxima instrução, sempre isola os parâmetros de comunicação e examina-os com o objetivo de verificar se é permitido continuar com a interpretação do programa corrente. Se o supervisor não pediu troca de usuário, a próxima instrução será processada. Mas se o supervisor pediu que a troca fosse feita, a página-contexto do usuário corrente será preservada e o supervisor será despertado.

Sempre que o interpretador for ativado, uma rotina "inicializará" as variáveis do interpretador com os valores

armazenados na página-contexto, e dependendo da variável "estado do interpretador" carregará o programa do usuário na memória, ou irá interpretar a instrução apontada pela variável "I.A.R.".

CAPÍTULO 6

CONCLUSÕES

Somente o uso em larga escala do sistema poderá informar-nos se o tamanho da fatia de tempo, se o número de terminais e se o número de páginas disponíveis (a todos os usuários) que estimamos devem ser modificados ou não. Esperamos que os dados estatísticos fornecidos pelo sistema (4.6) permitam-nos realimentá-lo com o objetivo de balanceá-lo.

Nosso sistema permite que apenas os programas escritos em linguagem-objeto sejam interpretados. Como é muito árdua a tarefa do usuário programar nessa linguagem, sugerimos que o sistema esteja capacitado a manter arquivos em disco. Com essa providência, os programadores poderiam ser beneficiados com as vantagens oferecidas pelas linguagens-fonte.

Dessa forma, os tradutores teriam a capacidade de armazenar os programas traduzidos em disco. Terminada a fase de tradução, os programas poderiam ser interpretados normalmente.

BIBLIOGRAFIA

|[1]| Shaw, Alan C. - 1974 - "The Logical Design of Operating Systems" - Prentice-Hall, Inc.

|[2]| Wilkes, M.V. - 1974 - "Time-Sharing Computer Systems" - Mac Donald-London and American Elsevier Inc.

|[3]| Denning, Peter J. - 1970 - "Virtual Memory" - Computing Surveys, vol. 2, nº 3.

|[4]| Knuth, D.E. - 1972 - "The Art of Computer Programming" - vol. 1, Fundamental Algorithms - Addison-Wesley Publishing Company.

|[5]| Mitra-15 - 1975 - "Manuel de Référence" - Tome 1 - Compagnie Internationale pour L'Informatique.

|[6]| Mitra-15 - 1976 - "Moniteurs" - Compagnie Internationale pour L'Informatique.

|[7]| Watson, Richard W. - 1970 - "Timesharing System Design Concepts" - McGraw-Hill Book Company.

|[8]| Daley, Robert C. and Dennis, Jack B. - 1967 - "Virtual Memory, Processes, and Sharing in Multics" - Communications of the ACM, vol, 11, nº 5.

|[9]| Barrow, D.W. - 1971 - "Computer Operating Systems" - Chapman and Hall Ltd.

|[10]| Ziegler, James R. - 1967 - "Time-Sharing Data Processing Systems" - Prentice-Hall, Inc.

|[11]| Rosen, Saul - 1967 - "Programming Systems and Languages" - McGraw-Hill Book Company.

|[12]| Madnick, Stuart E. and Donovan, John J. - 1974 - "Operating Systems" - McGraw-Hill Book Company.

|[13]| Burroughs - 1969 - "B 2500/B 3500 Systems Software Operational Guide" - Burroughs Corporation.

|[14]| Organick, Elliot I. - 1973 - "Computer System Organization"- The B 5700 / B 6700 Series - Academic Press.

ANEXO I

MÓDULO SUPERVISOR

```
*****************************************************
*                                                   *
*                                                   *
*    SISTEMA  DE  TEMPO  COMPARTILHADO              *
*                                                   *
*                                                   *
*                                                   *
*          MODULO    SUPERVISOR                     *
*                                                   *
*                                                   *
*****************************************************
*  DESCRICAO DA  ZONA  DE  DADOS  COMUNS            *

SUPSTS CDS
       RES  36          *AREA DE TRABALHO DO MONITOR
MIX    DATA 999         *PTR CABECA LISTA U.C.P.
USUPAG DATA 999         *IDEM DOS AGUARDANDO PAGINA
USUFES DATA 999         *IDEM AGUARDANDO E/S
USUNIN DATA 0           *IDEM DOS USUARIOS NAO INICI
QTOMIX DATA 0           *NUMERO USUARIOS LISTA U.C.P
QTOPAG DATA 0           *IDEM AGUARDANDO PAGINA
QTOFES DATA 0          *IDEM AGUARDANDO E/S
QTONIN DATA 32          *IDEM DOS  QUE ESTAO FORA DE
MODCOR DATA 0           *VARIAVL MODILO CORRENTE
*                       *0 = NENHUM MODULO ATIVADO
*                       *1 = INTERPRETADOR ATIVADO
*                       * 2 =  GERENCIA MEMO.ATIVADO
USUCOR DATA 999         *USUARIO CORRENTE
*LOADER CARREGARA:
PABSOL RES  1          *END.ABS.PRIMEIRO BLOCO MEMOR
DDMMAA RES  3           *DATA DO DIA EBCDIC
PTRELO DATA 0           *PTR P/32 BITS RELOGIO N025
**********************
PTRUSU DATA MXPGES
LAMBDA DATA 999
RECONH RES  1
USUBEN RES  1           *USUARIO QUE RECEBERA PAGINA
NUMBLO RES  1           *NUMERO DE UM BLOCO LIBERADO
ESOPDK RES  1           *ESTADO OPERACIONAL DO DISCO
*                        =0 LIVRE;  =1 OPERANDO
DISCO  DATA &8028,&0000,#BUF,256,0,0,4,0
*
*
CB01   DATA &8008,&8080,#BUF,74,0,0,4,0
CB02   DATA &8008,&8081,#BUF+74,74,0,0,4,0
CB03   DATA &8008,&8082,#BUF+148,74,0,0,4,0
CB04   DATA &8008,&8083,#BUF+222,74,0,0,4,0
*
```

```
CB31    DATA 8,&0380,#RECONH,&003F,0,0,4,0
CB32    DATA 8,&0381,#RECONH,&003F,0,0,4,0
CB33    DATA 8,&0382,#RECONH,&00BF,0,0,4,0
CB34    DATA 8,&0383,#RECONH,&003F,0,0,4,0
*
* LISTA DOS 32 POSSIVEIS USUARIOS INICIALMENTE
*TODOS OS USUARIOS ESTAO FORA DE USO CAMPO INFO=0
*SE =1 ENTAO FILA DE U.C.P.; =2 FILA AGUARD.MEM.
*FILA DE E/S =3,4,5,6
MXPGES  DATA,1 31,0,1,0,0,2,1,0,3,2,0,4,3,0,5,4,0,6
        DATA,1 5,0,7,6,0,8,7,0,9,8,0,10,9,0,11,10
        DATA,1 0,12,11,0,13,12,0,14,13,0,15,14,0
        DATA,1 16,15,0,17,16,0,18,17,0,19,18,0
        DATA,1 20,19,0,21,20,0,22,21,0,23,22,0
        DATA,1 24,23,0,25,24,0,26,25,0,27,26,0
        DATA,1 28,27,0,29,28,0,30,29,0,31,30,0,0
DIRETO  DO     32          *DIRETORIO DOS USUARIOS
        DATA   999
BUF     RES    148         *"BUFFERS"  DOS USUARIOS
        FIN

*   DESCRICAO  DA  SECAO   DE   DADOS  LOCAIS    *

SUPLDS  LDS
        RES    2
RELCON  RES    1           *TEMPO  QDO CONGELADO
PTRCB0  DATA   CB01        *PTR BLOCO CONTROLE E/S
PTRCB3  DATA   CB31        *PTR BLOCO CONTROLE "BREAK"
PTRDIR  DATA   DIRETO      *PTR DIRETORIO DOS USUARIOS
PTRBUF  DATA   BUF         *PTR "BUFFERS" DOS USUARIOS
CTE     DATA   806         *INDICE PARAMETROS ZONA COM.
PBYTZC  RES,1  1           *BYTES DA ZONA COMUM
SBYTZC  RES,1  1
TBYTZC  RES,1  1
QBYTZC  RES,1  1
MENOS1  DATA   &FFFF       *CONSTANTE -1
KONT    DATA   0           *VARIAVEL TEMPORARIA
PTR     RES    1           *APONTADOR PARA USO GENERALIZADO
ZEROS2  DATA   0,0
MSKINT  DATA   &01DQ       *MASCARA DO SUP/INT
NTER    DATA   4           *NUMERO DE TERMINAIS NO SIST
DELAY   DATA   3840        *FATIA DE 300MS
TEMP    RES    1           *VARIAVEL TEMPORARIA
ARRET   DATA   &0403       *PARAR RELOGIO NIVEL 24
USUCON  RES    1           *NUMERO USUARIO CONSULTADO
RAMI    DATA   PAZERO
PTRCTX  RES    1
POINTE  RES    1
```

```
INTE    TEXT "INTE"
SIGA    TEXT "SIGA"
PARE    TEXT "PARE"
DUMPE   TEXT "DUMP"
MUDE    TEXT "MUDE"
TRAC    TEXT "TRAC"
DESC    TEXT "DESC"     *MENSAGEM P/DESCONTINUAR
MSKDEG  DATA &1000      *MASCARA:=GER.DEVOLVA PAGINS
ENDMAX  DATA 512
        FIN
*****************************************************
*                                                   *
*    ESTE  E  O  MODULO CENTRAL DO SUPERVISOR       *
*    AQUI AS INTERRUPCOES SAO TRATADAS              *
*                                                   *
*****************************************************

SUPLPS LPS  SUPLDS
*TODOS TERMINAIS FICARAO AGUARDANDO "BREAK"
BKT     LDA  =0
        CLS  BREAKT
        ADD  =1
        CMP  NTER
        BCF  $-3
*MODULO CORRENTE SERA ATIVADO
PAZERO LDX  CTE
        DLD  PBYTZC
        DST  @#6,X      *ATUALIZA PARAMETROS
        CLS  ATMOCO     *ATIVA MODULO CORRENTE
*SUPERVISOR ESTA DESATIVADO
*QUANDO DESPERTADO, MODULOS INFERIORES SERAO
*DESATIVADOS E OS PARAMETROS SERAO GUARDADOS
*PARA FUTURA INVESTIGACAO

* PRIMEIRO PASSO: RELOGIO PARALISADO

        LDE  =&13
        RD
        STA  RELCON     *TEMPO  CONGELADO
        LDE  =&31
        LDA  ARRET
        WD              *RELOGIO PARALISADO
        CLS  DESATI     *DESATIVA OUTROS MODULOS
        LDX  CTE
        DLD  @#6,X
        DST  PBYTZC     *PARAMETROS RESGUARDADOS
        DLD  ZEROS2
        DST  @#6,X
```

```
*SEGUNDO PASSO: FOI O FINAL DE UMA OPERACAO E/S?

        LDA   =0
VARRES  SLLS  =4
        XAX
        LBR   @PTRCB0,X
        CMP   =&80          *TERMINOU E/S?
        BCF   FINES         *SIM->VAI INVESTIGAR
        XAX
        SRLS  =4
INCES   ADD   =1            *PREPARA P/PROXIMO TERMINAL
        CMP   NTER          *TODOS TERMINAIS VISITADOS?
        BCF   VARRES        *NAO VAI P/O PROXIMO TERMIN.

* TERCEIRO PASSO: FOI A RECEPCAO DE "BREAK" ?

TERPAS  LDA   =0
VARREB  SLLS  =4
        XAX
        LBR   @PTRCB3,X
        CMP   =&80
        BCF   FIMBRK        *TERMINOU "BREAK"
        XAX
        SRLS  =4
INCBK   ADD   =1
        CMP   NTER          *FALTA TERMINAL?
        BCF   VARREB        *SIM-->VOLTA
        BRU   QUARPA        *TERMINOU-->VAI P/O PASSO 4
*ENTRADA OU SAIDA TERMINADA:CAMPO SERA INVESTIGADO
FINES   LBR   =&80
        SBR   @PTRCB0,X     *RESTAURA IDENTIFICADOR
        XAX
        SRLS  =4            *AC:=NUMERO DO USUARIO
        STA   USUCON        *USUARIO CONSULTADO:= "AC".
        CLS   BREAKT        *PREPARA P/RECEBER PROXIMO
        MUL   =3
        ADD   =1
        XAX                 *X APONTA INFO@USUARIO
        LBR   @#PTRUSU,X    *AC:=INFO@USUCONA
*TESTA DE QUEM ERA A OPERACAO DE E/S ENCERRADA:
        CMP   =5
        BGE   TERSTS        *SE PARA O STS->RAMIFICA
*E/S ENCERRADA FOI PEDIDA PELA TAREFA DO USUARIO
        CMP   =3            *TESTA SE SAIDA
        BCT   $+6           *ENTRADA:->RAMIFICA
*APOS SAIDA, BRANCOS SERAO MOVIDOS PARA O "BUFFER"
        LDA   USUCON
        MUL   =74
        ADD   PTRBUF        *AC:=END.ABS.DO BUFFER
```

```
        LDE   =74
        CLS   LIMPA
*E/S TERMINADA, USUARIO IRA  IARA A FILA U.C.P.
        LDX   USUCON      *X:=NUMERO DO USUARIO
        LDA   =1          *FILA DESEJADA:=U.C.P.
        CLS   MUDLIS      *MUDA USUARIO DE LISTA
        XAX               *AC:=NUMERO DO USUARIO
        BRU   INCES       *VAI CONTINUAR VARREDURA
TERSTS  BCF   $+4         *SE SAIDA--->RAMIFICA
        CLS   ENTSUP
        LDA   USUCON      *AC:=NUMERO DO USUARIO
        BRU   INCES
        CLS   SAISUP
        LDA   USUCON
        BRU   INCES
*FIM TRANSMISSAO CARACTERE "BREAK"--->SE USUARIO
*DIFERENTE DE USUCOR E SE NAO ESTA NA LISTA DOS
*QUE ESTAO AGUARDANDO PAGINA ENTAO BREAK SERA
*ATENDIDO. CASO CONTRARIO: BLOCO SERA PULADO
FIMBRK  XAX
        SRLS  =4
        STA   USUCON      *USUCON:=NUMERO DO USUARIO
        MUL   =3
        ADD   =1
        XAX               *"X" APONTA P/INFOSPA
        LBR   3#PTRUSU,X  *AC:= FILA DO USUARIO
        CMP   =2          *FILA = AGUARDANDO PAGINA?
        BCT   $+4         *SIM-->ESQUECE BREAK
        LDA   USUCON
        CMP   #USUCOR     *E0 O USUARIO CORRENTE?
        BCF   $+3         *SIM--> ESQUECE BREAK
        LDA   USUCON
        BRU   INCBK       *VOLTA PARA VARRER "BREAK"
*SE PAGINA CONTEXTO EM DISCO ENTAO SERA CARREGADA
        CLS   ENDCON      *CALCULA END.ABS.PAG-CTX
        CMP   =0
        BGE   $+6         *VAI PARA RECONHECIMENTO
        LDA   =2        *PREPARA PARAMETORS PARA
        LDX   USUCON      *TROCAR USUARIO DE LISTA
        CLS   MUDLIS
        LDA   USUCON
        BRU   INCBK       *VAI EXAMINAR OUTRO BLOCO
*BREAK FOI RECONHECIDO, USUARIO IRA PARA A FILA
*DOS QUE ESTAO AGUARDANDO ENTRADA PARA O STS
        LDA   USUCON
        CLS   LERTER      *LER TERMINAL
        LDX   =5          *FILA:=ENTRADA P/STS
        XAX
        CLS   MUDLIS      *MUDA USUARIO DE LISTA
```

```
        XAX
        SLLS =4          *AC APONTA BLOCO BREAK USUAR
        XAX
        LBR  =&80
        SBR @PTRCB3,X    *SIMULA BRFAK EM CURSO
        LDA  USUCON
        BRU  INCBK       *VOLTA VARRER BLOCO BREAK

*QUARTO PASSO: FIM DE LEITURA OU GRAVACAO DISCO?

QUARPA LBR #DISCO        *AC:=BYTE EVENTO DISCO
        CMP  =&80        *TERMINOU E/S EM DISCO?
        BCT  QUINPA      *NAO-->VAI P/O QUINTO PASSO.
        LBR  =&80
        SBR #DISCO
        LDA  =0
        STA #ESOPDK      *FAZ ESTADO OPERACIONAL P/DK
        LBR #DISCO+2     *AC:=TIPO DE OPERACAO
*TESTA SE FOI LEITURA QUE TERMINOU.
        CMP  =0
        BCF  DISAID      *VAI P/FIM SAIDA DE DISCO
*TERMINOU LEITURA. PAGINA SERA ASSOCIADA AO USUARI
        LDA #USUPAG
        STA #USUBEN
        CLS  DESPAG      *ASSOCIA NUMBLO AO USUBENEFI
        BRU  QUINPA      *VAI PARA O QUINTO PASSO
*A OPERACAO TERMINADA NO DISCO FOI SAIDA
DISAID LDA #USUPAG       *BLOCO LIBERADO SERA DESTINA
        STA #USUBEN      *DO A UMA DAS PAGINAS  DO
        CLS  DESPAG      *USUARIO CABECA

*QUINTO PASSO: FOI O MODULO CORRENTE?

QUINPA LDA #MODCOR
        CMP  =0          *HA ALGUM?
        BCT  SEXPAS      *NAO-->VAI P/O SEXTO PASSO
        CMP  =1          *FOI O INTERPRETADOR?
        BCF  GERENC      *NAO-->RAMIFICA

*ATENDIMENTO AO MODULO INTERPRETADOR

        LDA  PBYTZC      *PARAMETROS SERAO ISOLADOS
        AND  =&0F
        STA  KONT        *KONT:=PARAMETRO
        CMP  =2          *TESTA SE INT.TRANSMITIU
        BGE  EVINTE      *SIM-->RAMIFICA
*INTERPRETADOR NAO ATIVOU STS.RELOGIO SERA VISTO
INTERP LDA  RELCON       *AC:=TEMPO RESTANTE
        CMP  =100
```

```
        BGE   $+6
        LDA   DELAY      *MAIS FATIA DE TEMPO
        STA   RELCON
        LDA   MSKINT     *AVISA INT.P/SALVAR USUARIO
        IOR   PBYTZC
        STA   PBYTZC
        BRU   PAZERO     *VAI ATIVAR INTERPRETADOR

*MENSAGEM DO INTERPRETADOR SERA ANALISADA

EVINTE  CMP   =2         *TESTA SE INT.JA MUDOU USUAR
        BCF   ETINT      *OUTRAS MENSAGENS-->RAMIFICA
        LDA #MIX
        CLS SUCESS       *RODA LISTA
        STA #MIX         *  U.C.P.
        LDA #USUCOR      *TESTA SE BREAK ESQUECIDO
        SLLS =4
        XAX
        LBR @PTRCB3,X    *AC:=BYTE EVENTO DO "BREAK"
        CMP   =X80       *PEDIDO  "BREAK"?
        BCT   SEXPAS     *NAO--->RAMIFICA
        LDX #USUCOR
        LDA   =5
        CLS   MUDLIS
        LDA #USUCOR
        CLS   LERTER     *FAZ ENTRADA P/O STS
        BRU   SEXPAS

*TESTA SE TAREFA QUER FAZER UMA OPERACAO DE ENTRAD

ETINT   CMP   =3
        BCF   SAINTE     *VAI VER SE SAIDA
*SERA FEITO UM COMANDO DE ENTRADA P/A TAREFA
        LDA #USUCOR
        CLS   LERTER
        LDX #USUCOR
        LDA   =3
        CLS   MUDLIS
        BRU   SEXPAS

*TESTA SE TAREFA QUER FAZER UMA OPERACAO DE SAIDA

SAINTE  CMP   =4
        BCF   PAGINT     *VAI VER SE PEDIDO DE PAGINA
*SERA FEITO UM COMANDO DE SAIDA
        LDA #USUCOR
        CLS   ESCTER
        LDX #USUCOR
        LDA   =4
```

```
          CLS   MUDLIS
          BRU   SEXPAS

    *TESTA SE TAREFA FEZ PEDIDO DE PAGINA

    PAGINT CMP   =5
          BCF   FINTER     *VAI VER SE "END-OF-JOB"
    *USUARIO IRA PARA A LISTA DOS QUE ESTAO AGUARD.PAG
          LDX #USUCOR
          LDA   =2
          CLS   MUDLIS
          BRU   SEXPAS

    *TESTA SE A EXECUCAO DA TAREFA ENCERROU

    FINTER CMP   =6
          BCF   STOPIN     *VAI TESTAR SE "BREAKPOINT".
    *USUARIO CUJA TAREFA ENCERROU IRA P/LISTA AGUARD.
    * DEVOLUCAO DE PAGINAS
    FIMPRO LDA #USUCOR
          CLS   ENDCON
          STA   PTRCTX
          LDX   =154
          LDA   MENOS1
          STA @PTRCTX,X
          LDX #USUCOR
          LDA   =2
          CLS   MUDLIS
          BRU   SEXPAS     *VAI VER SE USU.AGUARD.PAGS.

    *"BREAKPOINT" FOI ATINGIDO. SERA IMPRESSA A MENSA-
    *GEM:"ENDERECO XXXXX ATINGIDO". ESTADO PASSARA PARA
    *"BREAKPOINT"; NO FINAL DA IMPRESSAO PASSARA
    *PARA O ESTADO "SIGA".

    STOPIN CMP   =7          *TESTA SE BREAKPOINT
          BCF   FIMLOA     *NAO-->RAMIFICA
          LDA #USUCOR
          CLS   IMPSTP
          BRU   SEXPAS

    *"FINAL DO "LOADER"; SERA IMPRESSO NO TERMINAL
    *A MENSAGEM: "SIGA, PARE P, DUMP I J ...

    FIMLOA CMP   =8          *FINAL DO LOADER?
          BCF   FIMPRO     *NAO-->RAMIFIQUE FIM DE PROG
          LDA #USUCOR
          CLS   IMPSIG
          BRU   SEXPAS
```

```
*MODULO CORRENTE E O DE GESTAO DE MEMORIA

GERENC DLD   PBYTZC
       SRLD  =20
       AND   =&0F          *PARAMETROS ISOLADOS
       CMP   =0            *TESTA SE GESTAO ACABOU
       BCT   PAZERO        *SIM-->VAI ATIVAR MODULO GER
       CMP   =1
       BCT   SEXPAS        *SE PAGINA DEVOLVIDA->RAMIFI
       CMP   =2
       BCF   LRU

*PAGINA PEDIDA JA ESTA DISPONIVEL

       LBR   QBYTZC
       STA #NUMBLO
       LDA #USUCOR
       STA #USUBEN
       LBR   =&80
       SBR   #DISCO+2
       CLS   DESPAG
       BRU   SEXPAS

*NA TENTATIVA DE RETIRAR UMA PAGINA, GESTAO VIU
*QUE A LISTA-DE-ESPACO-DISPONIVEL ESTAVA VAZIA.
*POR ISSO, A PAGINA DEVERA SER COPIADA NO DISCO.
*O NUMERO DO USUARIO ESTA NO QBYTZC; E O NUMERO
*DA PAGINA NO TERCEIRO BYTE. SE NUMERO DA PAGINA
* = 255 ENTAO PORQUE TRATA-SE DE UMA PAG-CONTEXTO

LRU    LDE   =0
       LDA   TBYTZC
       SLLD  =8            *E:=NUMERO DA PAG
       XAA                 *A:=NUMERO USUARIO
       STA   KONT          *NUMERO DO USUARIO
       STE   TEMP          *NUMERO DA PAGINA
       STE #NUMBLO         *NUMBLO:=NUMERO DA  PAG.
       CLS   ENDCON
       STA   PTR
       XAE
       CMP   =255          *E' UMA PAGINA-CONTEXTO?
       BCF   CALPAG        *NAO-->VAI COPIAR
*NUMERO DA FILA SERA GRAVADO NA PAG.CTX
       LDA   KONT
       MUL   =3
       ADD   =1
       XAX
```

```
        LBR a#PTRUSU,X
        LDX  =162
        STA aPTR,X      *FILA EM QUE ESTAVA A TAREFA
        LDA  PTR
        STA #DISCO+4    *INDICA BLOCO CONT.END.GRAV.
        LDA  MENOS1
        STA  TEMP
        LDX  KONT
        ICX  =0,X
        LDA  PTRDIR
        STA  PTR
        BRU  CHAVE
CALPAG  ADD  =1
        SLLS =1
        XAX
        LDA aPTR,X
        AND  =&FF
        SLLS =8
        ADD #PABSOL     *END.ABS.DO BLOCO
        STA #DISCO+4    *INDICA NO BLOCO CONTR.END.
CHAVE   LDA  =1
        ADM  TEMP
        LDA  KONT
        MUL  =10
        ADM  TEMP
        STA #DISCO+12   *INDICA QUAL O SETOR DO DISCO
        CNA             *FAZ SETOR IGUAL O SIMETRICO
        STA aPTR,X
        LDA  =0
        STA #DISCO+2    *COPIA PAGINA
        LEA #DISCO      *AC:=END.ABS.BLOCO CONTR.DIS
        CSV  M:IO
        LDA  =1
        STA #ESOPDK     *DISCO EM OPERACAO

*SEXTO PASSO:ATENDIMENTO AOS USUARIOS AGUARDANDO
*                                                PAGINA

SEXPAS  LDA #LAMBDA
        STA #USUCOR
        DLD  ZEROS2
        DST  PAYTZC
        STA #MODCOR
        LDA #QTDPAG
        CMP  =1         *TESTA SE HA USUARIOS NA FILA
        BL  SETPAS      *SE LISTA VAZIA-->VAI P/ PAS
*TESTA SE HA USUARIOS AGUARDANDO DEVOLUCAO DE PAGS
        LDA #USUPAG
        STA  KONT
```

```
         STA   TEMP
DESCON   CLS   ENDCON
         STA   PTRCTX      *PTRCTX:=END.ABS.PAG.CTX
         CMP #LAMBDA       *USUARIO INICIALIZADO?
         BCT   GIRADE      *NAO---->RAMIFICA
         BAN   GIRADE      *SE CTX.EM DISCO-->RAM.
         LDX   =154
         LDA @PTRCTX,X     *AC:=NUM.PAG.FALTOSA
         CMP   MENOS1      *HA ALGUMA?
         BCF   GIRADE      *NAO--->RAMIFICA
*PAGINAS DE UMA TAREFA ENCERRADA SERAO DEVOLVIDAS
         LDX   TEMP        *X:=NUMERO DO USUARIO
         ICX   =0,X        *X:=INDICE DO USUARIO NO DIR
         LDA @PTRDIR,X     *AC:=NUM.DA PAG.CTX
         LBL @PTRCTX       *AC:=MAX.PAG. + NUM.BLO.CTX
         LDE   MSKDEG      *E:=GERENCIA DEVOLVA
         DST   PBYTZC
         LDA #LAMBDA
         STA @PTRDIR,X     *DIRSUSUA:=LAMBDA
         LDA   =0
         LDX   TEMP
         CLS   MUDLIS      *COLOCA USUARIO COMO NAO-INI
         LDA   TEMP
         CLS   BREAKT      *PREPARA TERMINAL P/+ 1 USU.
         BRU   MEMATI      *VAI ATIVAR MODULO GERENCIA
GIRADE   LDA   TEMP
         CLS   SUCESS
         STA   TEMP        *TEMP:=PROXIMO USUARIO
         CMP   KONT        *LISTA JA FOI VARRIDA ?
         BCF   DESCON      *NAO-->EXAMINA OUTRO USUARIO
         LDA #QTOPAG       *VERIFICA SE AINDA HA USUAR.
         BAZ   SETPAS      *NAO-->ENTAO RAMIFICA
         LDA #ESOPDK       *SE DISCO OPERACIONAL ENTAO
         CMP   =0          *UM USUARIO SERA ATENDIDO
         BCF   SETPAS      *NAO-->VAI P-SETIMPO PASSO
*PELO MENOS UM USUARIO SERA ATENDIDO.
         LBL   =#10        *PARAMETRO:GERENCIA,PRECISO
         SBL   PBYTZC      *DE UMA PAGINA
         LDA #USUPAG
MEMATI   STA #USUCOR
         LDA   =2
         STA #MODCOR
         BRU @RAMI

*SETIMO PASSO: ATENDIMENTO AOS USUARIOS AGUARDANDO
*                                        TEMPO DE UCP

SETPAS   LDA #MIX
         STA KONT
```

```
TESTAD LDA #QTDMIX     *TESTA SE HA USUARIOS NA
       CMP  =1         *FILA AGUARDANDO UCP
       BL   @RAMI
*ATENDIMENTO AO USUARIO-CABECA
       LDA #MIX
       CLS  ENDCON
       CMP  =0
       BGE  $+5
       LDA  =2
       LDX #MIX
       CLS  MUDLIS
       BRU  SETPAS
       STA  PTR
       LDX  =140       *X APONTA PARA O ESTADO DA
       LDA @PTR,X      *TAREFA
       CMP  =0         *ESTADO = INTERPRETADOR ?
       BCF  $+11       *NAO-->PULA USUARIO
*USUARIO-CABECA SERA BENEFICIADO
       LDE  =0         *PREPARA PARAMETROS P/ INT.
       LDA  PTR
       DST  PBYTZC     *
       LDA  =1
       STA #MODCOR
       LDA #MIX
       STA #USUCOR
       LDA  DELAY
       STA  RELCON
       BRU @RAMI
GIRA   LDA #MIX
       CLS  SUCESS
       CMP  KONT
       BCT @RAMI
       STA #MIX
       BRU TESTAD
       FIN  BKT
```

```
*MODULO CUJA FINALIDADE E O DE ATIVAR O MODULO
*CORRENTE ( SE HOUVER ALGUM) --------------------
ATMOCO LPS  SUPLDS
INI    LDA #MODCOR
       CMP  =0
       BCT  $+12        *SE NENHUM STS  DORMIRA
       CMP  =1
       BCF  $+12

       LDA  =1
       LDX  =4          *INTERPRETADOR SERA  ATIVADO
       CSV  M:IT
       LDA  =1
       LDX  =6
       CSV  M:IT
       LDA  RELCON       *TEMPO CONGELADO (OU DELAY)
       LDE  =%23        *VOLTARA
       WD               * A SER CONTADO

       DIT              * STS  DESATIVADO
       RTS              *QDO ATIVADO VOLTARA

       LDA  =3          *GERENCIA SERA ATIVADA
       LDX  =4
       CSV  M:IT
       LDA  =3
       LDX  =6
       CSV  M:IT
       BRU  $-8
       FIN  INI
```

```
*ESTE MODULO ANALISA  ENTRADA PARA O SUPERVISOR

ENTSUP LPS   SUPLDS
INI    LDA   USUCON
       MUL   =74
       ADD   PTRBUF
       STA   PTR          *APONTA P/O "BUFFER"DO USUAR
       LDA   USUCON
       CLS   ENDCON
       STA   PTRCTX       *APONTA P/PAGINA CONTEXTO
       CMP #LAMBDA        *SE USUARIO INICIALOZADO
       BCF   TESPRE       *ENTAO-->RAMIFICA
*UNICA PERMITIDA Eõ "INTERPRETE"
       LDX   =0
       LDA @PTR,X
       CMP   INTE
       BCF   ERRMSG
       ICX   =2           *AVANCA P/PROXIMOS 2 BYTES--
       LDA @PTR,X
       CMP   INTE+2
       BCF   ERRMSG
MEMAGU LDA   =2
MUDALI LDX   USUCON
       CLS   MUDLIS
       RTS
ERRMSG LDA   USUCON
       CLS   INVMES
       RTS
*TESTA SE PAG-CTX DO USUARIO ESTA PRESENTE
TESPRE CMP   =0
       BGE   PAGPRE       *PAG.PRESENTE-->RAMIFICA
*TESTA SE A PRESENCA JA FOI SOLICITADA
       LDA   USUCON
       SLLS  =4
       XAX
       LDA   =0
       SBR @PTRCBO,X   *FAZ BLOCO E/S->FIM DE OPERC
       LDA   USUCON
       MUL   =3
       ADD   =1
       XAX
       LBR @#PTRUSU,X
       CMP   =2
       BCF   MEMAGU
       RTS
* A PAGINA-CONTEXTO ESTA PRESENTE
PAGPRE LDX   =0
       LDA @PTR,X
* TESTA SE FOI TRANSMITIDO : "SIGA"                    *
```

```
        CMP   SIGA
        BCF   SEGUND
        ICX   =2
        LDA @PTR,X
        CMP   SIGA+2
        BCF   ERRMSG
ANDA    LDA   =0
        LDX   =140      *X APONTA PARA ESTADO TAREFA
        STA @PTRCTX,X
        LDA   =1
        BRU   MUDALI
* TESTA SE FOI TRANSMITIDO :  "PARE"                *
SEGUND  CMP   PARE
        BCF   TERCEI
        ICX   =2
        LDA @PTR,X
        CMP   PARE+2
        BCF   ERRMSG
        ICX   =2
        LEA @PTR,X
        CSV   M:DCBN
        XAE
        CMP   ENDMAX
        BGE   ERRMSG
        LDX   =150
        STA @PTRCTX,X
        BRU   ANDA
* TESTA SE FOI TRANSMITIDO :  "DUMP"                *
TERCEI  CMP   DUMPE
        BCF   QUARTA
        ICX   =2
        LDA @PTR,X
        CMP   DUMPE+2
        BCF   ERRMSG
        ICX   =2
        STX   TEMP
        LEA @PTR,X
        CSV   M:DCBN
        XAE
        CMP   ENDMAX
        BGE   ERRMSG
        LDX   =142
        STA @PTRCTX,X
        LDX   TEMP
        ICX   =4
        LEA @PTR,X
        CSV   M:DCBN
        XAE
        CMP   ENDMAX
```

```
        BGE   ERRMSG
        LDX   =144
        STA aPTRCTX,X
        DCX   =2
        CMP aPTRCTX,X
        BL    ERRMSG
        LDX   =140
        LDA   =2
        STA aPTRCTX,X
        LDA   USUCON
        CLS   DUMP
        RTS
* TESTA SE FOI TRANSMITIDO :  "MUDE"                *
QUARTA  CMP   MUDE
        BCF   QUINTA
        ICX   =2
        LDA aPTR,X
        CMP   MUDE+2
        BCF   ERRMSG
        ICX   =2
        STX   TEMP
        LEA aPTR,X
        CSV   M:DCBN
        XAE
        CMP   ENDMAX
        BGE   ERRMSG
        LDX   =158
        STA aPTRCTX,X
        LDX   TEMP
        ICX   =4
        LEA aPTR,X
        CSV   M:DCBN
        XAE
        LDX   =160
        STA aPTRCTX,X
        LDA   =3
        LDX   =140
        STA aPTRCTX,X
        LDA   USUCON
        CLS   AGUMOD
        RTS
* TESTA SE FOI TRANSMITIDO :  "TRACE"                *
QUINTA  CMP   TRAC
        BCF   SEXTA
        BCF   ERRMSG
        ICX   =2
        LDA aPTR,X
        CMP   TRAC+2
        BCF   ERRMSG
```

```
        ICX   =2
        STA   TEMP
        LEA  @PTR,X
        CSV   M:DCBN
        XAE
        CMP   ENDMAX
        BGE   ERRMSG
        LDX   =146
        STA  @PTRCTX
        LDX   TEMP
        ICX   =4
        LEA  @PTR,X
        CSV   M:DCBN
        XAE
        CMP   ENDMAX
        BGE   ERRMSG
        LDX   =148
        STA  @PTRCTX,X
        DCX   =2
        CMP  @PTRCTX,X
        BL    ERRMSG
        LDA   =1
        LDX   =140
        STA  @PTRCTX,X
        LDA   USUCON
        CLS   IMPSIG
        RTS
*TESTA SE USUARIO QUER DESCONTINUAR SEU PROGRAMA
SEXTA   CMP   DESC
        BCF   ERRMSG      *NAO-->VAI P/ MENSAGEM INVALI
        ICX   =2
        LDA  @PTR,X       *AC:= OUTRO PEDACO DA MENSAG
        CMP   DESC+2      *TESTA SE VALIDO
        BCF   ERRMSG      *NAO-->IMPRIME MENSAG.INVAL.
*USUARIO IRA PARA A LISTA DOS QUE ESTAO AGUARD.MEM
        LDA   =2
        LDX   USUCON
        CLS   MUDLIS      *MUDA USUARIO DE LISTA
        LDX   =154        *X APONTA PARA PAG.FALTOSA
        LDA   MENOS1      *FLAG:=TAREFA ENCERRADA
        STA  @PTRCTX,X    *GRAVA INFORMACAO P/DEVOL.DE
        RTS               *DE PAGINAS.  VOLTA
        FIN   INI
```

```
*ESTE MODULO RECOLOCA O USUARIO EM ALGUMA FILA,TAO
*LOGO UMA OPERACAO DE SAIDA TENHA SIDO ENCERRADA.
SAISUP LPS SUPLDS
INI    LDA  USUCON
       CLS  ENDCON
       STA  PTRCTX
       CMP #LAMBDA
       BCT  $+7          *SE CONTEXTO AUSENTE-->RAMIF
       CMP  =0           *CONTEXTO ESTA EM DISCO?
       BGE  $+7          *NAO--->RAMIFICA
       LDA  =2
MUDA   LDX  USUCON
       CLS  MUDLIS
       RTS
       LDA  =0           *CONTEXTO AUSENTE, USUARIO
       BRU  MUDA         *VOLTARA PARA LISTA NAO INI.
*ESTADO DA TAREFA SERA EXAMINADO
       LDX  =140
       LDA @PTRCTX,X     *AC:=ESTADO DA TAREFA
       CMP  =0
       BCF  $+3
       LDA  =1
       BRU  MUDA
       CMP  =2
       BCF  $+4
       LDA  USUCON
       CLS  DUMP
       RTS
       CMP  =4
       BCF  $+7
       LDX  =140
       LDA  =1
       STA @PTRCTX,X
       LDA  USUCON
       CLS  IMPSIG
       RTS
       CMP  =1
       BCF  $-2
       LDA  USUCON
       CLS  LERTER
       LDA  =5
       BRU  MUDA
       FIN  INI
```

```
*MODULO CUJA FINALIDADE E A DE DESATIVAR OS OUTROS
DESATI LPS  SUPLDS
INI    LDA  =1
       LDX  =3
       CSV  M:IT
       LDA  =1
       LDX  =5
       CSV  M:IT
       LDA  =3
       LDX  =3
       CSV  M:IT
       LDA  =3
       LDX  =5
       CSV  M:IT
       RTS
       FIN  INI
```

```
*ESTE MODULO INICIALIZA A PAGINA-CONTEXTO DE UM
*USUARIO (CUJO NUMERO ESTA NO AC) NO BLOCO
*CUJO NUMERO ESTA NO INDEXADOR.
TRABAL LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
PTRBUF DATA BUF
PTRCTX RES  1
ORDEM  DATA &8080
CB     DATA 0,&8080,#TRANS,26
TRANS  TEXT "SISTEMA À SUA DISPOSIÇÃO"
       DATA &0D15
       FIN
INPGCT LPS  TRABAL
INI    DST  SALVAE
       STX  SALVAX
       CLS  REPOUS     *LINHA EM REPOUSO P/E.S.
       IOR  ORDEM      *AC:=ORDEM ESCR.TER.DO USU.
       STA  CB+2
       LEA  CB
       CSV  M:IO
       CSV  M:WAIT
       DLD  SALVAE
       CLS  BREAKT
       LDX  SALVAX
       XAX
       SLLS =8
       ADD  #PABSOL
       STA  PTRCTX
       XAX
       MUL  =74
       ADD  PTRBUF
       LDX  =156
       STA  @PTRCTX,X
       LDA  #LAMBDA
       LDX  =128
       STA  @PTRCTX,X
       LDE  =126
       LDX  =4
       LEA  @PTRCTX,X
       DCX  =2
       MVS  @PTRCTX,X
       LDA  =0
       LDX  =140
       STA  @PTRCTX,X
       LBL  =2
       STA  @PTRCTX
```

```
LDA #LAMBDA
LDX  =152
STA @PTRCTX,X
LDA  =1
LDX  =162
STA @PTRCTX,X
ICX  =2
STA @PTRCTX,X
ICX  =2
LDA #LAMBDA
STA @PTRCTX,X
LDA  =1          *AC:=1
LDX  =142        *X APONTA LINFDUMP
STA @PTRCTX,X    *LINFDUMP:=1
CNA              *AC:=-1
ICX  =2          *X APONTA P/LIM.SUP.DUMP
STA @PTRCTX,X    *LIM.SUP.DUMP:=-1
RTS
FIN  INI
```

```
*ESSA SECAO ASSOCIA UM BLOCO DE MEMORIA COM A PAG.
*SOLICITADA PELA TAREFA DE UM USUARIO. A ENTRADA
*DO DIRETORIO (OU DO MAPA DE PAGINAS) RECEBERA O
*NUMERO INICIAL DE ACESSOS E O NUMERO DO BLOCO.
*O USUARIO BENEFICIADO EGAQUELE APONTADO PELA VARI
*USUBEN. E O NUMERO DO BLOCO ESTA NA VAR.NUMBLO.

LDESPG LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
PTRDIR DATA DIRETO     *PTR DIRETORIO
APONTA DATA 0          *PTR GERAL
INDICE RES  1
ENTRAD RES  1          *ENTRADA QUE SERA  INSERIDA
PTR    RES  1          *PTR ONDE A ENTRADA SERA INC
       FIN
DESPAG LPS  LDESPG
INI    DST  SALVAE
       STX  SALVAX
       LDA #NUMBLO
       LBL  =64        *NUMERO INICIAL DE ACESSOS
       STA  ENTRAD
*TESTA SE BLOCO ABRIGARA PAGINA-CONTEXTO
       LDA #USUBEN
       CLS  ENDCON     *PEGA END.ABS.PAG-CONTEXTO.
       CMP #LAMBDA     *USUARIO INICIALIZADO ?
       BCF  BLOENT     *NAO-->RAMIFICA
*PAGINA - CONTEXTO  SERA  ANALISADA
       LDA #USUBEN
       LDX #NUMBLO
       CLS  INPGCT     *INICIALIZA PAGINA-CONTEXTO.
*PREPARA VARIAVEIS PARA INSERIR ENTRADA NO DIRETOR
PREPAR LDX #USUBEN
       ICX  =0,X
       LDA  PTRDIR
       STA  PTR        *PTR APONTA P/DIRETORIO
       LDA  ENTRAD     *AC:=NUMERO DA ENTRADA
       STA @PTR,X
*TAREFA QUE OBTEVE SUA PAG-CONTEXTO IRA PARA FILA
*QUE ESTAVA ANTES DE SER REMOVIDA PARA O DISCO
       LDA #USUBEN
       CLS  ENDCON
       STA  PTR
       LDX  =162       *X APONTA FILA ANTERIOR
       LDA @PTR,X      *AC:=NUMERO DA FILA ANTERIOR
       LDX #USUBEN
       CLS  MUDLIS     *INCLUI USUARIO FILA ANTERIO
```

```
          BRU  EXAME      *VAI EXAMINAR
  *TESTA SE PAG.CTX NO DISCO
  BLOENT CMP  =0          *PG-CTX NA MEM-->RAMIFICA
          BGE  ASMAP
  *PAG-CTX NO DISCO. TESTA SE ELA FOI LIDA P/ O BLOC
          LBR #DISCO+2    *AC:=TIPO DE OPERACAO  DSK.
          CMP  =0         *FOI LEITURA?
          BCT  PREPAR     *SIM-->ENTAO INCLUI
  *BLOCO LIBERADO RECEBERA PAGINA-CONTEXTO
  LERDSK LBR  =0          *ORDEM-->LEITURA
          SBR #DISCO+2
          LDA #NUMBLO
          SLLS =8
          ADD #PABSOL
          STA #DISCO+4    *END.ABS.DO BLOCO
          LDA aPTR,X      *AC:= -(CHAVE)
          CNA
          STA #DISCO+12   *NUMERO DO SETRO A LER
          LEA #DISCO      *AC:=E.E. DO BLOCO CONTR.DSK
          CSV  M:IO       * LEIA  SETOR (PAGINA) P/ BL
          LDA  =1
          STA #ESOPDK     *ESTADO DO DISCO:=OPERACIONAL
          BRU  VOLTA
  *USUARIO ESTA PEDINDO PAGINA. PAG.FALTOSA OLHADA
  ASMAP  STA  PTR         *PTR:=END.ABS.DA PAG-CONTEXT
          LDX  #154       *X APONTA P/ PAG FALTOSA
          LDA aPTR,X     *AC:=NUMERO PAGINA FALTOSA
          ADD  =1
          SLLS =1
          XAX             *X APONTA ENTRADA MAPA PAGFA
          LDA aPTR,X      *AC:=ENTRADA PAG.FALTOSA
  *TESTA SE PAG.FALTOSA E DO TIPO - I
          CMP #LAMBDA
          BCF  $+4
          LDA  =1
          ADM aPTR        *INCREMENTA PAG.REFERENCIADA
          BRU  ASSOC      *VAI ASSOCIAR PAG.NO MAPA
  *TESTA SE PAGINA FALTOSA JA FOI LIDA
          LBR #DISCO+2      *AC:=TIPO OPER.TERM.DISCO
          CMP  =0         *A PAGINA FOI LIDA  ?
          BCF  LERDSK     *NAO-->VAI LER PAGINA
  *ENTRADA SERA ASSOCIADA
  ASSOC  LDA  ENTRAD
          STA aPTR,X
  *ESTADO DA TAREFA SERA EXAMINADO
  EXAME  LDA #USUBEN
          CLS  ENDCON
          STA  PTR        *PTR APONTA END.CONTX.
          LDX  =140
```

```
        LDA @PTR,X      *AC:=ESTADO DA TAREFA
        CMP  =0         *E INTERPRETADOR ?
        BCF  $+5
        LDA  =1
        LDX #USUBEN
        CLS  MUDLIS     *COLOCA TAREFA NA FILA U.C.P.
        BRU  VOLTA
        CMP  =3
        BCF  $+4
        LDA #USUBEN
        CLS  AGUMOD     *CHAMA SECAO P/MODIFICAR
        BRU  VOLTA
        CMP  =2
        BCF  VOLTA
        LDA #USUBEN
        CLS  DUMP       *INICIA "DUMP" DA TAREFA
VOLTA   DLD  SALVAE
        LDX  SALVAX
        RTS
        FIN  INI
```

```
*MODULO RESPONSAVEL PELA MODIFICACAO DA PALAVRA
*DE UM USUARIO. NO REG.AC O NUMERO DO USUARIO.
*NOS CARACTERES (158 E 160) DA PG-CTX ESTAO O END.
*E  O SEU NOVO CONTEUDO.

LDSMOD LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
APONTA RES  1
PAGNUM RES  1
       FIN
AGUMOD LPS  LDSMOD
INI    DST  SALVAE
       STX  SALVAX
       CLS  ENDCON
       STA  APONTA
       CMP #LAMBDA     *TESTA SE PAG-CTX INICIALIZD
       BCF  $+4
       LDA  SALVAC
       CLS  INVMES     *MENSAGEM INVALIDA
       BRU  VOLTA
       CMP  =0         *TESTA SE CTX NO DISCO
       BGE  $+5
PAGFAL LDX  SALVAC
       LDA  =2
       CLS  MUDLIS     *CONTEXTO AUSENTE; TAREFA
       BRU  VOLTA      *IRA P/FILA AGUARD.PAGINA
*ENTRADAS (158 E 160) SERA EXAMINADAS
       LDX  =158
       LDA @APONTA,X   *AC:=ENDERECO A MODIFICAR
       LDE  =0
       SLLD =8
       XAA
       XAE             *AC:=PAG  ; E:=LINHA
       STA  PAGNUM     *GUARDA NUMERO DA PAG.
       ADD  =1
       SLLS =1
       XAX
       LDA @APONTA,X   *AC:=NUMERO DO BLOCO
       CMP #LAMBDA     *A PAGINA EH INICIALIZADA ?
       BCF  $+5        *SIM-->RAMIFICA
       LDX  =154
       LDA  PAGNUM     *FAZ CTX%PAGFALA:=NUM.PAG
       STA @APONTA,X
       BRU  PAGFAL
       CMP  =0
       BL   $-5        *PEDE PAGINA FALTOSA
```

```
*MODIFICACAO SERA PROCESSADA
        SLLS =8
        ADD #PABSOL
        STE  PAGNUM
        ADM  PAGNUM     *END.AB DA POSICAO
        LDX  =160
        LDA @APONTA,X
        STA @PAGNUM     *MODIFICACAO FEITA
        LDA  =1
        LDX   =140      *FAZ ESTADO DA TAREFA IGUAL
        STA @APONTA,X   *ESTADO:="SIGA"
        LDA  SALVAC
        CLS  IMPSIG
VOLTA   DLD  SALVAE
        LDX  SALVAX
        RTS
        FIN  INI
```

```
*ESSE MODULO EORESPONSAVEL PELO DUMP DA MEMORIA
*DE UM USUARIO. A IMPRESSAO SERA EM HEXADECIMAL.
*OS 4 PRIMEIROS CARACTERES DA LINHA ESPECIFICA O
*ENDERECO DO PRIMEIRO CONJUNTO DE CARACTERES.
*CADA LINHA EXIBE O CONTEUDO DE 8 PALAVRAS.
*SE A PAGINA-CONTEXTO NAO ESTIVER INICIALIZADA
*ENTAO SERA IMPRESSO "MENSAGEM INVALIDA".AC:=NUMUSU

LDUMP  LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
PTRCTX RES  1
LINF   RES  1
LSUP   RES  1
BUFSAI RES  1
PTRPAG RES  1
NEGATI DATA &FFFF
TEMP   RES  1
XBUF   RES  1
ENTVAL RES  1
       FIN
DUMP   LPS  LDUMP
INI    DST  SALVAE
       STX  SALVAX
       CLS  ENDCON     *PEGA ENDERECO PAG.CONTEXTO.
*SE ENDERECO PAG-CTX = LAMBDA ENTAO->"MENS.INVALD"
       CMP #LAMBDA
       BCF  $+4
       LDA  SALVAC
       CLS  INVMES     *IMPRIME "MSG INVLDA"
       BRU  VOLTA
*SE PAG-CTX NO DISCO ENTAO:MUDA P/FILA AGARD.MEM.
       CMP  =0
       BL   MUDA
*DUMP SERA OPERADO
       STA  PTRCTX     *PTR PAGINA CONTEXTO
       LDX  =156
       LDA @PTRCTX,X
       STA  BUFSAI     *APONTA INICIO DO "BUFFER"
       LDX  =142
       LDA @PTRCTX,X
       STA  LINF
       ICX  =2
       LDA @PTRCTX,X
       STA  LSUP
       CMP  LINF       *TESTA SE "DUMP" TERMINADO
       BGE  NORMAZ     *NAO--->RAMIFICA
```

```
***DUMP ENCERRADO
        LDA  NEGATI
        STA @PTRCTX,X
        DCX  =2
        STA @PTRCTX,X
        LDA  SALVAC
        CLS  IMPSIG     *IMPRIME"SIGA,PARE,DUMP...".
        LDA  =1
        LDX  =140
        STA @PTRCTX,X  *FAZ ESTADO DA TAREFA:="SIGA
        BRU  VOLTA
*NORMALIZACAO DO ENDERECO
NORMAZ LDA  LINF
        SRLS =1
        SLLS =1         *NORMALIZACAO
        STA  TEMP
        XAE
        LEA @BUFSAI
        CSV  M:BNHX
        LDA  =8
        STA  XBUF
        LDA  TEMP
        LDE  =0
        SRLD =8        *AC:=PAG ; E:=LINHA
        STA ENTVAL
        ADD  =1
        SLLS =1
        XAX
        LDA @PTRCTX,X  *AC:=NUM.BLOCO
        CMP #LAMBDA    *PAG.INICIALIZADA?
        BCF  $+8       *SIM---> RAMIFICA
        LDA  ENTVAL
        LDX  =154
        STA @PTRCTX,X  *INDICA NUM.PAG.FALTOSA
MUDA    LDA  =2
        LDX  SALVAC
        CLS  MUDLIS
        BRU  VOLTA
*TESTA SE PAGINA ESTA NO DISCO
        CMP  =0
        BL   $-8       *PAG.FALTOSA SERA SOLICITADA
*OS CONTEUDOS DE  8 PALAVRAS SERAO IMPRESSOS
        STA  PTRPAG    *APONTA P/INCIIO DA PAG
LOOP  LDX  LINF
        LDE @PTRPAG,X
        LDX  XBUF
        LEA @BUFSAI,X
        CSV  M:BNHX
        LDA  =6
```

```
        ADM   XBUF
        CMP   =62
        BCT   PRINT
        LDA   =2
        ADM   LINF
        BRU   LOOP
PRINT   LDA   SALVAC
        CLS   ESCTER
        LDA   =6
        LDX   SALVAC
        CLS   MUDLIS
VOLTA   DLD   SALVAE
        LDX   SALVAX
        RTS
        FIN   INI
```

```
*MODULO RESPONSAVEL PELO COMANDO DE UM "BREAK"
*NO TERMINAL CUJO NUMERO ESTA NO REGISTRO ACUMUL.

TERLDS LDS
       RES  2
PTRCB0 DATA CB01
PTRCB3 DATA CB31
PTRBUF DATA BUF
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
TEMP   RES  1
LFRC   DATA &0D15
       FIN

BREAKT LPS  TERLDS
INI    DST  SALVAE     *SALVA REGISTROS "A" E "E"
       STX  SALVAX     *  E O REGISTRO  "X"
       CLS  REPOUS     *FAZ LINHA REPOUSAR
       SLLS =4
       XAX
       LEA  @PTRCB3,X
       CSV  M:IO
       DLD  SALVAE
       LDX  SALVAX
       RTS
       FIN  INI


*MODULO RESPONSAVEL PELA LEITURA DO TERMINAL CUJO
*NUMERO ESTA NO REGISTRO ACUMULADOR

LERTER LPS  TERLDS
INI    DST  SALVAE
       STX  SALVAX
       CLS  REPOUS     *POE LINHA EM REPOUSO
       SLLS =4
       XAX
       ICX  =2
       LDA  =0
       SBR  @PTRCB0,X
       LDA  =72
       ICX  =4
       STA  @PTRCB0,X
       DCX  =6
       STX  TEMP
       LDA  SALVAC
       MUL  =74
       ADD  PTRBUF
```

```
LDE   =74
CLS   LIMPA
LDX   TEMP
LEA @PTRCBQ,X
CSV   M:IO
DLD   SALVAE
LDX   SALVAX
RTS
FIN   INI
```

```
*ESTE MODULO IMPRIME O "BUFFER" ASSOCIADO AO TERM.
*CUJO NUMERO ESTA NO ACUMULADOR. OS CARACTERES
*BRANCOS NAO SERAO IMPRESSOS, E HAVERA A INSERCAO
* DE 2 CARACTERES QUE CONTROLARAO O ESP. E O RET.

ESCTER LPS  TERLDS
INI    DST  SALVAE
       STX  SALVAX
       CLS  REPOUS     *POE LINHA EM REPOUSO
       SLLS =4
       XAX
       ICX  =2
       LBR  =&80
       SBR  @PTRCBO,X
       ICX  =4
       STX  TEMP
       LDA  SALVAC
       MUL  =74
       ADD  =71
*BRANCOS SERAO ELIMINADOS
LOOP   XAX
       LBR  @PTRBUF,X
       CMP  =&40
       BCF  $+6
       XAX
       SUB  =1
       CMP  =0
       BCF  LOOP
       XAX
       ICX  =1
       LBR  LFRC
       SBR  @PTRBUF,X
       ICX  =1
       LBR  LFRC+1
       SBR  @PTRBUF,X
       XAX
       LDE  =0
       DIV  =74
       XAE
       ADD  =1
       LDX  TEMP
       STA  @PTRCBO,X
       DCX  =6
       LEA  @PTRCBO,X
       CSV  M:IO
       DLD  SALVAE
       LDX  SALVAX
       RTS
       FIN  INI
```

```
*ESTE MODULO JOGA BRANCOS (CONFORME O C(E)) NA
*AREA CUJO END.ABS. ESTA NO REG. "A"

LIMLDS LDS
       RES  2
BRANCO DATA &4040
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
       FIN
LIMPA  LPS  LIMLDS
INI    DST  SALVAE
       STX  SALVAX
       XAE
       SUB  =1
       XAX
       LBR  BRANCO
       SBR @SALVAC,X
       LDA  SALVAE
       SUB  =1
       XAE
       LEA @SALVAC
       ADD  =1
       MVS @SALVAC
       DLD  SALVAE
       LDX  SALVAX
       RTS
       FIN  INI
```

```
*MODULO RESPONSAVEL PELA ROTACAO DE UMA LISTA
*NO ACUMULADOR:ANTES NUMERO DO  USUARIO
*                DEPOIS NUMERO DO SUCESSOR.

RODLDS LDS
       RES  2
SALVAE RES  1
SALVAX RES  1
       FIN
SUCESS LPS  RODLDS
INI    STE  SALVAE
       STX  SALVAX
       MUL  =3
       ADD  =2
       XAX                 *X APONTA RLINK
       LBR @#PTRUSU,X *AC:=RLINK&PA
       LDE  SALVAE
       LDX  SALVAX
       RTS
       FIN  INI
```

```
*ESTA SECAO MUDA O USUARIO (CUJO NUM.ESTA NO "X")
*PARA A LISTA CUJO NUMERO ESTA NO REGISTRO "A"
*SE A LISTA ANTERIOR ERA AGUARDANDO MEMORIA E A
*ATUAL E U.C.P. ENTAO O NO SERA INSERIDO NO
*INICIO DA LISTA.TODOS OS OUTROS CASOS A INSERCAO
*SERA EXECUTADA NO FINAL DA LISTA.

MUDLDS LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
TABELA DATA Z,U,D,T,T,T,T
QUE    RES  1
PE     RES  1
XIS    RES  1
       FIN
MUDLIS LPS  MUDLDS
INI    DST  SALVAE
       STX  SALVAX
*NO DO USUARIO SERA RETIRADO DA LISTA ANTERIOR
       CLS  RETNOF
* SE LISTA ATUAL VAZIA ENTAO TRATAMENTO ESPECIAL
       SLLS =1
       XAX
       LDA  =1
* LISTA SERA INCREMENTADA
       BRX @TABELA
Z      ADM #QTONIN
       CMP  =1
       BCF  $+10
       LDA  SALVAX     *TRATAMENTO PRIMEIRO NO  LIS
       MUL  =3
       XAX
       LDA  SALVAX
       STA #USUNIN
       SBR @#PTRUSU,X *LLINKGPA:=P
       ICX  =2
       SBR @#PTRUSU,X *RLINKGPA:=P
       BRU  VOLTA
       LDX  SALVAX
       LDA #USUNIN
       BRU  LIGAR
U      ADM #QTOMIX
       CMP  =1
       BCF  $+10
       LDA  SALVAX
       MUL  =3
       XAX
```

```
          LDA   SALVAX
          SBR @#PTRUSU,X *LLINK&PA:=P
          STA   #MIX
          ICX   =2
          SBR @#PTRUSU,X *RLINK&PA:=P
          BRU   VOLTA
* SE USUARIO VEIO DA LISTA AGUARDANDO PAG.INS.ESQ.
          LDA   SALVAX
          MUL   =3
          ADD   =1
          LBR @#PTRUSU,X
          CMP   =2
          BCF   $+7
          LDA #MIX
          MUL   =3
          XAX
          LBR @#PTRUSU,X
          LDX   SALVAX
          BRU   LIGAR
          LDX   SALVAX
          LDA   #MIX
          BRU   LIGAR
D         ADM #QTDPAG
          CMP   =1
          BCF   $+10
          LDA   SALVAX
          MUL   =3
          XAX
          LDA   SALVAX
          STA #USUPAG
          SBR @#PTRUSU,X *LLINK&PA:=P
          ICX   =2
          SBR @#PTRUSU,X *RLINK&PA:=P
          BRU   VOLTA
          LDX   SALVAX
          LDA #USUPAG
          BRU   LIGAR
T         ADM #QTDFES
          CMP   =1
          BCF   $+10
          LDA   SALVAX
          MUL   =3
          XAX
          LDA   SALVAX
          STA #USUFES
          SBR @#PTRUSU,X *LLINK&PA:=P
          ICX   =2
          SBR @#PTRUSU,X *RLINK&PA:=P
          BRU   VOLTA
```

```
        LDX   SALVAX
        LDA  #USUFES
LIGAR   STX   XIS
        STA   PE
        MUL   =3
        ADD   =2
        XAX
        LBR  a#PTRUSU,X
        STA   QUE
        LDA   XIS
        SBR  a#PTRUSU,X
        MUL   =3
        XAX
        LDA   PE
        SBR  a#PTRUSU,X
        ICX   =2
        LDA   QUE
        SBR  a#PTRUSU,X
        STX   PE
        MUL   =3
        XAX
        LDA   XIS
        SBR  a#PTRUSU,X
        LDX   PE
VOLTA   DCX   =1
        LDA   SALVAC
        SBR  a#PTRUSU,X
        DLD  SALVAE
        LDX   SALVAX
        RTS
        FIN   INI
```

```
*ESTA SECAO RETIRA O NO DO USUARIO (X) DA FILA
*O NUMERO DE USUARIOS DA LISTA  SERA DECREMENTADO.
*SE O USUARIO FPR O CABECA A LISTA SERA GIRADA
*SE O USUARIO FOR O UNICO ENTAO PTRLIS:=LAMBDA

RETLDS LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
TABELA DATA Z,U,D,T,T,T,T
FLAGSO RES  1          *SE = 1 ENTAO UNICO USUARIO
RLINK  RES  1
INFO   RES  1
LLINK  RES  1
UMENOS DATA &FFFF
       FIN
REINOF LPS  RETLDS
INI    DST  SALVAE
       STX  SALVAX
       XAX
       MUL  =3
       XAX
       LDA  =1
       STA  FLAGSO
       LBR a#PTRUSU,X
       STA  LLINK
       ICX  =1
       LBR a#PTRUSU,X
       STA  INFO
       ICX  =1
       LBR a#PTRUSU,X
       STA  RLINK
       CMP  SALVAX     *TESTA SE USUARIO E O UNICO.
       BCT  DECREM     *UNICO USUARIO--> PTRILAMB
       LDA  =0
       STA  FLAGSO
*NO@PA  SERA RETIRADO DA ESTRUTURA
*RLINK@LLINK@PAAIRLINK@PA, LLINK@RLINK@PAAILLINK@PA
       LDA  LLINK
       MUL  =3
       ADD  =2
       XAX
       LDA  RLINK
       SBR a#PTRUSU,X
       MUL  =3
       XAX
       LDA  LLINK
       SBR a#PTRUSU,X
```

```
DECREM LDX   INFO
       ICX   =0,X
       LDA   UMENOS
       BRX   @TABELA
Z      ADM   #QTONIN     *QTONIN:=QTONIN-1
       LDA   #USUNIN
       CMP   SALVAX
       BCF   VOLTA
       LDA   FLAGSO      *ATUALISA NO
       CMP   =1
       BCF   $+4
       LDA   #LAMBDA
       STA   #USUNIN
       BRU   VOLTA
       LDA   SALVAX
       CLS   SUCESS
       STA   #USUNIN
       BRU   VOLTA
*USUARIO E DA LISTA U.C.P.
U      ADM   #QTOMIX
       LDA   #MIX
       CMP   SALVAX
       BCF   VOLTA
       LDA   FLAGSO      *ATUALISA NO-CABECA
       CMP   =1
       BCF   $+4
       LDA   #LAMBDA
       STA   #MIX        *UNICO FAZ PTRILAMBDA
       BRU   VOLTA
       LDA   SALVAX
       CLS   SUCESS
       STA   #MIX        *PTR:=SUCESSOR
       BRU   VOLTA
*USUARIO E DA LISTA AGUARDANDO PAGINA
D      ADM   #QTOPAG      *DECREMENTA NUMERO-USUARIOS
       LDA   #USUPAG
       CMP   SALVAX
       BCF   VOLTA
       LDA   FLAGSO      *ATUALISA NO-CABECA
       CMP   =1
       BCF   $+4
       LDA   #LAMBDA
       STA   #USUPAG
       BRU   VOLTA
       LDA   SALVAX
       CLS   SUCESS
       STA   #USUPAG
       BRU   VOLTA
*ATENDIMENTO =NO=  DA LISTA E/S
```

```
T       ADM #QTOFES
        LDA #USUFES
        CMP  SALVAX
        BCF  VOLTA
        LDA  FLAGSO     *ATUALISA NO-CABECA
        CMP  =1
        BCF  $+4
        LDA #LAMBDA
        STA #USUFES
        BRU  VOLTA
        LDA  SALVAX
        CLS  SUCESS
        STA #USUFES
VOLTA   DLD  SALVAE
        LDX  SALVAX
        RTS
        FIN  INI
```

```
*TODA VEZ QUE A TAREFA DE UM USUARIO JA FOI ATENDI
*DA POR UMA DAS ROTINAS DO SUPERVISOR ELA MUDARA
*PARA O ESTADO "SIGA". NESSE ESTADO, UM COMANDO DE
*"SIGA" PASSARA O CONTROLE DA TAREFA PARA O INTER-
*PRETADOR. OUTROS COMANDOS, PODERAO FAZER COM QUE
*O USUARIO SEJA SERVIDO POR OUTRAS TAREFAS (SE OS
*COMANDOS FOREM VALIDOS). QUANDO EXECUTADA, ESTA
*SECAO COLOCA O USUARIO NA LISTA DOS QUE ESTAO
*AGUARDANDO SAIDA PARA O "STS",

LDSIGA LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
PTRBUF DATA BUF
TEXTO  TEXT "ESCREVA:SIGA,PARE P,DUMP I S,TRACE"
       TEXT " I S,MUDE E V, DESC "
       FIN
IMPSIG LPS  LDSIGA
INI    DST  SALVAE
       STX  SALVAX
       MUL  =74
       XAX
       LDE  =54
       LEA  TEXTO
       MVS  @PTRBUF,X
       LDA  SALVAC
       CLS  ESCTER
       XAX
       LDA  =6
       CLS  MUDLIS
       DLD  SALVAE
       LDX  SALVAX
       RTS
       FIN  INI
```

```
*ESTE MODULO IMPRIME A MENSAGEM "ENDERECO XXXXX
* FOI ATINGIDO". O ESTADO DA TAREFA PASSARA PARA
*AGUARDANDO IMPRESSAO DE BREAKPOINT, E O USUARIO
*IRA PARA A LISTA DOS QUE AGUARDAM SAIDA PARA STS

LDSTOP LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
BUFSAI RES  1
PTR    RES 1
TEXTO  TEXT "ENDERECO        ATINGIDO"
       FIN
IMPSTP LPS  LDSTOP
INI    DST  SALVAE
       STX  SALVAX
       CLS  ENDCON
       STA  PTR
       LDX  =140
       LDA  =4
       STA  @PTR,X
       LDX  =150
       LDE  @PTR,X
       LDX  =156       *X APONTA END BUFFER
       LDA  @PTR,X     *AC:=END.ABS.BUFFER
       STA  BUFSAI
       LEA  TEXTO+10
       CSV  M:BNDC
       LDE  =24
       LEA  TEXTO
       MVS  @BUFSAI
       LDA  SALVAC
       CLS  ESCTER
       LDX  SALVAC
       LDA  =6
       CLS  MUDLIS
       DLD  SALVAE
       LDX  SALVAX
       RTS
       FIN  INI
```

```
*ESSE MODULO PEGA O ENDERECO ABSOLUTO DA PAGINA
*CONTEXTO DE UM USUARIO, CUJO NUMERO ESTA NO AC.
*NA VOLTA: SE AC < 0 ENTAO A PG-CTX ESTA NO DISCO
*          SE AC =999  "   "  " NAO ESTA INICIALIZ
*          OUTROS CASOS->AC:=END.ABSOLUTO CTX.

LDSABS LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
PTRDIR DATA DIRETO
       FIN
ENDCON LPS  LDSABS
INI    DST  SALVAE
       STX  SALVAX
       SLLS =1
       XAX
       LDA  @PTRDIR,X
       CMP  #LAMBDA
       BCT  VOLTA
       CMP  =0
       BL   VOLTA
       AND  =&FF
       SLLS =8
       ADD  #PABSOL
VOLTA  LDX  SALVAX
       LDE  SALVAE
       RTS
       FIN  INI
```

```
*ESSA ROTINA TEM COMO FINALIDADE IMPRIMIR NO TERM.
*DO USUARIO( CUJO NUMERO ESTA NO ACUMULADOR) O
*O SEGUINTE TEXTO "MENSAGEM INVALIDA". A ROTINA
*MUDA O USUARIO PARA LISTA:AGURD.SAIDA STS

LDSINV LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
PTRBUF DATA BUF
TEXTO  TEXT "MENSAGEM INVALIDA."
       FIN
INVMES LPS  LDSINV
INI    DST  SALVAE
       STX  SALVAX
       MUL  =74
       XAX
       LDE  =18
       LEA  TEXTO
       MVS  aPTRBUF,X
       LDA  SALVAC
       CLS  ESCTER
       XAX
       LDA  =6
       CLS  MUDLIS
       DLD  SALVAE
       LDX  SALVAX
       RTS
       FIN  INI
```

```
*ANTES DE UMA OPERACAO DE E/S, ESTE MODULO E CHAMA
*DO. SUA FINALIDADE E A DE COLOCAR A LINHA (CUJO
*NUMERO ESTA NO ACUMULADOR) EM REPOUSO, PARA QUE A
*OPERACAO DE E/S POSSA SER EXECUTADA.

REPLDS LDS
       RES  2
SALVAR RES  2          *PARA SALVAR "A"  "E"
SALVAX RES  1           *PARA SALVAR REGISTRO "X"
ORDEMR DATA &8280      *COMANDO DE REPOUSO
CBREPO DATA 0,0,0,2,0,0,0,0
       FIN
REPOUS LPS  REPLDS
INI    OST  SALVAR     *SALVAR "A"  "E"
       STX  SALVAX     *SALVA REGISTRO INDEXADOR
       IOR  ORDEMR     *AC:=NUM.USU. + ORDEM REPOUS
       STA  CBREPO+2   *PREPARA BLOCO DE CONTROLE
       LEA  CBREPO     *AC:=END.EFETIVO BLOCONTROLE
       CSV  M:IO       *COMANDA UMA ORDEM DE REPOUS
       OLD  SALVAR     *RESTAURA "A"  "E"
       LDX  SALVAX     *RESTAURA   "X"
       RTS             *VOLTA
       FIN  INI
       END  SUPLPS
```

ANEXO II

MÓDULO GERENCIA DE MEMÓRIA

```
**************************************************
*MODULO RESPONSAVEL PELA GESTAO DA MEMORIA
* INICIALMENTE : O VLR ABSOLUTO DO ENDERECO
* DA PRIMEIRA PAGINA E PASSADA AO SUPERVISOR
* (COM ISTO, CONSEGUIMOS MANIPULAR COM NUMEROS
* QUE VAO DE  ZERO (PRIMEIRA) ATE TOTPAG-1 (UL
*TIMA)); TODAS AS PAGINAS SAO INSERIDAS NA
* LISTA AVAIL; O NIVEL DO SUPERVISOR E ATIVADO.
* TODA VEZ QUE O PRESENTE NIVEL FOR ATIVADO, O
* PRIMEIRO BYTE DA ZONA COMUM E CONSULTADO
* SE PBYTZC = 0 ENTAO NADA E FEITO - DIT
* SE PBYTZC = 1 ENTAO UMA PAGINA SERA RETIRADA P/SPV
* SE PBYTZC = 2 ENTAO AS PAGINAS (CUJO TOTAL
* ESTA NO TBYTZC) DE 1 USUARIO SERAO INSERIDAS
* NA LISTA AVAIL; O SPV CARREGA NO QBYTZC O
* NUMERO DA PAGINA CTX DESTE USUARIO
* NOS 4 PRIMEIROS BITS DO SEGUNDO BYTE DA Z.C.(SBYTZC)
* A CADA MOMENTO ESTA A RESPOSTA DA GESTAO AO SUPERVISOR
* SE SBYTZC = 0 AINDA NAO TERMINEI
* SE SBYTZC = 1 TERMINEI INSERCAO
* SE SBYTZC = 2 RETIREI PAG CUJO END. ESTA NA SEG.WZC
* SE SBYTZC = 3 OVERFLOW FACA SAIDA
* SE SBYTZC = 4 OVEFLOW FACA ENTRADA
**************************************************
```

```
GERCDS CDS
       RES 3
PARAM RES 1            *PTR DA Z.C.
       RES  12
LAMBID DATA 999
AVAIL  DATA  0          *APONTA CABECA LISTA ESP.DIS
PTRPAG RES  1           *APONTADOR INICIO PRIM.BLOCO
PTRDIR RES  1           *END.ABS.DIRETORIO DOS USUAR
PTRUSU RES  1           *END.ABS.FILA DOS USUARIOS
LAMBDA EQU  LAMBID
       FIN
GERLDS LDS
BASEGE RES  1          * G DESTE MODULO
PBYTZC DATA,1 0        *PRIMEIRO BYTE DA Z.COMUM
SBYTZC DATA,1 0        *SEGUNDO BYTE DA ZONA  " "
TQBYZC DATA   0        *TER/QUA BYTES  "  "    "
MSKINT DATA &FF0F      *MSK P/ZERAR PARAMETROS
PTR    RES  1          *APONTADOR DIVERSOS
KONT   DATA 0          *CONTADOR
CTE    DATA 806        *INDEX APONTANDO PRIM.N.Z.C.
MSK1   DATA &FF1F      *MSK INFORMANDO SPV FIM INSERCAO
MSK2   DATA &0020      * IDEM IDEM FIM RETIRADA PAGINA
MSK3   DATA &0030       *IDEM IDEM FAZ ENTRADA
MSK4   DATA &FF4F      *IDEM IDEM IDEM  OVRFLW/ SAIDA
       FIN

GERLPS LPS  GERLDS
VOLTAR LDA =4
       LDX  =4
       CSV  M:IT
       LDA =4
       LDX  =6
       CSV  M:IT       *ATIVA NIVEK DO SUPERVISOR
LOOP   LDX  CTE
       DLD  @#PARAM,X
       DST  PBYTZC
       LDE  =0
       LDA  =0
       DST @#PARAM,X
       DLD  PBYTZC
       SRLD =28
       CMP  =0
       BCT  VOLTAR
TESTE  CMP  =1         *TESTA SE PAGINA FOI PEDIDA
       BCT  RETPAG
```

```
*PAGINAS DE UM USUARIO QUE TERMINOU TAREFA
*SERAO DEVOLVIDAS A LISTA "AVAIL"

        LDE  TOBYZC     *E:=PARAMETROS
        LDA  =0
        SRLD =8
        SRLS =8         *AC:=QBYTZC; E:=TBYTZC
        CMP  =255       *SE CTX=LAMBDA-->VOLTA
        BCT  ZEPASA     *PARA SUPERVISOR ATIVANDO P.
        STA  KONT       *KONT:=QBYTZC
        SLLS =8         *AC:=ENDERECO RELATIVO DO CTX
        ADD  #PTRPAG    *AC:=END ABS DO CTX
        STA  PTR        *PTR APONTA PRIM W DO CTX
        XAE             *AC:=TOT PAG USADA INTERP
        SLLS =1         *MULTIPL 2 PARA USAR NO INDEX

*PARA I VARIANDO DE X-1 ATE O CTX?IA SERA DEVOLVIDA
* A LISTA AVAIL

PONTO   XAX             *X APONTA IESIMA +1 WORD CTX
        LDA  @PTR,X     *AC:=P
        CMP  #LAMBID    *SE P=LAMBIDA  NAO DEVOLVE
        BCT  PROX
        CLS  INSERE     *PAGINA P SERA DEVOLVIDA
        DCX  =2         *X APONTA IESIMA WORD DO CTX
PROX    XAX             *TESTA SE TODAS AS PAGINAS
        CMP  =0         *JA FORAM DEVOLVIDAS
        BCF  PONTO      *SE NAO VOLTA

*TODAS PAGINAS JA FORAM DEVOLVIDAS FALTA SO PAG CTX

        LDA  KONT       *AC:=CTX
        CLS  INSERE     *CONTEXTO DEVOLVIDA A LISTA AVAIL

*PARAMETROS DE SAIDA INFORMARAO:
*TODAS AS PAGINAS JA ESTAO NA LISTA AVAIL

ZEPASA  LDA  =0
        LDE  MSK1
        LDX  CTE
        DST  @#PARAM,X
        BRU  VOLTAR     * VAI PARA ATIVACAO DO NIVEL DO SUPERVISOR
RETPAG  CLS  RETIRA
        CMP  #LAMBID
        BCT  OVRFLW
        STA   TQBYZC
        LDA  MSK2
        STA  PBYTZC
```

```
        DLD   PBYTZC
        LDX   CTE
        DST   @#PARAM,X
        BRU   VOLTAR
OVRFLW  CLS   LRU
        STA   TQBYZC
        LDA   MSK3
        STA   PBYTZC
        BRU   $-8
        FIN   LOOP
```

```
* MODULO RESPONSAVEL PELA INSERCAO  NA LISTA AVAIL
* DE UMA PAGINA CUJO ENDERECO  ESTA NO ACUMULADOR

LOSTEM LDS
       RES  2
PONTEI RES 1
TEMP   RES 1
       FIN

INSERE LPS  LDSTEM
INI    STA TEMP
       BAN  $+8        *SE PAG.EM DISCO--->VOLTA
       SLLS =8         *P:=P*128 EM WORDS
       ADD  #PTRPAG    *P:=P+BASE
       STA PONTEI
       LDA  #AVAIL     *P:=AVAIL
       STA @PONTEI
       LDA  TEMP
       STA  #AVAIL     *AVAIL:=P
       RTS
       FIN  INI
```

```
*MODULO RESPONSAVEL PELA RETIRADA DE UMA PAGINA
* DA LISTA AVAIL; NO FIM, NUMERO DA PAGINA ESTARA
*NO ACUMULADOR; SE OVERFLOW AC=999=LAMBIDA

RETIRA LPS   LDSTEM
INI    LDA   #AVAIL
       CMP   #LAMBID
       BCT   FINAL      *SE AVAIL=LAMBDA  VAI P/FINAL
       STA   TEMP       *TEMP:=AVAIL
       SLLS  =8
       ADD   #PTRPAG
       STA   PONTEI
       LDA   @PONTEI
       STA   #AVAIL     *AVAIL:=LINK@AVAIL@
       LDA   TEMP       *AC:=TEMP (PAGINA RETIRADA)
FINAL  RTS
       FIN   INI
```

```
*ESTE MODULO ELEGE A PAGINA QUE SERA REMOVIDA.
*A PAGINA ESCOLHIDA EH AQUELA QUE NAO FOI REFEREN-
*CIADA JA HA ALGUM TEMPO. UMA PAGINA CONTEXTO SO -
*MENTE SERA REMOVIDA SE O SEU MAPA  NAO CONTIVER
*APONTADORES PARA BLOCOS DE MEMORIA E SE A TAREFA
*NAO ESTIVER AGUARDANDO MEMORIA. NA SAIDA O ACUMU-
*LADOR FORNECERA O NUMERO DA PAGINA E O NUMERO  DO
*USUARIO. SE A PAGINA FOR A CONTEXTO, O NUMERO=255

LRULDS LDS
       RES  2
PAGREF RES  1
USUCON RES  1
ENTMIN RES  1
ENTCON RES  1
USUMIN RES  1
ACECON RES  1
BLOCON RES  1
FLAG   RES  1
ACEMIN RES  1          *VARIAVEL ACESSO MINIMO
PTR    RES  1
DIRPTR RES  1
MENOS1 DATA &FFFF
       FIN
LRU    LPS  LRULDS
INI    LDA  =255
       STA  ACEMIN
       LDA #PTRDIR
       STA  DIRPTR     *DIRPTR APONTA INICIO DIRETO
       LDA  MENOS1
       STA  USUCON
NOVUSU LDA  =0
       STA  FLAG
CICLO  LDA  =1
       ADM  USUCON
       CMP  =32        *TESTA SE TODOS USUARIOS
       BCF  $+4        *NAO-->PROSSEGUE
       LBR  ENTMIN
       LBL  USUMIN
       RTS
       LDX  USUCON
       ICX  =0,X       *X APONTA DIRETORIO?USUARIO?
       LDA @DIRPTR,X
       CMP  #LAMBID    *PAG CONTEXTO INICIALIZADA?
       BCT  CICLO      *NAO-->VAI P/PROX.USUARIO
       CMP  =0         *CONTEXTO ESTA NA MEM.PRINC?
       BL  CICLO       *NAO-->VAI VER OUTRO USUARIO
*ENTRADA DO DIRETORIO SERA SEPARADA
       SBL  ACECON+1   *ACECON:=NUMERO DE REFERENCIAS
```

```
        SBR   BLOCON+1
*END.ABS.PAG-CTX. SERA DETERMINADO
        AND   =&FF          *AC:=NUMERO DO BLOCONTEXTO
        SLLS  =8
        ADD   #PTRPAG       *AC:=BASE+DESLOCAMENTO
        STA   PTR           *PTR APONTA INICIO DA PAG.CT
        LDA   @PTR          *AC:=ACESSOS+NUMBLOCO
        AND   =&FF
        ADD   =1
        STA   PAGREF        *PAGREF:=PAGS.REFEREMCIAD+1
        LDX   =0
*MAIS UMA DAS ENTRADAS DO MAPA SERA CONSULTADA
VISMAP  LDA   MENOS1        *AC:=-1
        ADM   PAGREF        *DECREMENTA NUMERO DE VISITAS
        CMP   =0            *TESTA SE TODAS ENTRADS.VISI
        BCF   VISITA        *NAO-->VISITA
*TESTA SE USUARIO POSSUI ALGUMA PAGINA ALEM DA CTX
        LDA   FLAG
        CMP   =0            *TESTA SE HA PAGINA
        BCF   NOVUSU        *SIM-->VAI VER OUTRO USUARIO
*TESTA SE E POSSIVEL ESCOLHER A PAGINA CONTEXTO
        LDA   USUCON
        MUL   =3
        ADD   =1
        ADD   #PTRUSU
        STA   PTR           *PTR APONTA INFO&USUARIOA
        LBR   @PTR          *AC:=FILA DE ESPERA DO USUAR
        CMP   =2            *TESTA SE FILA AGUARDANDO MM
        BCT   CICLO         *SIM-->VAI VER OUTRO USUARIO
*COMPARA ACESSO DA PAG-CTX COM O MINIMO ATE AGORA
        LDA   ACECON
        CMP   ACEMIN
        BOF   CICLO         *SE ACESSO CTX > ENTAO VOLTA
        STA   ACEMIN        *ATUALIZA VARIAVEL ACESSO-MI
        LDA   =255
        STA   ENTMIN
        LDA   USUCON
        STA   USUMIN
        BRU   CICLO
*ENTRADA DO MAPA SERA VISITADA
VISITA  ICX   =2        *
        LDA   @PTR,X        *AC:=MAPA&XA
        STA   ENTCON        *ENTCON:=MAPA&XA
        CMP   #LAMBDA       *TESTA SE PAGINA  TIPO I
        BCT   VISITA
        CMP   =0            *TESTA SE PAG.NA MEM.PRINCIP
        BL    VISMAP        *NAO-->VAI DEC.NUM.DE VISIT.
        LDA   =1
        ADM   FLAG          *FLAG:=NUM.DE PAGS.NA MEM.PR
```

```
LDA   ENTCON
SBR   BLOCON+1
SBL   ACECON+1
LDA   ACEMIN
CMP   ACECON      *TESTA ACESSO MINIMO E CONSU
BL  VISMAP        *SE MENOR*->RAMIFICA
LDA   ACECON
STA   ACEMIN
LDA   USUCON
STA   USUMIN
LDR   =5          *AC:=(X)
SRLS   =1         *AC:=X/2
SUB   =1          *AC:=NUM.PAGINA CONSULTADA
STA   ENTMIN
BRU   VISMAP
FIN   INI
END   GERLPS
```

ANEXO III

MÓDULO INTERPRETADOR

```
*MODULO INTERPRETADOR: RESPONSAVEL PELA INTERPRETA
*CAO DAS TAREFAS DOS USUARIOS. TODA VEZ QUE ATIVA-
*DO, A PAGINA-CONTEXTO DO USUARIO E CARREGADA  NUM
*TRECHO DESTE MODULO, COM O PROGRESSO DA TAREFA HA
*VERA MODIFICACOES DESTES DADOS. QUANDO A OCORREN-
*CIA DE UM EVENTO FORCAR A SUSPENCAO DA EXECUCAO
*DA TAREFA ATUAL, ENTAO OS DADOS DAQUELE TRECHO,SE
*RAO COPIADOS (PRESERVADOS) NA PAGINA-CONTEXTO COR
*RESPONDENTE. OS EVENTOS RESPONSAVEIS PELA PARALI-
*ZACAO DA INTERPRETACAO PODEM SER:EXTERNOS (FIM DO
*TEMPO DE UCP P/EXPL) OU INTERNOS (TAREFA QUER FA-
*ZER E/S; FIM DE SERVICO; FALTA DE UMA PAGINA ETC)

INTCDS CDS
       RES  32
PTRPAG DATA 0          *END.ABS.DO PRIM.BLOCO
PTRDIR RES  1          *PONTEIRO DIRETORIO SUPERVIS
TAMAPG DATA 256        *TAMANHO DE UMA PAGINA (BYT)
CTE    DATA 806        *PTR PARA ZONA COMUM
PBYTZC RES,1 1         *PRIM.BYTE ZONA COMUM
SBYTZC RES,1 1         *SEGUNDO     "      "    "
TBYTZC RES,1 1         *TERCEIRO    "      "    "
QBYTZC RES,1 1         *QUARTO      "      "    "
LAMBDA DATA 999
PTRCTX RES  1          *END.ABS.DA PAG-CTX USUARIO
PONCON DATA CONTEX     *PONTEIRO CONTEXTO NA CDS
```

```
*DESCRICAO DA PAGINA CONTEXTO.ESSE TRECHO RECEBERA
*A PAGINA CONTEXTO DA TAREFA QUE SERA INTERPRETADA

CONTEX EQU  6           *INICIO DA PAGINA CONTEXTO
MAXPAG RES.1 1          *NUM.MAX.PAGINAS DO PROGRAMA
PAGREF RES.1 1          *NUM.DE PAGS.JA REFERENCIADA
MAPAGI RES  64          *MAPA DAS PAGINAS DO USUARIO
REGACU RES  1           *REGISTRO ACUMULADOR DO USU.
REGIND RES  1           *REGISTRO INDEXADOR   "   "
REGIAR RES  1           *CONTADOR PROGRAMA (I.A.R.)
INDCOM RES  1           *INDICADOR COMPAR.L=0;E=1;G=2
DIV    RES  1           *FUTURAS IMPLEMENTACOES
ESTARE RES  1           *ESTADO DA TAREFA VISTA P/SU
LINFDP RES  1           *LIMITE INFERIOR DO DUMP
LISUDP RES  1           *LIMITE SUPERIOR DO DUMP
LINFTR RES  1           *LIMITE INFERIOR DO TRACE
LISUTR RES  1           *LIMITE SUPERIOR DO TRACE
BREAKP RES  1           *ENDERECO DO BREAKPOINT
ESTINT RES  1           *ESTADO DO INTERPRETADOR
*ESTINT=LAMBDA (LOADER); 0=TESTA SE BREAKPOINT
* 1=TESTA SE TRACE; 2=INCREMENTA "IAR"; 3=INTERPRE
NUPGFA RES  1           *NUMERO DA PAGINA FALTOSA
ENDBUF RES  1           *END.ABS.DO BUFFER
PALALT RES  1           *PALAVRA QUE SERA ALTERADA
CONTEU RES  1           *COM O NOVO CONTEUDO
FILANT RES  1           *FILA ANTERIOR ANTES REM.CT
LEILOA RES  1           *LOADER AGUARD.LEI=0; SAID=1
ULTLOA RES  1           *ULT.END.CARREGADO;LAMBDA=NA
*                       *DA; -1=TIPO "H"
FUTURO RES  44          *AREA DISPONIVEL P/ O FUTURO
       FIN
```

```
*ESTA SECAO CARREGA A PAGINA-CONTEXTO DO USUARIO
*NA AREA DA "CDS" RESERVADA PARA TAL. O END.ABS.
*DA PAG-CTX SERA "BUSCADO" NA ZONA COMUM.E ARMAZE-
*NADO NO PONTEIRO. ESTE PONTEIRO SERA REUTILIZADO
*QUANDO DA PRESERVACAO DA PAG-CTX.

CARLDS LDS
       RES  2
PTR    RES  1          *PONTEIRO P/USO GENERALIZADO
       FIN
CARCTX LPS  CARLDS
INI    LDX  #CTE       *X APONTA P/ZONA COMUM
       DLD  a#6,X      *AC:=END.ABS.PAG.CTX.
       STA  PTR        *PTR APONTA /PAGINA CTX.
       STA  #PTRCTX    *PTRCTX  "   "    "
       LDE  #TAMAPG    *E:=NUM.BYTES P/TRANSMITIR
       LEA  aPTR       *AC:=END.ABS. ORIGEM
       MVS  #MAXPAG    *COPIA PAG-CTX NA "CDS"
       LDA  =1
       IOR  a#6,X      *AVISA ZONA-COMUM:PRESERVADA
       STA  a#6,X      *AINDA NAO TERMINEI
       RTS
       FIN  INI

*ESTA SECAO PRESERVA OS DADOS DA "CDS" NA PAGINA-
*CONTEXTO DO USUARIO.

PRESER LPS  CARLDS
INI    LDE  #TAMAPG    *E:=NUM.BYTES A COPIAR
       LEA  #CONTEX    *AC:=END.INICIAL DE TRANSMIS
       MVS  aPTR       *COPIA DADOS NA PAG-CTX.
       RTS
       FIN  INI
```

```
*ESTA SECAO TRANSFORMA UM ENDERECO LOGICO (INICIAL
*MENTE NO REG.AC.) NO NUMERO DA PAGINA+O DESLOCA -
*MENTO. O NUMERO DA PAGINA SERA USADO COMO INDEXA-
*DOR PARA ACESSAR A ENTRADA DO MAPA. SE A ENTRADA
*DO MAPAASSINALAR A AUSENCIA DA PAGINA ENTAO NA
*PAGINA-CONTEXTO SERA INSERIDA O NUMERO DA PAGINA
*FALTOSA.SE PRESENTE, O NUMERO DE ACESSOS SERA IN-
*CREMENTADO, E O END.ABS.DA POSICAO ESTARA NO ACU-
*MULADOR. SE ACUMULADOR=LAMBDA ENTAO FALTA-PAGINA.
*NO CASO DE OVERFLOW A ROTINA QUE DIVIDE OS CONTA-
*DORES DE REFERENCIAS SERA SOLICITADA.

LDSFOR LDS
       RES  2
SALVAE RES  1
SALVAC RES  1
SALVAX RES  1
APONTA RES  1
LINEND RES  1          *PARTE RELATIVA A LINHA
PAGEND RES  1          *PARTE RELATIVA A PAGINA
ACESSO RES  1
PTRBLO RES  1
       FIN
FORMAR LPS  LDSFOR
INI    DST  SALVAE     *SALVA REGISTROS: "E" "A"
       STX  SALVAX     *                 "X"
       SBR  LINEND+1   *GUARDA NUMERO DA LINHA
       SBL  PAGEND+1   *GUARDA NUMERO DA PAGINA
       LDX  PAGEND
       ICX  =2,X       *X APONTA P/MAPA DAS PAGINAS
       LDA  @#PONCON,X *AC:=ENTRADA DO MAPA
       CMP  #LAMBDA    *TESTA SE PAGINA TIPO-I
       BCF  $+7        *NAO-->RAMIFICA
*PAGINA FALTOSA *PAGINA CONTEXTO SERA NOTIFICADA
FALTOU LDA  PAGEND     *AC:=NUMERO-PAGINA-AUSENTE
       STA  #NUPGFA    *ASSINALA NA CTX, NUM.PAG.FA
       LDA  #LAMBDA    *FLAG:=LAMBDA (=PAG.AUSENTE)
VOLTA  LDE  SALVAE     *RESTAURA REGISTRO "E"
       LDX  SALVAX     *     "     "     "X"
       RTS             *RETORNA
       BAN  FALTOU     *SE PAGINA NO DISCO-->RAMIF.
*NUMERO DE ACESSOS A PAGINA REFERENCIADA SERA INCR
       SBL  ACESSO+1   *ACESSO:=NUMERO DE ACESSOS
       SBR  PTRBLO+1   *PTRPAG:=NUMERO DO BLOCO
       LDA  ACESSO     *AC:=NUMERO DE ACESSOS
       CMP  =127       *TESTA SE ACESSO MAXIMO
       BCF  INCACE      *NAO-->RAMIFICA
*OVERFLOW NO CONTADOR-->CONTADORES SERA DIVIDIDOS
```

```
         CLS   OVRFLW
         SRLS  =1          *FAZ ACESSO=ACESSO/2
INCACE   ADD   =1          *INCREMENTA ACESSO
         STA   ACESSO
         LBR   PTRBLO+1
         LBL   ACESSO+1    *AC:=ACESSO+NUMERO DO BLOCO
         STA @#PONCON,X    *ATUALIZA MAPA
*ENDERECO ABSOLUTO DO INICIO DA PAG.SERA CALCULADO
         LDA   PTRBLO      *AC:=NUMERO DO BLOCO
         SLLS  =8          *NUM.BLOCO * 256
         ADD #PTRPAG       *AC:=END.ABS.INICIO DA PAGINA
         ADD   LINEND      *AC:=END.ABS.DA POIC.REFER.
         BRU   VOLTA       *--->V O L T A
         FIN   INI
```

```
*SEMPRE QUE O CONTADOR DE ACESSOS ULTRAPASSAR "127"
*ESTE MODULO SERA CHAMADO. ELE DIVIDE TODOS OS CON
*TADORES POR 2. SOMENTE SE A PAGINA ESTIVER PRESEN
*TE (OU SEJA SE NAO FOR TIPO I NEM TIPO II).

OVRLDS LDS
       RES  2
DIRETO RES  1
PONTEI RES  1
USUCON RES  1
MENOS1 DATA &FFFF
CONTAR RES  1
SALVAE RES  2
SALVAX RES  1
PTRSGP DATA 808        *APONTADOR SEGUNDA PALAV.Z.C
       FIN
OVRFLW LPS  OVRLDS
INI    DST  SALVAE     *SALVA
       STX  SALVAX     *          REGISTROS
       LDA #PTRDIR
       STA  DIRETO     *DIRETO APONTA DIRETORIO
*PAG-CTX DO USUARIO SERA ATUALIZADA
       CLS  PRESER
*DIRETORIO SERA VARRIDO
       LDA  MENOS1
       STA  USUCON
OUTROU LDA  =1
       ADM  USUCON      *INCREMENTA NUMERO USUARIO
       CMP  =32          *TESTA SE TODOS
       BCF  $+8         *NAO--->RAMIFICA
VOLTA  LDX  PTRSGP      *  X APONTA SEG.PAL.DA ZONA C
       LDA #PTRCTX      *AC:=END.DA PAG.CTX USU.COR.
       STA @#6,X        *PREPARA END.PARA ROTINA CAR
       CLS CARCTX       *RESTAURA "CDS" C/PAG-CTX
       DLD  SALVAE      *SALVA REGISTROS "E"  "A"
       LDX  SALVAX      *  SALVA REGISTRO "X"
       RTS              *RETORNA
*    ATENDIMENTO A UM USUARIO
       LDX  USUCON
       ICX  =0,X        *X APONTA ENTRADA DO DIRETORIO
       LDA @DIRETO,X    *AC:=ENTRD. DIRETORIO
       CLS  DIVIDE      *DIVIDE CONTADOR POR 2
       STA @DIRETO,X    *GUARDA ENTRADA
       CMP #LAMBDA      *TESTA SE CTX E DO TIPO II
       BCT  OUTROU      *SIM-->VAI VER OUTRO USUARIO
       BAN  OUTROU      *SE CTX EM DISCO--> "  "  "
*ENTRADAS DO MAPA SERA VISITADAS
       AND  =#FF        *AC:=NUM.BLOCO
       SLLS  =8         *AC:=NUM.BLOCO*256
```

```
        ADD  #PTRPAG      *AC:=BASE+DESLOCAMENTO
        STA  PONTEI       *PONTEI APONTA INICIO DA CTX
        LDA  @PONTEI      *AC:=INICIO DA PAG.CTX.
        SRLS  =8          *AC:=NUM.MAX DE PAGINAS
        ADD  =1
        STA  CONTAR       *CONTAR:=NUM.MAX.+1
LOOP    LDA  MENOS1       *AC:=-1
        ADM  CONTAR       *DECREMENTA NUMERO DE PAGINAS
        BAZ  OUTROU       *SE TODAS-->OUTRO USUARIO
        LDX  CONTAR
        ICX  =0,X         *X APONTA MAPA$CONTARA
        LDA  @PONTEI,X    *AC:=MAPA$CONTARA
        CLS  DIVIDE       *DIVIDE ACESSO
        STA  @PONTEI,X    *ATUALIZA MAPA
        BRU  LOOP         *VAI VISITAR OUTRA ENTRADA
        FIN  INI
```

```
*ESTA SECAO TEM COMO OBJETIVO DIVIDIR O CAMPO DE
*ACESSOS DE UMA PAGINA POR 2. SE A PAGINA NAO
*ESTIVER PRESENTE ENTAO NADA SERA FEITO

DIVLDS LDS
       RES  2
TEMP   RES  1
       FIN
DIVIDE LPS  DIVLDS
INI    STA  TEMP
       LDE  =0          * ZERAR REGISTRO "E"
       CMP #LAMBDA
       BCT  $+6         *SE PAG.TIPO I-->RETORNA
       BAN  $+5         *SE PAG.TIPO II-->RETORNA
       SRCD =8          *AC:=NUMERO DE ACESSOS
       CMP  =0          *SE CONTADOR NULO ENTAO
       BCF  $+3         *----->RETORNA
       LDA  TEMP        *RESTAURA
       RTS              *RETORNA
       SRLS =1          *DECREMENTA CONTADOR
       SLCD =8          *CONCATENA ENTRADA MAPA
       RTS               * RETORNA
       FIN  INI
```

```
*****************************************************
*                                                   *
*NUCLEO CENTRAL DO MODULO INTERPRETADOR.            *
*                                                   *
*****************************************************

INTLDS LDS
       RES  2
PIR    RES  1          *PONTEIRO PARA USO GERAL
MENOS1 DATA &FFFF
MAXEND DATA  1000      *END.MAX.PERMITIDO
BRANCS DATA &4040      *BRANCOS EBCDIC
TEMPOR RES  1          *VARIAVEL TEMPORARIA
APONTA RES  1
MESS01 TEXT "TIPO""H""JA FOI TRANSMITIDO"
MESS02 TEXT "LIMITE TRACE INVALIDO"
MESS03 TEXT "FALTA TIPO""H"""
MESS04 TEXT "END.DE CARGA INVALIDO"
MESS05 TEXT "APENAS OS TIPOS""I""E""R""SAO CERTOS"
MESS06 TEXT "END.INICIO DE EXECUCAO INVALIDO"
       END
CODIOP RES  1          *CODIGO DE OPERACAO
OPERAN RES  1          *OPERANDO DA INSTR.DECODIFIC
TABELA DATA INV,LOAD,STO,LEI,SAI,HAL,ADD,SUB,MUL
       DATA DIVI,AND,SHL,SHR,CMP,BL,BE,BG,BRU,OU
MIL    DATA 1000
       FIN
INTLPS LPS  INTLDS
ATIVA  SBR #SBYTZC     *PREPARA PARAM.DO INT/SUP
       LDX #CTE        *APONTA P/ZONA-COMUM
       OLD #PBYTZC     *CARREGA PARAMETROS
       DST @#6,X       *ARMAZENA " NA ZONA-COMUM
       CLS  PRESER     *MOVE AREA-TRAB.P/PAG.CTX.
       LDA  =4         *****                  *****
       LDX  =4         *                          *
       CSV  M:IT       *  =ATIVA SUPERVISOR=      *
       LDA  =4         *                          *
       LDX  =6         *                          *
       CSV  M:IT       *                      *****

*INTERPRETADOR AGUARDANDO USUARIO--> SE ATIVADO:

INI    LDX #CTE        *X APONTA P/ ZONA-COMUM
       LDA @#6,X       *AC:=PARAMETROS DO SUP/INT
       BAZ  NOVUSU     *NOVO USUARIO-->RAMIFICA
LOOP   LDX #CTE        *APONTA P/ZONA-COMUM
       LDA @#6,X       *AC:=PARAMETROS DO SUP/INT
       SRLS =8         *ISOLA PARAMETROS
```

```
        AND   =&0F          *TESTA SE SUP.QUER TROCAR
        BAZ   ROTINA        *NAO--> VAI P/ ROTINA

* CONTROLE PASSARA PARA O SUPERVISOR

        LDA   =2            *FLAG:= JA SALVEI.MUDE.
        BRU   ATIVA

*NOVO USUARIO: CONTEUDO DA CONTEXTO SERA COPIADO

NOVUSU CLS    CARCTX        *CARREGA PAGCTX NA "CDS"
*ROTINA DE INTERPRETACAO

ROTINA LDA #ESTINT          *AC:=ESTADO TAREFA P/INTERP.
       CMP #LAMBDA          *ESTADO = "LOADER"  ?
       BCF  $+9             *NAO--> VAI VER ESTADOS
*PROGRAMA EM FASE DE CARREGAMENTO

       LDA #LEILOA          *TESTA SE LOADER PEDIU LEIT.
       BNZ  $+3             *NAO-->PEDE LEITURA
*JA HA NO BUFFER MAIS UMA LINHA FONTE

       CLS   LOADER         *MANDA LOADER ANALISA-LA
       BRU   ATIVA          *VAI ATIVAR SUPERVISOR
       LDA   =0             *
       STA #LEILOA          *FAZ-->"LOADER" AGUARD.LEIT.
       LDA   =3             *FLAG:=QUERO LEITURA
       BRU   ATIVA          *VAI ATIVASUPERVISOR
*PROGRAMA JA FOI CARREGADO:TESTE DAS OUTRAS FASES

       CMP   =0             *TESTA SE FASE=0(BREAKPOINT)
       BCF   $+8            *NAO-->RAMIFICA
       LDA   =1             *PREPARA P/NOVA FASE
       STA #ESTINT          * (1=TESTA SE TRACE)
       LDA #REGIAR          *AC:=I A R
       CMP #BREAKP          *TESTA SE BREAKPOINT ATINGIDO
       BCF   $+5            *NAO-->VAI P/OUTRA FASE
       LDA   =7             *FLAG:=BREAKPOINT ATINGIDO
       BRU   ATIVA          *VAI ATIVAR O SUPERVISOR
*TESTA SE TRACE OPERATIVO

       CMP   =1             *TESTA SE FASE=1("TRACE"?)
       BCF   $+11           *NAO-->VAI VER SE OUTRA FASE
       LDA   =1             *ZC:=1
       ADM #ESTINT          *ESTINT:=ESTINT+1
       LDA #REGIAR          *AC:= I A R
       CMP #LINFTR          *IAR < LIM.INF.TRACE ?
       BL   $+8             *SIM-->VAI INCREMENTAR IAR
       CMP #LISUTR          *LINF<=IAR<=SUP DO TRACE?
```

```
        BGE   $+6          *NAO-->VAI INCREMENTAR I A R
        CLS   TRACE        *PREPARA LINHA P/ TRACE
        LDA   =4           *FLAG:= QUERO SAIDA
        BRU   ATIVA        *ATIVA SUPERVISOR
*TESTA SE FASE E A DE INCREMENTAR O " I A R "

        CMP   =2           *FASE = 2? (2=INCREM.IAR)
        BCF   $+5          *NAO-->RAMIFICA
        LDA   =1           *AC:=1
        ADM #ESTINT        *ESTINT:=FASE DE INTERPRET.
        LDA   =2
        ADM #REGIAR        *IAR:=IAR+2
        LDA #REGIAR        *AC:=REGIAR
        CMP   MAXEND       *TESTA IAR COM O END.MAXIMO
        BL    $+3          *SE MENOR-->RAMIFICA

INV     LDA   =9           *FLAG:=ERRO NA EXECUCAO
        BRU   ATIVA        *VAI ATIVAR SUPERVISOR
*ENDERECO LOGICO SERA TRANSFORMADO EM END.FISICO

        CLS   FORMAR       *CONVERTE ENDERECO
        CMP #LAMBDA        *TESTA SE PAGINA AUSENTE
        BCF   $+3          *NAO-->RAMIFICA
PAGFAU  LDA   =5           *FLAG:=PAG.FALTOU
        BRU   ATIVA        *ATIVA SUPERVISOR
        STA   PTR          *PTR APONTA P/INSTRUCAO
        LDA @PTR           *AC:=INSTRUCAO
        CLS   DECODI       *DECODIFICA
        CLS   FORMAR       *CONVERTE OPERANDO
        STA   PTR          *PTR APONTA PARA O E.E.
        LDA   CODIOP       *AC:=CODIGO DE OPERACAO
        CMP   =19          *TESTA SE VALIDO
        BGE   INV          *NAO--> INVALIDO
        LDA   PTR          *AC:=END.EFETIVO
        CMP #LAMBDA        *PAGINA PRESENTE ?
        BCT   PAGFAU       *NAO-->PAGINA FALTOSA
        LDA   =0           *
        STA #ESTINT        *FASE:= INICIAL
        LDX   CODIOP       *X := CODIGO DE OPERACAO
        ICX   =0,X         *X:= 2$CODIGO DE OPERACAO
        BRX   @TABELA      *RAMIFICA SEGUNDO CODIOP

LOAD    LDA   @PTR         *AC:=(OPERANDO)
        STA #REGACU        *REGACU:=(OPERANDO)
        BRU   LOOP

STO     LDA #REGACU        *AC:=REGACU
        STA   @PTR         *(EE):=REGACU
        BRU   LOOP
```

```
*ROTINA DE LEITURA
*VARIAVEL LEILOA INDICA SE LEITURA JA FOI PEDIDA

LEI     LDA   =3          *RESTAURA ESTADO DO INTERPR.
        STA #ESTINT       *PARA QUE POSSA VOLTAR SE
        LDA   =1          *TESTA SE JA PEDIU LEITURA
        CMP #LEILOA       *LEITURA JA FOI  PEDIDA?
        BCT   $+4         *SIM-->RAMIFICA
*LEITURA AINDA NAO FOI PEDIDA
        ADM #LEILOA       *LEILOA INDICA LEITURA FEITA
        LDA   =3
        BRU   ATIVA
*LEITURA JA FOI REALIZADA
        LDA   =0
        STA #LEILOA       *RESTAURA P/PROXIMA LEITURA
        LDA #ENDBUF       *AC:=END-DO-BUFFER
        STA   APONTA      *APONTA PARA O BUFFER DO USU
*TESTA SE OS 72 CARACTERES PODEM SER MOVIMENTADOS
        LDA   =70         *AC:=NUMERO DE CARACTERES
        ADM   OPERAN      *AC:=OPERAN + 70
        CMP   MAXEND      *TESTA SE ENDERECO PERTENCE
        BGE   INV         *NAO ENCERRA
*DADOS SEAO MOVIMENTADOS
        LDX   =70         *X APONTA P/ O ULTIMO DADO
        LDA   OPERAN      *AC:=DESTINO
        CLS   FORMAR      *TRANSF.END.LOG/FISICO
        CMP   #LAMBDA     *TESTA SE POSICAO ESTA PRESENTE
        BCT   PAGFAU      *NAO-->VOLTA PEDINDO PAGINA
        STA   PTR         *PTR APONTA ENDERECO DESTINO
        LDA @APONTA,X     *AC:= + 2 BYTES
        STA @PTR          *MEMGOPERANA:=BUFFERGXA
        LDA   OPERAN
        SUB   =2          *
        STA   OPERAN      *OPERAN APONTA POSICAO ANTER
        DCX   =2          *X APONTA POSIC.ANT.DO BUFF.
        BCT   $-11        *SE X NAO-NEGATIVO-->VOLTA
        LDA   =0          *INFORMA QUE CONTROLE IRA P/
        STA #ESTINT       *A PROXIMA INSTRUCAO
        BRU   LOOP        *VAI P/PROX.INTRU.

*ROTINA RESPONSAVEL PELA SAIDA

SAI     LDA   =3          *RESTAURA ESTADO DO INTERPRE
        STA #ESTINT
        LDA #ENDBUF
        STA   APONTA      *APONTA PARA O BUFFER
        LDA   =70
        ADM   OPERAN      *OPERAN APONTA PARA FINAL AR
        CMP   MAXEND      *FINAL PERTENCE AO PROG?
```

```
        BGE   INV          *NAO-->INSTR.INVALIDA
        LDX   =70          *X APONTA P/O FINAL BUFFER
MOVSAI  LDA   OPERAN
        CLS   FORMAR       *TRANSF.END.LOGICO/FISICO
        STA   PTR          *PTR APONTA P/END.FISICO
        CMP #LAMBDA
        BCT   PAGFAU
        LDA @PTR           *AC:= 2 CARACTERES
        STA @APONTA,X      *BUFFER(X):=+ 2 CARACTERES
        LDA   OPERAN
        SUB   =2
        STA   OPERAN       *OPERAN APONTA P/ 2CHR.ANTER
        DCX   =2           *X APONTA 2 CHR.ANT.NO BUFF
        BCT   MOVSAI       *TODOS JA MOVIMENTADOS ?N.->
*CARACTERES JA FORAM MOVIDOS. ESTADO SERA RESTAURA
        LDA   =0
        STA #ESTINT
        LDA   =4
        BRU   ATIVA        *VAI PEDIR AO SUPERVISOR SAI

HAL     LDA   =6           *HALT-->SUPERVISOR SERA ATIV
        BRU   ATIVA

ADD     LDA #REGACU        *AC:=REACU
        ADD @PTR           *AC:=REGACU+(EE)
        STA #REGACU        *REGACU:=REGACU+(EE)
        BRU   LOOP

SUB     LDA #REGACU        *AC:=REGACU
        SUB @PTR           *AC:=REGACU-(EE)
        STA #REGACU        *REGACU:=REGACU-(EE)
        BRU   LOOP

MUL     LDE   =0
        LDA #REGACU
        MUL @PTR
        STA #REGACU        *REGACU:=REGACU*(EE)
        BRU   LOOP

DIVI    LDA @PTR           *AC:=DIVISOR
        BAZ   INV          *SE DIVISOR=0-->INVALIDO
        LDA #REGACU
        DIV @PTR
        STA #REGACU        *REGACU:=REGACU/(EE)
        BRU   LOOP

AND     LDA #REGACU
        AND @PTR
        STA #REGACU        *REGACU:=REGACU AND (EE)
```

```
        BRU   LOOP

SHL     LDX @PTR
        LDA #REGACU
        SLLS =0,X
        STA #REGACU     *REGACU:=REGACU SHIFT<-(EE)
        BRU   LOOP

SHR     LDX @PTR
        LDA #REGACU
        SRLS =0,X       *REGACU:=REGACU SHIFT-->(EE)
        STA #REGACU
        BRU   LOOP

CMP     LDA   =0
        STA #INDCOM
        LDA #REGACU
        CMP @PTR
        BL    LOOP
        BE    $+3
        LDA   =1
        ADM #INDCOM
        LDA   =1
        ADM #INDCOM
        BRU   LOOP

BL      LDA #INDCOM
        CMP   =0
TESTIN  BCF   LOOP

BRU     LDA   OPERAN    *AC:=END.ONDE RAMIFICARA
        SUB   =2
        STA #REGIAR
        BRU   LOOP

BE      LDA #INDCOM
        CMP   =1
        BRU   TESTIN

BG      LDA #INDCOM
        CMP   =2
        BRU   TESTIN

OU      LDA #REGACU
        IOR @PTR
        STA #REGACU     *REGACU:=REGACU  OU  (EE)
        BRU   LOOP
        FIN   INI
```

```
*ESTA SECAO E RESPONSAVEL PELA DECODIFICACAO DE
*UMA INSTRUCAO EQUE ESTA NO REGISTRO ACUMULADOR)
*O CODIGO DE OPERACAO FICARA NA VARIAVEL CODIOP
* E O OPERANDO NA VARICEL OPERAN******************

DECODI LPS  INTLDS
INI    LDE  =0
       DIV  MIL
       XAE
       STA  OPERAN
       STE  CODIOP
       RTS
       FIN  INI
```

```
*SECAO RESPONSAVEL PELO CARREGAMENTO DO PROGRAMA
*DE UM USUARIO

LOADER LPS  INILDS
INI    LDA #ENDBUF
       STA  PTR          *PTR APONTA INICIO DO BUFFER
       LBR @PTR          *AC:=CODIGO DA LINHA FONTE
       CMP  =&C8         *E TIPO "H" ?
       BCF  DIFAGA       *NAO-->VAI EXAMINAR SE TIPOT
*CARTAO TIPO "H" TESTES SERAO EXECUTADOS

       LDA #ULTLDA
       CMP #LAMBDA       *E O PRIMEIRO TIPO "H"?
       BCT  TIPOH        *SIM-->VAI ATENDER TIPO "H"
*IMPRIME: TIPO "H" JA FOI TRANSMITIDO"
       LDE  =27
       LEA  MESS01
MOVMES MVS @PTR
       LDA  =1
       STA #LEILDA
       LDA  =4
       RTS

*ATENDIMENTO CARTAO DO TIPO "H"

TIPOH  LDX  =2
       LDA @PTR,X
       CMP  BRANCS       *TESTA SE TRACE INEXISTENTE
       BCT  HLIDO
       LEA @PTR,X
       CSV  M:DCBN       *CONVERTE
       XAE               *AC:=LIM.INF.TRACE
       CMP  MAXEND       *TESTA END.C/MAXIMO
       BL   $+4          *MENOR-->RAMIFICA
INVTRA LDE  =21
       LEA  MESS02       *IMPRIME MENSAGEM INVALIDA
       BRU  MOVMES
       STA #LINFTR       *LINFTR:=LIM.INF.TRACE
       LDX  =8
       LEA  @PTR,X
       CSV  M:DCBN        *CONVERTE
       XAE               *AC:=LIM.SUP.TRACE
       CMP #LINFTR       *TESTA SE INF<SUP
       BL  INVTRA        *NAO-->INVALIDO
      CMP  MAXEND        *COMPARA COM ENDERECO MAXIMO
       BGE  INVTRA       *SE MAIOR-->INVALIDO
       STA #LISUTR       *LISUTR:=LIM.SUP.TRACE
HLIDO  LDA  MENOS1       *AC:=-1
       STA #ULTLDA       *FLAG:=JA FOI LIDO O "H"
```

```
FONTLI LDA   =0
       STA #LEILDA     *AGUARDANDO LEITURA
       LDA   =3        *PEDE CARTAO
       RTS

*CARTAO NAO E DO TIPO "H"

DIFAGA LDA #ULTLDA     *AC:=ULTIMO ENDERECO LIDO
       CMP   #LAMBDA   *TIPO "H" JA FOI LIDO?
       BCF   $+4       *SIM-->RAMIFICA
       LDE   =13       *NAO-->IMPRIME;
       LEA   MESS03    *FALTA CARTAO DO
       BRU   MOVMES    *  TIPO  "H"
       LBR   @PTR      *AC:=TIPO DA LINHA FONTE
       CMP   =%E3      *E TIPO "T"
       BCF   TIPOR     *NAO-->VAI TESTAR SE TIPO R

*ATENDIMENTO CARTAO TIPO "T"

     LDX   =2
       LDA   @PTR,X    *AC:=END. DE CARGA
       CMP   BRANCS    *TESTA SE CAMPO PREENCHIDO
       BCF   CNVEND    *SIM-->RAMIFICA
       LDA   =2
       ADD #ULTLOA     *INCREMENTA END.DE CARGA
TESTEM CMP   MAXEND    *TESTA SE END.PERMITIDO
       BL    CONVEL    *SIM-->RAMIFICA
       LDE   #21       *NAO-->IMPRIME:
       LEA   MESS04    *ENDERECO DE CARGA
       BRU   MOVMES    *   INVALIDO
CNVEND LEA @PTR,X
       CSV   M:DCBN    *CONVERTE ENDERECO DE CARGA
       XAE
       BRU   TESTEM

*ENDERECO LOGICO SERA CONVERTIDO

CONVEL STA   TEMPOR
       CLS   FORMAR
       CMP #LAMBDA     *TESTA SE PAGINA PRESENTE
       BCF   $+3       *SIM-->RAMIFICA

*PAGINA FALTOSA SERA REQUERIDA

       LDA   =5
       RTS
       STA   APONTA    *APONTA P/ END.DE CARGA
       LDX   =8
       LEA @PTR,X
```

```
        CSV   M:DCBN      *CONVERTE INSTRUCAO
        XAE
        STA aAPONTA       *CARREGA NA MEMORIA
        LDA  TEMPOR
        STA #ULTLOA       *ATUALIZA ULTIMO END.CARREF
        BRU  FONTLI       *VAI PEDIR + FONTE

*TESTA SE CARTAO E DO TIPO "R"

TIPOR   CMP  =&D9         *E DO TIPO  "R"
        BCT  $+4          *SIM-->RAMIFICA
        LDE  =36
        LEA  MESS05
        BRU  MOVMES
        LDX  =2
        LDA aPTR,X
        CMP  BRANCS
        BCF  $+3
        LDA  =0
        BRU  $+8
        LEA aPTR,X
        CSV  M:DCBN       *CONVERTE ENDERECO INICIO EXE
        CMP  MAXEND
        BL   $+4
        LDE  =31
        LEA  MESS06
        BRU  MOVMES

*PREPARACAO DO REGISTRO IAR

        SUB  =2
        STA #REGIAR
        LDA  =0
        STA #ESTINT       *FASE : INICIO INSTRUCAO
        STA #LEILOA       *FLAG:=AINDA NAO FIZ LEITURA
        LDA  =1           *FAZ ESTADO DA TAREFA
        STA #ESTARE       *:= AGUARDANDO SIGA
        LDA  =8           *FLAG:=FIM DO LOADER
        RTS
        FIN  INI
TRACE LPS INTCDS
INI RTS
 FIN INI
        END  INTLPS
```

ANEXO IV

CARREGADOR DO SISTEMA

```
*MODULO RESPONSAVEL PELO CARREGAMENTO DO SISTEMA.
* E NECESSARIO (ANTES DE CHAMA-LO) FAZER FO,Q&3200
*OU SEJA DESLOCAR O FOREGROUND DE 3200 (HEXADECIM)
*O PRIMEIRO PASSO DESTE PROGRAMA,  CARREGA  NA
*  MEMORIA O PROGRAMA QUE CONTROLARA O RELOGIO
* (ASSOCIADO AO NIVEL 25). APOS CARREGADO E CONEC-
*TADO SERA SOLICITADA A DATA DO DIA NO FORMATO
*=DDMMAA= . ESTA DATA SERA POSTERIORMENTE ARMAZENA
*DA NO SUPERVISOR, PARA QUE POSSA SER OFERECIDA.
*APOS SERA SOLICITADA A HORA CORRENTE NO FORMATO
* HH.MM. A HORA FORMECIDA SERA CONVERTIDA EM UM
*BINARIO DE 32 BITS,QUE SERA ATUALIZADO DE 2,56 EM
*2,56 SEGUNDOS. COM ESSE BINARIO, PODEMOS SABER A
*CADA MOMENTO QUAL A HORA CORRESPONDENTE.
* O SEGUNDO PASSO CONSISTE NO CARREGAMENTO EM MEMO
*RIA, DO MODULO QUE FICARA ASSOCIADO AO RELOGIO
* (DO NIVEL 24) QUE DEMARCARA OS "DELAYS" DE U.C.P
* O TERCEIRO PASSO CARREGA O MODULO SUPERVISOR E O
*ASSOCIA AO NIVEL 4 DE INTERRUPCAO.
* O QUARTO PASSO CARREGA O MODULO DE GESTAO, E CO-
*NECTA-O AO NIVEL 3. EM SEGUIDA O NUMERO DE PAGI-
*NAS E O SOLICITADO. ESSE NUMERO SERA ARMAZENADO
*NA VARIAVEL ADEQUADA DO MODULO DE GESTAO. O ENDE-
*RECO LOGO APOS O MODULO DE GESTAO SERA ARMAZENADO
*NA VARIAVEL PABSOL.
* O QUINTO PASSO CARREGA O MODULO INTERPRETADOR
*NA REGIAO ONDE TERMINAM AS PAGINAS DO  SISTEMA.
*E ASSOCIA-O AO NIVEL 1. O ENDERECO ABSOLUTO DA
*PRIMEIRA PAGINA TAMBEM  E ARMAZENADO NESSE MODULO
* O SEXTO PASSO CONSISTE EM CHAMAR O PROGRAMA QUE
*INICIALIZARA OA INTERFACE DAS LINHAS ASSINCRONAS.
* O SETIMO PASSO CONSISTE NA LIGACAO DAS PAGINAS
*A LISTA  AVAIL  QUE ESTA SITUADA NA GESTAO).
*FINALMENTE, O MODULO SUPERVISOR SERA ATIVADO, E
*O PRESENTE PROGRAMA SERA ABANDONADO.
```

```
INICDS CDS
       RES  32          *AREA TRABALHO MONITOR
DDMMAA RES  3           *DATA FORMATO DD-MM-AA
Q      RES  1           *TOTAL DE PAGINAS
RESERV RES  128
       FIN
*
INILDS LDS
       RES  2
TAMINT DATA &0200
PROG1  TEXT "N25   "
PROG2  TEXT "N24   "
PROG3  TEXT "SUPSTS"
PROG4  TEXT "GERSTS"
PROG5  TEXT "INTSTS"
PROG6  TEXT "ESTKRP"
ENDINI DATA &3010
FALTA  TEXT "O PROGRAMA          AUSENTE"
       TEXT " PROCESSO ABANDONADO"
FALCB  DATA 0,&8007,#FALTA,46
TEXTO  DATA FALTA
PTRCAB DATA RESERV
T8     DATA &00A0,&100E,0,0,0
T13    RES  1
T14    DATA  #RESERV
T15    DATA 256
T16    DATA &0004
T17    DATA &0100,0,0
T20    RES  1
T21    RES  1
TEMP   RES  1
TAMAN  RES  1           *TAMANHO DO PROG.CARREGADO
PTRELO DATA 32          *END.REL.CONTADOR DE TEMPO
PAGSUP DATA 92          *END.ABS.PAGINAS P/SUPERVISOR
PTRDAT DATA 94          *END.ABS.DATA-DO-DIA
RELSUP DATA 100         *END.ABS.NO SUP.DO CONTADOR
PAGEND DATA 0           *END.ABS.ONDE INICIAM PAGS.
PAGERE DATA 36          *END.ABS.PAGINAS P/GERENCIA
PAGINT DATA 64          *END.ABS.PAGINAS P/INTERPRET
PTRDIR DATA 354         *END.ABS.DIRETORIO P/SUPERV.
PTRUSU DATA 258         *END.ABS.FILAS USUARIOS
CTBIN DATA &0514,&E3E8,0,0
CTBOU  DATA &0510,&E3E8,0,0
TOT    DATA  1          *LIGACAO DAS PAGINAS
LAMBDA DATA 999         *CONSTANTE LAMBDA
CT256  DATA 256
       FIN
LIGAR  LPS  INILDS
```

```
INI     LDA   PAGEND
        STA   TEMP
        LDA   TOT
LOOP    STA @TEMP
        LDA   TEMP
        ADD   CT256
        STA   TEMP
        LDA   =1
        ADM   TOT
        CMP #0
        BCF   LOOP
        LDA   LAMBDA
        STA @TEMP
        RTS
        FIN   INI
INILPS LPS   INILDS
INI     XAE
        LEA #RESERV
        XAE
        LDX   =14
        CSV   M:BIB
        CMP   =0
        BGE   $+9
        LDX   =30
        LDE   =6
        LEA   PROG6
        MVS @TEXTO,X
        LEA   FALCB
        CSV   M:IO
        CSV   M:WAIT
        CSV   M:EXIT
*PROGRAMA SERÁ CARREGADO
        STX   PTRCAB
        LDR   =2
        ADM   PTRCAB
        LDX   =12
        LDA @PTRCAB,X
        STA   T13
        LDA   ENDINI
        CNA
        STA   T20
        LDX   =10
        LDA @PTRCAB,X
        STA   T21
        STA   TAMAN
        LDE   =28
        LEA   T8
        MVS   #32
        CSV   M:LOAD
```

```
LDA   ENDINI
SUB   TAMAN
STA   ENDINI      *NOVO ENDERECO CARREGAMENTO
LDA   #62         *AC:=END.REL.PROG.CARREGADO
RTS
FIN   INI
```

```
*PROGRAMA "LOADER" - CARREGAMENTO DO SISTEMA -

LOADER LPS   INILDS
*LINHAS DOS TERMINAIS SERAO INICIALIZADAS
INI     LEA   CTBIN
        CSV   M:ASGN
        LEA   PROG6
        CLS   INILPS
        LDX   =4
        CSV   M:CNEC
        LDA   =4
        CLS   ATIVAR
        LDA   =4
        CSV   M:KILL     *ABANDONAR TAREFA -
        LDA   TAMAN
        ADM   ENDINI     *RESTAURA END.INI.CARREGAMTO
*CARREGAMENTO PROGRAMA MARCA-TEMPO
        LEA   PROG1
        CLS   INILPS
        LDX   =25
        CSV   M:CNEC
        LDA   #58
        ADM   PTRELO     *PTRELO:=END.ABS.CONTEMPO
*DATA DO DIA SERA REQUERIDA
        CLS   PEDATA
*HORA CORRENTE SERA SOLICITADA
        CLS   PEHORA
        DST   @PTRELO    *ARMAZENA HORA NO CONTEMPO
        LDA   =25
        CLS   ATIVAR     *ATIVA NIVEL 25
*PROGRAMA DEMARCADOR DOS DELAYS SERA CARREGADO
        LEA   PROG2
        CLS   INILPS
        LDX   =24
        CSV   M:CNEC
*MODULO SUPERVISOR SERA CARREGADO
        LEA   PROG3
        CLS   INILPS
        LDX   =4
        CSV   M:CNEC
        LDA #58          *AC:=BASE G SUPERVISOR
        ADM   PAGSUP     *PAGSUP:=EED.ABS.PTR PRIM.PA
        LDA   #58
        ADM   PTRDAT     *PTRDAT:=END.ABS.DATA NO SUP
        LDA   #58
        ADM   RELSUP     *RELSUP:=END.ABS.NO SUP.DO R
        LDA   #58
        ADM   PTRDIR     *PTRDIR:=END.ABS.NO SUP.DIRE
        LDA   #58
```

```
        ADM  PTRUSU     *PTRUSU:=END.ABS.FILAS USUAR
*MOVE 6 CARACTERES DA DATA DO DIA P/ O SUPERVISOR.
        LDE  =6
        LEA #DDMMAA
        MVS @PTRDAT
*ATUALIZA APONTADOR NO SUP.DA HORA CORRENTE
        LDA  PTRELO
        STA @RELSUP
*MODULO GERENCIA SERA CARREGADO
        LEA  PROG4
        CLS  INILPS
        LDX  =3
        CSV  M:CNEC
        LDA  #58
        ADM  PAGERE
*END.ABS.DO PRIM.BLOCO DE MEMORIA SERA PRESERVADO
        LDR  =2
        SUB  ENDINI
        STA  PAGEND     *PAGEND:=END.ABS.PRIM.BLOCO
*TOTAL DE PAGINAS SERA SOLICITADO
CARA    CLS  PEDPAG     *PEDE NUMERO DE BLOCOS
        CNA
        ADD  ENDINI     *CALCULA
        CMP  TAMINT     *TESTA SE HA ESPACO P/INTERP
        BGE  CONTIN
*NUMERO DE PAGINAS EXCEDE TOTAL MEMORIA DISPONIVEL
        LNE
        LDA  =&10
        XAE
        WD
        LDE  =&20
        WD
        DIT
        BRU  CARA       *PEDE NUMERO DE PAGINAS NOVO
CONTIN STA  ENDINI
        CLS  LIGAR      *LIGA AS PAGINAS NA LISTA AV
*MODULO INTERPRETADOR SERA CARREGADO
        LEA  PROG5
        CLS  INILPS
        LDX  =1
        CSV  M:CNEC
        LDA  #58
        ADM  PAGINT

*ENDERECO  PRIM.BLOCO SERA ARMAZENADO NOS MODULOS
        LDA  PAGEND
        STA @PAGSUP
        STA @PAGERE
        STA @PAGINT
```

```
        LDX   =2
        LDA   PTRDIR
        STA @PAGINT,X
        STA @PAGERE,X  *ATUALIZA NA GESTAO PTRDIR
        LDX   =2
        LDA   PTRUSU
        STA @PAGERE,X  *ATUALIZA NA GESTAO PTRUSU
*SISTEMA SERA ATIVADO
*PRESENTE PROGRAMA SERA ABANDONADO
        LEA   CTBOU
        CSV   M:ASGN
        LDA   =4          *AC:=NIVEL DO SUPERVISOR
        CLS   ATIVAR      *SISTEMA SERA ATIVADO
        CSV   M:EXIT
        FIN   INI

ATIVAR LPS   INILDS
INI     STA   TEMP
        LDX   =4
        CSV   M:IT
        LDA   TEMP
        LDX   =6
        CSV   M:IT
        RTS
        FIN   INI
```

```
DEDATA LDS
       RES  2
CBIN   DATA 0,&0005,#BUFIN,6
BUFIN  RES  3
CBOUT  DATA 0,&8007,#BUFOUT,42
BUFOUT TEXT "CONFIRME : DIA=  ,MES=  ,ANO=  , S OU  N ?"
CB     DATA 0,&8007,#BUF,34
BUF    TEXT "INFORME DATA DO DIA FORMATO DDMMAA"
CBRESP DATA 0,&0005,#RESP+1,1
RESP   RES  1
       FIN
PEDATA LPS  DEDATA
INI    LEA  CB
       CSV  M:IO
       CSV  M:WAIT
       LEA  CBIN
       CSV  M:IO
       CSV  M:WAIT
       LDA  BUFIN
       SBL  BUFOUT+15
       SBR  BUFOUT+16
       LDA  BUFIN+2
       STA  BUFOUT+22
       LDA  BUFIN+4
       SBL  BUFOUT+29
       SBR  BUFOUT+30
       LEA  CBOUT
       CSV  M:IO
       CSV  M:WAIT
       LEA  CBRESP
       CSV  M:IO
       CSV  M:WAIT
       LDA  RESP
       CMP  =*E2
       BCF  INI
       DLD  BUFIN
       DST  #DDMMAA
       LDA  BUFIN+4
       STA  #DDMMAA+4
       RTS
       FIN  INI
```

```
DEHORA LDS
       RES  2
BUFIN  TEXT "INFORME HORA FORMATO HHMM "
BUFCON TEXT " CONFIRME : SAO    HORAS E     MINUTOS S OU N ? "
HORA   RES  2
RESP   RES  1
CBINF  DATA 0,&8007,#BUFIN,28
CBINHO DATA 0,&0005,#HORA,4
CBCONF DATA 0,&8007,#BUFCON,48
CBINCO DATA 0,&0005,#RESP+1,1
TEMP   RES  1
CT375  DATA 375
       FIN
PEHORA LPS  DEHORA
INI    LEA  CBINF
       CSV  M:IO
       CSV  M:WAIT
       LEA  CBINHO
       CSV  M:IO
       CSV  M:WAIT
       LDA  HORA
       STA  BUFCON+16
       LDA  HORA+2
       STA  BUFCON+28
       LEA  CBCONF
       CSV  M:IO
       CSV  M:WAIT
       LEA  CBINCO
       CSV  M:IO
       CSV  M:WAIT
       LDA  RESP
       CMP  =1E2       *RESPOSTA=S ?
       BCF  INI        *SE NAO VOLTA PARA INICIO
       LEA  BUFCON+16
       CSV  M:BNDC
       LDA  =0
       XAE
       MUL  =60
       STA  TEMP
       LEA  BUFCON+28
       CSV  M:BNDC
       LDA  =0
       XAE
       ADD  TEMP
       SLLD  =4
       RTS
       FIN  INI
```

```
PEDLDS LDS
       RES  2
TEXTO  TEXT "INFORME NUMERO DE PAGINAS NNN "
CBPAG  DATA 0,&8007,#TEXTO,30
CBNUM  DATA 0,&0005,#NUMERA,3
NUMERA RES 2
       FIN
PEDPAG LPS  PEDLDS
INI    LEA  CBPAG
       CSV  M:IO
       CSV  M:WAIT
       LEA  CBNUM
       CSV  M:IO
       CSV  M:WAIT
       LEA  NUMERA
       CSV  M:DCBN
       XAE
       STA #0          *GUARDA TOTAL DE PAGINAS
       SLLS =8
       RTS
       FIN  INI
       END  LOADER
```

ANEXO V

MANUTENÇÃO DA HORA CORRENTE

```
*****************************************************
*                                                   *
*                                                   *
*    MODULO RESPONSAVEL PELA MANUTENCAO DA HORA     *
*    CORRENTE.  SEMPRE QUE O RELOGIO ACABAR DE      *
*    DECREMENTAR 2,56 SEGUNDOS, NOVA ORDEM DE       *
*MARCHA SERA DADA, O CONTADOR DE TEMPO SERA ATUA-*
* LIZADO E O NIVEL SERA DESATIVADO                  *
*                                                   *
*****************************************************

N25  CDS
 RES 16
 FIN
LDS LDS
TOT1 RES 2
R25 DATA &223
CTE DATA 12800
 FIN
LPS LPS LDS
INI DLD TOT1
 ADD =100
 ACE
 DST TOT1
 LDA CTE
 LDE R25
 WD
 DIT
 LDA TOT1
 LDE =&10
 WD
 LDA TOT1+2
 LDE =&20
 WD
 BRU INI
 FIN INI
 END LPS
```

ANEXO VI

PREPARAÇÃO DAS LINHAS ASSÍNCRONAS

```
*****************************************************
*                                                   *
*                                                   *
*PROGRAMA RESPONSAVEL PELA PREPARACAO DAS LINHAS*
*ASSINCRONAS.                                       *
*                                                   *
*****************************************************
CDS CDS
 RES 32
 FIN
LOCAL LDS
SALVEX DATA 0
KR DATA &8D0D,&8D0D
VIT DATA &5858,&5858
PINDX DATA &44
PADR RES 1
 FIN
PROG LPS LOCAL
INI LDX SALVEX
 LDA @PINDX
 STA PADR
 LDA @PADR
 STA PADR
 DCX =4
 STX SALVEX
 DLD KR
 DST @PADR,X
 ICX =42
 STX SALVEX
 XAX
 CMP =76
 BCF INI
 DIT
 CSV M*EXIT
 FIN INI
 END PROG
```

ANEXO VII

DEMARCADOR DE "DELAYS"

```
*****************************************************
*                                                   *
*                                                   *
* MODULO ASSOCIADO AO NIVEL 24. TODA VEZ QUE O      *
* RELOGIO  ERESPONSAVEL PELA DEMARCACAO DAS FATI-*
* AS DE TEMPO DE U.C.P) PASSAR POR ZERO ESTE MODU*
* LO APOS ATIVAR O SUPERVISOR, SE DESATIVARA.       *
*                                                   *
*****************************************************
NIV24   CDS
        RES   16
        FIN
PROG    LPS   NIV24
INI     LDA   =4
        LDX   =4
        CSV   M:IT
        LDA   =4
        LDX   =6
        CSV   M:IT
        DIT
        BRU   INI
        FIN   INI
        END   PROG
```