UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SOFTWARE PARA PROCESSAMENTO DE TEXTOS

por

ROLF HARM HINRICHS

Tese submetida como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Prof. Roberto Tom Price

Orientador

Porto Alegre, 25 de fevereiro de 1980.

AGRADECIMENTOS:

- À Companhia de Processamento de Dados do Rio Grande do Sul, PROCERGS, pelo incentivo e pela liberação do horário de trabalho para a realização de todos os créditos do curso;

- À Eletrônica Digital S.A. , EDISA, pela liberação de certos horários nos meses finais da elaboração da dissertação, e por ceder o seu equipamento ED-100 para realizar a implementação;

- Ao professor Roberto Tom Price pelas horas de lazer sacrificadas para questionar, revisar, sugerir e orientar o trabalho;

- À banca examinadora, pelas melhoras sugeridas;

- À Ruth, por toda a sua expressiva colaboração;

- À Jussara e ao Rolf pela datilografia, ao Hélio pelos desenhos;

- Aos que ainda não foram mencionados explicitamente e que merecem os meus agradecimentos.

SINOPSE

O presente trabalho descreve um sistema orientado para o processamento de textos. É composto por

- um núcleo, que coordena a execução de várias tarefas e realiza as operações de entrada e saída,
- um conjunto de utilitários, que visa apoiar as tarefas normais de processamento de textos,
- um editor, orientado especificamente para realizar manipulação de textos, permitindo macros dinâmicas, manipulação sobre diversos arquivos, edição em qualquer sentido, e outros,
- um formatador de textos, que automaticamente realiza a marginação, alinhamentos, quebra de página, tabulações e outras funções.

É discutido também a implementação deste projeto em um sistema ED-100, da EDISA, composto por microprocessador, disquete, terminal e impressora.

ABSTRACT

This work describes a word processing oriented system. Its main modules are:

- a nucleus, which controls the execution of several tasks, and does the I/O operation,
- a set of utilities, which help the normal text editing work,
- a text editor, specifically tailored for word processing, which has dynamic macros, editing of several files, backward edition, and other features,
- a text formatter, which does numeration, alignment, pagination, tabulation, and other functions.

There is also a description of the implementation of this project on a ED-100, a microprocessor based system, manufactured by EDISA, with floppy disk, printer, CRT and keyboard.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

1 INTRODUÇÃO

1.1 Histórico

Desde o fim do século passado, a máquina de escrever conquistou uma posição definitiva na sociedade moderna. Onde existiam pessoas especializadas em escrever e transcrever textos, passaram a ser empregadas máquinas com velocidade e legibilidade bem superiores aos manuscritos.

Em meados deste século foram introduzidas as máquinas de escrever elétricas, que trouxeram como benefício uma maior produtividade, menor esforço do operador e qualidade de impressão superior à máquina mecânica. Embora o custo daquela seja consideravelmente superior ao desta, as aplicações comerciais são, hoje em dia, realizadas quase que exclusivamente com máquinas elétricas.

O maior problema das máquinas de escrever é a ausência total de inteligência - a cada operação manual corresponde um movimento mecânico. Isto traz, por exemplo, a inconveniência de, por omitir uma palavra, toda uma página necessitar de nova datilografia. Também em mala direta, onde o conteúdo das cartas é sempre o mesmo, todas devem ser datilografadas para se obter um grau de personalização desejável.

O primeiro passo para uma solução adequada deste problema foi dado no início da década de 1970, com o lançamento de uma máquina de escrever elétrica, dotada de fita magnética, onde podiam ser armazenadas cartas e textos padronizados. A escrita automática podia ser alternada com datilografia manual, possi

bilitando maior qualidade e velocidade de impressão.

À medida que a tecnologia eletrônica e a ciência da computação foram evoluindo, o custo dos equipamentos foi baixando, aumentando sua popularidade. Hoje em dia os recursos de edição e de formatação de textos são tantos e tão poderosos, que talvez só a aplicação destes equipamentos tenha algo em comum com a máquina de escrever original. Estes equipamentos são denominados equipamentos de "processamento de textos" ou "processamento de palavras". Cumpre, entretanto ressaltar que o segundo termo é ambíguo. Por "processamento de palavras" se entende qualquer equipamento que manipule palavras, sejam elas faladas, escritas ou micrografadas. Um ditafone, uma fotocopiadora e um processador de textos são, portanto, considerados equipamentos de processamento de palavras.

1.2 Configurações de equipamentos de processamento de textos

1.2.1 Processadores

Embora o panorama do mercado de equipamentos de processamento de textos evolua a um ritmo muito intenso, é possível caracterizar três grandes grupos:

a)- Processador não dedicado

Neste grupo são enquadrados todos os equipamentos que, além do processamento normal, realizam tarefas de processamento de textos. Costumeiramente trata-se de um equipamento de grande porte, que não foi adquirido com a finalidade específica de processamento de textos, mas que, através de terminais remotos, oferece a seus usuários facilidades de arquivamento e recuperação, aliada à sua capacidade de processamento.

As principais características deste sistema são:

- Crescimento

  O crescimento é fácil de realizar: basta colocar mais um terminal na rede.

- Versatilidade

  O terminal remoto não só pode ser utilizado para o processamento de textos, mas também para outras aplicações.

- Armazenamento

  A capacidade de armazenamento é grande.

- Espaço físico

  A estação de trabalho consiste unicamente de um terminal, eventualmente acrescido de um modem e uma impressora.

- Custo

  O custo por estação de trabalho tende a diminuir à medida que o número de estações cresce.

b)- Processador dedicado, estações múltiplas

Neste grupo enquadram-se todos os equipamentos que possuem uma unidade central de processamento dedicada exclusivamente ao processamento de textos, com uma série de estações de trabalho, nas quais os operadores realizam suas tarefas. Normalmente trata-se de um minicomputador, programado para executar exclusivamente este tipo de serviço.

As principais características deste sistema são:

- Crescimento

  O número de terminais é limitado (8 a 16 terminais).

- Aquisição do equipamento

Como a aquisição da UCP representa um custo elevado, deve ser adquirido um número de terminais suficientemente grande para tornar o custo por estação vantajoso em comparação com um equipamento de estação única. Com isto, o equipamento tende a apresentar uma ociosidade inicial grande.

- Recursos

Os recursos disponíveis são melhores - impressoras mais rápidas, discos rígidos, bancos de dados - do que nos equipamentos isolados. Por outro lado são compartilhados no tempo e nem sempre estão disponíveis a um usuário no momento em que este deles necessita.

c)- Processador dedicado, estação única

Neste grupo se enquadram aqueles equipamentos constituidos de unidade central de processamento, dispositivos para armazenamento, terminal interativo e impressora. Normalmente este tipo é o resultado de uma programação específica de um microcomputador, e é fornecido como "turn-key".

As principais características deste sistema são:

- Disponibilidade

O equipamento, pelo fato de não necessitar de manu

tenção preventiva, tal como os computadores de grande porte, estará mais tempo disponível ao operador. Em um grupo, ao acontecer alguma pane a um equipamento, apenas uma estação estará parada. O grupo torna-se, portanto, mais imune a falhas.

- Segurança

Por ser uma estação autônoma, o equipamento pode ser trancado em uma sala. Pelo fato de não existirem terminais remotos, nenhuma medida de segurança de transmissão de informações precisa ser tomada.

- Facilidade de instalação

Pode ser instalado em qualquer lugar, sem levar em conta fatores como modens e linha telefônica. A instalação normalmente consiste em abrir a embalagem, dispor o equipamento da forma física mais conveniente e ligá-lo na tomada elétrica.

1.2.2 Periféricos

Na parte de periféricos, é possível caracterizar três grandes grupos:

a)- Entrada de textos e comandos

O terminal interativo é o principal meio de entrada. Neste, o operador digita comandos e textos. No mesmo equipamento recebe as respostas do sistema.

Um tipo de terminal largamente difundido é um teclado com impressora, similar a uma teletipo. Diferença básica é, entretanto, que a qualidade de impressão equivale à de uma máquina de escrever elétrica.

Atualmente o terminal mais difundido é o terminal de vídeo. Em alguns equipamentos é empregado como uma teletipo: o usuário escreve os seus comandos e o sistema responde com o resultado das operações. Entretanto, uma forma de operação mais adequada é a do operador trabalhando diretamente sobre o texto exibido na tela. Este tipo de interação permite a visualização do estado atual do texto, sendo de uma facilidade de operação muito maior.

Além dos citados e dos meios de entrada tradicionais (leitora de cartões, fita de papel, etc..), outros estão se tornando mais populares:

- as leitoras OCR (OPTICAL CHARACTER READER), que já possuem um lugar de destaque no processamento de dados, processando principalmente documentos com

retorno (contas de água, luz, telefone). A expectativa é de que sua aplicação no processamento de textos se amplie nos próximos anos. Aplicações típicas são arquivamento de correspondência, artigos de jornal e livros.

- os equipamentos VIP (VOICE INFORMATION PROCESSING), tanto na parte de entrada, onde são utilizados na transformação de ditados em trechos datilografados sem a intervenção de um operador, como na parte de saída, onde a voz humana é sintetizada para as mais diversas finalidades (por exemplo uma máquina de escrever capaz de "falar" um certo texto, orientada para o uso de pessoas cegas).

b)- Meios de armazenamento

Um dos primeiros meios magnéticos a ser empregado foi a fita em cassete. O equipamento era orientado mais para aplicações de mala direta e circulares personalizadas. A edição do texto implicava em realizar uma cópia de todo o cassete, alterando as partes necessárias. Outra alternativa consistia em trazer todo o texto para a memória e alterá-lo, o que implicava em memórias razoavelmente grandes; após o término da edição o texto voltava a ser armazenado.

As grandes vantagens dos meios de acesso direto, popularizados pelo disquete (disco flexível), trouxeram uma nova série de recursos ao processamento de textos. Com o custo do meio baixo e um desempenho e capacidade de armazenamento bons, o dis-

quete passou a existir em praticamente todo o equipamento de processamento de textos atual.

c)- Saída

O principal meio de saída dos equipamentos de processamento de textos é o papel. As impressoras podem enpregar vários tipos de tecnologia - correia, tambor, bolinha, "daisy-wheel "- mas têm como principais características:

- letras maiúsculas e minúsculas
- caracteres de acentuação
- subscritos e superscritos (a_i , a^i)
- espacejamento vertical e horizontal variáveis e programáveis
- excelente qualidade de impressão.

1.3 Recursos oferecidos por um equipamento de processamento de textos

O software dos equipamentos de processamento de textos pode ser caracterizado como tendo três partes principais: um módulo básico, um editor de textos e um formatador de textos. Os equipamentos que também possuem capacidade de processamento de dados incluem os módulos: sistema operacional, utilitários, compiladores e programas do usuário.

O módulo básico consiste de todas as funções de apoio à operação do equipamento: manutenção de diretórios, criação e exclusão de arquivos, cópias e outras tarefas auxiliares. Este conjunto permite a operação básica do equipamento, tarefas de preparação do serviço de processamento de textos, cópia para fins de segurança e outros.

O editor serve para criar e alterar textos. Cumpre, entretanto, ressaltar que o termo "editor de textos", amplamente difundido na comunidade brasileira de processamento de dados, se refere na realidade a um "editor de registros" - consiste em localizar e alterar registros. Em um "editor de textos" o texto é considerado como uma cadeia virtualmente infinita de caracteres. As entidades são capítulos, páginas, parágrafos, sentenças e palavras. Embora seja possível usar um editor de registros para manipular textos, é preferível utilizar um editor de textos real, considerando as facilidades que oferece.

O formatador de textos é o elemento incumbido de dispor o texto sobre a folha de papel. Realiza a paginação (podendo numerar as páginas automaticamente), separa as linhas e os pa

rágrafos, tabula, além de muitas outras funções. Como comandos para o formatador são utilizadas palavras devidamente identificadas, embutidas no próprio texto. Ao escrever um texto, portanto, só são inseridos aqueles comandos que separam uma página ou um parágrafo, sem preocupação com o leiaute físico na página. O formatador, quando executado, controla toda a impressão. Após formatar um texto, qualquer alteração sobre o mesmo pode ser executada. A nova "datilografia", contendo as alterações, pode ser realizada em pouco tempo.

1.4 Aplicações

Procurando exemplificar as potencialidades de um equipamento de processamento de textos, são citadas três aplicações características:

a)- Correspondência comercial

Em organizações, todas (ou quase todas) cartas devem receber uma resposta. Existe um certo número de respostas padronizadas, em que alguns dados podem ser preenchidos no instante em que a carta é escrita. Após examinar a carta consulta, esta é entregue à central de correspondência, juntamente com as informações do padrão de resposta e dos dados a serem preenchidos.

b)- Confecção de propostas comerciais

Uma empresa de fornecimento de equipamentos deve apresentar propostas para os clientes. Em função das opções necessárias, esta será diferente para cada um dos clientes. Todas as propostas devem ser personalizadas, isto é, o cliente não deve perceber que está recebendo algo genérico, apenas adaptado para sua situação.

O processamento de textos oferece uma excelente solução para o caso. Ao confeccionar a proposta, o representante de vendas fornece a configuração aconselhável ao operador, que as transmite ao equipamento. Os valores numéricos necessários podem ser calculados pela própria máquina, a numeração de páginas e o

espacejamento são automáticos.

c)- Manuais técnicos

A atualização da documentação de um produto em evolução tecnológica constitui, normalmente, um grande problema. Exemplos de produtos sujeitos a muita modificação durante o seu desenvolvimento são software (básico, de apoio e de aplicação) e hardware (alteração nas configurações, melhorias de desempenho).

Quando os manuais são confeccionados através de um equipamento de processamento de textos, as retificações são realizadas, à medida que o produto recebe os retoques finais. A inserção de uma frase ou exclusão de um parágrafo não implicam em redatilografar uma série de páginas, mas apenas em algumas interações com um terminal.

Com os manuais editados e o produto lançado no mercado, existe a necessidade de atualizar constantemente a documentação, para mantê-la sempre em dia. Cada alteração é realizada sobre o texto original, armazenado num meio magnético. Quando surgir a necessidade de uma nova edição, basta imprimir o texto atualizado. Nesta impressão, todas as alterações realizadas sobre a edição anterior podem ser assinaladas com uma barra na margem, facilitando ao leitor da nova versão a visualização do que foi modificado.

1.5 Utilização

O maior mercado do equipamento de processamento de textos atualmente é o da substituição da máquina de escrever. Entretanto, devido a seu custo, que é cerca de dez vezes superior, a sua aquisição deve ser justificada em termos de um aumento de produtividade, rapidez de atendimento e qualidade de impressão.

Empresas de um certo porte costumam ter, como apoio às tarefas administrativas, um centro de datilografia, onde todos os documentos são datilografados como minutas, em papel timbrado da empresa, etc.. Todos os serviços maiores são executados por esta central, sendo que tarefas pequenas podem ser realizadas por secretárias e recepcionistas.

O centro de datilografia pode ser visto como um estágio embrionário de um centro de processamento de textos. O baixo nível de automação pode ser elevado com a introdução de um equipamento de processamento de textos, que reduzirá o número de tarefas a serem realizadas manualmente.

Este centro pode ser comparado ao centro de processamento de dados. O primeiro apresenta serviços como transcrição de manuscritos, acertos diversos, composição de textos, minutas, etc. O segundo emite relatórios da folha de pagamento, faturamento, contabilidade e custos.

O elemento chave do centro de processamento de textos é o supervisor (denominado "Word Processing Manager"), que é incumbido da distribuição de tarefas aos operadores e equipamentos, além de ser o responsável pelo arquivamento e pelo fluxo das informações dentro da empresa.

1.6 Tendências

Na área de equipamentos de processamento de dados já pôde ser observado o seguinte fenômeno: à medida que o número de usuários aumenta, o custo do equipamento diminui, mesmo que apresente desempenho e capacidade de armazenamento superiores.

Na área de processamento de textos vem ocorrendo um fenômeno similar: à medida que o número de usuários cresce, o custo dos equipamentos diminui, trazendo um maior número de clientes para os fabricantes. A tendência do equipamento de processamento de textos é de se tornar muito atrativo em termos de custos.

Na parte de processadores, com a diminuição de custos aliada ao aumento de capacidade, as facilidades para o usuário tendem a crescer cada vez mais.

Na parte de periféricos existe grande expectativa na evolução (isto é, na melhora de qualidade, confiabilidade e velocidade, junto com uma diminuição de custo e de tamanho), dos seguintes equipamentos:

- micrográficos, que permitam a microfilmagem direta das informações, sem etapas intermediárias. O arquivamento seria amplamente facilitado.
- impressoras, mais rápidas e com melhor qualidade de impressão que as atuais.
- OCR, com custo reduzido, podendo inclusive ler microfichas.
- VIP, capazes de "entender" ditados e transformá-los em documentos.
- fotocomposição, podendo compor diretamente matri-

zes para o processo "off-set".

- terminais gráficos, com copiadora, para a produção e atualização de desenhos e gráficos.
- redes de comunicação, para a transmissão de correspondência.

Supondo, quanto ao hardware, igualdade de condições para todos os fabricantes, a grande batalha do mercado se desenvolverá em termos de custo e facilidades para o usuário.

Na parte de software, algumas aplicações estarão certamente presentes:

- correspondência eletrônica, interligando vários equipamentos de processamento de textos. A correspondência é transmitida por linhas telefônicas ou linhas especiais, sob forma digital. É armazenada do outro lado em um meio magnético. Quando for solicitada, a correspondência é exibida carta por carta em uma tela, podendo o usuário desprezá-la, respondê-la ou arquivá-la. Com isto estará sendo criado o "escritório sem papel".
- terminais remotos, que, pela mesma rede de comunicação da correspondência, poderiam solicitar informações e pedir serviços a computadores.
- capacidade de processamento de dados, podendo o equipamento, além de processamento de textos, realizar todas as tarefas administrativas de uma empresa.

2 DEFINIÇÃO

2.1 Equipamento Portador

Como equipamento portador para o software de processamento de textos foi selecionado o ED-100, da EDISA, ainda em fase de prototipia. Apresenta todos os elementos necessários para a realização de processamento de textos. UCP e memória, de uma a duas unidades de disquete, um terminal interativo e impressora.

A UCP possue um processador Motorola 6800, de 8 bits. Existem, ainda, agregadas:

- lógica de interrupção, que permite leitura do conjunto de interrupções que estão atuando naquele instante, e uma máscara que permite a habilitação /desabilitação individual de cada uma das interrupções.
- lógica de temporização, que permite disparar um temporizador, que após a decorrência do intervalo de tempo programado, gera uma interrupção.

A memória do sistema é composta por:

- 3Kbytes de REPROM, que armazenam, entre outras, a rotina de carga do sistema para a memória RAM. Esta capacidade pode ser expandida até um máximo de 16K.
- 4 Kbytes de RAM, que armazenam o sistema propriamente dito. Esta capacidade pode ser ampliada para 32Kbytes.

As interfaces foram colocadas em endereços de memó -

ria, podendo, ser acessadas com as instruções normais de referência à memória.

O disquete, com formato compatível ao IBM 3740, permite a gravação de 1898 registros físicos, em 73 trilhas de 26 setores cada uma. O acesso pode ser realizado de maneira randômica, como em disco magnético.

O terminal interativo é composto por duas partes:

- um vídeo, com memória local, que permite a escrita em qualquer posição da tela.
- Um teclado específico para entrada de dados, gerando caracteres em EBCDIC. Existem 16 teclas de função, cujo código hexadecimal é inferior a 20.

A impressora é em ASCII, podendo representar 96 caracteres diferentes. Possui um "buffer" local de 1Kbyte, e é ligada por uma interface série tipo RS-232 ao processador. A transmissão para a impressora é realizada a uma taxa de 9600 bits por segundo.

2.2 O Sistema visto pelo usuário

Após ligar a máquina e inserir o disquete contendo o Sistema de Processamento de Textos (SPT), a carga do programa é iniciada automaticamente.

O operador é consultado sobre qual a atividade que ele deseja exercer:

- uma execução de utilitário
- uma edição de textos
- uma formatação.

A seguir será fornecida uma visão suscinta de cada uma destas atividades.

2.2.1 Utilitários

O usuário pode executar uma série de tarefas utilitárias, que devem auxiliá-lo nas suas atividades relacionadas com o processamento de textos. A seguir são enumeradas estas tarefas.

a)- Inicializar meio de armazenamento

Esta função inicializa um disquete para o processamento de textos. Como parâmetro é fornecido o nome do volume de disquete. Após esta função, todo o disquete está disponível para armazenamento de textos.

Fisicamente a inicialização consiste apenas em alterar as informações do disquete de tal forma, que para o SPT o vo

lume está vazio.

b)- Renomear arquivo

Esta função troca o nome de um arquivo. Como parâmetros são fornecidos o nome atual e o novo nome. Após esta função, existe apenas o arquivo novo.

Fisicamente, existe apenas uma troca no nome do arquivo em um rótulo.

c)- Incluir arquivo

Esta função cria um arquivo em um volume. Como parâmetros são fornecidos o nome do arquivo e o nome do volume. Após esta função, estará criado um novo arquivo para o processamento de textos.

Fisicamente, a criação consiste na alocação de uma trilha para este arquivo e a alocação de um rótulo. A indisponibilidade de qualquer um destes recursos invalida a operação.

d)- Excluir arquivo

Esta função excluium arquivo de um volume. Como parâmetro é fornecido o nome do arquivo. Após a execução desta função, o arquivo não existe mais.

Fisicamente, o nome do arquivo é trocado por brancos, e as trilhas que pertenciam a este arquivo são liberadas para serem utilizadas por outros arquivos.

e)- Listar

Esta função lista um arquivo ou um volume, dependendo do que é fornecido como parâmetro. Em se tratando de um arquivo, o seu conteúdo é listado. Em se tratando de volume, são listados os nomes de todos os arquivos que nele existem.

2.2.2 Editor

O editor de textos é a parte mais importante em um sistema de processamento de textos, pois é através deste módulo que ocorre a maior interação entre homem e máquina.

As características mais importantes do editor em questão são:

a)- Edição em qualquer sentido

O usuário, ao realizar uma pesquisa, pode especificar sentido positivo (do início ao fim do arquivo) ou negativo (do fim ao início do arquivo).Todas as alterações já realizadas permanecem. Quando é especificado sentido negativo, a pesquisa é realmente realizada no sentido negativo (isto é, não se trata de um retorno ao início, seguido de uma pesquisa no sentido positivo).

b)- Existência de macros

O usuário pode especificar qualquer seqüência de caracteres digitados pelo teclado como sendo uma macro. Ao chamar a macro, o editor automaticamente "digita" todas as teclas necessárias. Estas macros podem ser alteradas dinâmicamente pelo próprio conjunto de instruções do editor.

c)- Edição sobre vários arquivos

A edição do texto não se restringe a um arquivo de entrada; em qualquer ponto do texto podem ser incluídos trechos de outros arquivos.

d)- Exibição de texto atualizado

A tela sempre mostra o estado atual na qual o texto se encontra. Desta forma, cada alteração realizada é imediatamente exibida, sendo conferida pelo operador. As teclas de função foram utilizadas para facilitar o manuseio rápido dos textos, permitindo diversos tipos de deslocamento de cursor, além de inserção e exclusão de caracteres.

e)- Existência de marcas

O texto pode ser marcado em lugares que forem julgados necessários. Isto facilita o posicionamento em arquivos (POSICIONE o arquivo na marca X), inclusão (INCLUA o arquivo X des-

de a marca Y até a marca Z), exclusão (EXCLUA da marca Y até a marca Y) e movimentação (inclusão e posterior exclusão) de trechos de arquivo.

Existem, basicamente, dois modos de operação no editor: o modo comando e o modo dados.

No primeiro modo são digitados os comandos para o editor. Após digitar a tecla de fim dos comandos, o editor analisa e executa comando por comando. Após a execução com sucesso de todos os comandos, passa ao modo de dados.

No modo de dados, o texto é exibido em seu estado atual. Um cursor físico é colocado na tela, sublinhando um certo caracter. Um cursor lógico aponta internamente para a posição correspondente.

No estado normal, digitando qualquer caracter, este aparece no lugar apontado pelo cursor, apagando aquele que lá se encontrava. Certas teclas de função permitem o rápido deslocamento do cursor, facilitando o posicionamento em novos elementos do texto. Uma determinada tecla de função é utilizada como inversor do estado de inclusão (isto é, quando acionada, troca o estado de inclusão para normal e vice-versa), sendo que no estado de inclusão todos os caracteres digitados são inseridos antes do caracter apontado pelo cursor. Outra tecla de função é utilizada como comando de exclusão de caracter, retirando do texto o caracter apontado pelo cursor. Ainda outra tecla de função permite a passagem do modo de comandos ao modo de dados e vice-versa.

2.2.3 Formatador

O formatador é o elemento que dispõe o texto sobre a folha de papel, de acordo com as instruções que estão embutidas no texto e com certos parâmetros referentes à diagramação da página.

As características mais importantes do formatador são:

a)- Alinhamento

O alinhamento é realizado de forma automática. Títulos podem ser alinhados à direita ou à esquerda, além de poderem ser centrados no meio da linha. As linhas normais são alinhadas à direita e à esquerda.

b)- Paginação

A paginação pode ser realizada de forma automática, ou por comando. A impressão de rodapés e cabeçalhos é automática, podendo ter um lado par e um lado ímpar (como é o caso em livros, onde o lado esquerdo nem sempre possui os mesmos cabeçalhos e rodapés que o direito).

c)- Inclusão de vários arquivos

À medida que o texto for sendo impresso, trechos de outros arquivos podem ser incluidos, recursivamente (isto é, no

meio de um texto pode aparecer um comando INCLUA arquivo X de Y até Z. No meio desta inclusão pode aparecer novo INCLUA arquivo X'de Y'até Z').

d)- Macros

Dentro do texto podem ser criadas, alteradas e excluídas macros. Estas podem ter sido criadas tanto na fase de edição como na própria formatação.

2.3 Comandos

Os comandos que existem à disposição do usuário são divididos em dois grandes grupos:

- Os comandos imediatos, que são aqueles executados pelo editor.
- Os comandos deferidos, que são aqueles executados pelo formatador.

Os comandos imediatos são escritos no modo de comandos do editor, ou então correspondem a uma tecla de função no teclado.

Os comandos deferidos são aqueles que estão embutidos no texto, e que só serão executados no instante em que o formatador "descobrí-los". Todos os comandos digitados no modo de dados do editor, são, portanto, comandos deferidos.

Como existem alguns comandos que podem ser executados tanto de forma imediata como deferida, passou-se a denominá-los de comandos gerais.

A seguir é fornecida uma descrição sintética dos comandos. No apêndice 1 pode ser encontrada uma descrição formal.

As chaves ($\{\}$) representam a escolha de um dos elementos contidos em seu interior. As palavras maiúsculas representam uma palavra conhecida ao sistema. As palavras minúsculas entre $<\ >$ representam um elemento sintático, cujo detalhamento pode ser estudado no apêndice 1.

2.3.1 Comandos gerais

Estes comandos são compostos por três tipos:

- controle, que podem escolher e repetir comandos ou dados por um certo número de vezes.
- inclusão, que permite a inclusão de um trecho de outro arquivo no texto em processamento.
- manipulação de macros, que permitem a criação, alteração e exclusão de macros de sistema.

A seguir, um resumo da sintaxe e das ações de cada um dos comandos:

TIPO	SINTAXE	EFEITO
CONTROLE	REPITA <exp a><seqüência>	A <seqüência> é repetida <exp a> vezes. Se <exp a> é inferior a 1, a <seqüência> não é executada. <exp a> é avaliada antes da execução
	SE<nome> <seqüência>	A <seqüência> é repetida até encontrar o fim de arquivo
	[illegible] <exp b> <seqüência>	Enquanto a <exp b> for verdadeira, a <seqüência> é repetida
	SE <exp b> <seqüência>	Se a <exp b> for verdadeira, a <seqüência> é executada
INSUMO	[illegible] <arquivo><intervalo>	No ponto em que se encontra o cursor é incluido o <intervalo> especificado do <arquivo>
MACROS	[illegible] <nome> <seqüência>	Cria uma macro <nome> que corresponde ao conjunto de caracteres de <seqüência>
	MODIFICA <nome> <seqüência>	Altera a macro <nome> para o conjunto de caracteres de <seqüência>
	ATRIBUA <nome> <exp a>	Atribui à macro <nome> o resultado da <exp a>
	[illegible] <nome>	Exclui a macro <nome>

Tabela 1 - Comandos gerais

2.3.2 Comandos Imediatos

Os comandos imediatos são compostos por três grupos:

- comandos de pesquisa, que posicionam o cursor em uma certo conteúdo desejado.
- comandos de manipulação de texto, que permitem a inclusão/alteração/exclusão de cadeias de caracteres.
- comandos de processamento de macros, que podem estabelecer e deslocar marcas, bem como realizar o posicionamento sobre macros existentes no texto.

Segue um resumo da sintaxe e das ações provocadas por cada um dos comandos:

TIPO	SINTAXE	EFEITO
TEXTO	PRÓXIMO / ÚLTIMO SEQUÊNCIA <seqüência> / PERÍODO / PARÁGRAFO / PÁGINA / CAPÍTULO	O texto é posicionado sobre a próxima ou última ocorrência desejada.
	INSERE <seqüência>	A <seqüência> é inserida no ponto apontado pelo cursor
	ALTERE <seqüência>	Tantos caracteres quantos forem a extensão da <seqüência> são substituídos no texto apontado pela <seqüência>.
	EXCLUA <seqüência>	Tantos caracteres quantos forem a extensão da <seqüência> são retirados do texto apontado pelo cursor
	TROQUE <seqüência 1> <seqüência 2>	Todo arquivo é percorrido substituindo em todos os pontos, onde ocorrer, <seqüência 1> por <seqüência 2>.
	ELIMINE <intervalo>	Elimine o <intervalo> especificado do texto
MARCAS	MARQUE <nome>	A posição indicada pelo cursor recebe o <nome> (criação de uma marca)
	REMARQUE <nome>	Uma certa posição é "esquecida", sendo que o <nome> passa a referenciar a nova posição apontada pelo cursor
	POSICIONE <nome>	O cursor é posicionado na marca <nome>.

Tabela 2 - Comandos imediatos

2.3.3 Comandos deferidos

Estes comandos podem ser agrupados em dois tipos:

- singulares, que, por si só, causam um certo efeito
- delimitadores, que atribuem uma certa propriedade a um trecho do texto.

Segue um resumo da sintaxe e das ações provocadas por cada um dos comandos:

TIPO	SINTAXE	EFEITO
SINGULARES	– NOVA { LINHA / PARÁGRAFO / PÁGINA }	Este comando faz o formatador avançar para um nova linha, iniciar um novo parágrafo ou uma nova página.
	TABULA { DIREITA / CENTRO / ESQUERDA } < seqüência >	Alinha o texto, compreendido entre o último comando TABULA e o atual, entre as duas marcas de tabulação existentes. O alinhamento pode ser à esquerda, à direita ou no centro. O espaço restante é preenchido com a < seqüência >.
DELIMITADORES	{ INÍCIO / FIM } { ALINHAMENTO { DIREITA / CENTRO / ESQUERDA } < seqüência > / MAIÚSCULOS / NÃO QUEBRA / CAPÍTULO / NÃO FORMATA }	Este comando atribue uma certa propriedade a um trecho do texto.

Tabela 3 - Comandos deferidos

3 Projeto

3.1 Estrutura do sistema

Nesta parte é descrita a estrutura do sistema sob dois aspectos importantes: o funcional, ou seja, que função cada parte exerce, e o operacional, ou seja, como funciona o conjunto. Um exemplo permite a visualização dos conceitos propostos.

3.1.1 Estrutura funcional

A figura 1 mostra a estrutura funcional do sistema.

FIGURA 1 - ESTRUTURA FUNCIONAL DO SISTEMA

As MACROROTINAS tornam o conjunto de instruções do processador mais "poderoso". Ao invés de o programador ter que programar certas rotinas, este conjunto permite que com uma chamada de subrotina tarefas mais complexas possam ser realizadas. Como exemplo, pode ser citada a rotina de indexação que, dados uma base, o tamanho do elemento e o número do elemento desejado, aponta para o endereço de memória do elemento solicitado.

O NÚCLEO consiste das rotinas necessárias à implemen

tação de um sistema operacional. É composto por rotinas que permitem a coordenação de diversos processos, ativação e finalização de tarefas de E/S e outras.

As rotinas de E/S realizam todo tipo de operação com os periféricos. Iniciam a operação, habilitando interrupções e acertando endereços de atendimento da interrupção, coordenam a operação em si, monitorando o funcionamento do periférico, e concluem a operação, desativando o periférico e avisando que houve conclusão e quais os resultados obtidos.

As rotinas de MANIPULAÇÃO DE TEXTOS realizam todas as tarefas relacionadas com esta atividade. A pesquisa de uma certa seqüência de caracteres, a inserção e exclusão de caracteres, bem como a manipulação das estruturas criadas para trabalhar sobre os textos são elementos característicos deste grupo de rotinas.

Os UTILITÁRIOS são um conjunto de rotinas destinadas a realizar atividades relacionadas com o processamento de textos. Inicializar um disco, listar ou exibir o diretório de um disco e excluir arquivos são algumas das tarefas típicas.

O EDITOR de textos permite a criação e posterior atualização de textos. Além dos comandos de pesquisa, existe uma série de recursos que permite manipular textos, incluir e excluir trechos de arquivos, e outras operações.

O FORMATADOR imprime os textos, após a sua edição. Realiza a quebra de páginas, com numeração, quebra de linhas e de parágrafos, alinhamento de textos e marginação.

Na estrutura apresentada, as MACROROTINAS, o NÚCLEO e as tarefas de E/S formam um conjunto de atividades que poderiam servir como base não só para um equipamento de processamento de

textos, mas também para sistemas de processamento de dados. O EDITOR, o FORMATADOR, os UTILITÁRIOS selecionados e as rotinas de MANIPULAÇÃO DE TEXTOS, por sua vez, formam um conjunto de atividades orientadas, quase que exclusivamente, para o processamento de textos.

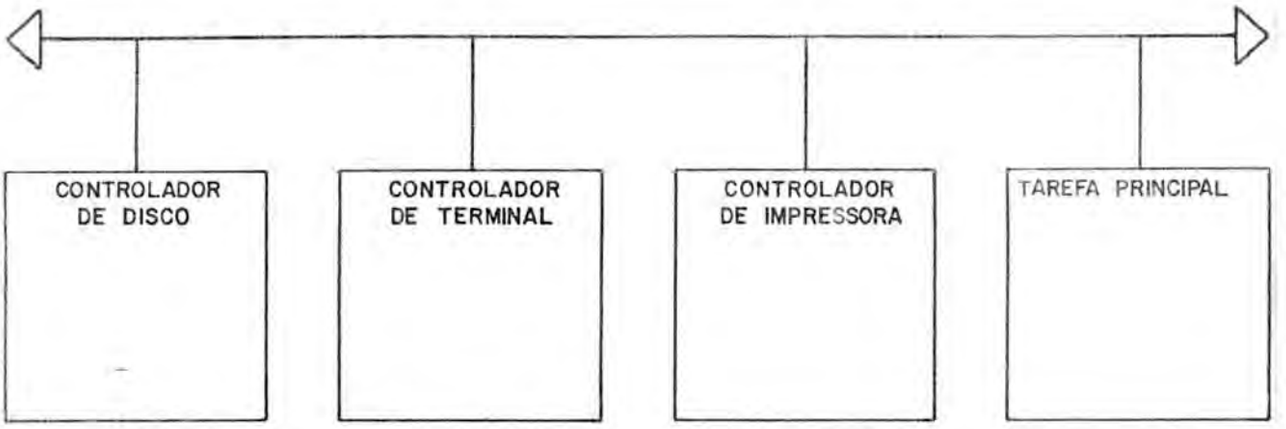


FIGURA 2 - ESTRUTURA DE VARIAS TAREFAS

A figura 2 mostra como o sistema foi estruturado em diversas tarefas. Existe, à direita, a TAREFA PRINCIPAL, que interage com o operador, podendo ser tanto um UTILITÁRIO, o EDITOR ou o FORMATADOR. Os outros módulos são controladores de periféricos. Cada um controla um periférico específico. Estes controladores são ativados pela TAREFA PRINCIPAL. Após concluir a operação de E/S solicitada, DESATIVAm-se a si próprios.

As setas sugerem a expansibilidade do sistema, permitindo maior número de tarefas em execução alternada. Assim como podem ser colocados controladores de outros periféricos, podem ser adicionadas novas tarefas executando alternadamente. Em equipamentos de processamento de textos isto traz pelo menos uma grande vantagem: à medida que um texto é impresso, o próximo já pode ser editado. Este paralelismo de atividades, além de trazer

um enorme aumento de produtividade ao equipamento, não cansa o operador com esperas pela conclusão de certas tarefas.

A comunicação entre as rotinas de atendimento e os controladores é realizada através de variáveis globais, sendo inicializadas e alteradas por um, testadas pelo outro.

Durante a execução das diversas tarefas, os periféricos vão exigindo atenção da UCP, assincronamente. Cada periférico pode interromper o atendimento de outro periférico, mas não a si mesmo. Assim, por exemplo, o teclado pode interromper a rotina de atendimento da impressora, mas jamais quando o próprio teclado está sendo atendido (o que causaria o atendimento do segundo caracter digitado antes do processamento do primeiro).

Uma outra alternativa seria de fazer com que os próprios controladores realizassem a E/S, sem o uso de interrupções. O sério agravante de que todas as rotinas de TAREFA PRINCIPAL teriam que prever um tempo máximo de execução, ou então a necessidade de um temporizador para monitorar as diversas atividades, fez com que a primeira alternativa fosse adotada.

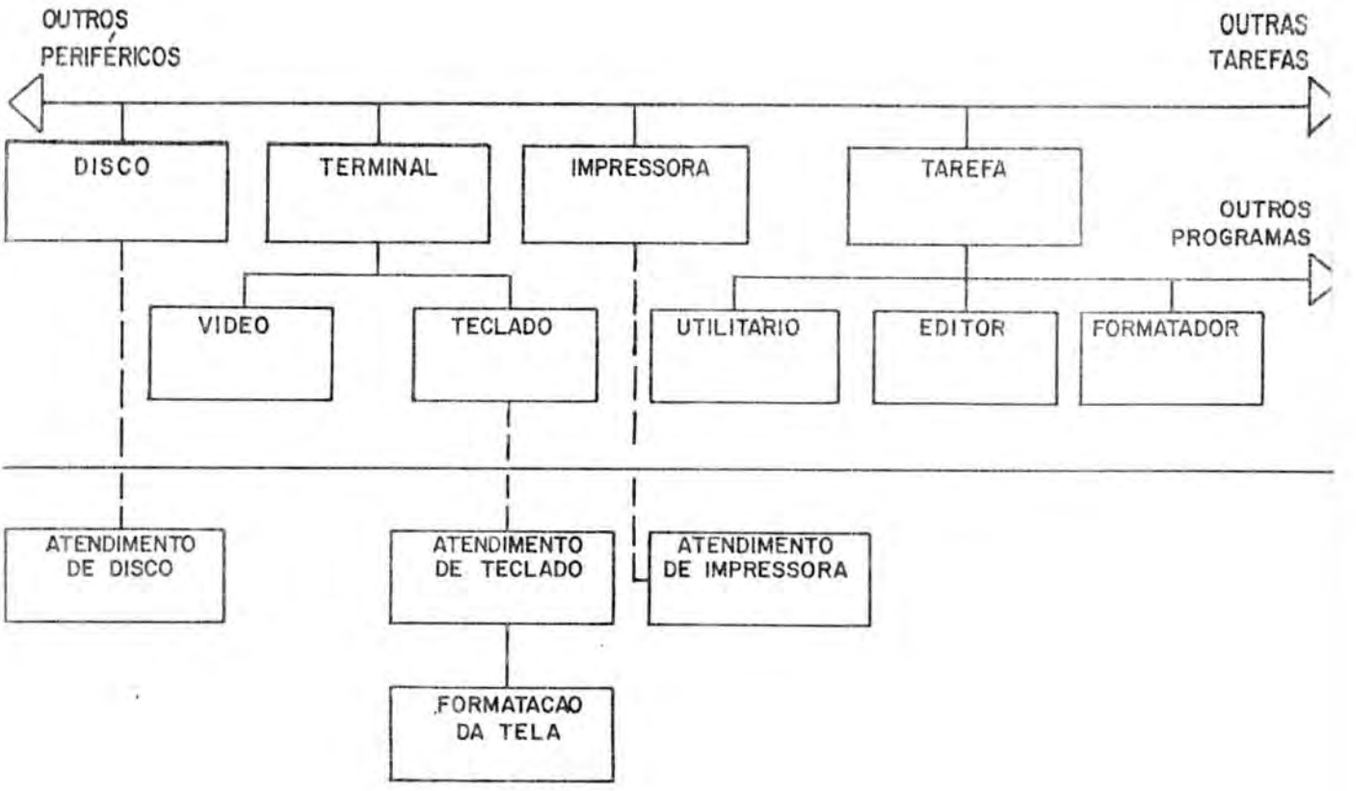


FIGURA 3 - ESTRUTURA GERAL DO SISTEMA

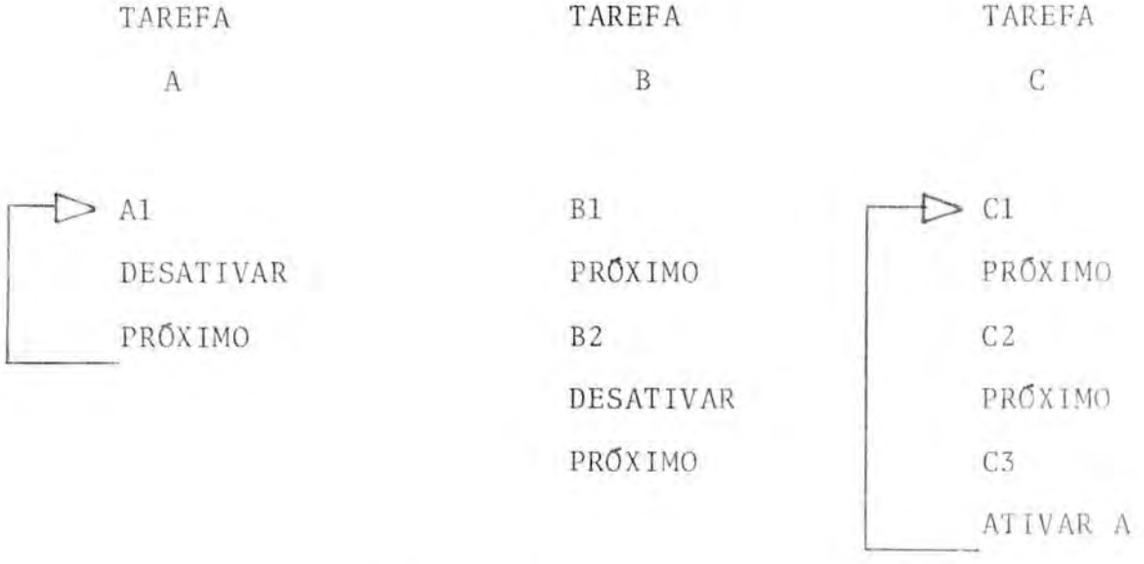
A figura 3 mostra, na sua parte superior, a parte síncrona, ou seja, as tarefas atuadas pelo selecionador da próxima tarefa ativa. Na parte inferior está a parte assíncrona, ou seja, aquelas tarefas atuadas por interrupções dos periféricos.

3.1.2 Estrutura operacional

Para implementar o sistema foi escolhida uma estrutura em multi-tarefas, isto é, o sistema é composto por uma série de tarefas executadas (logicamente) ao mesmo tempo. Este tipo de de estrutura facilita a definição e programação das diversas tarefas, além de permitir a expansão futura do sistema, incorporando novos módulos.

As tarefas podem estar em um estado ativo ou inativo. A passagem de um estado para outro é realizado pelas primitivas ATIVAR e DESATIVAR. A primitiva PRÓXIMO seleciona a próxima tarefa ativa.

Considere-se, como exemplo, três tarefas - A,B e C - que realizassem as seguintes atividades:



Supondo A, B e C inicialmente ativas, a tabela mostra qual a tarefa selecionada para execução em um determinado intervalo de tempo variável, qual a atividade exercida neste intervalo e qual o estado das tarefas após a execução (I= INATIVO, A= ATIVO), visto pelo selecionador da tarefa a executar (primi-

tiva PRÓXIMO).

INTERVALO	TAREFA SELECIONADA	ATIVIDADE EXECUTADA	ESTADO APÓS EXECUÇÃO		
			A	B	C
1	A	A1	I	A	A
2	B	B1	I	A	A
3	C	C1	I	A	A
4	B	B2	I	I	A
5	C	C2	I	I	A
6	C	C3	A	I	A
7	C	C1	A	I	A
8	A	A1	I	I	A
9	C	C2	I	I	A
10	C	C3	A	I	A
11	C	C1	A	I	A
12	A	A1	I	I	A

3.1.3 Exemplo

Para exemplificar o funcionamento das estruturas acima, será exposto uma operação de carga de uma rotina em que, ao mesmo tempo, alguns parâmetros são digitados pelo teclado. A operação de carga consta de uma leitura de 4 setores, com transferência para uma certa área de memória. A solicitação de parâmetros consiste na exibição de um texto na tela, seguida da leitura dos caracteres digitados.

O programa teria as seguintes intruções:

SOLICITE A CARGA (1º PEDIDO)
SOLICITE A EXIBIÇÃO DA TELA (2º PEDIDO)
SOLICITE A LEITURA DE CARACTERES DO TECLADO (3º PEDIDO)
AGUARDE O ATENDIMENTO DO 3º PEDIDO
SOLICITE A EXIBIÇÃO DA TELA (4º PEDIDO)
SOLICITE A LEITURA DE CARACTERES DO TECLADO (5º PEDIDO)
AGUARDE O ATENDIMENTO DO 5º PEDIDO
AGUARDE O ATENDIMENTO DO 1º PEDIDO

Utilizando o código abaixo

- nada a fazer
P tarefa principal
T tarefa terminal
D tarefa disquete,

a execução do programa acima ocasiona a seguinte seqüência de acontecimentos síncronos:

TAREFA	ATIVIDADE
P	Solicita a carga
D	Comanda a leitura do 1º setor
P	Solicita a exibição da tela
T	Exibe a tela
D	-
P	Pedido de digitação
T	Prepara a leitura de caracteres do teclado
D	Transfere o 1º setor, comanda a leitura do 2º setor
P	-
T	-
D	-
P	-
T	Encerra a leitura de caracteres do teclado
D	Transfere o 2º setor. Comanda a leitura do 3º setor
P	Solicita a exibição da tela
T	Exibe a tela
D	-
P	Pedido de digitação
T	Prepara a leitura de caracteres do teclado
D	Transfere o 3º setor. Prepara a leitura do 4º setor
P	-
T	-
D	-
P	-
T	Encerra a leitura de caracteres do teclado
D	-
P	-
D	Transfere o 4º setor. Encerra a atividade de disco
P	Prossegue na execução do programa

Para o operador do equipamento, tudo se passa como se, à medida que estava digitando, as operações de disco fossem realizadas. O programador de sistemas, por outro lado, só se preocupou em solicitar operações de E/S sem necessariamente aguardar o seu término. As rotinas de E/S desdobravam o pedido original em uma série de atividades elementares de E/S, comandando-as e aguardando o seu término. Os periféricos, à medida que realizam os comandos solicitados, são atendidos e reprogramados ou para nova atividade, ou então, para o repouso.

3.2 Macros

Uma macro é considerada um conjunto de teclas digitadas. Trata-se de um recurso muito poderoso para o processamento de textos.

As macros são recursivas, isto é, dentro de uma macro podem ser referenciadas outras.

Para o usuário, a macro pode ter as mais diversas aplicações - desde a simples abreviatura até a criação de novos comandos. A abreviatura funciona da seguinte forma: é criada uma macro (por exemplo, SPT = Sistema de Processamento de Textos); cada vez que for encontrada em um texto, é substituida pelo seu conteúdo. Um novo comando pode ser criado, agrupando-se em uma macro um ou mais comandos, sejam eles imediatos, deferidos ou gerais.

As macros também podem ser utilizadas para a padronização de formulários e textos dentro de uma empresa. A primeira pode ser conseguida, definindo os parâmetros da diagramação de cada um dos formulários em uso. A segunda pode ser obtida, criando uma série de padrões de texto, que podem ser usados mediante uma simples referência à macro.

Ao longo do texto, o hexadecimal FF, que identifica uma referência a uma macro, será representado por um "!", para facilitar a sua representação em uma máquina de escrever.

A seguir são apresentadas a descrição das principais características, a descrição das operações internas sobre as macros, e alguns exemplos.

3.2.1 Características de Macros

As macros tem algumas peculiaridades, que merecem atenção:

a)- uma macro possui tamanho variável

b)- o nome da macro possui tamanho variável

c)- uma macro deve poder ser excluída, incluída e alterada com facilidade.

A hipótese a) acarretou na escolha de uma área contínua de memória, onde as macros novas e alteradas são acrescentadas. Sempre que a área for insuficiente para acomodar a nova macro, é realizada a compactação da área.

Pensou-se também na hipótese de utilizar como estrutura áreas interligadas, tanto de tamanho fixo como de tamanho variável. Entretanto, as fragmentações interna e externa foram um argumento convicente para abandoná-la.

A hipótese b) acarretou em uma solução similar à da adotada para as macros. Ainda, para cada macro, foram acrescentados vários atributos, tais como tamnho da macro, tamanho do nome, tipo de macro e também um elo de ligação que será visto logo abaixo.

A hipótese c) acarretou na escolha de uma árvore para implementar o acesso aos elementos. Para tanto, foi escolhida uma área à parte, que serve unicamente para acessar os elementos.

3.2.2 Estrutura

A estrutura pode ser representada graficamente da seguinte forma:

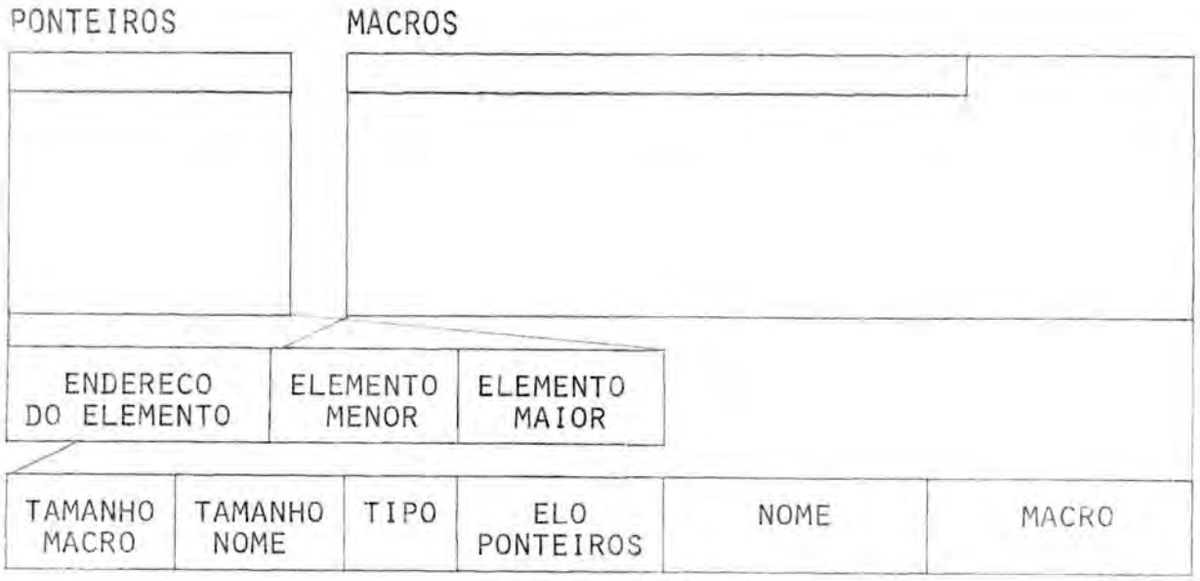


FIGURA 4 - ESTRUTURA INTERNA DAS MACROS

Na tabela PONTEIROS, ENDEREÇO DO ELEMENTO aponta para um endereço da tabela MACROS. ELEMENTO MENOR aponta para o elemento com NOME menor que de em PONTEIROS, ELEMENTO MAIOR aponta para o com NOME maior. Se algum destes elos for zero, significa que não existe o elemento.

Na tabela MACROS, TAMANHO MACRO indica, em bytes, qual o tamanho da MACRO. TAMANHO NOME indica, em bytes, qual o tamanho do nome. TIPO informa que tipo de macro é: comando, parâmetro, marca, macro, disponível, etc. (maiores esclarecimentos sobre os tipos serão dados na seção 3.5, Execução de comandos). ELO PONTEIROS aponta para o número do elemento PONTEIROS que aponta para esta macro. MACRO contém o conteúdo correspondente.

O elemento de índice Ø contém algumas informações especiais:

- O ENDEREÇO DO ELEMENTO aponta para uma macro especial, denominada sentinela. Esta é sempre a última macro dentro da área MACROS. Pode ser ampliada, ou até o fim da área, ou até alcançar o tamanho máximo de uma macro. Esta propriedade é importante, pois desta forma os nomes e as macros propriamente ditas são criadas.
- O ELEMENTO MENOR aponta para a raiz da árvore.
- O ELEMENTO MAIOR aponta para o topo da pilha de elementos disponíveis da tabela PONTEIROS. O ELEMENTO MAIOR deste elemento, por sua vez, aponta para o próximo disponível.

3.2.3 Operações

Existe um conjunto de operações que podem ser realizadas sobre as macros. Estas manipulam a estrutura lógica, sem se importar com a estrutura física dos dados.

a)- PESQUISA

Realiza a pesquisa de uma chave dentro das macros, percorrendo a árvore. Se não encontar o elemento, mostra o ponto em que este deverá ser inserido.

b)- INCLUSÃO

É criado um novo elemento na árvore. Este aponta para o que era a sentinela. Nova sentinela é criada.

c)- ALTERAÇÃO

Um determinado elemento da árvore passa a apontar para a sentinela. A macro apontada anteriormento é tornada disponível. Nova sentinela é criada.

d)- EXCLUSÃO

Um determinado elemento da árvore é excluido. A sua área apontada é tornada disponível.

Cumpre ressaltar que, à medida que as operações são realizadas, o espaço final da tabela tende a diminuir, ao mesmo tempo que os espaços disponíveis tendem a se tornar mais numerosos dentro do espaço utilizado. A rotina que resolve este problema, juntamente com um outro grupo de rotinas que manipulam as estruturas físicas, será discutida no capítulo 4 - Implementação.

3.2.4 Exemplos

Serão vistos alguns exemplos de como as macros podem ser utilizadas pelo usuário e pelo próprio sistema.

a)- ABREVIATURA

Ao invés de escrever a palavra "software", o usuário cria uma macro SOF, de conteúdo "software". Cada vez que colocar !SOF em seu texto, aparecerá, no lugar, a palavra "software".

Como a utilização de macros é recursiva, podem ser imaginadas aplicações mais complexas. Supondo as seguintes opera

ções:

CRIE SP / sistema de processamento de/;

CRIE SPT / !SP textos/;

CRIE SPD / !SP dados/;

Ao referenciar SPT, !SP será substituído pelo seu conteúdo, e o texto final será "sistema de processamento de textos".

Como as macros podem ser utilizadas tanto pelo editor como pelo formatador, as referências (a macros) podem ser inseridas no texto e as macros definidas posteriormente.

b)- CRIAÇÃO DE UM NOVO COMANDO

O usuário deseja criar um novo comando. Este será, por exemplo, uma instrução que conta o número de períodos em um texto. Supondo que a cada caracter ponto (.) está associado um período, a solução poderia ser:

```
CRIE CONTAPERIODOS :1 MARQUE AQUI;
                        POSICIONE INICIO;
                        CRIE CONTAPONTOS /Ø/;
                        SEMPRE  :2 PESQUISA /./;
                                   ATRIBUA CONTAPONTOS
                                   CONTAPONTOS+1;
                                :2
                        POSICIONE AQUI;
                        EXCLUA AQUI;
                    :1
```

Observa-se que, no exemplo acima, :1 e :2 são considerados delimitadores de seqüências.

Após a criação do comando acima, qualquer referência a !CONTAPERIODOS colocará na macro CONTAPONTOS o número de períodos contidos no texto.

c)- REDUÇÃO

As macros são utilizadas pelo próprio sistema para a realização de reduções. A redução consiste em desdobrar um comando em uma série de comandos elementares, facilitando muito a programação do sistema.

Por exemplo, o comando

```
ENQUANTO  !CONT < 10  /!MEUCOMANDO/
```

é reduzido para

```
CRIE DEF1  /!MEUCOMANDO/;
CRIE DEF2 :1 SE !CONT < 10 :2!DEF1 !DEF2:2:1:
!DEF2;
DESTRUA DEF2;
DESTRUA DEF1;
```

3.3 Organização de arquivos

O meio de armazenamento e a organização dos dados nele contidos representam um fator importante em um sistema de processamento de textos. Na presente seção são discutidos os principais objetivos da organização de arquivos e o formato padrão de gravação em disquete.

3.3.1 Objetivos

Os principais objetivos que determinaram a escolha da organização de arquivos foram:

a)- Fácil inclusão/exclusão

Um sistema de processamento de textos tem como característica um grande número de inclusões / exclusões. A organização escolhida deve permitir muitas inclusões e exclusões no arquivo.

b)- Bidirecional

Como uma das propostas básicas é o fato de poder percorrer o texto nos dois sentidos, a organização escolhida deve admitir a possibilidade de leitura do início ao fim do arquivo e vice-versa.

c)- Expansão

Um arquivo ocupa, ao ser criado, um certo espaço físico. Como o usuário não precisa saber de antemão qual o espaço final necessário, o sistema deve permitir a expansão dinâmica de arquivos sem a intervenção do operador.

d)- Compatibilidade

Os dados gravados pelo SPT devem poder ser lidos por outros equipamentos.

3.3.2 Estrutura dos Arquivos

Tendo em vista o objetivo a), fácil inclusão /exclusão de registros, ficou fora de cogitação a possibilidade de utilizar uma estrutura sequencial. Por outro lado, o objetivo b), leitura bidirecional, obrigou a adoção de uma estrutura duplamente encadeada. Como o texto não é algo divisível em registros (como um código fonte, por exemplo), houve ainda a necessidade de incluir em cada setor um contador do número de caracteres válidos.

A alocação de área dinâmica, objetivo c), sugeriu a utilização de uma pilha de registros disponíveis. A desvantagem deste método, porém, consiste em uma inicialização obrigatória dos disquetes. Outra desvantagem é o número elevado de deslocamentos de braço que é necessário à medida que o disquete vai sen

do reutilizado.

Uma outra forma de alocação, de dois níveis resolve os problemas acima enumerados. O primeiro nível consite em alocar o espaço disponível trilha a trilha. A inicialização de um disquete consiste, portanto, em tornar todas as trilhas disponíveis - em outros termos, inicializar um vetor de 73 elementos. O segundo nível consiste em alocar o setor dentro da trilha. Isto pode ser realizado sem uma inicialização prévia da trilha.

A liberação de registros ocorre mediante a sua anexação a uma pilha de disponíveis. Nesta pilha também serão acrescentadas os registros disponíveis de uma trilha alocada.

O objetivo d), compatibilidade com outros equipamentos, fez com que se optasse pelo padrão da indústria - o formato IBM 3740. Este padrão pode ser lido e processado por um grande número de equipamentos brasileiros. Algumas alterações, entretanto, foram introduzidas para permitir a realização dos itens mencionados acima, sem, no entanto, fugir do formato padrão.

3.3.3 Descrição do formato padrão

a)- Trilhas

O disquete possui 77 trilhas, 26 setores por trilha, 128 caracteres por setor.

As trilhas são subdivididas da seguinte forma:

TRILHA 0 - trilha de índices

TRILHAS 1 - 73 - trilhas de dados

TRILHA 74 - sem uso específico

TRILHA 75 - 76 - trilhas reserva.

A trilha de índices possui a seguinte subdivisão:

SETORES 1 - 4 : sem uso específico

SETOR 5 : Índice de trilhas defeituosas

SETOR 6 : sem uso específico

SETOR 7 : identificação do volume

SETORES 8 - 26 : Índices de arquivo.

As trilhas reserva são utilizadas quando uma das trilhas de dados não permite mais a gravação/leitura de dados. Em uma nova inicialização do disquete o usuário pode especificar as trilhas defeituosas. O processo de inicialização indica estas trilhas como defeituosas e realiza um deslocamento das outras em direção ao centro do disquete.

O processo de leitura, ao detectar uma trilha defeituosa,desloca-se à trilha adjacente.

b)- Índices

Um disquete admite um máximo de 19 índices de arquivos, distribuídos do setor 8 ao setor 26 da trilha 0. Cada um dos índices de arquivo possui os seguintes campos importantes:

COLUNA	CONTEÚDO
1	H/D
2 - 4	DR1
6 - 13	nome do arquivo
25 - 27	tamanho do registro
29 - 33	início da área do arquivo
35 - 39	fim da área do arquivo
75 - 79	próximo registro disponível

c)- Dados

Os setores são áreas de 128 posições úteis, possuindo ainda uma marca associada e dois caracteres de CRC. A marca pode ser do tipo DAM - Data Adress Mark, identificando dados válidos, ou então do tipo DDAM - Deleted Data Adress Mark, identificando dados excluídos. Os dois caracteres de CRC permitem a detecção de falhas nos dados gravados.

3.3.4 Formato de disco utilizado

Dentro do formato de gravação padrão foram realizadas algumas alterações descritas abaixo:

a)- Trilha 0, Setor 1

Neste setor específico estará contida a relação de pertinência das trilhas. Como este setor é reservado ao uso do sistema, mas não tem nenhuma finalidade, esta alteração não despadroniza o disquete.

O formato deste setor é o seguinte:

VETOR DE TRILHAS	

FIGURA 5 - FORMATO DO SETOR 1, TRILHA 0

Nas primeiras 73 colunas está colocado um byte que, se estiver contido no intervalo [8,26] indica que a trilha foi apropriada ao índice cujo número está nesta posição. Se o caracter tiver um valor fora do intervalo acima, a trilha está disponível e pode ser apropriada.

Na inicialização do disquete padrão este setor é gravado com caracteres espaço, cujo valor decimal (64) está fora do intervalo acima. Todo o disquete inicializado de forma padrão estará automaticamente inicializado para o SPT.

b)- Setor de índice

Este setor permaneceu com o mesmo conteúdo como no formato padrão. O ínicio da área do arquivo aponta para a menor trilha utilizada pelo arquivo. O fim da área do arquivo aponta para a maior trilha utilizada. O próximo registro disponível aponta para o próximo registro após o fim da área.

Este formato de gravação permite que todo o conteúdo do arquivo possa ser lido por outro equipamento. Entretanto, deve ser lembrado que nem todas as trilhas precisam necessariamente pertencer aquele arquivo.

c)- Setor inicial

O setor inicial, isto é, o primeiro setor da menor trilha, possui um formato especial, contendo informações que são importantes para o SPT. Seu formato é:

VETOR DE TRILHAS	ELO MARCAS	ELO MACROS	ELO D	EXT	ELO A	ELOP

FIGURA 6 - FORMATO DO SETOR INICIAL

O VETOR DE TRILHAS permite a realização da tradução das trilhas lógicas (empregadas nos elos) para trilhas físicas (gravadas nos campos de endereço do disquete). Esta tradução visa facilitar o gerenciamento do disquete - trocar duas trilhas de posição só afeta o vetor de trilhas; os elos de ligação não precisam ser percorridos, realizando os acertos.

O ELO D aponta para o primeiro setor disponível.

A EXTENSÃO contém uma constante, 73.

O ELO A aponta para o primeiro setor de dados. O ELO P aponta para o último setor de dados.

d) Setor de dados

O setor de dados possui o seguinte formato:

DADOS	EXTENSÃO	ELOA	ELOP

FIGURA 7 - FORMATO DO SETOR DE DADOS

Os DADOS são um conjunto de, no máximo, 123 caracteres. O número de caracteres válidos do registro é fornecido pelo campo EXTENSÃO.

OS ELOS A e P apontam, respectivamente, para o setor anterior e para o próximo. Cada elo é formado por dois bytes, sendo que o primeiro indica o número da trilha lógica e o segundo o número do setor.

Os campos de controle foram colocados no fim do registro para facilitar eventuais futuras ampliações do sistema - se houver a necessidade de ler um arquivo seqüencial de tamanho de registro fixo, basta, após a leitura de cada registro, inserir a extensão do registro (que deve ser menor ou igual a 123) e apontar ELO P para o próximo registro.

3.3.5 Arquivos usados pelo SPT

Tanto o formatador como o editor possuem dois arquivos sobre os quais é realizado todo o processamento. Um arquivo é o de entrada, de onde são extraídos os dados a processar. Outro arquivo é o de saída, para onde os dados processados são transferidos.

No editor, existem duas possibilidades de utilização dos arquivos:

a)- Cópia de arquivos

Ao realizar a edição de um arquivo, o usuário pode optar pela cópia do arquivo. Neste caso, todas as alterações realizadas no texto só aparecem no novo arquivo. Se houver queda de luz, não se perdem arquivos.

b)- Edição no próprio arquivo

O usuário realiza alterações no próprio arquivo. Neste caso, o arquivo de entrada é o mesmo que o de saída. Ao faltar energia, entretanto,o arquivo pode ser perdido.

No formatador, existem duas possibilidades de utilização de arquivos:

a)- Com saída

Esta modalidade é utilizada sempre que o usuário qui

ser escrever o texto que está sendo formatado.

b)- Sem saída

Esta modalidade deve ser utilizada sempre que um texto sofreu uma série grande de modificações, ou nunca foi formatado. Como os comandos para o formatador não foram submetidos a uma consistência pelo editor, podem aparecer referências e macros inexistentes, comandos inválidos e outros erros. Para evitar que haja desperdício de papel, pode ser realizada uma formatação sem saída, só para detectar erros.

3.3.6 Operações

As principais operações que podem ser realizadas sobre os arquivos são:

NOME	FUNÇÃO
ABRIR	realiza as tarefas relacionadas com a abertura de um arquivo
FECHAR	realiza o fechamento do arquivo.

A operação ABRIR consiste em trazer para a memória todos os atributos do arquivo. Além disto, transfere as marcas dos arquivos para a tabela de marcas. Lê, ainda, o setor 1, onde estão armazenados os dados referentes à tradução de trilhas, primeiro setor de dados, etc.

A operação FECHAR consiste em gravar os dados atualizados no rótulo e no setor 1 do arquivo, se este for de saída. Além disto, todas as marcas deste arquivo são excluídas da tabela de marcas.

3.4 Janela

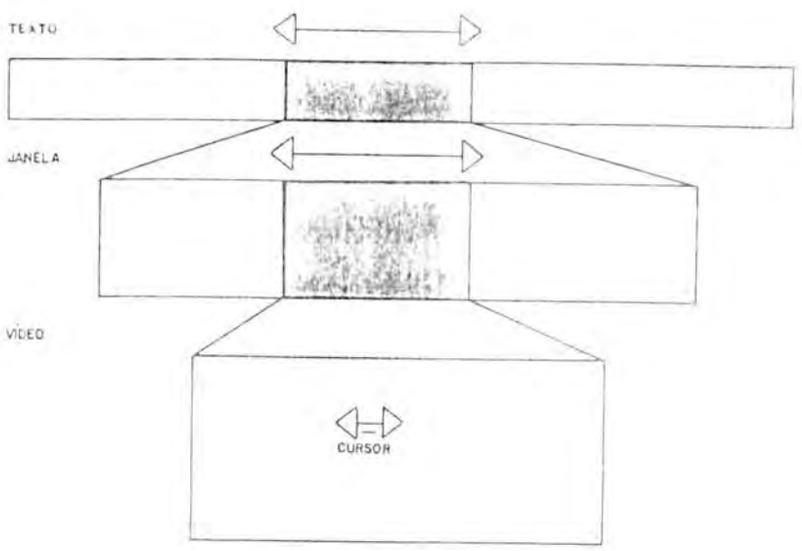
O texto a ser editado é um conjunto de caracteres virtualmente infinito. O operador , bem como os dispositivos de entrada e saída, só possuem a capacidade de trabalhar sobre um trecho limitado do texto.

Este trecho específico é denominado de janela. Nela é que são realizadas todas as operações de edição do texto.

A seguir, serão estudados alguns aspectos relativos ao seu funcionamento.

3.4.1 Apresentação

A janela pode ser representada da seguinte forma:

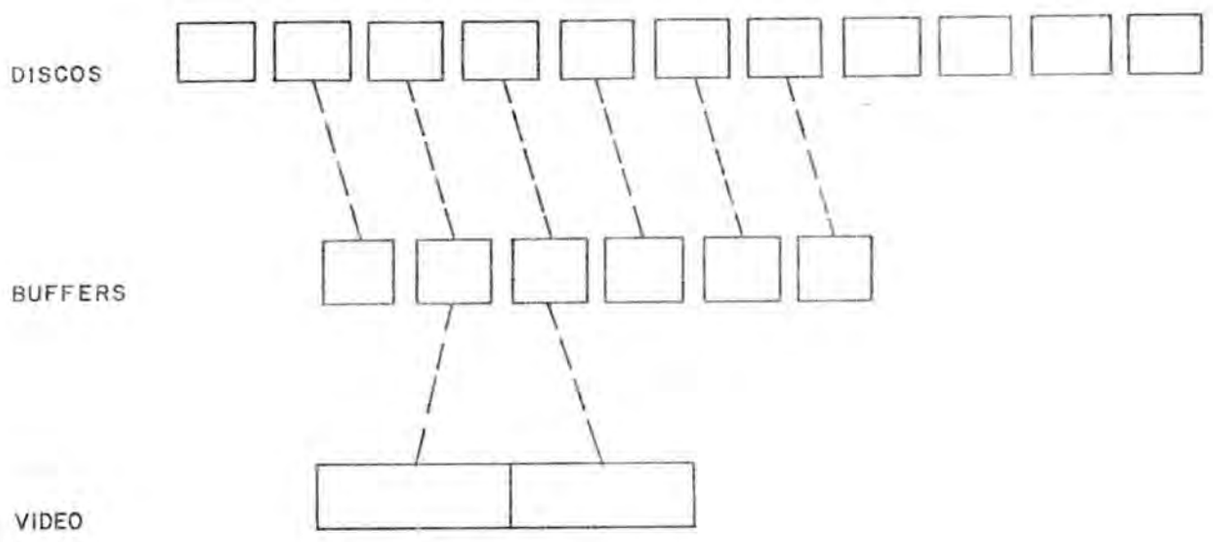


A janela pode se deslocar sobre o texto nos dois sentidos. O vídeo pode se deslocar sobre a janela em ambos sentidos, exibindo o seu conteúdo. No vídeo, um cursor pode ser deslocado para frente e para trás, dentro das limitações da tela.

Uma das grandes vantagens que este método traz é que o vídeo reflete sempre o estado atual do texto. As alterações sobre o texto podem ser realizadas na própria tela, sem necessidade de comandos especiais. À medida que o usuário realiza as alterações, o texto é exibido em seu estado atual; desta forma, as alterações podem ser conferidas visualmente à medida em que são efetuadas.

3.4.2 Estrutura

Para implementar a janela, várias estruturas de dados podem ser adotadas. Uma das possibilidades é a seguinte:



Usando esta estrutura, os buffers são a imagem de seus setores correspondentes em disco. A estrutura é tornada linear quando o texto é exibido.

Quando existe um certo número de inclusões em um determinado buffer , um novo setor precisa ser alocado. Este é inserido entre aquele, no qual se encontra o cursor, e o próximo. Por outro lado, quando existe um certo número de exclusões em um determinado buffer , este pode tornar-se vazio e ser liberado.

Algumas características resultantes do emprego desta estrutura devem ser destacadas:

- A grande facilidade de inclusão e exclusão de caracteres no texto.
- O conjunto de buffers constitui a própria área de trabalho, não necessitando de outras áreas para a manipulação de textos.
- Se for adicionado um atributo ao buffer , que diz se o seu conteúdo está gravado em disco ou não, os setores que não sofreram alterações não precisam ser regravados em disco. Isto economiza considerável tempo de E/S em uma operação de pesquisa, por exemplo.

3.4.3 Marcas

A marca é um elemento que aponta para uma certa posição do texto. É utilizada para fins de:

- Posicionamento, pois o cursor pode se deslocar diretamente para uma posição marcada, sem a necessi-

dade de percorrer o arquivo.

- Inclusão, quando o trecho a ser incluído de outro arquivo é delimitado por marcas.
- Exclusão, quando o trecho a ser excluído é delimitado por marcas. Entretanto, se o trecho for pequeno, a exclusão pode ser realizada sem a utilização de marcas.

As marcas são atualizadas da seguinte forma:

- Enquanto a marca aponta para um setor em disco, ela sempre estará apontando para a mesma posição.
- Quando um setor apontado por uma marca é lido, a marca é atualizada; ao invés de indicar um setor em disco, aponta para um buffer na memória.
- Ao ocorrer um deslocamento de texto no buffer apontado por uma marca, e esta apontar para o trecho do texto que sofreu deslocamento, a marca acompanha o deslocamento.
- Após serem escritas e terem o seu buffer desalocado, as marcas que apontam para o setor voltam a apontar para o disco.

Sob o ponto de vista do sistema, uma marca se comporta de maneira similar a uma macro. Em primeiro lugar, possui um nome de tamanho variável. Em segundo lugar, pode ser incluída, excluída e alterada dinâmicamente. Em terceiro lugar, possui um certo conteúdo associado.

Tirando proveito das rotinas já existentes de manipulação de nomes de tamanho variável, a estrutura escolhida para as marcas foi uma tabela de MARCAS, formada pelos seguintes ele

mentos:

INDICE MACROS	NUMERO DO ARQUIVO	TRILHA	SETOR	DESLOC

FIGURA 10 - ESTRUTURA DAS MARCAS

O ÍNDICE MACROS aponta para o elemento de tabela MACROS que contém o nome da marca. O NÚMERO DO ARQUIVO indica o nível do arquivo (veja parágrafo seguinte). A TRILHA e o SETOR compõe o endereço lógico do setor no qual se encontra a marca, e o DESLOCAMENTO fornece a distância, em caracteres, da marca ao primeiro byte do setor.

Quando numa inclusão de arquivo for comandada nova inclusão, vários arquivos de entrada estarão abertos ao mesmo tempo. As marcas de cada arquivo precisam estar presentes na memória, para permitir o controle da inclusão. Ao encerrar a inclusão, o arquivo poderá ser fechado, e suas marcas removidas da tabela acima, onde não mais são necessitadas. Para saber que marcas podem ser removidas, é necessário incluir o campo NÚMERO DO ARQUIVO.

Quando a TRILHA possui valor zero, o SETOR aponta para o número do buffer no qual se encontra a marca. O deslocamento será a distância entre a marca e o primeiro byte do buffer.

Na tabela MACROS existe, além do nome, um ponteiro para o elemento da tabela MARCAS, para permitir que a marca seja acessada pelo seu nome.

3.4.4 Operações

Existe um conjunto de operações que podem ser realizadas sobre a janela. Estas operações trabalham com a estrutura lógica da janela, sem se importar com a estrutura física dos dados. São elas:

a)- PESQUISAR

Percorre o arquivo à procura de um certo conjunto de caracteres. O sentido pode ser tanto do início ao fim do arquivo como vice-versa.

b)- DESLOCARCURSOR

Desloca o cursor um certo número de caracteres para frente ou para trás. É utilizada para fins de formatação da tela.

c)- FECHAR

Fecha uma das alas da janela. A ala esquerda é composta por todos os caracteres à esquerda do cursor. Todos os buffers da ala são gravados. Todo o deslocamento do cursor causa, a partir deste ponto, perda de informações. É utilizada para fins de exclusão.

d)- ABRIR

Abre uma das alas da janela. Os buffers do lado aberto são preenchidos com dados.

e)- POSICIONAR

Fecha as duas alas da janela, desloca o cursor até a marca, e volta a abrir a janela no novo ponto.

f)-DESLOCARATEMARCA

Desloca o cursor até a marca, caracter a caracter. Se, por exemplo, a ala esquerda estiver fechada, os dados são excluídos.

g)- INSERIR

Insere um conjunto de caracteres na posição apontada pelo cursor. Após a operação, o cursor aponta para o primeiro caracter posterior ao conjunto de caracteres incluídos.

h)- ALTERAR

Altera um conjunto de caracteres na posição apontada pelo cursor. Após a operação, o cursor aponta para o primeiro caracter posterior ao conjunto de caracteres alterados.

i)- EXCLUIR

Exclui um conjunto de caracteres de posição apontada pelo cursor.Após a operação, o cursor aponta para o primeiro caracter posterior ao conjunto de caracteres excluídos.

j)- CENTRAR

Posiciona o cursor no meio da tela.

k)- EXIBIR

Exibe o texto, ou à esquerda, ou à direita do cursor,conforme parâmetro fornecido.

3.4.5 Exemplo

O comando ELIMINE DE MARCA1 ATÉ MARCA2 é desdobrado nas seguintes operações sobre a janela:

a)- POSICIONAR NA MARCA1

b)- FECHAR ala esquerda

c)- DESLOCARATEMARCA MARCA2

d)- ABRIR ala esquerda

Após a operação a), o cursor estará apontando para a MARCA1, que é o início do trecho que deve ser eliminado. A operação b), acarreta que todo o texto à esquerda do cursor, a partir deste ponto, será desprezado. A operação c) desloca o cursor até a MARCA2. Todo o texto deixado à esquerda do cursor é perdido. Após realizar a operação d), o antigo conteúdo à esquerda da MARCA1 é transferido para a esquerda da MARCA2.

3.5 Execução de comandos

Existem vários tipos de comandos, dependendo do programa em execução e do modo no qual este está operando.

3.5.1 Modos de Operação

Existem no SPT, dois modos de operação:

- interativo, no qual o operador interage com a máquina,
- contínuo, no qual a máquina executa uma série de atividades sem a intervenção do operador.

O editor utiliza os dois modos de operação. Possui dois tipos de tela para fins de interação: a de dados e a de comandos. Na tela de dados, o operador pode realizar alterações diretamente no texto, ou então deslocar o cursor rapidamente sobre a tela, posicionando-o no ponto desejado. Na tela de comandos, escreve e edita comandos, até apresentarem a forma desejada, quando são processados de forma contínua pelo editor.

O formatador sempre opera de forma contínua, processando o texto e os comandos nele embutidos.

3.5.2 Execução de comandos no modo interativo

A cada tecla digitada corresponde uma ação imediata que pode variar, dependendo da tecla que o operador digitou. A seguir, será visto que ações correspondem ao acionamento das te

clas no modo interativo.

É importante lembrar que uma tela de dados é um trecho de um texto maior. Ao chegar com o cursor no final da tela, a tela pode ser deslocada. No caso de estar com uma tela de comandos, a tela não pode ser deslocada por não fazer parte de um texto maior.

Se o operador digitar uma tecla normal (isto é, qualquer tecla que não seja de função), o caracter correspondente é exibido na tela. Se o cursor aponta para um trecho de texto, este é alterado. Em caso contrário, trata-se de uma adição de um caracter ao texto.

Se o operador digitar uma tecla de função, alguma ação especial acontecerá. As funções de movimentação disponíveis no SPT são:

a)- Movimentação do cursor/texto

A movimentação do cursor facilita o posicionamento na palavra desejada, para fins de edição. A movimentação do texto pode estar associada a um movimento do cursor ou não, dependendo do caso.

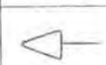
SÍMBOLO	FUNÇÃO NA TELA DE DADOS	FUNÇÃO NA TELA DE COMANDOS
	Retorna o cursor à primeira posição da tela. Caso já se encontre lá, exibe a tela anterior à atual.	Retorna o cursor à primeira posição da tela.
	Posiciona o cursor na última posição da tela. Caso já se encontre lá, exibe a tela posterior à atual.	Posiciona o cursor na última posição da tela.
	Avança o cursor. Caso se encontre na última posição da tela, posiciona o cursor no início da linha e sobe o texto uma linha.	Avança o cursor. Caso se encontre na última posição da tela, posicione o cursor no início da tela.
	Recua o cursor. Caso se encontre na primeira posição da tela, posiciona o cursor no fim da linha e desce o texto uma linha.	Recua o cursor. Caso se encontre na primeira posição da tela, posiciona o cursor no fim da tela.
	Sobe o cursor uma linha. Caso se encontre na primeira linha, desce o texto uma linha.	Sobe o cursor uma linha. Caso se encontre na primeira linha, posiciona o cursor na última linha.
	Desce o cursor uma linha. Caso se encontre na última linha, sobe o texto uma linha.	Desce o cursor uma linha. Caso se encontre na última linha, posiciona o cursor na primeira linha.
	Avança o cursor uma palavra.	Avança o cursor 8 posições.
	Recua o cursor uma palavra.	Recua o cursor 8 posições.
	Avança o cursor para o início do próximo período.	Avança para a primeira posição da próxima linha.
	Recua o cursor para o início do presente período. Caso já se encontre lá, recua ao início do período anterior.	Recua para a primeira posição da linha. Caso já se encontre lá, sobe o cursor uma linha.

Tabela 4 - Funções do cursor

b)- Inversão do modo inclusão

Esta tecla de função tem por finalidade colocar/tirar o editor do estado de inclusão. Quando se encontra neste estado, todos os caracteres de dados são inseridos no texto na posição apontada pelo cursor. O cursor e o texto após a inserção se deslocam uma posição para trás.

c)- Exclusão

Esta tecla de função tem por finalidade excluir o caracter apontado pelo cursor do texto.

d)- Troca de modo

Esta tecla de função tem por finalidade alternar o modo de dados com o modo de comandos e vice-versa. A cada modo está associada uma tela diferente.

e)-Execução de comandos

Esta tecla de função tem por finalidade encerrar a edição da tela de comandos e solicitar ao editor que os comandos digitados sejam executados.

3.5.3 Execução de comandos no modo contínuo

No modo contínuo de operação, os comandos são executados sucessivamente.

No caso do editor, após preencher a tela de comandos e digitar a tecla de execução de comandos, o editor inicia a interpretação. Cada um dos comandos digitados é desdobrado em uma série de microcomandos, que serão vistos logo abaixo.

No caso do formatador, após iniciar o processamento do texto, todos os comandos são identificados e executados, sendo, de forma similar ao editor, decompostos em microcomandos.

Existem muitos microcomandos. Para facilitar a sua compreensão e implementação, foram subdivididas nas seguintes famílias:

- janela
- macros
- arquivos
- interpretação.

As tres primeiras famílias já foram vistas anteriormente, bem como as operações elementares que podem ser realizadas com as suas respectivas estruturas. Os microcomandos apenas representam, para fins de execução dos comandos, uma tarefa elementar que deve ser realizada pelos outros módulos.

A família dos microcomandos de interpretação será vista com maiores detalhes na seção seguinte.

3.5.4 Microcomandos de interpretação

A linguagem utilizada foi escolhida de tal forma, que o interpretador sabe, de antemão, o que vem a seguir.

Ao iniciar a execução de uma tela de comandos, a primeira palavra deve ser necessariamente um comando. Após verificar qual o comando a executar, são obtidos os parâmetros. Se forem possíveis vários tipos diferentes, uma palavra chave identifica qual o escolhido.

O hexadecimal 81, representado no texto a seguir por ";" encerra o comando.

Desta forma, o interpretador não precisa obter um comando inteiro, para depois analisá-lo e executá-lo, mas pode realizar as operações à medida que surgem os elementos sintáticos. Isto facilita o emprego de macros, pois pode acontecer que, no caso de haver muita referência a macros dentro de macros, a análise se torne bastante complexa.

As principais rotinas do interpretador são, portanto, rotinas que procuram obter um certo elemento sintático no texto. Existem ainda as rotinas que verificam a presença/ausência da tabela de MACROS. Também existe uma instrução especial destinada à redução de comandos. São elas:

BUSCASEQUÊNCIACOM	Coloca, na área de macro da sentinela, uma seqüência com delimitadores.
BUSCASEQUÊNCIASEM	Coloca, na área de macro da sentinela, uma seqüência delimitada por brancos.
BUSCASOLUÇÃO	Obtém a solução de uma expressão. Se a expressão for aritmética, coloca seu valor resultante na área de macro da sentinela. Se a expressão for booleana seu valor será atribuida à macro do sistema "$COND".
BUSCAPARÂMETRO	Busca um parâmetro do comando.
BUSCANOME	Coloca, na área de nome da sentinela, um nome, delimitado por brancos ou por ponto-e-vírgula.
BUSCACOMANDO	Percorre a área de comando à procura de um comando.
VERIFICAPRESENÇA	Verifica se o nome contido na sentinela existe na tabela de MACROS.
VERIFICAAUSÊNCIA	Verifica se o nome contido na sentinela existe na tabela de MACROS.
EMPILHAMACRO	Faz com que o interpretador "leia" uma MACRO

Tabela 5 - Microcomandos de interpretação

Para ilustrar os microcomandos de interpretação, é mostrado, como exemplo, a decomposição do comando CRIE <nome> <seqüência>.

a)-BUSCANOME

b)-VERIFICAAUSÊNCIA

c)-BUSCASEQUÊNCIA

d)-CRIE (MACRO)

Após a execução de a), a sentinela conterá um nome. Como deve ser criada uma nova macro, a não existência do nome é assegurado pelo comando b). O comando c) coloca, na área da macro da sentinela, uma seqüência de caracteres. O comando d) introduz a nova macro na estrutura lógica das macros.

Em cada um dos passos acima é verificado se ocorreu algum erro, que acarreta o fim precoce da análise do comando.

3.5.5 Estrutura

Para realizar a interpretação de comandos foram utilizadas duas pilhas-uma de dados, outra de comandos. Ambas possuem a mesma estrutura, como está indicado na figura 11.

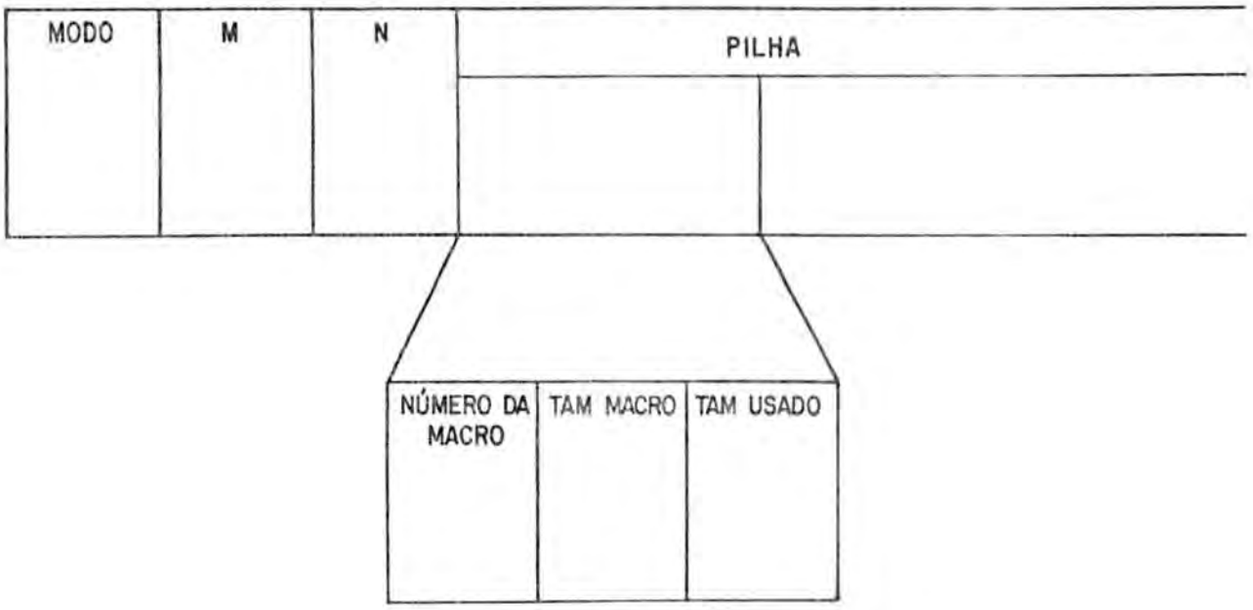


FIGURA 11 - ESTRUTURA DAS PILHAS DE INTERPRETACAO

O MODO é um atributo da pilha, dizendo se os bytes devem ser obtidos de modo transparente ou não (isto é, se referências a macros devem ser ignoradas ou não). M é o número de elementos úteis da pilha, N o número máximo de elementos. NÚMERO DA MACRO aponta para a macro que está sendo usada. TAMMACRO é o tamanho da macro e TAMUSADO é o número de bytes já processados desta macro.

Como exemplo, seja a seguinte referência a macros:

NÚMERO MACRO	TAMANHO MACRO	NOME	CONTEÚDO
4	11	X	"ABCD !Y EFG"
23	13	Y	"HIJK !L!Z MNO"
12	1	Z	"4"
10	4	L4	"PQRS"

O texto interpretado será:

"ABCD HIJK PQRS MNO EFG"

Quando estiver sendo apontada a letra Q, a situação da pilha será:

NÚMERO DA MACRO	TAMMACRO	TAMUSADO
4	11	7
23	13	9
10	4	2

3.5.6 Execução de comandos

A filosofia de execução dos comandos enquadrou-se na estrutura de macros - a cada comando corresponde uma série de microcomandos. Foi introduzida a macro do tipo "comando" e "parâmetro", que contém o conjunto de microcomandos que causam o efeito desejado.

Para não causar confusão com macros do usuário, as macros do sistema são prefixadas por um sinal especial (%). Como o usuário não pode criar uma macro que inicia por este caracter, não existe a necessidade de introduzir palavras reservadas ao sistema.

Desta forma, existe a seguinte macro do sistema:

```
% CRIE = BUSCANOME
        VERIFICAAUSÊNCIA
        BUSCASEQUÊNCIA
        CRIE (MACRO)
```

Ao ser executado o microcomando BUSCACOMANDO, este coloca um caracter "%" na área de nome da sentinela e percorre o texto à procura de uma palavra. Após transferi-la para a área de nome, é realizada uma pesquisa na tabela MACROS. Ao ser encontra

do, o conteúdo correspondente é empilhado na pilha de comandos. O próximo microcomando a ser executado será o primeiro microcomando do comando analisado.

Os parâmetros são obtidos de forma semelhante. Ao encontrar o microcomando BUSCAPARÂMETRO, a área de nome da sentinela é inicializada com o caracter "%". A seguir, é acrescentada a palavra chave do texto. O conjunto é pesquisado, e a macro correspondente é empilhada na pilha de comandos, desde que não haja erro de pertinência (um parâmetro que não é admitido com o comando em processamento).

Como exemplo pode ser citado o comando PRÓXIMO, que pode ter um entre uma série de parâmetros. A macro do sistema PRÓXIMO é:

```
%PRÓXIMO = BUSCAPARAMETRO
           PESQUISA PRÓXIMO (JANELA)
```

Por sua vez, o parâmetro PERÍODO é uma macro do sistema

```
%PERÍODO = EMPILHAMACRO  %PONTO
           BUSCASEQUENCIACOM
```

Na execução, sucede o seguinte:

- Ao buscar o comando PRÓXIMO, o primeiro microcomando é BUSCAPARAMETRO.
- O parâmetro buscado é PERÍODO, sendo que seus microcomandos são empilhados.
- A macro %PONTO é empilhada no topo da pilha de dados
- Ao buscar uma seqüência, o topo da pilha é o primeiro elemento a ser analisado. O conteúdo da ma-

cro %PONTO, sem os delimitadores, é transferido para a área de macros da sentinela.

- O topo da pilha de comandos é desempilhado, e o próximo microcomando a ser executado é PESQUISA PRÓXIMO, executado pela janela.

4 IMPLEMENTAÇÃO

4.1 O processador

O processador M6800 possui algumas características importantes para a sua programação:

- Dois registrados, A e B, de 8 bits, sobre os quais são realizadas as operações aritméticas e lógicas.
- Um registrador índice, X, de 16 bits, que serve para fins de endereçamento.
- Um registrador de pilha, SP, de 16 bits, utilizado para o manuseio da pilha do processador.
- Um contador de instruções, PC, de 16 bits, que aponta para a próxima instrução a ser executada.
- Um registrador de estado, CC, de 6 bits, onde estão os bits C (Carry), V (Overflow), Z(Zero), N (Negative), I (Interrupt Mask) e H (Half-carry).

Para referenciar a memória existem:

- Modo imediato onde o operando segue imediatamente o código de operação.
- Modo indexado, onde o byte após o código de operação é somado ao registrador X, formando o endereço do operando.
- Modo normal, onde o(s) byte(s) após o código de operação são o endereço do operando.

Os modos direto e estendido referenciados no manual do processador, são o modo normal com 1 e 2 bytes de endereço, respectivamente. O modo relativo, utilizado para instruções de

desvio, é uma variante do modo imediato. Quando há ausência de um operando, o modo de endereçamento é denominado inerente.

4.2 Núcleo

O núcleo do sistema possui rotinas denominadas MACRO ROTINAS. Estas servem tanto para realizar as atividades simples do sistema (movimentar, indexar, limpar), como para administrar a estrutura em multi-tarefas e coordenar as atividades de E/S.

Na implementação do núcleo, alguns cuidados tiveram que ser tomados, no tocante à reentrância. Como, com o processador utilizado, é praticamente impossível escrever código reentrante sem utilizar artifícios, dois aspectos mereceram atenção especial:

- Rotinas assíncronas, ou seja, de atendimento de interrupção, só podem utilizar MACROROTINAS se:

  a)- A macrorotina é reentrante (rotinas sem variáveis locais).

  b)- A macrorotina não é reentrante, mas desabilita/habilita as interrupções.

  c)- As variáveis locais da macrorotina são empilhadas externamente antes de sua chamada, e restauradas após.

- Rotinas síncronas quando chamarem a mesma macrorotina simultaneamente, devem empilhar suas variáveis locais no início e restaurá-las no seu término.

Abaixo é exibida a relação do conjunto de MACROROTINAS disponíveis.

A, B e X são os registradores da UCP (E) e (S) simbolizam argumentos de entrada e saída, respectivamente.

01	MOVERF	Transfere B(E) caracteres de ORIGEM (E) para DESTINO (F). Após a execução, ORIGEM e DESTINO apontam para a próxima posição a ser movida. O sentido da movimentação é do byte mais significativo para o menos significativo.
02	SOMAR	Soma BGA(E) em X(E,S). Esta operação é realizada em 16 bits, considerando B o byte mais significativo.
03	MULTIPLICAR	Multiplica A(E) por B(E), armazenando o resultado em BGA(S). A operação é feita em 8 bits sendo o resultado 16 bits com sinal. O byte mais significativo é o registrador B.
04	INDEXAR	Dados uma base X(E), o número do elemento A(E) e o tamanho do elemento B(E), retorna o endereço do elemento em X(S).
05	LIMPAR	Transfere o caracter A(E) para B(E) bytes a partir de X(E).
06	SALVATUDO	Esta rotina empilha o estado atual da UCP. Se é chamada durante a execução de uma subrotina, o retorno deverá ser realizado pela instrução RTI (Retorno de interrupção) ao invés de RTS (Retorno de subrotina).
07	DESABILITAR	Desabilita a interrupção de número A(E). As restantes podem continuar interrompendo, desde que habilitadas.
08	HABILITAR	Habilita a interrupção de número A(E).
09	LIGABIT	Liga o bit A(E) em B(S). São apenas considerados os 3 bits menos significativos de A.
10	ATIVAR	Ativa a tarefa de número A(E).
11	DESATIVAR	Desativa a tarefa em execução.
12	PRÓXIMO	Entrega o controle à próxima tarefa ativa.
13	CARGA	Carrega a rotina X(E).
14	EXECUTA	Executa uma determinada rotina, após esta ter sido carregada.
15	DESCARGA	Libera a área ocupada pela rotina A(E).
16	EMPILHA	Empilha B(E) bytes a partir de X(E).
17	DESEMPILHA	Desempilha B(E) bytes a partir de X(E).
18	INCREMENTASETOR	Incrementa a trilha B(E,S), setor A(E,S).
19	PEDIDOES	Inicia a operação de E/S descrita pela área de memória apontada por X(E).
20	ERRO	Exibe a mensagem de número A(E) e aguarda o operador acusar o recebimento.
21	ESATENDIDO	Armazena B(E) no A(E) -ésimo controle do pedido de E/S.
22	MOVERT	Transfere B(E) caracteres de ORIGEM (F) para DESTINO (F). ORIGEM e DESTINO apontam para a próxima posição a ser movida. O sentido da movimentação é do byte menos ao byte mais significativo.

Tabela 6 - Macrorotinas

A seguir são colocadas algumas observações importantes referentes à implementação das MACROROTINAS:

a)- INDEXAR

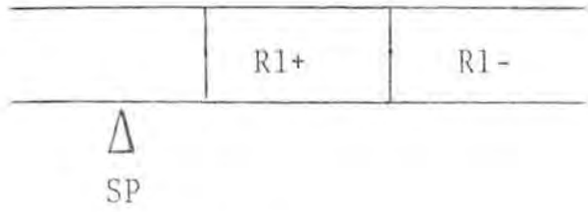
Esta rotina pode ser chamada várias vezes, conseguindo-se com isto vetores multidimensionais. Supondo que se deseja acessar o elemento I,J,K de um arranjo MxNxP, cujos elementos tenham o comprimento L. Se X iniciar em BASE, a seqüência abaixo realizará o acesso ao elemento desejado, desde que L*N*P < 256:

```
*                          PRIMEIRO NÍVEL
    LDA   A   I
    LDA   B   L*N*P
    LDX       #BASE
    JSR         INDEXAR
*                              SEGUNDO NÍVEL
    LDA   A   J
    LDA   B   L*P
    JSR       INDEXAR
*                                  TERCEIRO NÍVEL
    LDA   A   K
    LDA   B   L
    JSR       INDEXAR
```

b)- SALVATUDO

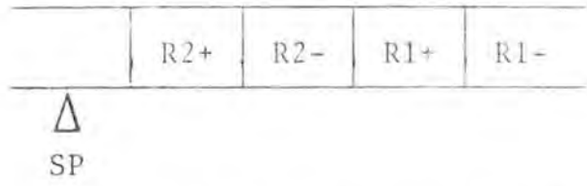
Esta MACROROTINA deverá ser sempre chamada de dentro de uma subrotina. Ao entrar em uma subrotina, o estado da pilha

é:

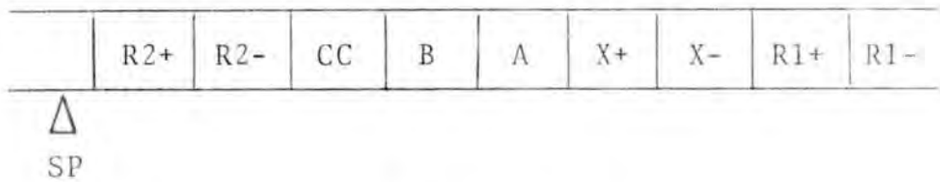


onde, + é o byte mais significativo do endereço, e - o byte menos significativo, e R o endereço de retorno.

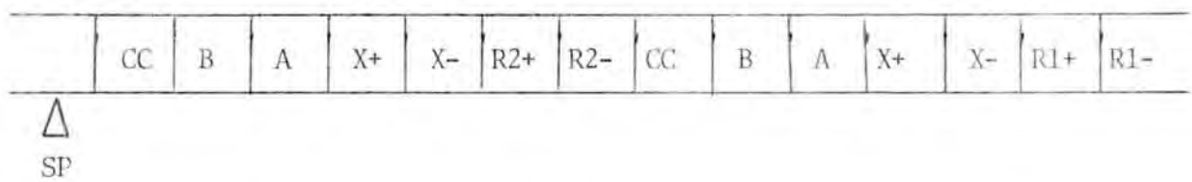
Ao chamar SALVATUDO, a pilha fica:



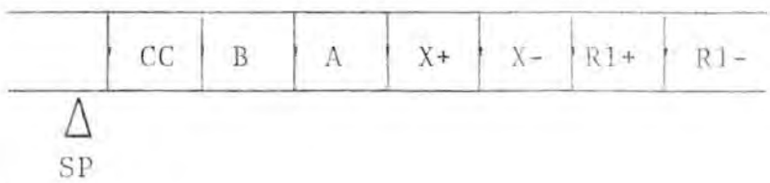
A subrotina empilha o estado atual da UCP e empilha R2 no lugar correto.



Logo após, os registradores são copiados:



Uma instrução RTI causa, agora, o retorno à primeira instrução após o JSR SALVATUDO. A pilha permanece com



Se for executada uma instrução RTI, o programa segue a execução no ponto em que foi chamada a primeira subrotina.

Para o programador de sistema, esta rotina é um recurso muito útil, pois permite a chamada de subrotinas sem se preocupar com a alteração do conteúdo de seus registradores.

c)- HABILITA

Esta rotina, ao habilitar a máscara de interrupção, permitiria o ocasionamento imediato de uma interrupção. Isto poderia, eventualmente, causar a execução de uma rotina antes de retornar da MACROROTINA.

Pelo fato de esta rotina ser utilizada por um atendimento de interrupção e, conseqüentemente, não poder ser interrompida novamente, esta rotina liga/desliga o bit que impede a interrupção (bit I). O retorno ao seu estado anterior ocorre de forma indireta ao retornar da interrupção e retirar o conteúdo do registrador CC da pilha.

d)- PRÓXIMO

Aqui será exposto, em detalhes, o chaveamento do processador de uma tarefa para outra.

A variável ATIVOS contém o estado de cada uma das tarefas: se o bit estiver ligado, a tarefa está ativa. Se, por exemplo, contiver o valor hexadecimal 81, as tarefas 7 e 0 estão ativas, as restantes estão inativas.

A variável NUMERODATAREFA contém o número da tarefa

que, de momento, está em execução.

A tabela ESTADOUCP contém o estado do processador em cada uma das tarefas, ou seja, o conteúdo de todos os seus registradores.

Ao chamar a subrotina PRÓXIMO, acontece o seguinte:

- O estado atual da UCP é empilhado (SALVATUDO).
- O elemento NUMERODATAREFA da tabela ESTADOUCP é indexado.
- O conteúdo dos registradores é copiado da pilha para o elemento junto com SP.
- É pesquisado o próximo bit correspondente a uma tarefa ativa.
- O número desta tarefa é armazenado em NUMERODATAREFA.
- A tarefa é indexada na tabela ESTADOUCP.
- O SP é carregado.
- O estado dos registradores é empilhado.
- O controle do processador é entregue à tarefa por um RTI.

O fato de armazenar SP traz como conseqüência que precisam existir tantas pilhas quanto tarefas. Embora parecendo ser uma desvantagem, isto facilita a programação do sistema: de qualquer ponto de uma tarefa pode ser chamada a subrotina PRÓXIMO, até mesmo de dentro de uma subrotina. Se, entretanto, não houver necessidade de chamar esta rotina de dentro de uma subrotina, várias tarefas podem compartilhar a mesma pilha, bastando que o registrador SP seja inicializado com o mesmo valor para as diversas tarefas.

e)- CARGA, EXECUTA, DESCARGA

Partindo do conteúdo da tabela ROTINAS, a rotina CARGA copia a rotina solicitada do disquete do sistema para a memória. Cada elemento desta tabela contém:

ENDEREÇO DISCO		TAMANHO	PARAMETROS	NÚMERO DE VARIAVEIS	ENDEREÇO	PRESENÇA
TRILHA	SETOR					

Uma segunda tabela, MEMÓRIA, contém o número e os endereços inicial e final de cada rotina:

NÚMERO DA ROTINA	ENDEREÇO INICIAL	ENDEREÇO FINAL

Ao iniciar a CARGA, é verificado em primeiro lugar se a rotina já não se encontra presente (PRESENÇA 0). A seguir é verificado se a carga não afeta nenhuma das outras rotinas presentes no sistema. Se, nesta verificação, for encontrado um lugar vago, o número da rotina é armazenado, junto com os seus endereços, inicial e final.

Após constatar que não existem problemas, a rotina solicitada é copiada do disquete e tornada presente (PRESENCA recebe o número do elemento da tabela MEMÓRIA).

A DESCARGA de uma rotina consiste em verificar se a subrotina está presente, torná-la ausente (PRESENÇA é zerada) e remover sua entrada da tabela MEMÓRIA.

- A rotina EXECUTA se encarrega de realizar a execução

de uma subrotina, incluíndo todo o serviço de transferência de parâmetros de entrada e saída.

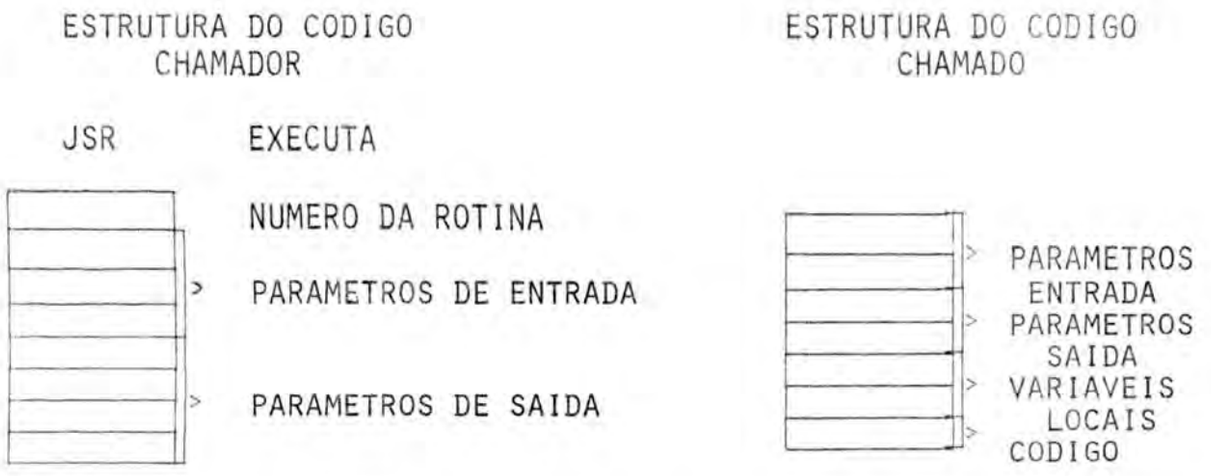


FIGURA 14 - FORMATO DO CÓDIGO DE USO DE SUBROTINAS

Conforme a figura 14 , o código chamador indica, logo após o JSR EXECUTA, a subrotina a ser executada. A rotina EXECUTA se encarrega de realizar a transferência dos parâmetros de entrada, transferir o controle à subrotina e, após a sua conclusão, transferir os parâmetros de saída da subrotina para o código chamador.

A rotina EXECUTA é reentrante, o que permite que uma subrotina executada chame outra. Se, por algum motivo, uma subrotina chamada executar a si própria, a reentrância do código fica por conta do programador do sistema.

f)- EMPILHA, DESEMPILHA

Estas rotinas são muito úteis quando se deseja escrever código reentrante. Basta o programador do sistema empilhar todas as variáveis no início da execução de uma rotina e desempilhá-las no final.

g)- PEDIDOES, ESATENDIDO

PEDIDOES inicia uma operação de E/S, apontada pelo registrador X.

O formato é o seguinte:

DISPOSITIVO	SENTIDO	NUMERO DE BYTES	ENDEREÇO DA MEMÓRIA	ENDEREÇO DE PERIFÉRICO	CONTROLE

O DISPOSITIVO pode ser: 0 - DISCO
1 - TERMINAL
2 - IMPRESSORA

O SENTIDO pode ser: 0 - ENTRADA
1 - SAÍDA

O NUMERO DE BYTES indica quantos bytes devem ser transferidos na operação de E/S.

O ENDEREÇO DE MEMÓRIA indica em que endereço estão/ficarão as informações.

O ENDEREÇO DE PERIFÉRICO indica qual o endereço dentro do periférico em que deverá ser lida/escrita a informação.

O CONTROLE fornece dados adicionais sobre a operação em si.

Maiores detalhes sobre a utilização de cada um destes dados podem ser encontrados nas rotinas específicas de E/S dos periféricos.

A rotina PEDIDOES, tomando por base o "programa de E/S" descrito por X, aciona a rotina solicitada e guarda na tabela PARES o endereço do descritor da operação.

Depois de concluída a operação de E/S, com ou sem sucesso, a rotina ESATENDIDO transfere o resultado da operação para a variável de controle.

h)- ERRO

Esta rotina é utilizada para acusar erros em geral, tanto de sistema, como de operação. Os erros de sistema, como por exemplo, a ativação de um dispositivo ativo (sobrepondo comandos), não devem acontecer na prática. Entretanto, facilitam ao programador do sistema a implementação e depuração de suas rotinas.

4.3 Rotinas de E/S

4.3.1 Descrição geral

Cada dispositivo de E/S possui sua própria rotina de controle. A lógica de controle, de maneira geral, pode ser descrita pelo fluxograma da figura 16 .

FIGURA 16 - FLUXOGRAMA GERAL DAS ROTINAS DE E/S

A inicialização é realizada da seguinte forma:

- A rotina é chamada em seu ponto inicial, por JMP.
- O número da tarefa é colocada na variável NUMERODA TAREFA.
- É chamada a primitiva PRÓXIMO.

PRÓXIMO, por sua vez, "entende" que a rotina em execução é aquela cujo número esta em NUMERODATAREFA. O estado atual da UCP é, portanto,armazenado na tabela ESTADOUCP. Com isto, ao ser ATIVAda pela próxima vez, a rotina de E/S entra em seu laço permamente de controle de operação.

A transferência de parâmetros transmite as informações necessárias às variáveis específicas da rotina. Pode também executar alguma tarefa inicial comandada pela variável CONTROLE.

A seleção da rotina a executar é realizada nos casos em que mais de um tipo de serviço é oferecido pela rotina de E/S. Após a seleção da rotina adequada e a transferência do controle para ela, esta atua o dispositivo de E/S.

O passo seguinte é a espera pelo fim da E/S comandada. Consiste de um teste da variável de resultado, verificando se o bit de "operação pronta" está ligado. Após a execução da primitiva PRÓXIMO, ou a espera prossegue, ou então a operação de E/S é encerrada.

4.3.2 Terminal

O terminal possui uma característica particular que o diferencia dos outros periféricos: a sua operação de saída en-

volve apenas o vídeo, cuja memória local precisa ser carregada; a operação de entrada, por sua vez, envolve o teclado e o vídeo, pois precisa ser dado o eco dos caracteres digitados para que o operador possa ver o que digitou.

As variáveis parâmetros na operação sobre o terminal são:

SENTIDO - define se a operação é sobre o teclado (E) ou o vídeo (S).

NUMERO DE BYTES - contém o número de caracteres a transferir na operação de E/S.

ENDEREÇO DE MEMORIA - aponta para a área de memória que deverá ser lida/escrita.

ENDEREÇO DE PERIFÉRICOS - posição do cursor na tela em que será dado eco ou escrito o primeiro caracter.

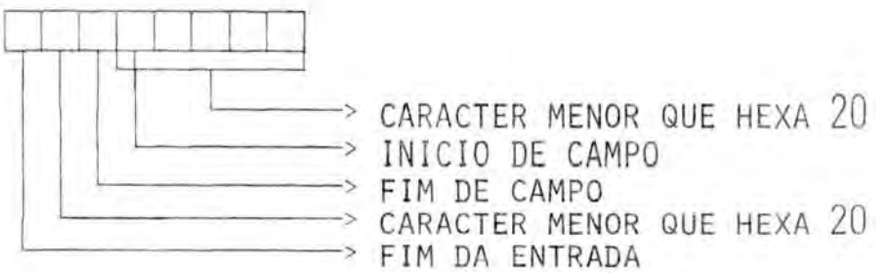
CONTROLE - possui os seguintes leiautes:

O bit 7 determina se a área a ser lida deve ou não ser limpada. Se for especificado 1, a área que deverá ser digitada será exposta com brancos ao operador. Em caso contrário, será exposto aquilo que estiver contido naquela área.

O bit 6 determina se, ao esgotar o campo previsto, o cursor retorna à primeira posição ou se é encerrada a operação.

O bit 2 determina se, após a exibição do texto, o cursor deve aparecer na tela ou não.

RESULTADO - possui os seguintes leiautes:

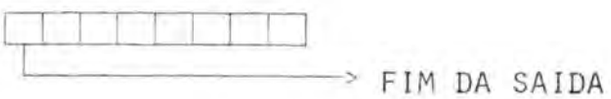


O bit 7 informa que a operação terminou.

O bit 6 informa que foi digitado um caracter inferior ao hexadecimal 20, e este caracter não pôde ser processado pela rotina de atendimento. O caracter se encontra nos bits 4-0.

O bit 4 informa que houve tentativa de passar do início do campo.

O bit 3 informa que houve tentativa de passar do fim do campo.



O bit 7 informa que a operação de saída terminou.

4.3.3 Disquete

A rotina de controle do disquete realiza a parte de E/S física sobre o disco. O processamento lógico, usando arquivos, volumes e encadeamento de registros, é realizado fora desta rotina.

As variáveis parâmetros na operação sobre o disquete são:

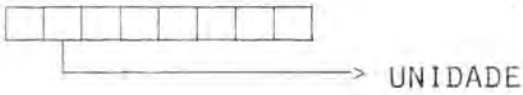
SENTIDO - define se a operação é de leitura ou de escrita.

NUMERO DE BYTES - define a quantidade de bytes a serem transferidos na operação de E/S.

ENDEREÇO DE MEMÓRIA - aponta para o endereço da memória onde estão os dados que serão lidos/gravados pela operação de E/S.

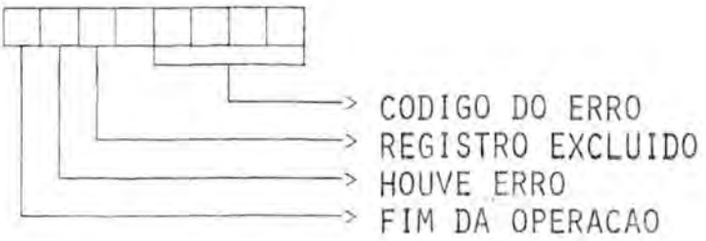
ENDEREÇO DO PERIFÉRICO - contém a trilha e o setor sobre o qual será realizada a operação de E/S.

CONTROLE - possui o seguinte leiaute:



O bit 6 determina sobre qual unidade de disquete será realizada a operação.

RESULTADO - possui o seguinte leiaute:



O bit 7 informa que a operação de E/S terminou.

O bit 6 informa se ocorreu um erro na operação. Em caso afirmativo, o código do erro está contido nos 4 bits menos significativos.

O bit 5 informa se o registro lido possuía a marca de registro excluído ou não.

4.3.4 Impressora

A impressora, no ED-100, é ligada por um interface série. A impressora ligada no protótipo não transmite nenhuma informação sobre o seu estado. Apenas dois sinais estão disponíveis para serem testadas por software:

O sinal PRONTO, que avisa que a impressora está pronta e à disposição do sistema.

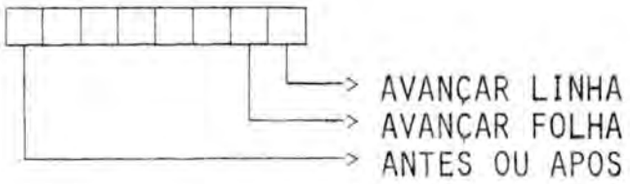
O sinal CHEIO, que avisa que o "buffer" de caracteres da impressora está cheio e que deve ser aguardado o seu esvaziamento.

Os parâmetros para a operação da impressora são:

NÚMERO DE BYTES - Define quantos bytes devem ser impressos.

ENDEREÇO DE MEMÓRIA - Aponta para o endereço de memória onde se encontram os dados.

CONTROLE - Possui o seguinte leiaute:

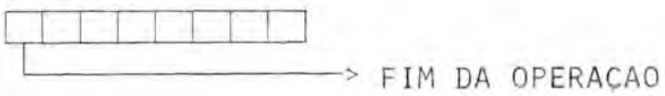


O bit 7 indica se o avanço de linha/folha deve ser realizado antes ou após a impressão do texto.

O bit 1 indica que deverá haver um salto de folha.

O bit 0 indica que deverá haver um salto de linha.

RESULTADO - Possui o seguinte leiaute:



A área de memória sempre deverá conter caracteres EBCDIC; a tradução para ASCII é realizada no própria rotina de impressão.

Se o programador do sistema deseja realizar uma simples movimentação da folha, basta colocar o NÚMERO DE BYTES em zero e chamar a rotina de impressão com o controle apropriado.

4.4 Macros

As rotinas que manipulam as macros podem ser subdivi
didas em três grupos:

- manipulação da estrutura lógica
- manipulação da estrutura física
- inicialização e finalização.

4.4.1 Rotinas que manipulam a estrutura lógica

São 4 as rotinas que operam sobre a estrutura lógica das macros. São elas:

a)- PESQUISA

Esta rotina procura na tabela MACROS uma macro que possua o mesmo nome que a chave. Se não for possível localizar tal elemento, retorna o número da macro que deverá ser o pai do novo elemento, bem como qual o lado em que deve ser realizada a inserção.

Os argumentos desta rotina são:

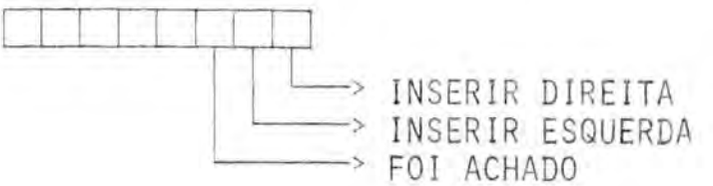
CHAVE (E) - Aponta para a chave que deve ser pesquisada.

PAI (S) - Aponta para o pai do elemento após sua localização na tabela.

NUMERO (S)- Aponta para o elemento achado na tabela, ou então para o elemento que deverá ser

o seu pai, em caso de inclusão.

RESULTADO(s)-



Os argumentos foram projetados de tal forma que, após uma pesquisa, pode ser executada qualquer uma das outras rotinas sem precisar alterar o formato dos parâmetros.

b)- INCLUSÃO

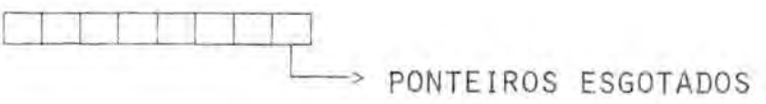
Esta rotina incorpora a sentinela à estrutura MACROS, coloca o seu ponteiro na árvore de PONTEIROS, e cria uma nova sentinela. O ponto de inclusão é obtido a partir dos dados fornecidos pela rotina PESQUISA.

Os argumentos da rotina são:

PAI (E) - Aponta para o pai do novo elemento.

LADO (E) - Contém o lado em que deve ser realizada a inclusão.

RESULTADO(S) -



c)- EXCLUSÃO

Esta rotina realiza a exclusão de uma macro. Exclui

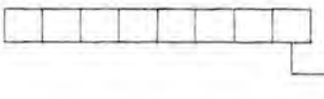
o elemento da árvore de PONTEIROS, e torna a área ocupada em MACROS disponível.

Os seus argumentos são:

PAI (E) - Aponta para o pai do elemento a ser excluído.

LADO (E) - Aponta para o lado do elemento no pai.

RESULTADO(S)



Avisa que houve violação da proteção de macros (por exemplo, tentativa de exclusão de uma macro do sistema).

d) ALTERAÇÃO

Esta rotina altera o conteúdo de uma macro para aquilo que está na sentinela. Consiste em mudar, na árvore de PONTEIROS, o elemento que aponta para a área de MACROS. A área anteriormente apontada é tornada disponível.

Seu único argumento é:

NUMERO (E) - Aponta para o número do elemento que deve ser alterado.

4.4.2 Rotinas que manipulam a estrutura física

São várias as rotinas que manipulam a estrutura física. Sua principal tarefa é a de tornar as rotinas lógicas inde-

pendentes da estrutura física. Além disto, realizam uma série de pequenas tarefas, repetidas muitas vezes durante a programação do sistema, facilitando desta forma a escrita do código.

Estas rotinas usam como parâmetros de entrada e de saída os registradores da UCP, sem utilizar a transferência de parâmetros normalmente empregada.

São elas:

NOME	FUNÇÃO
CRIASENTINELA	- Cria uma sentinela em MACROS. Retorna N em 1 se a área estiver esgotada.
TAMANHOMACRO	- Zera o tamanho do nome e da macro.
APENDICEASOMA	- Acrescenta o caracter contido em A no nome da sentinela. Retorna N em 1 se a área estiver esgotada. Retorna Z em 1 se o tamanho do nome estiver esgotado.
APENDICEAMACRO	- Acrescenta o caracter contido em A na macro. Retorna N em 1 se a área estiver esgotada. Retorna Z em 1 se o tamanho da macro estiver esgotado.
ACESSAPONT	- Retorna em X o endereço de PONTEIROS correspondente ao elemento em A.
ACESSAMACRO	- Retorna em X o endereço de MACROS correspondente ao elemento em A.
APONTAMACRO	- Retorna em X o endereço do primeiro caracter da macro apontada por X.
VERIFICAESPAÇO	- Verifica se na tabela MACROS há espaço para o número de bytes indicados em A.
COMPACTA	- Compacta a tabela MACROS. Retorna Z em 1 se não houver compactação.
EXCLUIMACRO	- Torna a macro apontada por X disponível.
INCLUIMACRO	- Transforma a sentinela em macro.
AUMENTA	- Acrescenta o conteúdo de A no final da tabela MACROS
SOMAMACRO	- Retorna em B e A o tamanho total da macro apontada por x.

Tabela 5 - Microcomandos de interpretação

Entre as rotinas acima, uma merece atenção especial: COMPACTA. Esta realiza, a partir do menor espaço disponível, a compactação da tabela MACROS. Após a sua execução, o espaço no final da tabela apresenta-se como uma área contínua de memória e não existem mais espaços disponíveis no interior da tabela.

Na rotina EXCLUIMACRO, o X é comparado com a variável MENOREXCLUSÃO. Caso for inferior, é realizada a substituição do conteúdo. Após uma série de exclusões, a variável MENOREXCLUSÃO indicará o espaço disponível com o menor endereço. Com isto a área inicial da memória não precisa passar pelo algorítmo de compactação.

4.4.3 Inicialização e finalização

Para que o usuário possa utilizar macros criadas em uma sessão anterior, estas tem que ser guardadas em disco, junto com o texto.

Por outro lado, ao iniciar, tanto o formatador como o editor precisam de um conjunto de macros. Os comandos e os parâmetros, por exemplo, são macros sem as quais o sistema não pode funcionar.

Ainda deve ser levado em conta que a programação do sistema pode mudar. Os textos editados anteriormente, porém, devem poder ser processados pela nova versão de programação do sistema.

A rotina de finalização do editor está encarregada de despejar o conteúdo da tabela MACROS para o disco, desprezando as

macros do sistema. O formato de armazenamento é idêntico ao dos dados. São criados dois conjuntos em disco, sendo um de macros e outro de marcas.

A rotina de inicialização do editor contém uma parte cuja função é de carregar todas as macros do sistema e do usuário para a memória. À medida que a carga é realizada, a árvore de PONTEIROS é inicializada.

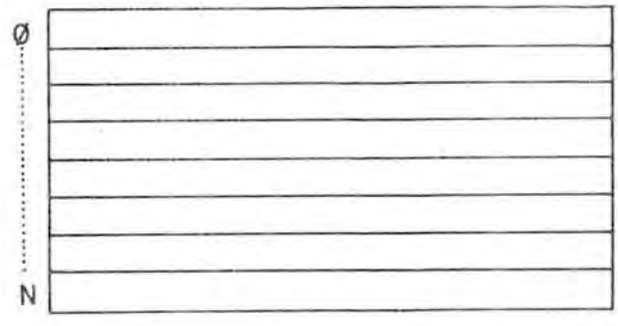
4.5 Janela

A janela, conforme o anteriormente descrito, é um conjunto de buffers sobre o qual é realizado o processamento do texto. O seu conteúdo, da mesma forma que os setores em disco correspondentes, é um conjunto encadeado de segmentos de texto. Para se reconstruir o texto em sí, é preciso percorrer as cadeias, juntando os seus pedaços em um trecho único.

Seguem-se alguns aspectos da sua implementação.

4.5.1 Estrutura

A estrutura básica é formada por uma tabela, BUFFERS. Cada elemento desta tabela contém a imagem dos setores em disco.



Uma outra tabela, ATRIBUTOS, mantém os atributos de cada setor:

ENDEREÇO	ESTADO	PRÓXIMO	ULTIMO

O ENDEREÇO aponta para o endereço do setor em disquete (trilha lógica, setor). O ESTADO contém a informação se a imagem deste setor está gravada em disquete ou não. O PRÓXIMO e o

ÚLTIMO são os elos de uma fila duplamente encadeada.

Existem, ainda, alguns ponteiros especiais, que são:

- PRIMEIRO, que aponta para o primeiro elemento da fila,
- ÚLTIMO, que aponta para o último elemento da fila.
- INICIOT, que aponta para o par (buffer, deslocamento) onde se encontra o primeiro byte exibido na tela,
- FIMT, que aponta para o par (buffer, deslocamento) onde se encontra o último byte exibido na tela,
- CURSOR, que aponta para o par (buffer, deslocamento) correspondente ao caracter apontado pelo cursor na tela,
- CURSORT, que contém a posição física do cursor na tela.

4.5.2 Rotinas que manipulam a janela, logicamente

Pelo fato de a janela ser um elemento que pode ser percorrido nos dois sentidos, houve a necessidade de implementar certas rotinas duas vezes, porém com sentidos contrários. Como as operações nos dois sentidos são essencialmente simétricas, será descrito apenas o que ocorre em um dos casos - o sentido positivo, isto é, do início ao fim do arquivo.

As operações usam, como parâmetro, os dados contidos na sentinela, visto que nesta área foi realizada a análise sintática do comando. Em uma das operações é necessário um parâmetro

adicional.

As operações lógicas sobre a janela são:

a)- AVANÇARCURSOR (X,Y)

Avança o cursor X posições sobre o texto. Caso encontre o fim da tela, CURSORT é recuado de Y posições, ao mesmo tempo que INICIOT e FIMT são avançados por Y posições; esta operação é percebida pelo operador como "texto subindo na tela".

b)- EXIBIRTELA

Exibe toda a tela, do cursor para trás, do cursor para frente, ou apenas posiciona o cursor na tela. Uma operação especial, CENTRAR, coloca o cursor no centro da tela, para que o operador possa visualizar um trecho antes e outro após o cursor.

c)- INCLUIR

Inclui um texto na posição apontada pelo cursor. Se o espaço disponível no buffer do cursor não comportar a inserção do texto, os vizinhos são consultados para ver se há espaço disponível. Se houver, ocorre um deslocamento de textos; em caso contrário, novo buffer precisa ser alocado para poder conter os dados incluídos e/ou deslocados.

d)- EXCLUIR

Exclui um texto da posição apontada pelo cursor. Se o espaço que se tornar disponível no buffer do cursor comportar o texto do próximo, existe uma união dos dois buffers e a liberação de um deles.

e)- ALTERAR

Altera um texto a partir da posição apontada pelo cursor.

f)- FECHAR

Se tiverem que ser fechadas ambas as alas da janela, consiste em copiar o conteúdo dos buffers para o disco e marcar as alas como fechadas. Se tiver que ser fechada apenas uma das alas, o buffer, na posição em que se encontra o cursor, é "quebrado" em duas partes, de tal forma, que o cursor se encontre na primeira posição de um buffer.

g)- ABRIR

Abre a ala da janela, copiando o conteúdo dos setores para os buffers na memória.

h)- PESQUISAR

Consiste em avançar o cursor até encontrar ou um certo conjunto de caracteres no texto ou o fim do arquivo.

i)- DESLOCARATEMARCA

Avança o cursor até encontrar uma certa marca no texto.

j)- POSICIONAR

Consiste em fechar a janela, posicionar o cursor na marca apropriada, abrir a ala esquerda e a ala direita na nova posição.

4.5.3 Rotinas que manipulam a janela, fisicamente

Estas rotinas são chamadas pelas operações lógicas sobre a janela e visam dar facilidade de programação ao sistema.

São elas:

a)- QUEBRANOCURSOR

Realiza a quebra do buffer, no ponto em que se encontra o cursor; esta operação sempre deixa o cursor na posição 1 de um buffer.

b)- DESLOCAESQUERDA

Desloca o texto apontado pelo cursor um certo número de bytes para a esquerda (exclusão). Também é realizado todo o gerenciamento no caso de a operação resultar em um fluxo de informações para fora do buffer.

c)- DESLOCADIREITA

Desloca o texto apontado pelo cursor um certo número de bytes para a direita (inclusão). Também é realizado todo o gerenciamento no caso de a operação resultar em um fluxo de informações para fora do buffer.

d)- BALANCEADOR

Esta rotina está encarregada de manter o balanceamento da estrutura de buffers. Realiza a gravação dos buffers que já foram processados, e se encarrega de ler novos setores à medida que o fim da tela se aproxima do fim dos dados.

4.5.4 Marcas

As marcas precisam ser atualizadas sempre que houver uma alteração no contexto que estiverem apontando. Isto acontece em três condições:

a)- um setor marcado é lido

b)- um texto marcado sofre deslocamento

c)- um setor marcado é gravado

As situações a) e c) são controladas pelo balanceador. Sempre que um setor é lido ou gravado, a tabela de marcas é percorrida, verificando se há alguma marca apontando para este setor. Em caso afirmativo, a marca é atualizada.

A situação b) só ocorre nas rotinas de deslocamento de texto. As rotinas DESLOCADIREITA e DESLOCAESQUERDA, portanto, verificam se no trecho deslocado não existe uma posição sendo apontada por uma marca. Em caso afirmativo, a marca é relocada.

4.6 Interpretação

A filosofia geral da interpretação dos comandos digitados pelo operador já foi exposta na seção 3.5. Na presente seção serão abordados alguns aspectos práticos da implementação das rotinas.

4.6.1 Rotinas de busca

As rotinas de busca (BUSCASEQUÊNCIACOM, BUSCASEQUÊNCIASEM, BUSCASOLUÇÃO, etc) percorrem o texto à procura de um certo elemento sintático. Pode acontecer que, durante a busca apareça um erro sintático ou um fim precoce dos dados.

Por isso todas as rotinas de busca retornam um código que relata o que aconteceu. O SPT, quando detecta um erro na sintaxe, interrompe o processo e avisa o operador de que ocorreu uma falha.

Internamente, todas as rotinas utilizam a rotina de obtenção de bytes, que é a que devolve um byte do texto, resolvendo as referências a macros. Um estudo mais detalhado desta rotina será feito na próxima seção.

4.6.2 Lógica de obtenção de byte

No SPT foi adotado como filosofia de trabalho que um byte não está mais acessível ao sistema depois que foi processa-

do. Como em qualquer ponto do texto pode ocorrer uma referência à macro, esta metodologia simplificou a análise sintática e/ou execução de comandos.

Existem duas rotinas importantes para o processo de obtenção de bytes: BUSCABYTE e AVANÇABYTE. A primeira tem como função retornar o byte que está sendo apontado pelo topo da pilha de macros. Se esta rotina for chamada várias vezes, sempre retornará o mesmo caracter. A segunda rotina avança para o próximo caracter, e, se este não existir, desempilha até encontrar uma macro não totalmente utilizada, ou então o fim da pilha.

Enquanto a rotina AVANÇABYTE possui uma lógica simples, a rotina BUSCABYTE possui uma lógica mais complicada. Pode ser descrita da seguinte forma:

- Pega o byte apontado pelo topo da pilha;
- Se o modo for transparente, retorna;
- Se o caracter não for "!", retorna;
- Chama as rotinas BUSCABYTE e AVANÇABYTE, formando o nome da macro;
- Pesquisa o nome na tabela de MACROS;
- Empilha os ponteiros para a macro no topo da pilha;
- Chama a rotina BUSCABYTE.
- Retorna;

Não estão representados todos os erros que podem ocorrer durante a execução desta rotina. Importante é ressaltar que a rotina é executada de forma recursiva, o que permite resolver expressões tais como, por exemplo:

!ABC!D

onde primeiro !D é substituído por seu conteúdo, e

após !ABCXXX, onde XXX representa o conteúdo de !D.

Após buscar todos os caracteres que estiverem na pilha de dados, o editor passa automaticamente ao modo de dados.

O formatador, após esvaziar a pilha de dados, realiza a leitura do próximo setor e começa a processá-lo; caso não existirem mais dados, encerra suas atividades.

4.6.3 Exemplo

Como exemplo do exposto acima será vista a redução do comando:

```
REPITA <exp a> <seqüência>;
Existem as seguintes macros do sistema:
%SOBEPILHA = ATRIBUA %PILHA !%PILHA+1;
%CRIAE1    = CRIE %E1!%PILHA/0/;ATRIBUA %E1!%PILHA
%CRIAS1    = CRIE %S1!%PILHA
%MREPITA   = ENQUANTO %E1!%PILHA>0 /%S1!PILHA
               ATRIBUA %E1!%PILHA %E1!%PILHA-1;/
               DESTRUA %S1!%PILHA;
               DESTRUA %E1!%PILHA;
               ATRIBUA %PILHA !%PILHA-1;
%REPITA    = EMPILHA MACRO %SOBEPILHA          (a)
               BUSCA COMANDO                   (b)
               EMPILHA MACRO %CRIAE1           (c)
               BUSCA COMANDO                   (d)
               EMPILHA MACRO %CRIAS1           (e)
               BUSCA COMANDO                   (f)
```

EMPILHA MACRO %MREPITA (g)

BUSCA COMANDO (h)

Quando o interpretador inicia a execução do comando REPITA, busca, em primeiro lugar, um comando e empilha este no topo da pilha de comandos. Ao analisar o próximo microcomando, encontra (a). Este causa o empilhamento da macro %SOBEPILHA no topo da pilha de dados. O próximo microcomando, (b), busca um novo comando, que é "ATRIBUA %PILHA !%PILHA+1;". Este comando é executado. O próximo microcomando, (c), empilha a macro %CRIAE1. O microcomando (d) executa o comando "CRIE %E1!%PILHA /0/; ATRIBUA %E1!%PILHA <exp a>",onde a <exp a> é a original do comando REPITA. O microcomando (e) empilha a macro %CRIAS1. (f) executa o comando "CRIE %S1!%PILHA <seqüência>;",onde a <seqüência> é a original do comando REPITA. O microcomando (g) empilha a macro %MREPITA, executada pelo microcomando (h). Assim os microcomandos são executados e a redução final é alcançada.

É importante destacar a função da variável PILHA. No comando REPITA, a <seqüência> pode representar qualquer conjunto de comandos, incluíndo o próprio REPITA. Como este é reduzido para um comando ENQUANTO com um contador de repetições, se não fosse empregada esta técnica haveria dificuldade para diferenciar os contadores.

4.7 Editor

Todas as rotinas de edição,E/S e de interrupção necessárias para o funcionamento do editor foram explicadas em seções anteriores.

Ao iniciar a sua execução, o editor entra na fase de solicitação de parâmetros. Obtidos estes parâmetros, são abertos os arquivos de entrada (se este existir) e de saída. As estruturas de MACROS são inicializadas com as macros do sistema.

Após esta inicialização, o editor entra na fase de processamento de comandos. Esta consiste em aguardar as ações do operador:

- Se o editor estiver exibindo a tela de dados, os caracteres digitados são analisados e transformados em uma ação sobre a janela, ou num comando interno, diretamente.
- Se o editor estiver exibindo a tela de comandos, é aguardado o término da digitação dos comandos, que, então, são analisados e executados.

Se o operador digitar alguma tecla durante o processamento contínuo dos comandos, significa que o processamento contínuo deve ser interrompido. Para que a interrupção não ocorra acidentalmente, a condiçao de término é a tecla ANULA, juntamente com as duas chaves, INFERIOR e SUPERIOR. A interrupção pode ser necessária se, por algum motivo, o editor entrar em um "laço sem fim" (por exemplo, executando um comando ENQUANTO que não possui término).

Após detectar o comando de fim do editor, é realizado o procedimento final, gravando as macros e as marcas em dis-

co, e atualizando o setor 1 e o rótulo.

4.8 Formatador

Na maior parte o formatador é composto pelos mesmos módulos que constituem o editor.

A grande diferença está na rotina de saída. Onde o editor simplesmente grava em disco, o formatador formata os dados e os dispõe sobre o papel.

A rotina de saída possui muitos pontos que dependem do equipamento impressor: se tem ou não maiúsculas/minúsculas, subscritos e superscritos, espacejamento horizontal e vertical variável, grifo, sublinha, etc.

No ED-100 está disponível uma impressora serial, que forma os caracteres a partir de uma matriz de pontos. Das características enumeradas acima, só possui caracteres maiúsculos e minúsculos.

4.8.1 Macros especiais

O formatador possui quatro grupos de macros especiais, que tem uma certa finalidade na impressão:

- variáveis, que contam uma certa ocorrência,
- textos, que são impressos em certos instantes,
- tabulações, que formam um conjunto de colunas nas quais se deseja tabular,

- parâmetros de diagramação, que definem o leiaute do texto sobre a folha.

A seguir as macros de cada grupo serão apresentadas com mais detalhes.

a)- Variáveis

%NUMPAG - Contém o número da página que está sendo impressa.

%NIVELCAP - Contém o nível hierárquico do capítulo que está sendo impresso. A cada comando INÍCIO CAPÍTULO, esta variável é incrementada. A cada comando FIM CAPÍTULO é decrementada.

%NUMCAPn - Contém o número do capítulo de nível n= %NIVELCAP. No presente ponto do texto, por exemplo, %NUMCAP0 conteria 4, %NUMCAP1 conteria 8 e %NUMCAP2 conteria 1. Esta variável é inicializada em 0 a cada comando INÍCIO CAPÍTULO no nível superior.

%CONTLIN - Contém o número de linhas impressas na página atual.

b)- Textos

%CABEÇALHO- É o texto impresso no início de cada folha.

%RODAPÉ - É o texto a ser impresso no final de ca-

da folha.

%INICIOCAP- É o texto impresso no início de um capítulo.

%PARAGRAFO- É a formatação que deve ser dada ao início de um novo parágrafo.

c)- Tabulação

%TAB - Contém as colunas em que deve ser realizada tabulação, separadas por vírgula.

d)- Diagramação

%TAMLIN - Contém o número de posições úteis por linha.

%TAMPAG - Contém o número de linhas úteis por folha .

%MARGEMS - Contém o número de linhas em branco do topo da página até o cabeçalho.

%MARGEMI - Contém o número de linhas em branco do rodapé até o início da nova página.

%MARGEME - Contém o número de brancos na margem esquerda.

4.8.2 Funcionamento

O formatador baseia o seu funcionamento em uma área

que possui o dobro do tamanho da linha.

Esta área é preenchida com bytes até uma das condi - ções ocorrer:

- comando de quebra de linha
- esgotamento da primeira parte.

No segundo caso, a palavra que causou o esgotamento da linha é deslocada para a segunda parte da área, sendo substituída por brancos. Em ambos os casos, a linha é impressa. Após a impressão, o conteúdo da segunda área é deslocada para dentro da primeira.

Existem três níveis de processamento de textos, no formatador:

a)- Processamento a nível de caracter

Este é o responsável pelo controle das letras maiúsculas e minúsculas, bem como controle do preenchimento da área a imprimir.

b)- Processamento a nível de linha

Realiza o processamento da linha, realizando os espa cejamentos e alinhamentos necessários para a impressão da linha.

c)- Processamento a nível de página

Realiza o controle e o processamento de rodapés e cabeçalhos.

4.8.3 Processamento do texto

Os comandos de formatação estão embutidos no texto.À medida que este é processado, as referências a macros são resolvidas. A rotina de processamento de caracter recebe o texto byte por byte , não sabendo em que nível de referência se encontram as macros.

Todas as referências a macros devem estar encerradas por um delimitador nulo, representado no texto como o hexadecimal 81. Este byte é filtrado pela rotina de processamento do caracter. Isto evita que o delimitador seja considerado um elemento do texto.

Supondo, por exemplo o texto:

UTILIZADO NO !SPT.

O ponto não é considerado como sendo do texto, mas sim, como sendo do nome da macro. Para evitar este fato,deve ser escrito

UTILIZADO NO !SPT;.

Os comandos devem ser prefixados por ! e %- o primeiro byte é necessário para avisar de que se trata de uma macro; o segundo byte distingue a macro como sendo do sistema.

Ao ser encontrada em um texto, a macro é resolvida.A

rotina de obtenção de bytes, por sua vez, identifica que esta macro é do tipo "comando". Ao invés de passar o caracter à rotina de processamento de caracteres, o topo da pilha de dados é passado para a pilha de comandos, onde o interpretador inicia a execução os microcomandos.

Após concluir a execução, o texto volta a ser processado pela rotina de processamento de caracteres.

4.8.4 Exemplo

Como exemplos serão citados dois casos práticos:

- cabeçalhos diferentes par e ímpar,
- numeração da seção.

a)- Cabeçalhos diferentes em página par e ímpar

Supondo que nas páginas pares deve constar o nome do capítulo, e no lado ímpar o nome da seção, isto poderia ser resolvido pelos seguintes comandos.

```
!%ALTERE %CABEÇALHO:1!%NOVA LINHA;
                      !%NOVA LINHA;
                      !%INICIO ALINHAMENTO ESQUERDA;. .;
                      !%SE !%NUMPAG/2*2=!%NUMPAG
                            :2!CAPITULOØ:2;
                      !%SE !%NUMPAG/2*2≠!%NUMPAG
                            :2!CAPITULO!%NIVELCAP:2;
                      !%FIM ALINHAMENTO;
```

```
                    :%INICIO ALINHAMENTO DIREITA . .;
                     !%NUMPAG;
                    :%FIM ALINHAMENTO;
                    :%NOVA LINHA;
                    :1
```

Após este ponto, cada referência a %CABEÇALHO, realizada na impressão da página, imprimirá o cabeçalho par/ímpar. As macros CAPITULOn, onde n é o nível do capítulo/seção, devem conter o nome da seção. Isto pode ser conseguido por um comando

```
%ALTERE CAPÍTULO!%NIVELCAP /EXEMPLO/;
```

b)- Numeração de seção

Uma seção deve ser numerada e possuir um nome. Supondo o seguinte comando:

```
!%ALTERE %INICIOCAP :1!%NOVA LINHA;
                    !%NOVA LINHA;
                    !%ATRIBUA CONT /0/;
                    !%ENQUANTO !CONT<!%NIVELCAP
                              /!%NUMCAP!%CONT;./
                    !%NUMCAP!%NIVELCAP; - :
                    :CAPITULO!%NIVELCAP;
                    !%NOVA LINHA;
                   :1
```

Sempre que for iniciada nova seção, deverão ser realizadas antes as seguintes instruções:

```
%ATRIBUA NÍVEL !%NIVELCAP+1;
%ALTERE CAPÍTULO!NÍVEL /MACROS ESPECIAIS/;
```

Ao ser realizada a referência a %INICIOCAP, sairá impresso

Linha em branco

Linha em branco

4.8.1 - Macros especiais

Linha em branco

5 CONCLUSÕES

5.1 Conclusões sobre o SPT

Neste trabalho discutiu-se uma série de recursos orientados para o processamento de textos. Algumas observações merecem ser feitas sobre alguns pontos em particular:

- As macros, dinâmicas, possuem um potencial de aplicação muito grande. Além de serem um recurso muito poderoso para o usuário do sistema, o próprio programador do sistema pode usá-las na solução de muitos problemas.

- É fácil incluir um novo comando no sistema, ou então alterar a sua execução. Também é fácil mudar a linguagem. O sistema, como foi estruturado, apresenta, portanto, facilidade de manutenção.

- A interpretação do sistema pode sofrer melhorias. Ao invés de empilhar macros, para depois interpretá-las, pode ser introduzido um processo combinado, que reúne os dois.

5.2 Futuras ampliações

A seguir estão relacionadas as ampliações que podem ser realizadas no SPT, sem alterar profundamente a sua estrutura.

a)- Arquivo de comandos

Pode ser introduzido no SPT uma ampliação que traz uma melhoria muito grande na operação do sistema: o arquivo de comandos. Confeccionado pelo próprio editor de textos, este arquivo conteria todos os comandos que precisam ser efetuados sobre o texto.

Um novo comando, por exemplo EXECUTA <arquivo> se encarregaria de executar todos os comandos contidos naquele arquivo, de forma contínua.

b)- Melhorar a linguagem usada

A linguagem empregada procura ser de fácil compreensão para alguém que não conheça o sistema. Entretanto, se o usuário, para obter a próxima seqüência desejada, precisa inevitavelmente digitar PRÓXIMA SEQUÊNCIA <seqüência>, a operação do sistema tornar-se-á cansativa.

Também para usuários acostumados a outros sistemas pode ser interessante continuar usando os antigos comandos no novo sistema, para facilitar a adaptação.

Uma das possibilidades de modificar a linguagem é por intermédio de macros. Se o usuário criar uma macro PS = PRÓ-

XIMA SEQUÊNCIA, basta digitar !PS e o sistema entenderá "PRÓXIMA SEQUÊNCIA".

Uma outra alternativa é a de usar a execução de um arquivo de comandos, como definido em a).

Uma terceira hipótese é a de alterar a linguagem no arquivo de macros do sistema.

c)- Impressão durante a edição

É uma característica importante de um equipamento de processamento de textos realizar a impressão de um texto de forma simultânea com uma edição de um novo texto. O SPT possui toda a infraestrutura de software para anexar mais tarefas ao seu processamento. Teria que ser estudado, entretanto, de que forma o operador se comunicaria com a formatação, e como poderia descontinuá-la ou comandá-la.

d)- Formatação no vídeo

É importante para o susuário poder ver o texto de forma formatada no vídeo. Isto pode servir tanto para fins de conferência como para o estudo de um melhor leiaute sobre a folha de papel, sem, entretanto, imprimí-la realmente.

e)- Periféricos para o ED-100

O ED-100, embora possua os periféricos básicos necessários ao processamento de textos, deveria ter as seguintes me-

lhorias:

- vídeo de maior capacidade
- impressora especial para processamento de textos
- teclado especial para processamento de textos.

f)- Digitação sobreposta com execução

Atualmente,o SPT após iniciar a execução de comandos não admite que novos comandos sejam digitados. O operador, por - tanto, aguarda o resultado da operação sem ter o que fazer.

Com a estrutura em multi-tarefas, não é muito difícil permitir que o operador digite simultaneamente com a execução de comandos. Se terminar a digitação antes de terminar o processamento dos comandos anteriores, a nova tela pode ser inserida no fundo da pilha; após terminar de processar a tela anterior, o interpretador passa a executar automaticamente a nova tela. Enquanto isto, o operador pode estar digitando novos comandos.

g)- Macros interativas

As macros, atualmente, só podem ser criadas, alteradas e excluídas por intermédio de um processamento contínuo. É muito útil, porém, poder realizar as alterações de forma interativa. Com isto, o usuário poderia verificar quais as macros existentes e qual o seu conteúdo. Também a atual restrição de que uma macro, para ser alterada, precisa ser totalmente redigitada, poderia ser eliminada.

h)- Diagnóstico de erros

Atualmente, ao ocorrer um erro, apenas é exibida uma mensagem e o usuário precisa descobrir o porquê do erro. Normalmente, este processo é simples e não requer muito trabalho. Entretanto, à medida que são realizadas referências a macros dentro de macros, podem surgir dificuldades de diagnóstico do erro.

Para solucionar este problema, a pilha de dados tem disponível todo o histórico de referências, inclusive o ponto em que está o processamento de cada uma. A listagem ou exibição da pilha, bem como a possibilidade de mostrar o texto tal como é visto pelo editor /formatador, facilitariam muito o diagnóstico do erro.

BIBLIOGRAFIA

1. CARLS,C.B. Getting ready for word processing's second generation. Datamation, Barrington, 24(9):139-44, Sept. 1978.

2. CUMPSTON, C. Word processing. Administrative Management,New York, 38(10):124, Oct. 1977.

3. _____. _____. Administrative Management, New York, 38(12):92, Dec. 1977.

4. EDP ANALYZER. Word Processing, part 1. Vista, V.15, n.2, Feb. 1977.

5. _____. Word processing, part 2. Vista, V.15, n.3, Mar. 1977.

6. _____. The automated office, part 1. Vista, V16, n.9, Sept 1978.

7. FITZPATRICK, J. Outlook on word processing. In: AUERBACH Data Processing Management. Pennsanken, 1980. V.3, rep.n. 6-02-04.

8. LAWSON, B. Destroying the Myths about WP. Administrative Management, New York, 38(11):57-60, Nov. 1977.

9. LIPPOLD, W.A. Whys and Hows of WP system flexibility. Datamation, Barrington, 24(9):165-6, Sept. 1978.

10. MOSER, F.B. Livro. (Não publicado).

11. POLYMAX SISTEMAS E PERIFÉRICOS. Sistema Polymax de processamento da palavra; manual de formação do operador. Porto Alegre, s.d.

12. STEMER, J.E. & REDDY, T.R. The soft display word processor. Computer, Long Beach, 11(12):39-48, Dec. 1978.

APÊNDICE I - ANATOMIA DA LINGUAGEM

SUMÁRIO

APÊNDICE I - ANATOMIA DA LINGUAGEM

SUMÁRIO ALFABÉTICO

A1 ANATOMIA DA LINGUAGEM

A1.1 Introdução

O presente apêndice tem como objetivos:

- fornecer ao usuário do SPT uma descrição pormenorizada dos comandos, com detalhes sintáticos e semânticos, e
- fornecer ao programador de sistemas os detalhes da implementação dos comandos, mostrando como os módulos já descritos interagem na execução de um comando.

Foi subdividido em quatro divisões:

a) Comandos gerais, que podem ser executados tanto pelo Editor como pelo Formatador.

b) Comandos imediatos, que são executados pelo Editor, no instante posterior ao da sua digitação.

c) Comandos deferidos, que são executados pelo Formatador, à medida que surgem ao longo do texto.

d) Elementos sintáticos, utilizados pelos comandos.

Cada divisão é subdividida em seções, uma para cada elemento sintático ou comando.

Cada seção é subdividida em quatro ítens:

a) Sintaxe, fornecendo a descrição do comando em BNF,

b) Semântica, descrevendo os detalhes das ações provocadas pelos comandos,

c) Exemplo, mostrando uma aplicação do referido comando,

d) Implementação, descrevendo o desdobramento do comando nos seus microcomandos, ou então a sua redução para comandos mais simples.

A1.2 Elementos sintáticos

A1.2.1 Nome

A1.2.1.1 Sintaxe

<nome>::=<letra>|<conjunto alfanumérico>

<conjunto alfanumérico>::=<vazio>

<caracter alfanumérico><conjunto alfanumérico>

<caracter alfanumérico>::=<letra>|<dígito>

<letra>::=A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z

<dígito>::=0|1|2|3|4|5|6|7|8|9

<vazio>::={o conjunto vazio de caracteres}

A1.2.1.2 Semântica

Um nome serve para identificar elementos do SPT. Estes elementos podem ser uma macro, uma marca, ou um arquivo.

O nome deve começar, necessariamente, por uma letra.

O número de caracteres máximo de um nome é 255.

A1.2.1.3 Exemplo

POSICIONE TEMPORARIO;

TEMPORARIO é um nome. Neste caso, é considerado uma marca. O cursor é posicionado na marca especificada.

L.2.1.4 Implementação

Os nomes são extraídos da cadeia de caracteres pela rotina OBTERNOME. Sua função é obter caracteres e armazená-los no campo NOME da sentinela até que ocorra uma das seguintes condições:

a) o primeiro caracter não é letra

b) é encontrado um caracter não alfanumérico

c) o tamanho do nome é superior a 255 posições.

Um nome sempre representa um elemento ao SPT. Normalmente, após obter um nome, deve ser realizada a "tradução" para o elemento correspondente. Após a microinstrução OBTERNOME é freqüente a ocorrência da instrução PESQUISA.

A1.2.2 Expressão

A1.2.2.1 Sintaxe

<expressão>::=<expressão aritmética>|<expressão lógica>

<expressão aritmetica>::=<ea>

<expressão lógica>::=<eb>

<ea>::=<ea><opa><ea>|

(<ea>)|

<número>|

<eb>::=<ea><opr> <ea>|

(<eb>)|

<eb><op1[2]> <eb>|

<op1[1]><eb>|

<valor verdade>

<opa>::=*|/|+|-

<opr>::=<|>|=

<op1[1]>::=NAO

<op1[2]>::=OU|E

<número>::=<dígito>|

<dígito><número>

<valor verdade>::=V|F

A1.2.2.2 Semântica

Uma expressão sempre representa uma fórmula aritmética ou lógica. O resultado final, no caso de uma <expressão aritmética> é um <número>, e, no caso de uma <expressão lógica>, é um <valor verdade>.

A prioridade dos operadores é a seguinte:

* , /

\+ , -

< , > , =

NAO

E

OU

Quando existir empate, a expressão é resolvida da esquerda para a direita.

Se houver necessidade de alterar a prioridade dos operadores, a utilização de parenteses permite a criação de nova seqüência na solução de expressões.

A1.2.2.3 Exemplo

No SPT não existem variáveis. Além disto, as operações aritméticas só podem ser realizadas sobre números.

No entanto, o SPT oferece o recurso de macros, que podem ser utilizadas para realizar o papel de variáveis.

Por exemplo, o comando

ATRIBUA CONT /!CONT+1/:

Supondo que a macro CONT possua o valor 15, após a execução do comando seu valor passa a ser 16. A expressão aritmética vista pelo sistema foi 15+1.

A1.2.2.4 Implementação

Sempre que é esperada uma <expressão>, é chamada a rotina OBTERSOLUÇÃO, informando se o resultado final deve ser numérico ou lógico.

A subrotina OBTERSOLUÇÃO:

- busca os operandos e transforma em formato padrão (20 posições com sinal);

- busca os operadores e resolve as operações parciais de acordo com as prioridades estabelecidas;

- detecta o final da expressão e,
  .se o resultado esperado for aritmético, transforma o resultado final em formato de uso (zeros suprimidos, com sinal - se negativo), e armazena na área MACRO da sentinela, ou
  .se o resultado esperado for lógico, transfere o valor verdade resultante para uma determinada posição da memória (COND).

Normalmente, um microcomando OBTERSOLUÇÃO é precedido por um microcomando OBTERNOME. Este se encarrega de

colocar na área de NOME da sentinela o nome da macro em cujo conteúdo será armazenado o resultado da OBTERSOLUÇÃO. Logo após é colocado um microcomando ALTERAMACRO ou INCLUIMACRO, para atribuir o valor da solução a uma macro.

A1.2.3 Seqüência

A1.2.3.1 Sintaxe

```
<seqüência>::=<delimitador><seqüência sem delimitador>
                                        <delimitador>
<seqüência sem delimitador>::=<vazio>|
                              <caracter diferente de
                                        delimitador>
                              <seqüência sem delimitador>
<caracter diferente de delimitador>::={qualquer caracter
                                       EBCDIC exceto
                                       <delimitador>}
<delimitador>::=<delimitador singular>|
                <delimitador composto>
<delimitador singular>::={qualquer caracter EBCDIC exceto
                          <indicador de composição>}
<delimitador composto>::=<indicador de composição><nível
                                    de delimitação>
<indicador de composição>::=:
<nível de delimitação>::={qualquer caracter EBCDIC}
```

A1.2.3.2 Semântica

A <seqüência> permite especificar uma cadeia de caracteres e comandos. O tamanho mínimo é zero, o máximo é 255.

A seqüência é iniciada e encerrada pelo mesmo <delimitador>. O que estiver entre estes dois pontos é considerado como sendo a cadeia de caracteres.

Como <delimitador> pode ser empregado qualquer caracter exceto dois pontos(:), que indica um delimitador composto. Neste caso, é necessário um segundo caracter para indicar o nível de delimitação.

A1.2.3.3 Exemplo

Uma seqüência empregando um delimitador singular pode ser:

/SEQUENCIA/

A barra (/) é utilizada como o delimitador da cadeia de caracteres.

Entretanto, se for necessário pela utilização de macros, criar seqüências dentro de seqüências, a escolha do delimitador se tornaria mais complexa e ilegível, como por exemplo

CRIE A /ALTERE B .ATRIBUA C*!C+1*./;

Por este motivo foi criado o <delimitador composto>. O mesmo texto ficaria, empregando este recurso,

CRIE A :1ALTERE B :2ATRIBUA C :3!C+1;3:2:1;

A1.2.3.4 Implementação

As seqüências são extraídas do texto pelas rotinas OBTERSEQUENCIACOM e OBTERSEQUENCIASEM, que correspondem aos microcomandos BUSCASEQUENCIACOM e BUSCASEQUENCIASEM, respectivamente. A primeira extrai uma seqüência entre delimitadores, a segunda considera as extremidades como sendo brancos.

OBTERSEQUENCIACOM analisa o primeiro caracter do texto e verifica se é um indicador de composição (:); se for, busca e guarda o segundo caracter. A partir deste ponto, sai à procura de um delimitador. No caso de um <delimitador composto>, é tomada precaução contra uma cadeia do tipo ::4, onde o primeiro caracter é parte da seqüência, o segundo e o terceiro formam o <delimitador composto>, encerrando a seqüência.

OBTERSEQUENCIASEM considera a cadeia de caracteres encerrada por um branco. Neste caso, não há nenhuma preocupação com <delimitador>es.

Ambas as rotinas armazenam os caracteres extraídos no campo MACRO da sentinela. A microinstrução seguinte é, normalmente, uma ordem para as rotinas de macros ou uma rotina que usa o campo MACRO da sentinela para uma determinada finalidade.

Um dos comandos que emprega OBTERSEQUENCIACOM é ALTERA <nome> <seqüência>, desdobrado internamente em

a) BUSCANOME

b) VERIFICAPRESENCA

c) BUSCASEQUENCIACOM

d) ALTERAMACRO

O microcomando a) extrai o nome do contexto e o coloca na área NOME da sentinela. O comando b) verifica se existe a macro que contém o nome. O comando c) transcreve o texto extraído para a área MACRO. O comando d) altera a macro.

O comando ENQUANTO emprega OBTERSEQUENCIASEM da seguinte forma:

a) EMPILHAMACRO %E1!%PILHA

b) BUSCANOME

c) BUSCASEQSEM

d) CRIE

O microcomando a) cria o nome de uma macro a ser armazenada. O microcomando b) armazena este nome na área NOME da sentinela. O microcomando c) transfere os próximos caracteres até o primeiro branco. O microcomando d) inclui a nova macro no sistema.

Como resultado final, uma expressão lógica pode ser armazenada nas macros sem a necessidade de usar uma <seqüência>. O comando ENQUANTO <exp b> <seqüência> pode ser implementado e reduzido.

A1.3 COMANDOS GERAIS

A1.3.1 Repita

A1.3.1.1 Sintaxe

<comando REPITA>::=REPITA <exp a><seqüência>

A1.3.1.2 Semântica

O comando REPITA faz aparecer ao SPT uma <seqüência> por um certo número de vezes, determinada pela <exp a>. Pode ser utilizado tanto para repetir um grupo de comandos, um comando ou uma seqüência de caracteres.

A <exp a> é resolvida antes de iniciar a repetição. As macros que compõe a <exp a> podem ser alteradas livremente sem causar problemas.

A1.3.1.3 Exemplo

Para repetir um grupo de comandos, basta colocá-los em uma seqüência e realizar a repetição:

```
REPITA 5 :1ULTIMO /ABCD/;
                ALTERE /ABCD/ /XYZ/
            :1
```

As 5 últimas ocorrências de ABCD serão trocadas

por XYZ.

A1.3.1.4 Implementação

O comando REPITA é implementado como um comando ENQUANTO. Portanto, se cria um contador, cujo valor inicial é o da <exp a>, e que se decrementa a cada vez que a seqüência é executada. Quando o contador chegar a zero, a repetição é encerrada.

O comando REPITA é reduzido para:

```
ATRIBUA %PILHA !%PILHA+1
CRIE %E1!%PILHA /0/;
ATRIBUA %E1!%PILHA <exp a>;
CRIE %S1!%PILHA <seqüência>;
ENQUANTO %E1!%PILHA>0 :1%S1!%PILHA;
                          ATRIBUA %E1!%PILHA
                                  %E1!%PILHA-1;
                      :1
DESTRUA %S1!%PILHA;
DESTRUA %E1!%PILHA;
ATRIBUA %PILHA !%PILHA-1;
```

<exp a> e <seqüência> são os elementos sintáticos originais.

A1.3.2 Enquanto

A1.3.2.1 Sintaxe

ENQUANTO <exp b><seqüência>

A1.3.2.2 Semântica

O comando ENQUANTO permite repetir uma <seqüência> enquanto uma <exp b> for verdadeira. Pode ser utilizada para repetir um grupo de comandos, um comando ou uma seqüência de caracteres.

A expressão é resolvida em primeiro lugar. Se for verdadeira, a <seqüência> é processada. Em caso contrário, o comando seguinte é executado.

Se, inicialmente, a expressão for falsa, a <seqüência> não é processada nenhuma vez.

Deve se tomar o cuidado de, dentro da <seqüência> permitir a alteração do valor lógico da expressão, para evitar um laço permanente indesejável.

A1.3.2.3 Exemplo

Supondo que se queira repetir um comando um certo número de vezes sem utilizar o comando REPITA. Basta realizar o seguinte comando:

```
ATRIBUA CONT 7;
ENQUANTO !CONT>0 :1PROXIMO /ABCD/;
                   ALTERE /ABCD/ /XYZ/;
                   ATRIBUA CONT !CONT-1;
                 :1
```

A1.3.2.4 Implementação

O comando ENQUANTO é implementado como um comando SE. É criada uma macro contendo a <seqüência>, seguida da instrução SE, chamando a própria macro, recursivamente. A primeira chamada faz com que seja executada sucessivamente até a expressão lógica assumir valor falso.

Especial cuidado foi tomado para que estas chamadas sucessivas não deixem "resíduos" na pilha, o que poderia causar o esgotamento da pilha em certos casos. Ao terminar um comando SE, todas as pilhas devem estar no mesmo estado em que estiveram ao iniciar o comando.

O comando ENQUANTO é reduzido para:

```
ATRIBUA %PILHA !%PILHA+1;
CRIE %E1!%PILHA /<exp b>/;
CRIE %S1!%PILHA <seqüência>;
CRIE %S2!%PILHA :1SE %E1!%PILHA :2!%PILHA;
                                  !%S2!%PILHA;
                                :2
                  :1
!%S2!%PILHA;
```

```
DESTRUA %S2!%PILHA;
DESTRUA %S1!%PILHA;
DESTRUA %E1!%PILHA;
ATRIBUA %PILHA /!%PILHA-1/;
```

<exp b> e <seqüência>são os elementos sintáticos originais. A criação de %E1!%PILHA foi representada de maneira simplificada. Maiores detalhes encontram-se na exemplificação da rotina OBTERSEQUENCIASEM, na implementação da <seqüência>.

A1.3.3 Se

A1.3.3.1 Sintaxe

SE <exp b><seqüência>

A1.3.3.2 Semântica

O comando SE permite o processamento condicional de uma <seqüência>. Esta pode ser um grupo de comandos, um comando ou uma seqüência de caracteres.

A expressão é resolvida em primeiro lugar. Se for verdadeira, a seqüência é processada. Em caso contrário, o comando seguinte é executado.

A.1.3.3.3 Exemplo

Supondo que se queira imprimir a palavra SENHOR ou SENHORA em uma carta, dependendo do sexo da pessoa. Criando uma macro

CRIE SR ;1SENHOR!SE !SEXO=!FEM :2A:2:1;

Sempre que houver dentro do texto uma referência a esta macro (!SR), será processada a palavra SENHOR ou SENHORA, dependendo da macro SEXO.

A1.3.3.4 Implementação

O comando SE exige um cuidado especial na sua implementação para poder ser chamado recursivamente sem causar um esgotamento das pilhas de interpretação.

É decomposto nos seguintes microcomandos:

a) BUSCASOLUÇÃO (lógica)

b) BUSCASEQUENCIACOM

c) EMPILHAMACRO (Cond)

O microcomando a) busca a solução de uma expressão lógica, e armazena o seu resultado na macro %COND. O microcomando b) extrai uma seqüência do texto e armazena-a na área MACRO da sentinela. O microcomndo c) analisa o conteúdo da macro %COND; se verdadeiro, transforma a sentinela em uma macro temporária, e empilha a macro no topo da pilha de dados; se falso, nada acontece.

Após processar uma macro do tipo temporária no topo da pilha de dados, esta macro é excluida automaticamente pela rotina de interpretação.

A1.3.4 Inclua

A1.3.4.1 Sintaxe

INCLUA <arquivo><intervalo>

<arquivo>::=ARQUIVO <nome>|
<vazio>

<intervalo>::=<início do intervalo>|
<fim do intervalo>|
<início do intervalo><fim do intervalo>

<início do intervalo>::=DE <marca>

<fim do intervalo>::=ATE <marca>

<marca>::=<nome>

A1.3.4.2 Semântica

O comando INCLUA permite incluir no texto um trecho de um outro arquivo, delimitado por marcas.

Se o arquivo não for especificado, é subentendido que se trata do próprio arquivo que está sendo processado.

Se o início do intervalo não for especificado, é subentendido que a inclusão deve iniciar no início do arquivo.

Se o fim do intervalo não for especificado, é subentendido que a inclusão deve encerrar no fim do arquivo.

Pode, no formatador, ser realizada uma inclusão dentro de outra inclusão.

A1.3.4.3 Exemplo

Dentro de um CPT existe uma padronização dos diversos formulários empregados. Supondo que exista um arquivo PADROES, com todos os diferentes formatos arquivados, o comando

INCLUA ARQUIVO PADROES DE INICIOPX3 ATE FIMPX3;

Traria um conjunto de comandos que atribuiria aos parâmetros de diagramação do Formatador certos valores iniciais.

A1.3.4.4 Implementação

O comando INCLUA é composto pelos seguintes micro-comandos:

a) EMPILHAMACRO %SOBEPILHA
b) FECHAR ALA ESQUERDA
c) FECHAR ALA DIREITA
d) EMPILHA MACRO %ZERAMARCA
e) BUSCAPARAMETROS
f) ABRIR ARQUIVO
g) POSICIONAR %MARCA1%PILHA
h) ABRIR ALA ESQUERDA
i) DESLOCAR ATE %MARCA2%PILHA
j) FECHAR ARQUIVO
l) ABRIR ALA DIREITA
m) EMPILHA MACRO %DESCEPILHA

O microcomando a) empilha a macro "ATRIBUA %PILHA !%PILHA+1", para tornar o comando passível de ser usado recursivamente. Os microcomandos b) e c) fecham a janela. O microcomando d) empilha a macro

"MUDE %MARCA1!%PILHA /%INICIO/;
MUDE %MARCA2!%PILHA /%FIM/;
MUDE %NOME!%PILHA /!%ARQE/;",

inicializando as omissões. O microcomando e) altera as macros acima para um novo conteúdo, caso for especificado. O microcomando f) abre o arquivo cujo conteúdo deve ser incluído. O microcomando g) posiciona o arquivo na primeira marca (início de intervalo). O microcomando h) abre a ala esquerda, destruindo o conteúdo anterior à marca, sobrepondo os caracteres anteriores à posição apontada pelo cursor. O microcomando i) desloca o cursor até a segunda marca (fim de intevalo). Todos os caracteres percorridos são automaticamente incluídos no texto. O microcomando j) fecha o arquivo que estava sendo incluído. O microcomando 1) abre a janela à esquerda da última posição incluída, incorporando o final do trecho incluído no texto. O microcomando m) empilha a macro

"ATRIBUA %PILHA !%PILHA-1",

anulando o efeito da macro empilhada no microcomando a).

A1.3.5 Crie

A1.3.5.1 Sintaxe

CRIE <nome><seqüência>

A1.3.5.2 Semântica

O comando CRIE serve para criar uma macro, que, a partir deste comando, pode ser referenciada pelo seu nome. Ao ser referenciada, a seqüência é processada.

Não podem existir duas macros com o mesmo nome, pois a criação de uma macro já existente causa erro de execução. Caso for necessário alterar o conteúdo de uma macro já existente, pode ser utilizado o comando MUDE ou então o ATRIBUA.

A1.3.5.3 Exemplo

Para criar uma macro cujo valor inicial seja 5 basta executar o comando
CRIE VIAS /5/;.

A1.3.5.4 Implementação

O comando CRIE é decomposto nos seguintes micro-comandos:

a) BUSCANOME

b) VERIFICAAUSENCIA

c) BUSCASEQUENCIA

d) CRIEMACRO

O microcomando a) transfere o nome para o NOME da sentinela. O microcomando b) comanda uma pesquisa na tabela de macros e acusa erro caso a procura tiver sucesso. O microcomando c) transfere a seqüência para a MACRO da sentinela. O microcomando d) incorpora a nova macro na estrutura.

A1.3.6 Mude

A1.3.6.1 Sintaxe

MUDE <nome><seqüência>

A1.3.6.2 Semântica

O comando MUDE altera o conteúdo de uma macro. A partir deste comando, todas as referências a esta macro passam a utilizar o novo conteúdo.

A macro deve ter sido criada antes de ser alterada.

A1.3.6.3 Exemplo

Para alterar o conteúdo da macro CHAVE para ABCD basta realizar o comando
MUDE CHAVE /ABCD/;.

A1.3.6.4 Implementação

O comando MUDE é decomposto nos seguintes micro-comandos:

a) BUSCANOME

b) VERIFICAPRESENÇA

c) BUSCASEQUENCIA

d) ALTERE

O microcomando a) transfere o nome para o NOME da sentinela. O microcomando b) comanda uma pesquisa na tabela de macros e acusa erro caso a procura não tiver sucesso. O microcomando c) transfere a seqüência para MACRO, da sentinela. O microcomando d) substitui o antigo conteúdo da macro pelo da sentinela.

A diferença entre o comando CRIE e MUDE está no microcomando b).

A1.3.7 Atribua

A1.3.7.1 Sintaxe

ATRIBUA <nome><exp a>

A1.3.7.2 Semântica

O comando ATRIBUA serve para colocar como valor atual de uma macro o resultado de uma expressão aritmética. A partir deste comando, todas as referências a esta macro passam a referenciar o seu novo conteúdo.

A macro deve ter sido criada antes de ser executado este comando.

A1.3.7.3 Exemplo

Para se decrementar de uma unidade o valor da macro CONT basta realizar o comando
ATRIBUA CONT !CONT-1;.

A1.3.7.4 Implementação

O comando ATRIBUA é decomposto nos seguintes micro comandos:

a) BUSCANOME

b) VERIFICAPRESENCA

c) BUSCASOLUCAO (aritmética)

d) ALTERE

O microcomando a) transfere o nome para NOME, da sentinela. O microcomando b) comanda uma pesquisa na tabela de macros e acusa erro caso a procura não tiver sucesso. O microcomando c) armazena em MACRO, da sentinela, o resultado da expressão aritmética. O microcomando d) substitui o antigo conteúdo da macro pelo que foi colocado na sentinela.

A diferença entre o comando ATRIBUA e MUDE está no microcomando c).

A1.3.8 Destrua

A1.3.8.1 Sintaxe

DESTRUA <nome>

A1.3.8.2 Semântica

O comando DESTRUA serve para eliminar macros e marcas da tabela de macros. A partir deste comando, nenhuma referência pode ser realizada à macro ou marca especificada.

A macro deve ter sido criada antes de ser executado este comando.

A1.3.8.3 Exemplo

Para eliminar a macro CONT da tabela de macros, basta realizar o comando
DESTRUA CONT;.

A1.3.8.4 Implementação

O comando DESTRUA é decomposto nos seguintes micro comandos:

a) BUSCANOME

b) VERIFICAPRESENCA

c) DESTRUA

O microcomando a) transfere o nome para NOME, da sentinela. O microcomando b) comanda uma pesquisa na tabela de macros e causa erro caso a procura não tiver sucesso. O microcomando c) elimina a macro da tabela.

A1.4 COMANDOS IMEDIATOS

A1.4.1 <comando de pesquisa>

A1.4.1.1 Sintaxe

<comando de pesquisa>::=<sentido da pesquisa><argumento de pesquisa>

<sentido da pesquisa>::=PROXIMO|ULTIMO

<argumento de pesquisa>::=SEQUENCIA <seqüência>|
PERIODO|PARAGRAFO|PAGINA|CAPITULO

A1.4.1.2 Semântica

Uma pesquisa pode ser realizada em dois sentidos: do início ao final do texto e vice-versa. Em termos de exibição na tela, para a direita e para a esquerda, respectivamente. PROXIMO comanda uma pesquisa no primeiro sentido, ULTIMO no sentido inverso.

O argumento de pesquisa determina o que deve ser procurado e, caso for encontrado, o cursor é posicionado sob a primeira posição do elemento correspondente. O argumento de pesquisa pode ser:

a) SEQUENCIA <seqüência>, quando é procurado no texto um conjunto de caracteres que coincida em con

teúdo e extensão com a seqüência.

b) PERIODO, quando é procurado o caracter "." (ponto).

c) PARAGRAFO, quando é procurado o comando !NOVA PAGINA.

d) CAPITULO, quando é procurado o comando !INICIO CAPITULO.

A1.4.1.3 Exemplo

Supondo que se queira posicionar na próxima ocorrência de MACRA, deve se digitar o comando PROXIMA /MACRA/.

A1.4.1.4 Implementação

Os comandos de pesquisa são implementados usando o seguinte conjunto de microcomandos:

a) EMPILHAMACRO

b) BUSCAPARAMETROS

c) PESQUISA

O microcomando a) atribue à macro %SENTIDO o valor 1 ou 0, dependendo se o sentido da pesquisa é para a direita ou para a esquerda. O microcomando b) extrai o argumento de pesquisa, e armazena na área MACRO, da sentinela. O microcomando c) ativa a rotina de pesquisa da janela, que, usando como parâmetro a macro %SENTIDO e o conteúdo da sentinela, realiza a pesquisa.

A1.4.2 <comando de edição>

A1.4.2.1 Sintaxe

<comando de edição>::=<tipo de edição><seqüência>

<tipo de edição>::=INSIRA|ALTERE|EXCLUA

A1.4.2.2 Semântica

Os comandos de edição permitem realizar operações de edição elementares sobre o texto.

Uma inserção consiste em incluir uma seqüência antes do cursor. A posição apontada pelo cursor permanece a mesma.

Uma alteração consiste em sobrepor a seqüência ao texto, iniciando pela posição apontada pelo cursor.

Uma exclusão consiste em eliminar tantos caracteres do texto quantos existirem na seqüência.

A1.4.2.3 Exemplo

Supondo que o cursor já esteja posicionado e se deseja incluir um certo texto basta comandar

INSIRA /AMANHA /;.

A1.4.2.4 Implementação

Os comandos de edição possuem uma correspondência direta com as rotinas de edição de texto da janela. Os microcomandos são:

a) EMPILHAMACRO

b) BUSCASEQUENCIA

c) EDICAO

O microcomando a) atribue à macro %EDICAO os valores 0,1 ou 2, dependendo do tipo de edição. O microcomando b) armazena na área MACRO da sentinela a seqüência que será editada. O microcomando c) ativa a rotina de edição da janela que, usando como parâmentros a macro %EDICAO e a sentinela, realiza a operação desejada.

A1.4.3 Troque

A1.4.3.1 Sintaxe

TROQUE <seqüência><seqüência>

A1.4.3.2 Semântica

O comando TROQUE consiste na pesquisa da primeira seqüência e sua substituição pela segunda.

Este comando pode ser pensado como um comando PROXIMO, seguido de uma exclusão e posteriormente uma inclusão.

A1.4.3.3 Exemplo

Supondo que em um certo ponto do texto exista a palavra MACRA, que deve ser substituida por MARCA. O comando abaixo realiza esta função:
TROQUE /MACRA/ /MARCA/;.

A1.4.3.4 Implementação

O comando TROQUE é decomposto nos seguintes micro-comandos:

a) BUSCASEQUENCIA

b) PESQUISAPROXIMO

c) EXCLUIR

d) BUSCASEQUENCIA

e) INSERIR

O microcomando a) extrai uma seqüência do texto e a armazena na área MACRO da sentinela. O microcomando b) pesquisa a próxima ocorrência desta seqüência no texto em edição. O microcomando c) exclui esta seqüência. O microcomando d) extrai uma nova seqüência do texto e armazena na área MACRO da sentinela. O microcomando e) inclui esta nova seqüência no texto em edição.

A1.4.4 Elimine

A1.4.4.1 Sintaxe

ELIMINE <intervalo>

<intervalo>::=<início do intervalo>|
<fim do intervalo>|
<início do intervalo><fim do intervalo>

<início do intervalo>::=DE <marca>

<fim do intervalo>::=ATE <marca>

<marca>::=<nome>

A1.4.4.2 Semântica

O comando ELIMINE exclui um trecho do arquivo, delimitado por duas marcas.

Se o início do intervalo não for especificado, é subentendido que a exclusão deve principiar no início do arquivo.

Se o fim do intervalo não for especificado, é subentendido que a exclusão deve terminar no fim do arquivo.

Combinado com o comando geral INCLUA, pode ser utilizado para movimentar trechos dentro de um texto.

A1.4.4.3 Exemplo

Supondo que um trecho de texto seja devidamente identificado pelas marcas INICIOEXC e FIMEXC. Para eliminar-se o trecho, basta comandar
ELIMINE DE INICIOEXC ATE FIMEXC;.

A1.4.4.4 Implementação

O comando ELIMINE é composto pelos seguintes micro comandos:

a) EMPILHAMACRO
b) BUSCAPARAMETROS
c) POSICIONA %MARCA1
d) FECHE ALA ESQUERDA
e) DESLOCA ATE %MARCA2
f) ABRA ALA ESQUERDA

O microcomando a) empilha a macro
"MUDE %MARCA1 /%INICIO/;
MUDE %MARCA2 /%FIM/;",
atribuindo os valores por omissão às marcas de início e fim. O microcomando b) busca os parâmetros de início e fim da operação, caso eles existirem.O microcomando c) desloca o cursor até o ponto inicial da exclusão de texto. O microcomando d) faz com que, a partir deste ponto, todos os caracteres que ficarem à esquerda do cursor à medida que este se desloca para o final do arquivo, sejam destruidos. O microcomando e) desloca o cursor até a segunda marca, destruindo o texto à medida que o cursor

se desloca. O microcomando f) abre a ala esquerda, "fundindo" o texto anterior ao início ao texto posterior ao término da operação.

A1.4.5 Marque

A1.4.5.1 Sintaxe

MARQUE <nome>

A1.4.5.2 Semântica

O comando MARQUE cria uma marca que aponta para a posição atual do cursor. Sempre que em um comando for utilizada esta marca pelo seu nome, estará sendo referenciada a posição na qual o cursor se encontra neste instante.

A1.4.5.3 Exemplo

Supondo que se deseja marcar a presente posição para, após, excluir o trecho. Se o cursor já se encontrar na posição certa, basta comandar
MARQUE INICIOEXC;.

A1.4.5.4 Implementação

O comando MARQUE é composto pelos seguintes micro-comandos:

a) BUSCANOME

b) VERIFICAAUSENCIA

c) MARQUE

d) CRIE

O microcomando a) extrai um nome do texto e o coloca na área NOME da sentinela. O microcomando b) assegura que este nome ainda não existe na tabela de macros. O microcomando c) coloca na área MACRO da sentinela as informações referentes à marca. O microcomando d) inclui a sentinela na estrutura de macros.

A1.4.6 Remarque

A1.4.6.1 Sintaxe

REMARQUE <nome>

A1.4.6.2 Semântica

O comando REMARQUE atribui uma nova posição de cursor a uma marca já existente. A partir da execução deste comando todas as referências a esta marca passam a utilizar a nova posição do cursor.

A1.4.6.3 Exemplo

Supondo que exista uma marca de nome TEMPORARIO. Para apontar esta marca à posição em que se encontra o cursor basta comandar
REMARQUE TEMPORARIO;.

A1.4.6.4 Implementação

O comando REMARQUE é composto pelos seguintes microcomandos:

a) BUSCANOME

b) VERIFICAPRESENÇA

c) MARQUE

d) ALTERA

O microcomando a) extrai um nome do texto e o coloca na área NOME da sentinela. O microcomando b) assegura que este nome já existe na tabela de macros, causando erro em caso contrário. O microcomando c) coloca na área MACRO da sentinela as informações referentes à marca. O microcomando d) inclui a sentinela na estrutura de macros.

A1.4.7 Posicione

A1.4.7.1 Sintaxe

POSICIONE <nome>

A1.4.7.2 Semântica

O comando POSICIONE permite o deslocamento do cursor até uma determinada marca dentro do texto. Todos os comandos a seguir tomam como referência a nova posição do cursor.

A1.4.7.3 Exemplo

Supondo que se queira reposicionar o texto em uma posição anteriormente marcada, basta realizar o comando POSICIONE TEMPORÁRIO;.

A1.4.7.4 Implementação

O comando POSICIONE é decomposto nos seguintes microcomandos:

a) BUSCANOME

b) VERIFICAPRESENCA

c) POSICIONE

O microcomando a) extrai o nome do texto e o armazena na área NOME, da sentinela. O microcomando b) verifica a presença desta macro na tabela de macros. O microcomando c) posiciona o cursor na marca desejada.

A1.5 COMANDOS DEFERIDOS

A1.5.1 <comando de quebra>

A1.5.1.1 Sintaxe

<comando de quebra>::=NOVA <nível>
<nível>::=LINHA|PARAGRAFO|PAGINA

A1.5.1.2 Semântica

O comando de quebra força o encerramento de uma linha.

Se for comandada uma nova linha, o alinhamento é realizado dentro dos padrões estabelecidos, e a linha impressa.

Um novo parágrafo causa, além da impressão da linha, a execução da macro %PARAGRAFO. Nela devem constar quantas linhas devem ser deixadas em branco verticalmente, e quantos espaços devem constar no início da nova linha.

Uma nova página causa a execução da rotina de quebra de página.

A1.5.1.3 Exemplo

ENUMERADAS A SEGUIR:!%NOVA LINHA

A1.5.1.4 Implementação

O comando de quebra é decomposto no microcomando BUSCAPARAMETROS.

Se o parâmetro for LINHA, é comandada uma quebra de linha.

Se o parâmetro for parágrafo, é empilhada a macro %PARAGRAFO e comandada a quebra de linha.

Se o parâmetro for PAGINA, é comandada uma quebra de página.

A1.5.2 Tabula

A1.5.2.1 Sintaxe

TABULA <alinhamento>

<alinhamento>::=<lado><seqüência>

<lado>::=DIREITA|CENTRO|ESQUERDA

A1.5.2.2 Semântica

O comando TABULA permite que sejam realizadas tabulações dentro de uma linha. As posições que forem saltadas são preenchidas com caracteres da seqüência.

Se o alinhamento é à direita, o texto compreendido entre a última marca de tabulação e a posição atual dentro da linha é deslocado até atingir uma nova marca de tabulação.

Se o alinhamento é central, o texto compreendido entre a última marca de tabulação e a posição atual dentro da linha é deslocado até o seu centro atingir a média entre as marcas de tabulação.

Se o alinhamento é à esquerda, são preenchidas tantas posições quantas forem necessárias para atingir a próxima marca de tabulação.

A1.5.2.3 Exemplo

TABULA ESQUERDA /./;

A1.5.2.4 Implementação

O comando TABULA é decomposto nos seguintes micro-comandos:

a) BUSCANOME

b) BUSCASEQUENCIA

c) TABULA

O microcomando a) extrai o alinhamento do texto e armazena na área NOME da sentinela. O microcomando b) extrai a seqüência do texto e armazena na área MACRO da sentinela. O microcomando c) chama a rotina de tabulação do Formatador, que se encarrega de realizar a tabulação desejada.

A1.5.3 <comando composto>

A1.5.3.1 Sintaxe

<comando composto>::=<início de atribuição>|
<fim de atribuição>
<início de atribuição>::=INICIO <atribuição>
<fim de atribuição>::=FIM <atribuição>
<atribuição>::=<alinhamento>|
MAIUSCULOS|
NAOQUEBRA|
CAPITULO|
NAOFORMATA

A1.5.3.2 Semântica

O comando composto atribui uma certa característica a um trecho de texto, compreendido entre o ponto em que é dado INICIO de uma atribuição e o seu FIM.

O alinhamento faz com que todas as linhas, cuja quebra for forçada, sejam alinhadas dentro das margens esquerda e direita, e os caracteres restantes preenchidos com uma seqüência.

A atribuição MAIUSCULOS desaciona o algoritmo de maiúsculos automáticos dentro do Formatador. Este escreve apenas os caracteres após o ponto(.) em maiúsculo, e o restante minúsculo.

A atribuição NAOQUEBRA desativa o algoritmo de quebra de página do formatador. O conteúdo de linhas e o tamanho da página não são controlados, ficando a responsabilidade por conta do usuário.

A atribuição CAPÍTULO atualiza as variáveis %NIVELCAP e %NUMCAPn, podendo o usuário utilizá-las para a numeração de capítulos e seções.

A atribuição NAOFORMATA desativa todos os algoritmos de formatação. O usuário pode escrever um texto exatamente da forma no qual será impresso.

A1.5.3.3 Exemplo

```
INICIO CAPITULO;
```

A1.5.3.4 Implementação

Este comando é decomposto no microcomando BUSCAPARAMETRO.

Cada um dos parâmetros empilha uma macro que, por sua vez, atribui o valor 0 ou 1 a uma macro do sistema, que representa o atributo.

No caso do início de um capítulo é empilhada a macro %INICIOCAP, que é o texto impresso no início de um capítulo.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO DA UFRGS

SOFTWARE PARA PROCESSAMENTO
DE TEXTOS

TESE APRESENTADA AOS SRS.

Visto e permitido a impressão
Porto Alegre, 28/11/80

Coordenador do Curso de Pós-Graduação
em Ciência da Computação