UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA DE CAMPINAS

Departamento de Automação

INTEGRAÇÃO DOS SISTEMAS OPERACIONAIS DE DISCO E FITA MAGNÉTICA DO COMPUTADOR EAI-640

por

Daniel Camilo

Orientador

Yaro Burian Junior

Tese apresentada a Faculdade de Engenharia de Campinas da Universidade Estadual de Campinas como requisito parcial para a obtenção do Título de Mestre em Ciências.

Outubro/74

C145i

824/BC

AGRADECIMENTOS

Quero expressar meus sinceros agradecimentos ao colega Márcio Luiz de Andrade Netto, pelo seu apoio e dedicado auxílio na confecção deste trabalho.

RESUMO

O sistema de computação implantado no laboratório de simulação-híbrida da FEC, inclue duas unidades auxiliares de memória, disco e fita magnética "cassette", com monitoramento individual, o que, virtualmente, elimina a possibilidade de operação conjunta das duas unidades.

Este trabalho, trata da unificação do sistema de monitoramento dos dois periféricos, permitindo acesso e manipulações simultanea das duas unidades.

O principal objetivo deste monitoramento é colocar à disposição do usuário, a rapidez de operação do sistema de disco, aliada à, praticamente, infinita capacidade de armazenamento de informaçães do sistema de fita magnética. Como restrição foi imposta uma utilização eficiente da memória central, em termos de área ocupada pelo novo sistema operacional.

ÍNDICE

		Pagina
1.	INTRODUÇÃO	1
2.	A ESTRUTURA DO EAI-640	3
	2.1 Características Básicas	3
	2.2 Arquitetura do EAI-640	3
	2.2.1 Unidade Central de Processamento	3
	2.2.2 As Unidade do Sistema	4
	2.2.3 Estrutura de Endereçamento	5
	2.2.4 Sistema de Interrupção	7
	2.3 Representação do EAI-640	8
	2.4 Sistema de "Software" do EAI-640	11
	2.4.1 Processadores de Linguagem	11
	2.5 O Monitoramento do EAI-640	12
3.	CARREGADORES	19
	3.1 Discussão	19
	3.2 Descrição de Carregadores	20
	3.2.1 Ligações de Sub-rotinas	21
	3.2.2 Esquema de um carregador	21
	3.3 Carregadores e Montadores do EAI-640	25
	3.4 Considerações sobre seu Sistema Operacinal . . .	33
4.	MONITORAMENTO PROPOSTO	35
	4.1 Discussão	35
	4.2 Programação do Monitor Proposto	36
	4.2.1 Descrição das Sub-rotinas	37
	4.3 Diagrama de Blocos do Monitoramento Proposto . .	44

Pagina

4.4 Considerações sobre o monitor proposto 49

5. CONCLUSÕES : : : 50

REFERÊNCIAS 53

CAPÍTULO I

1. INTRODUÇÃO

Um requisito importante para um sistema de computação digital, é dispor de unidades auxiliares de memórias, de capacidades compatíveis com os tipos de trabalhos a serem realizados. Como exemplo, uma instalação cuja atividade é o processamentos de dados contábeis, também chamado processamento comercial, para o qual a sequência de operações a ser realizado pelo computador, ou o programa, é praticamente invariável durante longo periodo de tempo, - necessita unidades auxiliares que permitam o armazenamento de uma quantidade muito grande de dados alfanuméricos, tipicamente unidades de fitas magnéticas.

Por outro lado, um sistema cujo objetivo principal seja servir como ferramenta de trabalho, em diversas áreas de pesquisas científicas, necessita áreas auxiliares de memória suficientes, para o armazenamento de programas prontos para utilização (suporte - de programas), onde o requisito sobre, o tempo médio de acesso a uma informação não é tão crítico, mas , também, exige a existência de um meio de armazenamento, ainda que de pequena capacidade, seja suficientemente rápido para operação das diversaa fases constituintes , do processamento de um programa isolado. Isto porque a característica básica de funcionamento de um centro de processamento - científico, é a existência de uma infinidade de programas, todos diferentes, mas que se utilizam, em massa, de uma série de sub-rotinas ou subprogramas constantes das chamadas "bibliotecas de pro-

gramas científicos.

Claro está que, dispor de meios de armazenamento de grande capacidade e, simultaneamente, extremamente rápidos seria a solução ideal, que, entretanto, se mostra, atualmente, demasiada custosa.

Em particular, o sistema de computação digital EAI-640 do Laboratório de Simulação Híbrida da Faculdade de Engenharia de - Campinas, apresenta uma estrutura bastante interessante, contendo uma unidade de disco com capacidade de 360 k palavras, tempo médio de acesso a uma informação de 17 ms, e uma unidade de fita magnética "cassette", com quatro fitas independentes, de capacidade praticamente ilimitadas.

Dada a pequena capacidade de armazenamento do disco, é impraticável operar-se apenas com seu auxílio, e, por outro lado , dada a baixa velocidade da fita magnética, o processamento, tendo esta única opção, torna-se extremamente lento.

No entanto, esta é, precisamente, a situação atual do sistema, visto que o fabricante fornece sistemas operacionais específicos para cada tipo de periférico: o "MONITOR", exclusivamente para o disco e o "LIBRARIAN" para a fita magnética "cassette".

O objetivo deste trabalho é, então, realizar um novo - sistema operacional capaz, sem uma excessiva utilização da memória central disponível, de controlar simultaneamente e de uma forma eficiente os dois periféricos disponíveis, resolvendo também todas as implicações envolvidas com esta mudança, ou seja, reorganizando todo o sistema de "software" pré-existente.

CAPÍTULO II

2. A ESTRUTURA DO EAI-640

Neste capítulo, descreve-se o EAI-640. Não se trata, entretanto, de um manual de referências, para se usar o sistema, assim, não serão dados pormenores de como processar, manipular dados, ou ainda executar programas. A este respeito, será fornecida uma lista de referências no final do trabalho.

O propósito deste capítulo é simplesmente, fornecer um entendimento do sistema como um todo. Leva-se em conta que, é necessário ao leitor, ter alguns conceitos básicos, para um melhor entendimento do capítulo.

2.1 - CARACTERÍSTICAS BÁSICAS

O sistema digital EAI-640, possui basicamente duas características principais; processar dados, utilizando somente recursos do sistema digital; sua outra característica é servir como um controlador e manipulador de dados, para o sistema analógico.

2.2 - ARQUITETURA DO EAI-640

2.2.1 - O sistema básico de computação digital do EAI-640, é disponível com 8 K palavras de memória. O tamanho padrão de uma palavra é de 16-bits. Os dados são armazenados ou retirados da memória através de um registro denominado MDR (Memory Data Register), ou simplesmente registro M, com endereçamento através

do registro MAR (Memory Address Register), ou registro S. O EAI 640, conta com um registro de indexação X , usado para modificação de endereços; o registro X , possui 16-bits, onde o bit mais significativo é reservado para assinalar endereçamento direto ou indireto. O registro de contador de programa P , contendo 15-bits, é o registro que contém o endereço da instrução, que está sendo executada. A instrução é retirada da memória e passa para o registro de instrução I, sendo então executada. Ao final da execução o registro P , conterá o endereço da próxima instrução a ser processada. O registro P fornece, no entanto, uma sequência contínua de endereços, onde estão armazenadas as instruções a serem executadas.

2.2.2 - A unidade de controle dirige as operações requeridas para executar instruções e estabelecer os tempos necessários para executar essas operações sequencialmente, e é justamente quando a instrução entra no registro I, que a unidade de controle tem acesso a instrução que será executada, para o estabelecimento do tempo necessário a ser gasto na execução. Depois que esta instrução é interpretada a unidade de controle envia os comandos necessários as outras unidades, para execução, por exemplo, de operações de entrada e saída, proteção de programa, interrupção e processamento de dados na unidade aritmética ou lógica.

A unidade de memória do computador, consiste de dois registros (M e S) e a memória propriamente dita. O ciclo de memória pe de 1.65 microsegundos, e é definido como sendo o tempo mais curto possível, entre sucessivas operações de leitura e armazena-

mento de dados. A memória propriamente dita é do tipo de núcleos de ferrite com 17-bits por palavra, sendo que um bit é dispensável para a proteção da palavra.

Através do sistema de proteção de memória é possível prevenir alteração em qualquer posição, cujo bit de proteção tenha valor 1. Uma tentativa de violar um local protegido na memória, origina um sinal de interrupção.

A unidade aritmética e lógica, composta do registro comumente chamado de acumulador ou registro A, é onde se processam todas as operações aritméticas e lógicas, capazes de serem executadas pelo computador. O registro Q ou extensão do acumulador, é usado em algumas operações juntamente com o registro A. Estes são os principais registros da unidade lógica e aritmética.

A unidade de entrada e saída, é composta de um sistema de fita de papel (leitora e perfuradora), leitora de cartão perfurado, fita magnética de "cassette", disco e o teletipo usado como impressora e dispositivo de entrada para comandos de controle, através de seu teclado.

2.2.3 - Estrutura de Endereçamento

Para um melhor entendimento do processo de endereçamento do EAI-640, é necessário considerar o número de bits da instrução utilizada para o endereçamento.

O campo de deslocamento D, quando tratado como um número positivo de 9-bits, resulta em um intervalo de 0 a 511 endereços. Então, 512 localizações de 00000_8 a 00777_8, denominada de zona-zero, po-

dem ser diretamente usadas pelo programador. Esta forma de endereçamento é referida como endereçamento absoluto. Outra forma de endereçamento é o relativo, onde a localização é referida através do conteúdo do contador de programa (registro P), somado com o número contido no campo de deslocamento da instrução, considerada agora como um número com sinal e oito bits em complemento de dois ($\pm$ 256).

Esta forma de endereçamento, absoluto e relativo, não oferecem a flexibilidade de programação desejável. Entretanto, o sistema de computação digital EAI-640, oferece ainda o endereçamento através do registro de indexação X, e também endereçamento indireto, através de qualquer posição de memória. É também permitida qualquer combinação dos três modos de endereçamento acima descritos. Isto fornece ao programador acesso a qualquer localização e em qualquer parte de seu programa, de uma forma fácil de programação. A figura 2.1, ilustra o formato para a instrução de referência à memória e a palavra armazenada.

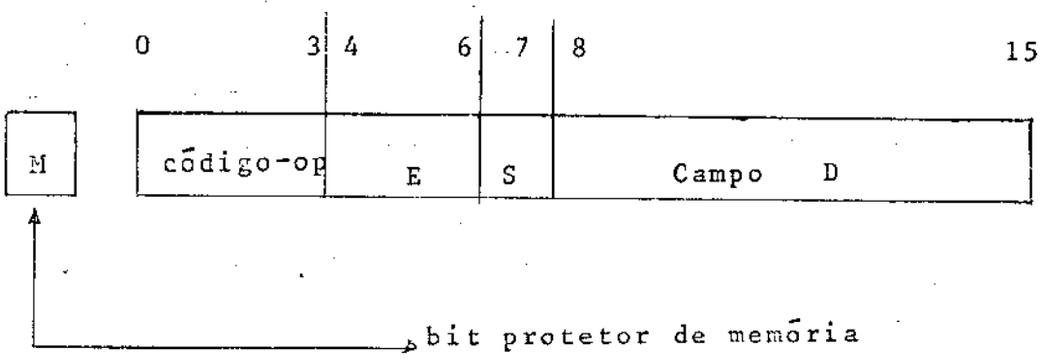


Fig. 2.1

A localização de cada palavra armazenada é identificada por um endereço de 15-bits. A instrução de referência à memória consiste de três campos, definidos como:

- campo de operação , com 4-bits de 0 a 3 na palavra de instrução este campo define o tipo de instrução a ser executada;
- campo de endereçamento efetivo E, com 3-bits de 4 a 6, os quais determinam a interpretação do campo de deslocamento D;
  bit 4, indica endereçamento indireto (I),
  bit 5, indica endereçamento indexado (X),
  bit 6, indica endereçamento relativo ao contador de programa P;
- Campo de deslocamento D, localiza um determinado endereço.

2.2.4 - Sistema de interrupção

Interrupção é um sinal que, interrompe a sequência normal de operação e transfere o controle para um programa especial na memória. Este programa especial é comumente chamado de Rotina de Serviço de Interrupção,sendo sua função semelhante a uma sub-rotina. A diferença principal, é que, a chamada para uma sub-rotina, ocorre em determinado ponto no programa, enquanto que uma interrupção,portanto, a execução da Rotina de Serviço de Interrupção, pode ocorrer em qualquer instante no programa.

A interrupção fornece ao programador, uma razoável capacidade de controle e manejo sobre a máquina.Um sinal de interrupção, em relação ao processador central, è um sinal gerado internamente, por exemplo, a execução de uma instrução de TRAP, ou externamente, causado por um dispositivo periférico, que , por exem

plo, terminou sua tarefa, originando o sinal no seu canal de entrada e saída. O programador tem completo domínio sobre a Rotina de Serviço de Interrupção, através da inibição da interrupção que desejar, mascarando-a; exceto na falta de energia elétrica.

Quando ocorre uma interrupção, o processador central - força o conteúdo do registro P a ser armazenado em uma posição da memória, indicada por um apontador; o conteúdo do registro A, ou acumulador, é armazenado na posição seguinte a do registro P; a próxima instrução a ser executada, a qual é considerada pela unidade de controle como a primeira instrução da Rotina de Serviço Interrupção sendo executada, normalmente, a seguir.

2.3 - REPRESENTAÇÃO DO EAI-640

Nesta secção, representa-se o EAI-640 em diagrama de - blocos, com uma descrição sumária de suas unidades.

O EAI-640, assim como a maioria dos mini-computadores, seguem basicamente o mesmo esquema, quanto à sua estrutura física. Compõe-se das seguintes unidades:

- unidade de controle;
- unidade aritmética e lógica;
- memória central;
- unidades de entrada e saída.

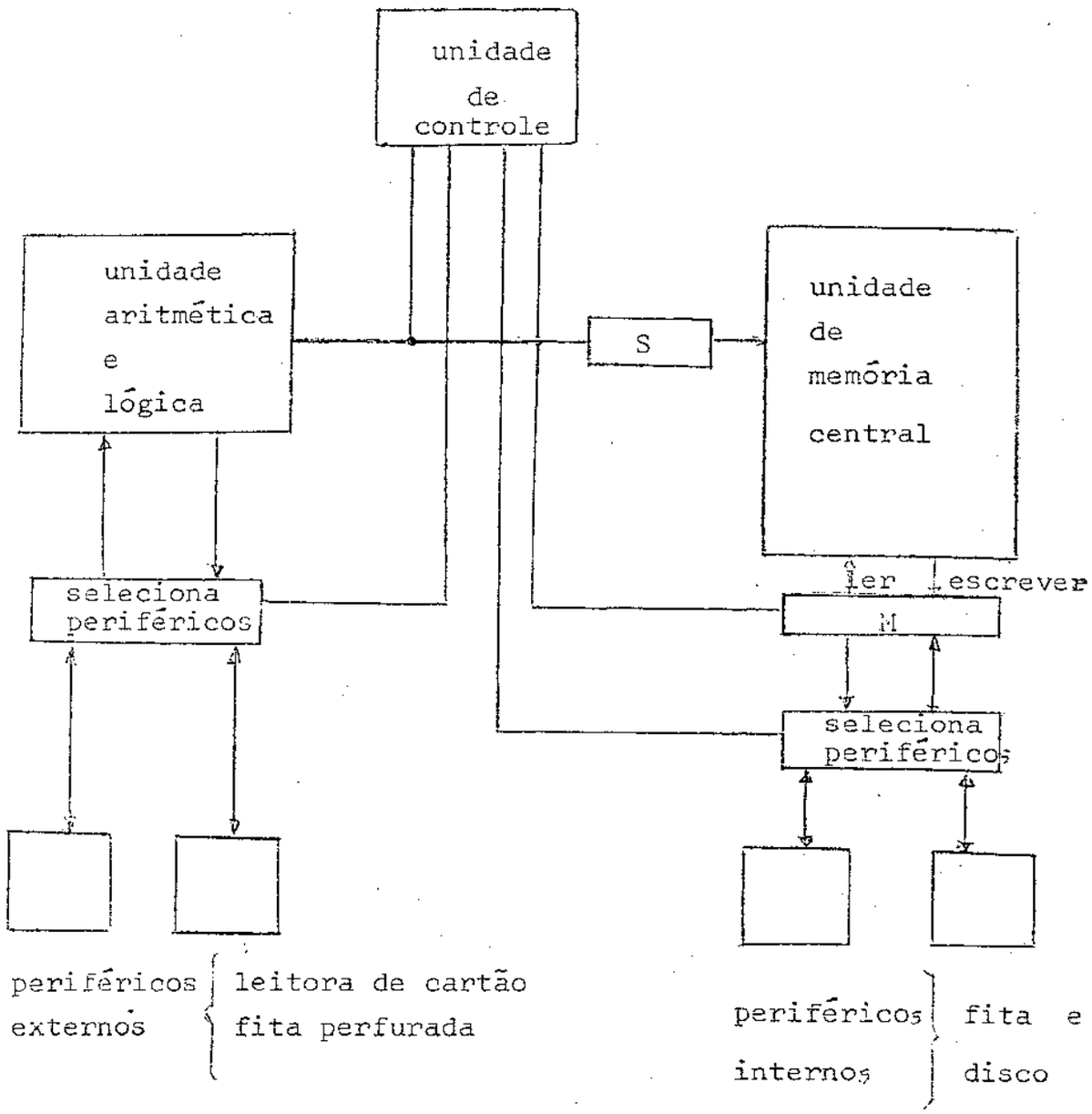
A unidade de controle, composta dos registros P, contador de programa, registro de indexação X , de estado K e re-

gistro de instrução I. Esta unidade, como o próprio nome diz, - coordena e controla as diversas operações da máquina.

A unidade aritmética e lógica, efetua os cálculos e operações comandadas pela unidade de controle. O acumulador A, e a extensão do acumulador, registro Q , pertencem a esta unidade.

A memória central, é onde se armazena programas e dados para a sua execução. Seus principais registros são: S, seleciona dor de endereços e o registro M de dados.

Existem dois tipos de unidades de entrada e saída: unidades de comunicações (leitora de cartão, fita perfurada, impressora), que permite diálogo com o exterior, e as memórias auxiliares (disco, fita magnéticas), que podem armazenar grande massa de informações.



- Diagrama de blocos das unidades do EAI-640

2.4 - SISTEMA DE SOFTWARE DO EAI-640

O objetivo desta secção, é apresentar alguns processadores de linguagens, assim como certos recursos disponíveis.

O EAI-640 é disponível basicamente com os periféricos de entrada e saída de dados: fita de papel perfurada, fita magnética "cassette", unidade de fita magnética, leitora de cartão e disco. Sendo que cada sistema básico é fornecido pelo fabricante com o seu sistema operacional.

2.4.1 - Processadores de linguagens

Como já foi dito anteriormente, o principal campo de aplicação do EAI-640, é a pesquisa. Propositalmente, o seu sistema operacioanl, vem munido de algumas linguagens de programação, como o FORTRAN IV, projetada para escrever programas científicos e aplicações em engenharia. Uma das grandes vantagens do compilador FORTRAN IV deste sistema de computação, é a possibilidade de aceitar comandos e gerar códigos em linguagem Assembler, o que permite a manipulação dos registros internos da máquina, praticamente inacessíveis em outra linguagem de programação. Esta prática é bastante usada, quando se faz programação híbrida.

O processador de linguagem "Hitran Operation Interpreter" (HOI), criado especialmente para oparação, controle, monitoramento e execução do processamento no sistema híbrido, é outra linguagem pertencente ao EAI-640.

O processador HOI , foi projetado, de modo a facilitar a computação digital, em uma configuração híbrida, oferecendo

um compatível entrosamento entre as partes analógica e digital.

Do ponto de vista sintático, as caractersíticas da linguagem HOI , são semelhantes ao FORTRAN VI . Entretanto, as seguintes propriedades tornam o HOI . mais poderoso e mais conveniente para a computação híbrida:

- programação conversacional;
- comunicação intrínsica com o sistema híbrido;
- modo de execução especial, para programação analógica.

Além dos processadores acima descritos, o "software" do sistema, ainda dispõe de programas editores, programas auxiliares para manipulação de dispositivos de entrada e saída, biblioteca - de rotinas científicas e rotinas de ligação com o sistema híbrido.

2.5 - O MONITORAMENTO DO EAI-640

O monitoramento do sistema digital, o programa monitor do disco e o "librarian" de fita magnética "cassette", são os programas que controlam todas as operações que envolvem estes periféricos da máquina. Esse controle que o sistema de monitoramento - dispõe ao usuário, é oferecido através do teclado do teletipo ou leitora de cartão (cartões de comandos e controles).

Devido a grande utilização do monitoramento, a sua disponibilidade deve ser constante no sistema, portanto ele deve ser residente em disco, ou outro periférico que oferece rápido e fácil acesso, para ser carregado nos núcleos de memória pelo "bootstrap".

A constituição do programa monitor do EAI-640, que será estudada num capítulo posterior, é composta de sub-rotinas e subprogramas, cada um com finalidades bastantes específicas.

A seguinte sequência de processamento de um programa de usuário, desde a fase de preparação da máquina pelo monitor, compilação do programa, ligações de sub-rotinas e subprogramas e finalmente a execução do programa, dada a seguir, ilustra a ação do monitor no sistema. Para maior simplicidade, suporemos que o programa fonte, esteja na forma de cartões perfurados. No processamento de um programa no EAI-640, temos duas alternativas: uma através do Processador de Controle de Opções (COP) , ou processamento sequencial (batch), a outra alternativa é aquela em que o operador toma as vezes do COP , isto é, assume o controle das opções.

O processamento com auxílio do COP , é usado quando se dispõe de memórias auxiliares de grande capacidade (discos, fitas, etc.) e que libera o operador de fornecer à máquina as opções que o programa do usuário requer. Esta alternativa de processamento - sequencial, inicia-se com o comando de controle # X, U, dado através do teletipo, (no capítulo que se estuda as sub-rotinas do monitor, define-se estes símbolos), fazendo com que, o monitor através de suas rotinas, carregue nos núcleos de memória o Processador de Controle de Opções. Este processador ou monitor sequencial, é um subprograma do monitor, que interpreta cartões de controle, inicializa os registros da máquina, carrega e transfere o controle ao processador de línguegem (compilador), em seguida, car

rega e transfere o controle ao montador de programa (montador) e finalmente carrega e executa o programa do usuário.

A figura abaixo, ilustra a sequência de alguns cartões de controle e o programa fonte a ser processado pelo COP . O primeiro cartão $JOB, é interpretado como sendo o início de um novo programa a ser processado.

O cartão $PROC , faz com que o COP , carregue e transfere o controle para o processador de linguagem (compilador FORTRAN IV , Assembler, e outros), que compila o programa fonte.

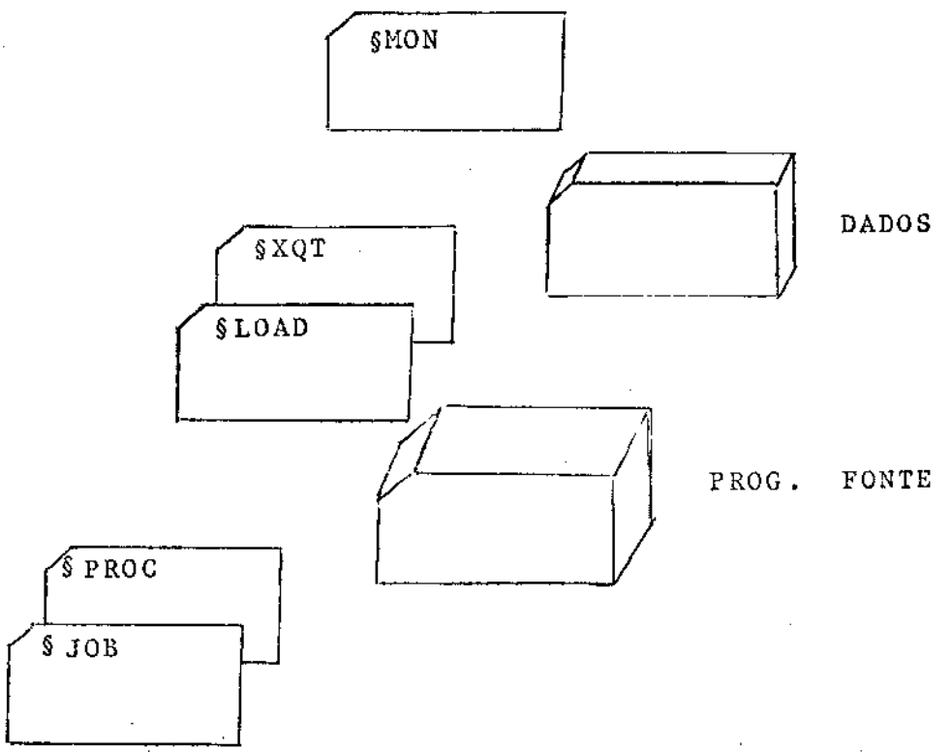


Figura 2.2

Ao final da compilação, o controle retorna ao monitor e subsequentemente ao COP. Este com o comando novamente, continuará a processar os cartões de controle, contidos logo após os cartões do programa fonte. Em seguida, o COP através das rotinas do monitor, carregará nos núcleos de memória o montador de programa.

No sistema de "software" do EAI-640, existem dois subprogramas montadores:

- o carregador de programa objeto (PL), que carrega e executa programas em formato objeto;
- o outro montador de programa é o gerador de programa (gerador de programa - CIG), que monta o programa objeto em formato de máquina, em uma memória auxiliar qualquer.

Os montadores, como o próprio diz, montam o programa fornecendo-lhes endereços absolutos, e construindo uma lista de todos os nomes externos (sub-rotinas e subprogramas), referenciados no programa do usuário. Esta lista, que o montador cria, dos nomes das sub-rotinas solicitadas neste programa, recebe o nome de vetor de transferência. Terminada esta operação do montador, o próximo passo a ser dado, será o carregamento das sub-rotinas e subprogramas, (este carregamento é feito pelas rotinas do monitor e controlado pelo COP), cujos nomes constam no vetor de transferência criado pelo montador. Estas sub-rotinas se encontram na biblioteca de sub-rotinas (Run Time Library, RTL), ou nas memórias auxiliares: disco, fita magnética de "cassette", fita de papel perfurada, etc. Depois de todas as sub-rotinas e subprogramas te

rem sido carregados e ligados (linking), o controle é retornado ao monitor e ao COP subsequentemente, que finalmente carrega o programa do usuário na memória e transfere o comando para êste programa para a sua execução.

Chama-se a tenção do leitor, neste ponto, com respeito - as sub totinas, que podem ser solicitadas no programa do usuário (rotinas científicas, da biblioteca, etc), que devem ser residente nos sistemas de memórias auxiliares. O monitoramento proposto - pelo fabricante, "librarian" da fita magnética de "cassette" e o monitor do disco, não permitem as comunicações entre os dois periféricos auxiliares de memória, trazendo o incoveniente de que, as rotinas e subprogramas residentes nestes periféricos, não serem acessíveis por monitores diferentes. Esta limitação é grave, observando que, a disponibilidade de recursos de memória auxiliar, fica bastante restringida. Tendo em vista que, o disco tem a vantagem sôbre a fita, no que diz respeito a velocidade e rapidez de processamento: Leitura, manipulação e armazenagem de arquivos. Mas com o aumento do número de sub-rotinas e subprogramas residentes (rotinas científicas, processadores de linguagem, programas editores , biblioteca de sub-rotinas e muitos outros), que o sistema deve dispor, ou ainda, com o próprio tamanho do programa do usuário, tem sua eficiência sensivelmente reduzida, dada a sua relativamente - pequena capacidade de armazenamento.

Por outro lado o sistema de fita magnética, tem capacidade de armazenar, grande massa de informações, em resumo, êste - sistema tem capacidade de guardar todo sistema de "software" da má

quina, incluindo com isto, as facilidades de manejos, trocas e a possibilidade de um grande estoque de fitas magnéticas de "cassette", devido a seu baixo custo. Mas este dispositivo auxiliar de memória é mais lento, a procura e consequentemente carregamento de um arquivo é demorado, perdendo para o disco em termos de rapidez.

O seguinte fluxograma ilustra o processamento de um programa de usuário, usando os recursos do processamento sequencial e dispensando estes recursos, i. é, operador fazendo o controle das opções:

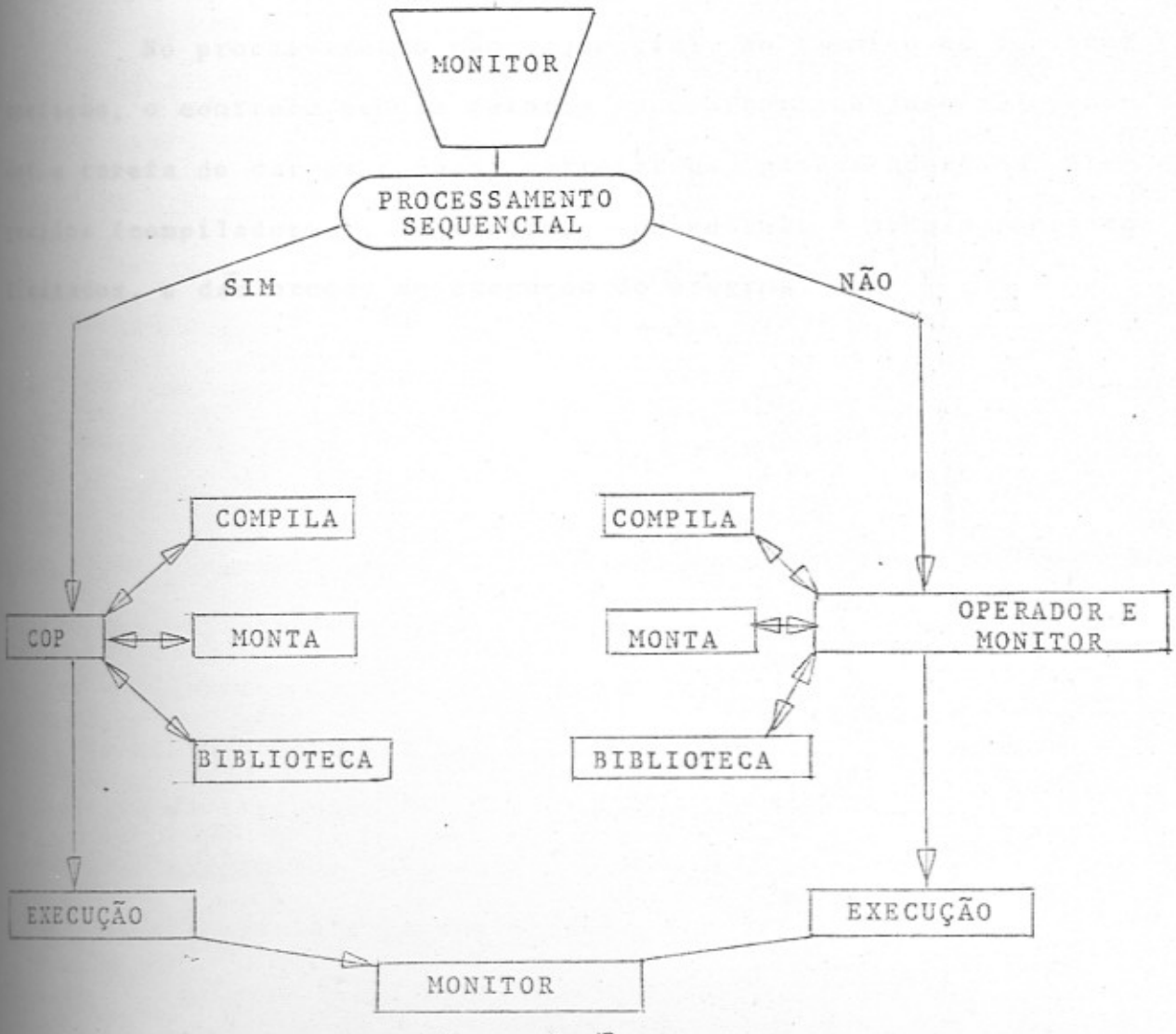


Fig. 2.3

Figura 2.3

A outra alternativa de processamento de um programa, é aquela em que o operador toma as vêzes do COP, como mostram os diagramas de blocos, onde o operador assume o controle das opções . Este tipo de processamento tem certas vantagens e desvantagens em relação ao processamento sequencial. Uma das desvantagens é que torna o processamento mais demorado e exige do operador, alguns conhecimentos sôbre a máquina, com a finalidade de fornecer as opções que o programa do usuário requer, como vantagem tem-se - que, o usuário dispõe de maiores, recursos, tanto de memória como sobre a máquina.

No processamento não sequencial, ao término de qualquer operação, o controle sempre retorna ao monitor, cabendo ao operador a tarefa de dar as opções: carregar os processadores de linguagens (compiladores), montadores, sub-rotinas e subprogramas solicitados, e dar ordens de execução do progrma.

CAPÍTULO III

3. CARREGADORES

3.1. DISCUSSÃO

O proposito deste capítulo, é apresentar e discutir alguns conceitos sobre carregadores, assim como, oferecer esquemas e funções para determinadas máquinas, não necessariamente o EAI - 640, com a finalidade de dar ao leitor, um encaminhamento para o principal objetivo deste trabalho.

Quando se inicia operação de qualquer programa, seja êle nas mais diversas formas de ser apresentado ao computador: - cartões perfurados, teclado do teletipo, enfim, um programa em linguagem fonte (FORTRAN, PL/1, e outras), o primeiro passo a ser dado no processamento deste programa, será a compilação. Em seguida o programa passa à fase de montagem, sendo armazenado em dispositivos auxiliares de memória (disco, fita magnética "cassette", etc). Na fase de compilação, o programa é convertido da linguagem fonte para a linguagem objeto. Na montagem do programa será transformado em linguagem de máquina. O carregador é um programa de máquina, que, diretamente, carrega êstes programas em formato objeto dos periféricos auxiliares para a memória principal, dando-lhe o controle para a sua execução, ou indiretamente, transforma o programa em linguagem de máquina, armazenando-o em periféricos auxiliares auxiliares, carregando-o, em seguida, na memória e passando-lhe o controle para sua execução.

Em particular, o carregador executa as seguintes tarefas:

1 - fixa espaço na memória, para os programas e sub-rotinas (Alocação) ;

2 - Constroi uma tabela de referências, simbólicas para o programa objeto, listando todas sub-rotinas solicitadas no programa e sub-programas (Ligação);

3 - ajusta todos os endereços, dependendo da localização, assim como os endereços das constantes definidas, para necessariamente corresponder aos espaços fixados (relocação)

4 - fisicamente, coloca as instruções de máquina e dados na memória (carregamento); direta ou indiretamente , como visto nteriormente.

5 - devolve o controle ao programa para a sua execução.

Os carregadores produzidos por diferentes fabricantes , diferem sensivelmente, tanto na sua extensão, quanto na execução de suas tarefas. Alguns deles, são tão simples que chegam a ser pouco mais que um simples "bootstrap" (inicializador ou partida - fria do computador). Outros carregadores incluem funções, que sã distribuidas aos outros programas do sistema, como é o caso do EAI - 640, ou outro mini-computador de seu porte. Enfim, o carregador é projetado levando em conta o sistema a que se destina, pequeno , médio ou grande porte.

3.2 DESCRIÇÃO DE CARREGADORES

Trata-se nesta secção de descrever com mais pormenores .

os quatros ítens enumerados na secção anterior. Antes porém, é conveniente definir o têrmo segmento. Segmento é uma unidade de informação, a ser tratado como uma entidade, sendo um programa ou dado. Usualmente corresponde a um simples programa em formato objeto.

3.2.1 Ligações de sub-rotinas

Nesta secção, faz-se uma breve discussão, do ponto de vista do programador de um mecanismo especial para a chamada de uma sub-rotina, na montagem de um programa.

O problema de ligação é o seguinte: Um programa principal A, pede a execução de um sub-programa B. O programador, no programa A, escreverá as instruções de transferência (digamos por exemplo: LINK B, (desvie para o sub-programa B). Entretanto, o compilador não conhece o valor deste símbolo de referência e o declara como um erro (símbolo indefinido), exceto se um mecanismo especial for implementado.

A pseudo-operação (pseudo-op) assembler EXTRN, seguido - por uma lista de símbolos, indica que tais símbolos, são definidos em um outro programa, mas referenciados no presente programa. Em correspondência, se um símbolo é definido em um programa e referenciado em outro insere-se tal símbolo numa lista seguido pelo - pseudo-op assembler ENTRY, ou seja, o compilador imformará ao carregador, que esses símbolos podem serem referenciados em outros - programas.

3.2.2. - Esquema de um carregador

Em alguns tipos de carregadores, a tarefa de relocação e

e ligação de subrotinas é executada pelo próprio programador. Para evitar este tedioso trabalho, pensou-se em um carregador capaz de executar as relocações e ligações de sub-rotinas e sub-programas.

Um exemplo deste tipo de carregador é o "Binary Symbolic Sub-routime" (BSS), usado nos computadores: IBM 1130, UNIVAC 1108 e GE 635.

O compilador fornece diversos segmentos, montando cada um deles separadamente, sendo que o carregador recebe com cada - segmento, informações e referencias sôbre possíveis interligações e relocações.

Um programa compilado no esquema BSS, é um programa objeto com informações sôbre todos outros subprogramas e referencias. Para cada programa fonte, existirá um program objeto pré-fixado por um vetor de transferência, que contém os nome das sub-rotinas referenciadas pelo programa fonte. Por exemplo, se uma rotina para extrair raiz quadrada (SQRT), foi mencionada e é a primeira sub-rotina chamada, a primeira localização no vetor de transferência contém o nome simbólico SQRT.

O comando chamando SQRT será traduzido em uma instrução de transferência que indica um desvio para a localização do vetor de transferência associado com a SQRT. O compilador fornece ao carregador informações adicionais como comprimento do programa e o comprimento do vetor de tranferência, isto é, todas as sub-rotinas nele contido. Depois de efetuado o carregamento do programa na memória, o carregador inicia a relocação de todas as sub-

rotinas no vetor de transferência, e então, a execução do comando CALL SQRT resutará em um desvio para a primeira localização do vetor.

A figura 3.2 ilustra um programa escrito na linguagem assembler, para um computador hipotético que usa o carregador BSS.

A função do programa não é importante, serve somente de ilustração de exemplo de chamada de uma sub-rotina SQRT, que extrai a raiz de 9 e se o resultado for diferente de 3 haverá uma chamada para uma subrotina de êrro "ERR". O pseudo-op EXTRN identifica o símbolo SQRT e ERR como nomes de sub-rotinas, desde que as localizações destes símbolos não estão definidas neste programa. Para cada símbolo externo o compilador coloca no inicio do programa a codificação EBCDIC correspondente ao nome da sub-rotina. As palavras assim geradas, formam o chamado vetor de transferência:

```
MAIN    INIC
        EXTRN     SORT, ERR
        .
        .
        STA       SAVE        armazena o endereço
                              de retorno
        .
        .
        .
        LA        9           carrega o valor a ser
                              testado
        .
        .
        .
        L         SQRT        chama a sub-rotina que
                              extrai raiz quadrada
        C         3           Compara com a respos
                              ta
        SNE
        L         ERR         Desvia para rotina de
                              mensagem de erro se
                              resultado é di erente
        .
        .
        .
        LA, I     SAVE        toma o endereço de re
                              retorno
        .
        .
        .
        J         SAVE        retorna
SAVE    BSS       1,0
        .
        .
        .
        END
```

Fig. 3.2

Usando a informação do programa principal, o carregador coloca as sub-rotinas SQRT e ERR, nas localizações seguintes da memória, respectivamente. Finalmente, as palavras do vetor de transferencia, são alteradas para conter instruções de desvio para as correspondentes sub-rotinas. Desta forma, as funções básicas do carregador, alocação. ligação, relocação, carregamento e execução do programa são todas processadas automaticamente pelo carregador BSS.

3.3 - CARREGADORES E MONTADORES DO EAI - 640

Os programas operacionais disponíveis no sistema de "software" do EAI - 640, como carregadores e montadores são : o carregador de programa objeto (PL), o gerador de programa (CIG) e o próprio monitor através de suas rotinas.

Para ilustrar a fase de montagem de um programa será considerado o gerador de programa. Este montador exige para a sua operação que inicialmente se definam três condições iniciais (quando o operador não fornece estas definições iniciais, o próprio CIG as define). A primeira é a área comum às variáveis dos diversos programas (COMMON). O segundo item que necessita de definição é o endereço inicial do programa (STARTING ADDRESS). Este é o endereço a partir do qual se inicia e monta o programa, na opção padrão, a montagem é inicializada a partir da posição 1000 da memória. O programador, entretanto poderá alterar posições, através do pseudo-op ABS e REL. O terceiro e último ítem que se define, é o inicio da ZONA-ZERO da memória (ZONE-ZERO POOL), que

é o endereço onde se colocará no final da montagem do programa, o vetor de transferência das sub-rotinas solicitadas no programa do usuário.

Para melhor ilustração, considere-se a seguinte sequência de programação em Linguagem Assembler:

```
START    REL      0
         EXTRN    C, B
         .
         .
         .
         L        C          link  com sub-rotina C
         .
         .
         .
         L        B          link com sub-rotina B
         .
         .
         .
         END
```

Fig. 3.3

Este programa depois de compilado, é armazenado em um dispositivo auxiliar de memória (por simplicidade de ilustração , considera-se este dispositivo auxiliar como sendo a fita), com uma identificação criada pelo COP (ou operador), por exemplo : OBJTAA.

Inicialmente a memória central se encontra no seguinte estado:

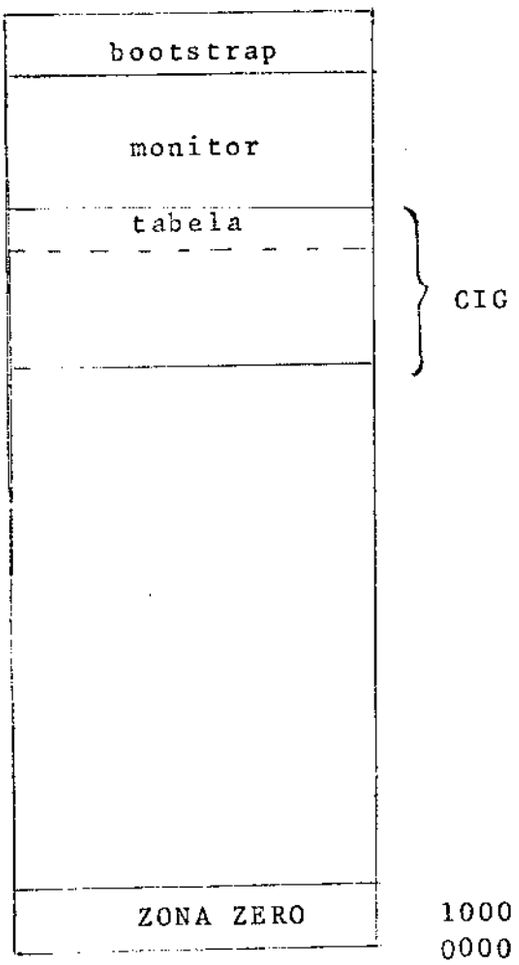
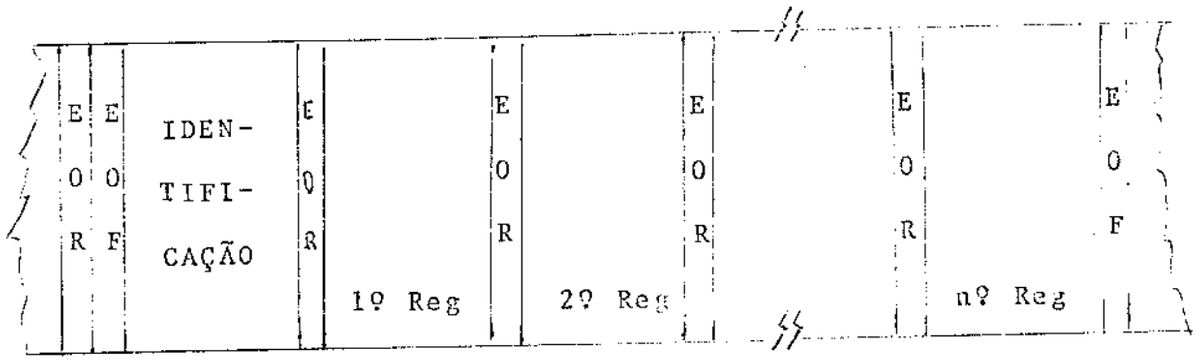


Fig. 3.4

O CIG constroi dentro de si, uma tabela que conterá os nomes das sub-rotinas referenciadas no programa do usuário, esta tabela recebe o nome de vetor de transferência. O programa objeto na fita, fica então com a seguinte disposição física:



onde

EOR - fim de registro

IDENTIFICAÇÃO - nome do arquivo (OBJTAA, no exemplo)

EOF - fim de arquivo

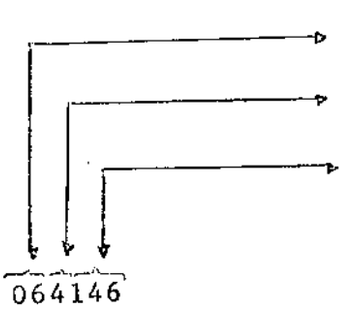
Fig. 3.5

O operador, ou o COP, cria, através do monitor um arquivo que deverá conter o programa em Linguagem de máquina, por exemplo CRMGAA (#N,CRMGAA,U,T).

Quando o operador, ou o COP, dá ao montador a ordem de execução, o CIG lê o arquivo OBJTAA, e começa a gerar a sequência em linguagem de máquina correspondente a este programa, colocando-a no arquivo CRMGAA recém-aberto.

Quando o CIG encontra a primeira instrução que referencie num símbolo externo no exemplo L C (link com a sub rotina C), coloca na primeira posição do vetor de transferência o símbolo C. Supondo que o início da ZONA-ZERO seja a posição 146, o

CIG fornecerá, então, a instrução de máquina absoluta (não relocá vel) 064146, onde



O CIG continuará a gerar as instruções de sequência de programação, até encontrar o nome da segunda sub-rotina, processando de maneira análoga a anterior, com a geração da instrução da máquina 064147, onde 147 é o endereço da segunda posição da ZONA -ZERO, colocando na segunda posição do vetor de transferencia o símbolo B. Novamente o CIG continua, entretanto, a geração do programa até encontrar o pseudo-op END, que finaliza o programa objeto, denominado OBJTAA, por exemplo, na posição 1576, e emitindo as mensagens pelo teletipo

LI

IP

A seguir o CIG processa o carregamento das sub-rotinas cujo nomes símbolicos se encontram na tabela construida por êle. Chama-se então o arquivo RTL ("Run Time Library" - biblioteca de rotinas) e começa-se a ler o nome das sub-rotinas lá existentes . A figura abaixo ilustra a biblioteca de sub-rotinas RTL.

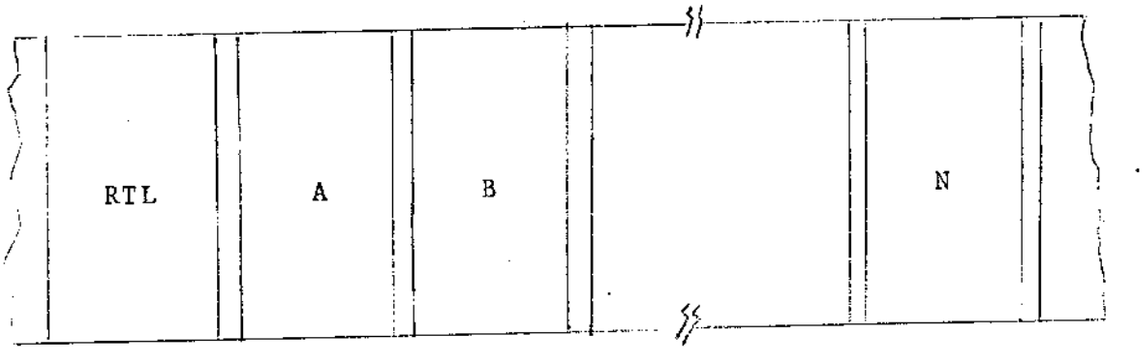


Fig. 3.6

O primeiro nome da rotina a ser lida no RTL, é o nome simbolico A, e o CIG compara-o com os nomes existentes no vetor de transferência. Se este nome não consta da tabela, a sub-rotina A não será necessária e, portanto, não será carregada. Lê-se o segundo nome RTL , que é B, comparando este nome simbólico com a sua tabela, o CIG verifica que este nome esta no seu vetor de transferência; B é então, carregado na posição seguinte ao término do programa principal OBJTAA, que será 1577. Neste ponto o nome simbólico B é apagado e colocado em seu lugar o endereço 1577, isto é, na tabela ficará 147 1577. O CIG continuará a ler os nomes das sub-rotinas restantes, comparando-as com os da tabela. A rotina C é carregada analogamente a B , e teremos na tabela : 146 1656 (onde 1655, é a última posição que a sub-roti

na B ocupou). Terminada esta fase de ligação e carregamento de sub-rotina, o CIG coloca na ZONA-ZERO o vetor de transferência, - isto é, os endereços absolutos das sub-rotinas. No final da montagem do programa na fita ficará da seguinte forma:

onde, e o r : indica fim de registro

e o f : indica fim de arquivo

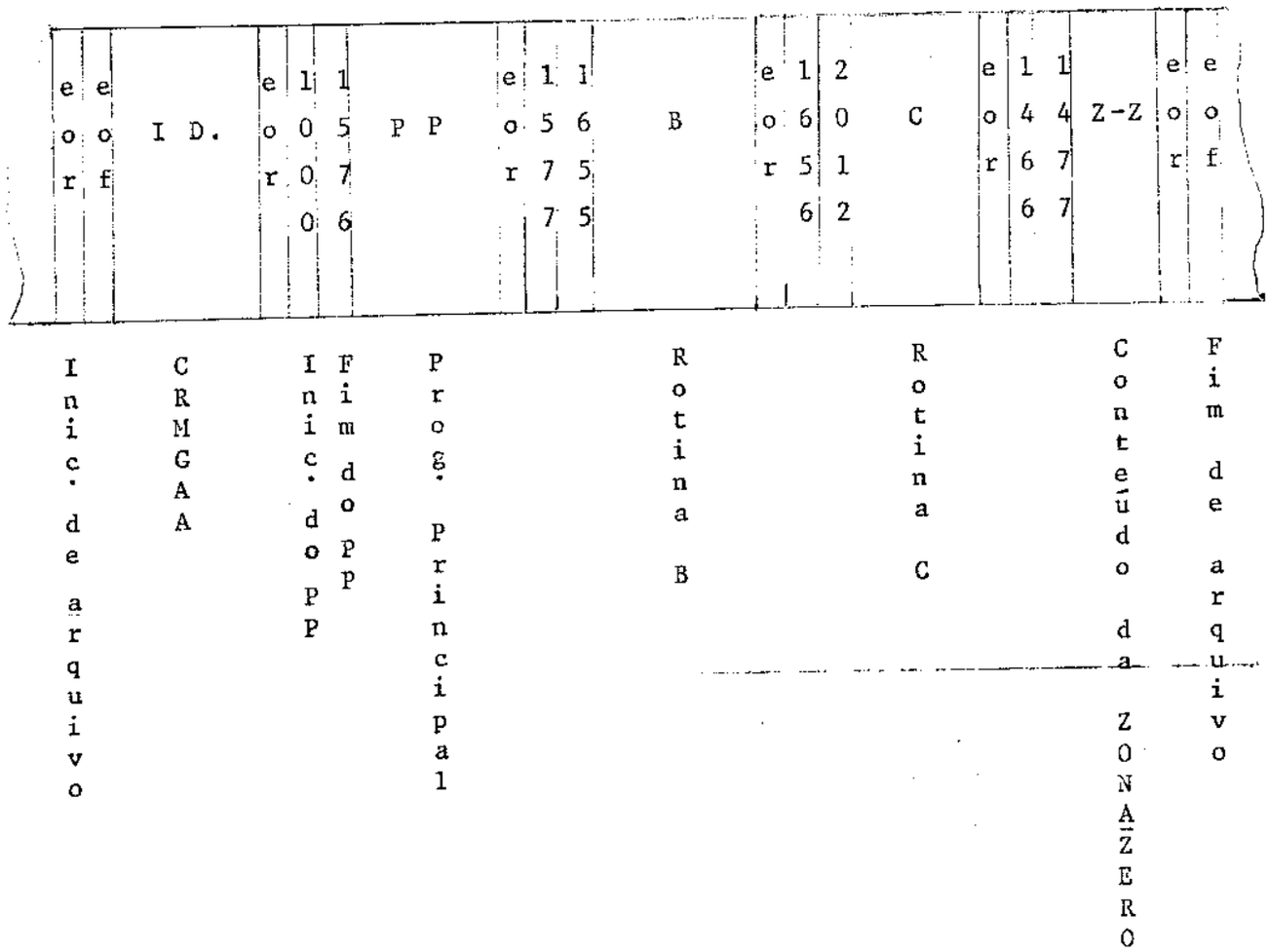


Figura 3.7

Para a execução do programa, o arquivo CRMGAA é carregado na memória através do monitor e ocupará as seguintes posições:

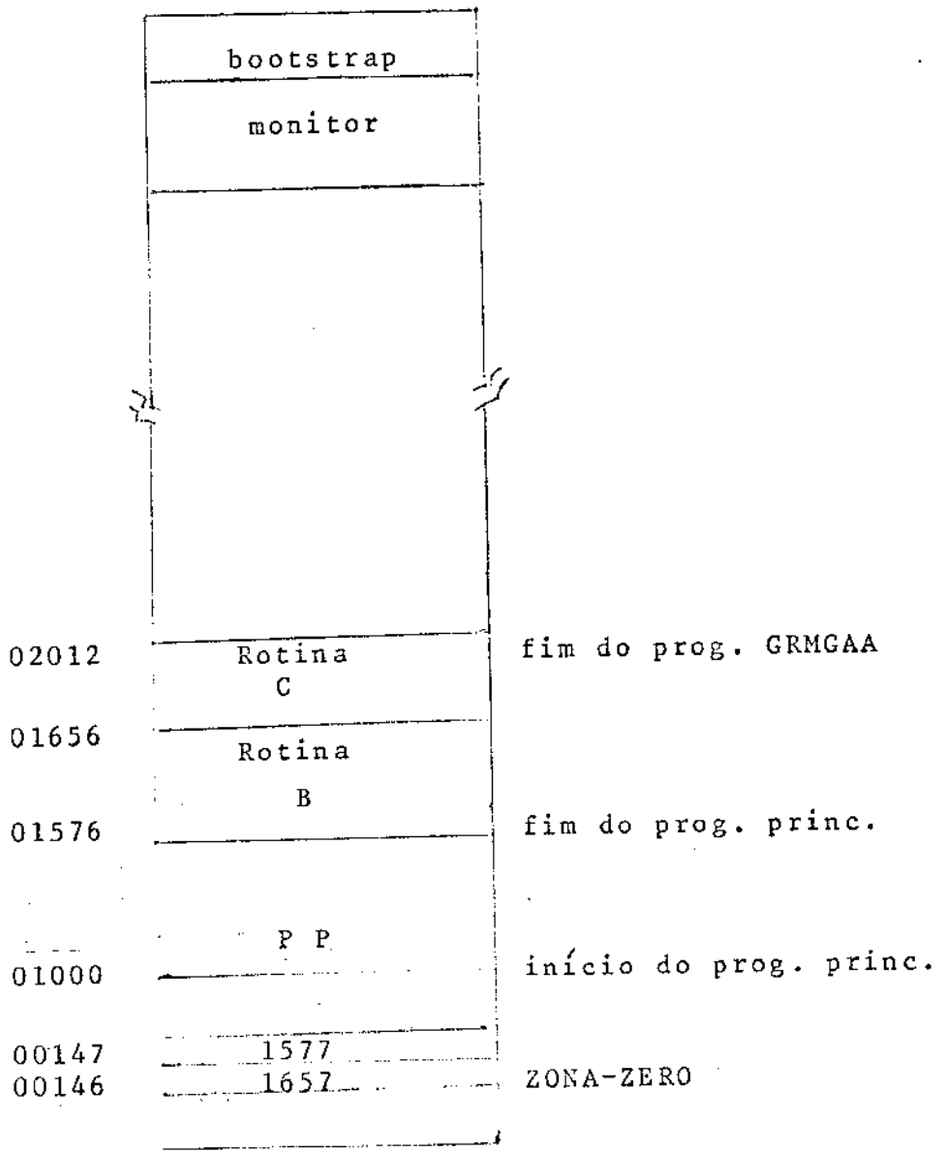


Fig. 3.8

3.4 - CONSIDERAÇÕES SOBRE O SISTEMA OPERACIONAL DO EAI-640

A montagem do programa em linguagem de máquina é realizada, por limitações do sistema operacional do EAI-640, no mesmo periférico onde se encontram o programa objeto e as sub-rotinas isoladas.

Este tipo de limitação não seria grave em sistema de computação como os usados nos setores comerciais, que necessitam uns poucos programas apenas, utilizados intensivamente sobre uma grande quantidades de informações, ou ainda,nos computadores usados para controle de processos industriais. De maneira geral pode-se afirmar que tal limitação, não é importante em processadores cuja utilização seja bastante específica.

Tal não é entretanto, o caso do sistema EAI-640, destinado primordialmente a ser utilizado em pesquisas científicas. Isto exige do sistema um suporte de programas e sub-rotinas para pronta utilização, bastante extenso e variado; o que implica na existência de meios auxiliares de armazenamento com grande capacidade. Além disso, é importante que a fase de preparação para a execução de um programa (compilação e carregamento), sejam muito rápidas e, quando possível, realizados automaticamente.

O requisito básico-meios sufientes de armazenamento de programas-esta satisfeito para o sistema de computação da Faculdade de Engenharia de Campinas, dada a existência de quatro unidades de fita magnnética "cassette".

O segundo requisito também esta satisfeito, no que diz respeito ao "hardware", pois existe uma unidade de disco de cabe-

ças fixas, extremamente rápida (tempo médio de acesso à informação 17 ms.).

Entretanto, no sistema atual é impossível operar simultaneamente as duas unidades, dada a limitação do sistema operacional existente.

A realização de um sistema operacional que permita a utilização conjunta das duas unidades é o objetivo deste trabalho e será exposta no capítulo que se segue.

CAPITULO IV

4. MONITORAMENTO PROPOSTO

4.1 - DISCUSSÃO

Nos capítulos precedentes, procurou-se oferecer conhecimentos sôbre as capacidades, limitações, enfim, uma certa familiaridade com a máquina, em uma tentativa de levar o leitor ao encontro do problema que se propos no capítulo introdutório.

Passa-se a ter agora, condições de se abordar o problema objetivo deste trabalho. Esta abordagem será uma minunciosa descrição do que se fez e do novo monitoramento, assim como em momento oportuno, chama-se a atenção às vantagens introduzidas, com suas respectivas justificativas. Um diagrama de blocos é também apresentado.

Tanto o monitor do disco, como o "librarian" da fita magnética de "cassette", são programas montados através de sub-rotinas e subprogramas, com funções específicas, as quais serão descritas com mais detalhes posteriormente. Muitas destas sub-rotinas são executadas através de comandos do próprio monitor ou requisitadas pelo sistema de "software", ou pelo programa do usuário. As sub-rotinas do monitor, são carregadas nas posições mais altas da memória através do "boots trap" (partida fria). Normalmente elas residem em um periférico que oferece rápido e fácil acesso para o seu carregamento (disco, por exemplo). Essas sub-rotinas não podem serem superpostas por outros programas, com mais rigor ainda o

"bootstrap", numca poderá ser superposto.

O principal subprograma do sistema de monitoramento é o Processador de Controle de Opções (COP). Este subprograma não é residente na memória principal, mas auxilia o processamento sequencial e é carregado nos núcleos de memória pelas rotinas do monitor.

4.2 - PROGRAMAÇÃO DO MONITOR PROPOSTO

Se o programa monitor do disco e o programa "librarian" da fita magnética de "cassette", são representados em fluxogramas ou se forem montados os algoritmos destes dois programas, tais representações exibiriam uma estreita semelhança, podendo mesmo serem tratados como um único programa, tamanha é a semelhança dos dois. Mesmo as rotinas que executam funções idênticas para disco e "cassette", recebem os mesmos nomes. Estes dois programas, realmente, desempenhan os mesmos objetivos, do ponto de vista operacional. Pois de um lado o monitor exerce o controle executivo entre memória-disco-memória-periféricos, enquanto de outro, o "librarian" exerce o controle executivo entre memória-fita-memória-periféricos.

Entretanto, apesar desta estreita semelhança em termos globais, êles apresentam sensíveis diferenças, quanto a programação de algumas de suas sub-rotinas, que executam tarefas em um determinado dispositivo. Estas diferenças, entretanto, são devidas entre outras coisas, às características físicas da unidade do "cassete" e do disco propriamente dito; isto faz com que os registros que compoem um arquivo, possuam estruturas de armazenagens di

ferentes. Por exemplo, os parâmetros de um arquivo em disco tem - ligeira diferença para os arquivos em "cassette", justificamdo então programações diferentes de sub-rotinas. Mas, por outro lado , existem aquelas sub-rotinas que executam suas tarefas independente do dispositivo disco ou fita, é o caso das subrotinas que leem e escrevem mensagens nos periféricos externos da máquina.

Com as considerações acima descritas, e informações até aqui apresentadas, pode-se falar da programação do novo monitor.

4.2.1. - Descrição das subrotinas

Em programas operacionais, ou programações que são relativamente longas, torna-se conveniente e melhor de serem desenvolvidas, programações seccionadas em um número conveniente de sub-rotinas e subprogramas mais ou menos independentes, os quais podem ser escritos e testados independentemente. Existirá, entretanto, uma referência-cruzada entre as sub-rotinas e subprogramas . Muitas destas refêrencia-cruzadas na programação do monitoramento proposto, são feitas através da instrução de TRAP. Esta instrução, causa uma interrupção normal no fluxo do programa, onde ele é desviado para uma rotina especial, que maneja interrupções causadas por este tipo de instrução. Nesta rotina são pesquisados os endereços das sub-rotinas solicitadas, que é para onde será desviado a execução normal do programa. É também nesta rotina, que maneja as interrupções causadas pelas instruções de TRAP, que, através de uma programação especial, é pesquisado o endereço de retorno ou para onde será desviado a sequência normal de execução do programa, obviamente, depois de ter executados as sub-rotinas solici

tadas.

A seguir dá-se uma descrição pormenorizadas de todas - sub rotinas do monitoramento proposto.

O programa que se propõe como monitoramento do sistema operacional do computador digital EAI - 640, é produto da reunião dos programas monitor do disco e "librarian" do sistema de fita "cassette". Obviamente, por se tratarem de programas, que executam funções idêntivas e terem razoáveis semelhanças, esta unificação não se limitou somente a simples união dos dois programas, i. é, pegar de um lado o "Librarian" e de outro o monitor e processá-los como um único programa. Para executar-se esta unificação, - criou-se rotinas e subprogramas, para o perfeito ajuste e encaixe dos dois programas, vizando sempre sua maior eficiência e o menor uso possível de áreas na unidade de memória.

- .LOAD, carrega as rotinas. CARD e .UNPK para a memória. Esta - sub-rotina executa operação tanto para "cassette" como disco , portanto independe do periférico. A rotina .CARD aciona a leitura do cartão. A rotina .UNPK interpreta os catões lidos.

  Sequencia de chamada de rotina:

  T .LOAD

  não existe êrro de retôrno, se ocorrer êrros causa uma pausa na máquina.

- .MNAM, cria registro de identificação para um novo arquivo, colocando no diretório do disco, o nome do novo arquivo. A rotina .CNAM, registra na fita "cassette", o nome do novo arquivo,-

depois de haver sido posicionada pela rotina .CPOS, um local vazio.

Sequência de chamada:

```
T      .MNAM
J      ERRO
```

para disco

```
T      .CNAM
J      ERROR
```

Para fita.

O retorno de erro existe sempre que o controlador dos periféricos detetar algum erro, quando a capacidade do disco ou fita tiverem sido excedidas e naturalmente, quando já existem arquivos com o mesmo nome deste novo arquivo.

- .MPOS e .CPOS, rotinas de posicionamento de arquivos, disco e fita magnética de "cassette", respectivamente.

Sequencia de chamada:

```
T     .MPOS
J     ERROR     , ou
T     .CPOS
J     ERROR     para disco e fita, respectivamente.
```

O retorno de erro para o disco acontece quando o fim do espaço definido pelo diretório foi atingido sem encontrar o nome especificado. Para a fita quando existem falhas ou erros de "hardware" ou ainda quando é detetado o fim da fita magnética.

- .MTTO, esta rotina causa a impressão de mensagens, pelo teleti

po ou impressora de linha.

Serve tanto para disco como a fita, podendo ser chamada pelo usuário. Sequência de chamada

```
LX    BUFP
T     .MTTO
ADR   END
```

Esta rotina não permite retorno de erro, os caracteres empacotados dois por palavra vem numa área identificada pelo endereço BUFP. A saída para impressão dos caracteres termina depois que se dá o comando de retorno do carro da impressora (CR) ou depois que foi impresso o caracter definido no endereço END.

- .MTKI, aceita a entrada de uma linha através de teletipo, finalizada com o comando de retorno CR, dado no carro da impressora. Aceita também a entrada de caracteres pela Leitora de cartões, com o auxilio de .CARD e .UNPK, e armazena estes caracteres sequencialmente, um caracter por palavra. Esta rotina é acessível ao usuário, podendo ser usada tanto para disco como para a fita. Sequência de chamada

```
LX    BUFP
T     .MTKI
ADR   END
```

As mesmas considerações apresentadas na rotina .MTTO para os endereços BUFP e END, tem validade aqui. Exemplo de como é usado o endereço BUFP que contém o endereço da mensagem a ser dada pelo teletipo:

```
BUFP    ADR   *+1
        BCI   4, mensagem
```

- .MSIN e .CSIN, lê uma informação de 16 bits em disco e fita respectivamente. É usada para pesquisar nomes de arquivos, ou identificações que ocupem algumas palavras de 16 bits.

- .MSOU e .CSOU, escreve uma informação de 16 bits.

Sequência de chamada para utilização com o disco

```
T     .MSIN  ou .MSOU
ADR   BEGIN
ADR   END
J     ERROR
```

Para a fita é idêntico, obviamente, mudando-se o nome das rotinas para CSIN e CSOU. O retorno de erro acontece quando existe tentativa de escrever em áreas protegidas, erro de paridade ou mal funcionamento de "hardware". Os endereços BEGIN e END, definem o número de palavras e sua localização na memória.

- .MSP5 e .CSP5, escreve no final de cada arquivo, a idenfificação EOF (end-of-fi e), para fechar o arquivo.

Sequência de chamada

```
T     .MSP5  ou .CSP5
J     ERRO
```

O retorno de erro acontece sempre que se define o número da unidade periférica não coincide com as do disco (21, 22,23, 24) e fita (10, 11, 12, 13) ou ocorra erro de paridade ou "hardware"

- .MSP4 e .CSP4, para o disco a rotina .MSP4, procura o início do

diretório, e na fita procura o seu início.

Sequência de chamada para disco e fita, respectivamente:

T .MSP4 ou .CSP4

J ERRO

Os erros ocorrem quando existem mal funcionamentos de hardware ou número impróprio da unidade.

- .MSP3 e .CSP3, rotinas usadas para pular um arquivo existente em disco ou fita.

T .MSP3 ou .CSP3

J ERRO

Logo após sua operação ela aponta para o primeiro registro do - proximo arquivo. Os erros de retorno, acontecem quando existe mal funcionamento de hardware, número improprio da unidade.

- .MSP2 e MSP1, posiciona o arquivo ou um registro, respectivamente, imediatamente anterior. Usada somente com o disco.

Sequencia de chamada

T .MSP2 ou .MSP1

J ERRO

Ocorrem erros quando há mal funcionamento de hardware ou número impróprio da unidade.

As rotinas definidas abaixo, são usadas somente pelo sistema de fita magnética de "cassette", sendo normalmente chamadas pelo usuário e recebem o nome de "rotinas ou programas de catalogações" (programa scheduller), desenpenhando as funções de ordenar ,

estabelecer prioridades e supervisionar a execução das tarefas exigidas nos processamentos de programas.

- Rotina PSINIT, esta rotina inicializa o sequenciador de programas, transferindo o controle para o programa de onde veio.
- Rotina PRGDEF, é uma rotina que define um novo programa para o sequenciador de programas.
- Rotina PRGACT, ativa um especificado programa. O controle sempre retorna, para a sequência de onde foi chamada.
- Rotina PRGDAC, desativa um especificado programa. O controle sempre retorna, para a sequência de onde foi chamada.
- Rotina PFETCH, é uma rotina que fornece ao programa ativado, um apontador que o identifica.
- Rotina PRGHCA, ativa um programa especificado armazena o estado da máquina, para o programa anterior e o torna inativo. Todos os registros são armazenados, exceto o acumulador.
- Rotina PRGHTC, armazena o estado da máquina e torna o corrente programa inativo. O programa continuará, quando for reativado. Todos os registros são armazenados, exceto o acumulador.
- Rotina SAVACT, esta rotina sendo chamada pelo usuário, ativa um determinado programa, dada a ocorrência de um sinal de interrupção. A rotina compara a prioridade do programa interrompido com o interruptor e transfere o controle para o de maior prioridade.
- Rotina PRGREL, torna o corrente programa ativado, mas transferindo o controle para o próximo programa de maior prioridade.

- Rotina PRGTRM, remove um nome da lista de programas a serem executados, finalizando, portanto, a sua execução.
- Rotina PSCAN, é usada pelo sequenciador para procurar os programa que tenham sido ativados e transfere o controle para o de maior prioridade.
- Rotina RSAVE, usada para armazenar o registro Q e posições da ZONA ZERO de memória.
- Rotina RESTOR, usada pelo sequenciador para transferir o controle, para um programa especificado pelo registro X e restaurar registros e as posições da ZONA ZERO De memória.

Todos os símbolos de comando de controle que dirigem o monitor ou o "librarian, para criar, apagar e manipular arquivos, foram mantidos inalterados no monitoramento proposto, tendo em - vista que,as alterações destes comandos, acarretaria mudanças em - todo sistema de "software" do EAI-640. Portanto, procurou-se de certa forma, um entrosamento entre os sistema de monitoramento implantado e o proposto,quanto aos seus comandos executivos e controle, sendo que, uma das únicas modificações introduzidas, no que diz respeito a comando de controle, é o símbolo de fechar um arquivo, i.é, o símbolo #C da fita magnética de "cassette", foi alterado para #F, Isto para distinguir no comando #C de fechar um arquivo no disco.

4.3 - DIAGRAMA DE BLOCOS DO MONITORAMENTO PROPOSTO

O diagrama de blocos que se apresenta a seguir, mostra

como o programa monitor, coordena a operação do EAI-640.

Os símbolos e comandos apresentados no diagrama, correspondem exatamente com os comandos que se dá à máquina. Neste fluxograma esta claro que, após o término de qualquer ramificação, o controle retorna sempre ao início do monitor, preparando-se para um novo comando.

As seguintes convenções de comandos são usadas:

- NOMEXX, de um a seis caracteres, em código ASCII, determina o nome de um arquivo. Se NOMEXX, tem menos que seis caracteres, os espaços não utilizados são preenchidos - com zeros.

- EOF sinal que identifica o final de um registro ou arquivo Na fita magnética de "cassette", dois sinais EOF seguidos identifica o final de um arquivo.

- T este símbolo designa o tipo de arquivo, onde:
  0 - arquivo fonte,
  1 - arquivo em formato objeto,
  2 - arquivo em linguagem de máquina,
  3 - arquivo em formato de dados.

- BLCT designa o tamanho de um registro de arquivo; aplicável somente para arquivo de dados.

-C indica para a leitora de cartão, a entrada de um cartão de controle:
  1 - controle retorna para a leitora de cartão,
  0 - controle retorna para o teclado do teletipo.

- END abreveatura de endereço. Como exemplo, no comando

#G,END este END é o endereço do início de execução - do programa carregado na memória;

END_I endereço inicial,

END_F endereço final,

O sistema de computação EAI-640, manipula endereços so em código octal.

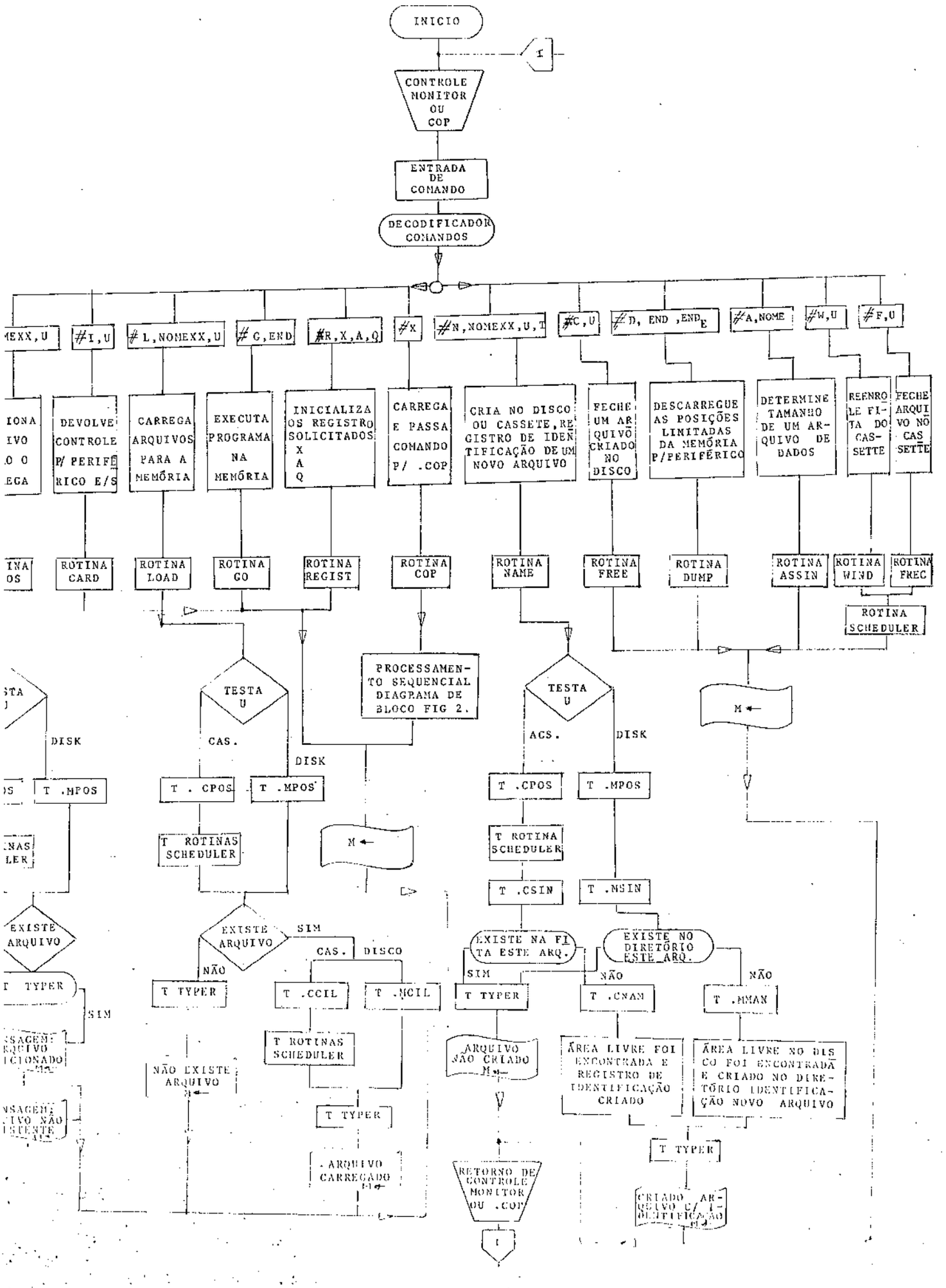
- A O conteúdo que deve ser colocado no acumulador ou registro A.
- X conteudo a ser colocado no registro de indexação X.
- Q conteudo a ser colocado na extensão do acumulador, ou registro Q.
- K indica o conteúdo a ser colocado no registro de estado da máquinaZou simplesmente registro K.
- # identifica um particular início de mensagem de controle ao monitor, através do teclado do teletipo.
- ↵ " return" (CR), símbolo de retorno do carro da impressora, é usado para terminar uma particular mensagem de controle e comando do monitor, ou a entrada - de uma linha pelo teclado do teletipo.
- U este símbolo identifica a unidade lógica do periferico de menória auxiliar que se usa:

  21, 22, 23 e 24 para a unidade de disco,

  10, 11, 12 e 13 para a unidade de fita magnética do - "cassette".

A notação T SUBR, onde SUBR, é o nome de uma sub--rotina, é uma instrução de TRAP, que gera uma interru

pção no fluxo normal do programa. desviando para a rotina TRAPP, que maneja as interrupções, sendo que, através de uma programação especial, encontra-se o endereço da sub-rotina SUBR, para onde será desviada a sequência de programação.


INICIO
1
CONTROLE MONITOR OU COP
ENTRADA DE COMANDO
DECODIFICADOR COMANDOS
#I,U
#L,NOMEXX,U
#G,END
#R,X,A,Q
#X
#N,NOMEXX,U,T
#C,U
#D, END ,END
#A,NOME
#W,U
#F,U
DEVOLVE CONTROLE P/ PERIFÉRICO E/S
CARREGA ARQUIVOS PARA A MEMÓRIA
EXECUTA PROGRAMA NA MEMÓRIA
INICIALIZA OS REGISTRO SOLICITADOS X A Q
CARREGA E PASSA COMANDO P/ .COP
CRIA NO DISCO OU CASSETE, REGISTRO DE IDENTIFICAÇÃO DE UM NOVO ARQUIVO
FECHE UM ARQUIVO CRIADO NO DISCO
DESCARREGUE AS POSIÇÕES LIMITADAS DA MEMÓRIA P/PERIFÉRICO
DETERMINE TAMANHO DE UM ARQUIVO DE DADOS
REENROLE FITA DO CASSETTE
FECHE ARQUIVO NO CASSETTE
ROTINA CARD
ROTINA LOAD
ROTINA GO
ROTINA REGIST
ROTINA COP
ROTINA NAME
ROTINA FREE
ROTINA DUMP
ROTINA ASSIN
ROTINA WIND
ROTINA FREC
ROTINA SCHEDULER
TESTA U
DISK
CAS.
ACS.
PROCESSAMENTO SEQUENCIAL DIAGRAMA DE BLOCO FIG 2.
M
T .MPOS
T . CPOS
T ROTINAS SCHEDULER
T ROTINA SCHEDULER
T .CSIN
T .NSIN
EXISTE ARQUIVO
SIM
NÃO
T TYPER
T .CCIL
T .MCIL
T .CNAM
T .MMAN
EXISTE NA FITA ESTE ARQ.
EXISTE NO DIRETÓRIO ESTE ARQ.
NÃO EXISTE ARQUIVO
ARQUIVO NÃO CRIADO
ÁREA LIVRE FOI ENCONTRADA E REGISTRO DE IDENTIFICAÇÃO CRIADO
ÁREA LIVRE NO DISCO FOI ENCONTRADA E CRIADO NO DIRETÓRIO IDENTIFICAÇÃO NOVO ARQUIVO
. ARQUIVO CARREGADO
RETORNO DE CONTROLE MONITOR OU .COP
CRIADO ARQUIVO C/ IDENTIFICAÇÃO

4.4 - CONSIDERAÇÕES SOBRE O MONITOR PROPOSTO

O procedimento usado na programação do monitor proposto consistiu de uma conveniente unificação dos dois programas: monitor do disco e "librarian" da fita magnética "cassette". Obviamente, como estes são programas que operam individualmente o sistema, todos os programas da máquina: COP, os processadores de linguagens montadores, enfim todo o sistema de "software" do EAI-640, se comunicam ou se utilizam de um dos dois programa monitores.

Os programas montadores (PL, CIG) quando carregados na memória para executar suas tarefas, ocupam áreas logo abaixo dos programas monitores. Posições idênticas, em relação aos programas monitores, também são tomadas por outros programas do sistema de "software", quando carregados na memória.

Com as considerações apontadas acima, para a implantação do monitor proposto é necessária uma reorganização de todo o sistema de "software", para compatibilizá-los. Esta reorganização constou de:

- relocação dos programas que se situam logo abaixo do monitor;
- modificação dos programas que se utilizam do monitor nos pontos onde esta ligação é requerida;
- adaptação do "bootstrap" para aceitar o novo monitor.

CAPÍTULO V

5. CONCLUSÕES

O sistema de computação digital EAI-640, dada a sua condição de mini-computador, apresenta características de arquitetura e confiabilidade que o tornam uma máquina de uso geral.

O conjunto de compiladores, editores, carregadores, enfim, o seu "software", permite a utilização efiente e fácil de todas as possibilidades da máquina.

Porém, provavelmente por problemas de ocupação de memória, o fabricante não oferece um sistema operacional,que permita a utilização conjunta dos vários tipos de unidades auxiliares de memória, que podem ser aclopadas ao computador.Como foi analisados nos capítulos precedentes,tal situação, deixa muito a desejar de um sistema,cuja utilização principal se faz em aplicações científicas.

Tendo por objetivo eliminar esta insuficiência, foi realizado, para o sistema de computação do Laboratório de Simulação Híbrida, da Faculdade de Engenharia de Campinas, um novo sistema operacional, descrito anteriormente, capaz de comandar simultaneamente os dois periféricos de memória existentes.

Os resultados obtidos com o monitoramento, propriamente dito, podem ser considerados satisfatórios do ponto de vista de utilização de memórias, uma vez que a sua versão final,operando os dois periféricos excedem o antigo monitoramento em apenas 950 palavras.

Para maior clareza, com respeito à ocupação de memórias temos os seguintes dados:

	ÁREA OCUPADA
MONITOR ANTIGO	1683 palavras
MONITOR ATUAL	2633 palavras

Figura 5.1

Além da programação do novo monitor, para a implantação do sistema operacional proposto foi necessária uma reorganização em praticamente todo o sistema de "software" do computador, para que este pudesse se beneficiar das novas possibilidades existentes. Como foi dito no capítulo IV, esta reorganização constou basicamente de relocações e adaptações, feitas sem, praticamente, nenhum aumento da área ocupada de memória.

Com respeito à distribuição dos vários programas constantes do "software" do sistema pelas duas unidades periféricas (disco e fita magnética "cassette"), o critério adotado foi o grau de utilização de cada um, confrontado com a capacidade de armazenamento limitada do disco. Assim, colocou-se nos discos os compilado

res FORTRAN, ASSEMBLER e o interpretador HOI, assim como a biblioteca de sub-rotinas (RTL) usadas implicitamente por estes programas , e também o carregador (CIG) e algumas sub-rotinas largamente utilizadas, como as de ligações com o sistema híbrido (LINKN) e as traçadoras de gráficos.

Em fitas magnéticas ficaram todos os outros programas , ou seja, bibliotecas de rotinas científicas, editor de texto, diagnósticos, depuradores, etc., que entretanto, tem utilização menos freguentes .

REFERENCIAS

1. Jean-Pierre Meinadier - "Structure et Fonctionnement des O Ordinateurs" - Paris, Larousse, 1971

2. Donovan - "Systems Programming" - New York, McGraw-Hill, 1972

3. D. W. Barron - "Assemblers and Loaders" - London, 2 ed., Macdonald, 1972.

4. Flores, I. - "Computer Software", N.J. Prentice-Hall, Inc. 1966.

5. D. Morris, G. D. Dettenfsen, G.R. Frank and T.J. Sweeney - "The Structure of the MU5 Operating System". The Computer Journal, vol 15, nº 2, 1971.

6. James D. Schoeffler and Ronald H. Temple - "Minicomputers : Hardware, Software, and Aplications". New York, IEEE Press, 1972.

7. Manuais de referências do sistema Digital EAI - 640 - publicações 1971.
   - Disk Operating System Manual
   - Cassette Tape Operating System Manual
   - Reference Handbook, EAI 640
   - Pacer 100 - Digital Computing System - Reference handbook - 1973.

8. Julius T. Tou - "Software Engineering" - New York, Academic Press, 1970

9. P.A. Kalinchenko, I.S. Lupashina, V.A. Makarov, A.S. Markov, I.V. Popova and S.N. Sokolov. "Report on a Monitoring System of Programming Automation and a Translator from Fortran for the MINSK-22 Computer". URSS, Computational Mathematics and Mathematical Physics, Vol 10, nº 2,1970.

10. Chu, Y. - "Computer Org. & Micro-Programming". Englewood Cliffs, N.J. : Prentice-Hall, Inc., 1970.

11. "Special Issue on minicomputers" - Proceedings of the IEEE, vol 61, nº 11, november 1973.