Wagner Ideali

Sistema Operacional

CP/M 80

ÉRICA

PUBLICAÇÕES EM ELETRÔNICA

Rádio — Propagação

Envolve propagação de ondas longas até microondas, ondas ópticas e seus meios, bem como, atmosfera, guias de onda, fibras óticas e os métodos de Propagação, através de estudos da:

Reflexão, Refração, Zonas de Fresnel, Princípio de Huygens, Critérios de Rayleigh, Antena, Radar, Satélites, etc.

N.º de páginas: 168

Autor: **Jaroslav Smit**

Física — Volume 01

Aborda Funções e Gráficos; Grandezas e Medidas; Movimento Retilíneo Uniforme e Movimento Retilíneo Uniformemente Variado; Queda dos Corpos; Vetores; Velocidade Relativa; Composição de Movimentos; Leis de Newton; Energia.

Em cada capítulo contém uma série de exercícios resolvidos e exercícios propostos. O Apêndice contém dezesseis experiências propostas de fácil execução.

N.º de páginas: 304 1.ª Edição.

Autor: **Rocco Lence**

Eletrônica de potência

O livro aborda o estudo dos Conversores Estáticos, implementados com Tirístores. Seqüencialmente são tratados: classificação dos Conversores, em forma resumida e com uma análise detalhada, fixados com exemplos numéricos e, aplicação de Conversores no acionamento de motores elétricos.

N.º de páginas: 300 — 1.ª Edição

Autor: **José Luiz Antunes de Almeida**

Sistema

Operacional

CP/M 80

Dados de Catalogação na Publicação (CIP) Internacional
(Câmara Brasileira do Livro, SP, Brasil)

Ideali, Wagner, 1957-
I22s Sistema operacional CP/M-80 / Wagner Ideali. --
São Paulo : Érica, 1986.

Bibliografia.

1. CP/M-80 (Sistema operacional de computador)
I. Título.

86-0902 CDD-001.6425

Índices para catálogo sistemático:

1. CP/M-80 : Sistema operacional : Computadores : Processamento de dados 001.6425

Wagner Ideali

Sistema Operacional

CP/M 80

LIVROS ÉRICA EDITORA LTDA.

1986

Produção Editorial:
PAULO ROBERTO ALVES

Revisão de Português:
ROSÂNGELA CRISTINA LIMA

Desenhos:
JOSÉ DE FARIA TOLEDO

Arte Final:
ROSANA ARRUDA DA SILVA

Supervisão:
ROSELI BARBARESCO DE OLIVEIRA

Capa:
JOSÉ DE FARIA TOLEDO

LIVROS ÉRICA EDITORA LTDA.
Rua Jarinu, 594 - Tatuapé - São Paulo
Fone: 294-8686 - C.G.C. 50.268.838/0001-39
Caixa Postal 15.617

Dedico à
Minha Esposa
Walkiria e
Meus Filhos
Thiago e Thayná.

PREFÁCIO

O objetivo básico deste livro é dar ao leitor inicia
te uma explicação detalhada sobre o sistema operacional CP/M-80. Dividido em sete capítulos, mostraremos as partes principais, ma
nuseio, aplicações e alterações neste sistema operacional.

No primeiro capítulo mostraremos o que é um sistema operacional e os mais comuns do mercado.

No segundo capítulo teremos o CP/M-80 com suas ver
sões, modificações, aplicações e etc.

No terceiro e quarto capítulos há explicação detalhada so
bre os comandos internos e externos do CP/M nas várias versões.

No quinto capítulo temos as informações sobre os apli
cativos mais comuns para CP/M (M8Ø, ED, FORMAT, MBASIC, PIP, etc).

O sexto capítulo é dirigido para parte puramente técni
ca do CP/M, ou seja, alterações, preparações para o computador "ro
dar" este sistema operacional e etc. Estas informações são impor
tantes, pois muitos computadores podem ser alterados para "rodar" o CP/M.

Lembramos que o sexto capítulo é dirigido aos leitores com bons conhecimentos de Software e Hardware.

No último capítulo montamos o que seria uma aplicação de um computador com CP/M, ou seja, criar um programa, editar, compilar e retirar possíveis defeitos e por fim sua utilização. Este capítulo é um apanhado geral do livro.

Lembramos aos leitores que este livro é dirigido àque
les que têm alguns conhecimentos sobre microprocessadores, das famílias 8Ø8Ø, 8Ø85 e Z8Ø. Procure na medida do possível ler este livro com um computador com CP/M à frente, pois ficará muito mais fácil você ao estudar um comando, um programa ou qualquer opera
ção do CP/M poder repetir (testar) no computador o aprendizado.

Espero, com este livro, alcançar o objetivo de levar ao leitor algum conhecimento básico sobre CP/M (Marca Registrada da Digital Research), colaborando assim com o preenchimento da lacuna existente na área de livros didáticos.

O autor

SUMÁRIO

INTRODUÇÃO

Antes de adentrarmos ao assunto CP/M-80 vamos abordar o que seja basicamente um computador, suas partes básicas e o sistema operacional, para assim formarmos uma idéia global do assunto. Desejamos com isso trazer você à nossa linha de raciocínio.

Diagramas em Blocos de um Computador

Um computador é formado basicamente de uma unidade de memória onde são armazenados os programas e dados transientes. Esta memória pode ser ROM (READY ONLY MEMORY), com programa armazenado ou RAM (RANDON ACCESS MEMORY) de acesso aleatório para programas e dados, sendo que a última é do tipo volátil.

O computador tem uma unidade central de processamento o qual podemos chamar de o "coração do computador". Esta unidade é a responsável pela interpretação das instruções existentes na memória de programa.

Para comunicação entre homem e máquina existem os periféricos, ou seja, teclado, vídeo, display, impressora, etc.

Os programas e dados são armazenados em memória de massa que são as fitas K-7, discos, etc. Outro periférico é a impressora para tirar listagens ou qualquer tipo de cópia escrita de informações.

Na figura 1.1 temos o diagrama em bloco onde visualizamos todo o conjunto. Podemos verificar que todos os periféricos e memórias estão ligados na CPU. Um programa especial deverá existir para controlar, gerenciar e ordenar todo o conjunto e este programa é o sistema operacional.

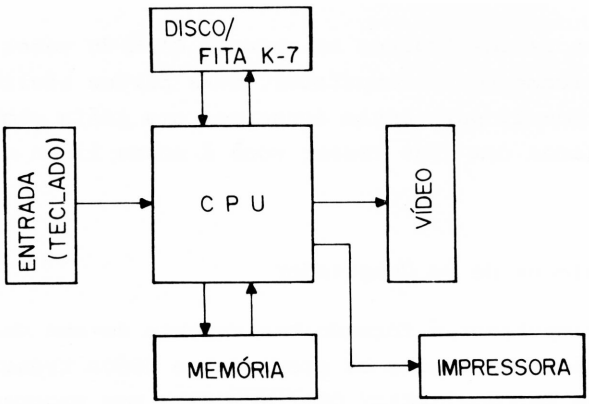


Figura I.1 - Diagrama em blocos de um computador.

O trabalho do sistema operacional é o de receber os dados vindos de um dispositivo de entrada (Ex. Teclado), processá-los, controlar o vídeo, organizar os arquivos em um disco, possibilitar a impressão de dados arranjando-os de forma conveniente, organizar devidamente os bancos de memória, etc., de onde concluímos ser sua função bastante complexa.

Antes de abordarmos os mais conhecidos sistemas operacionais (os mais populares), passemos a uma rápida revisão de alguns termos importantes à compreensão do que pretendemos desenvolver.

BIT

O Bit é a menor porção de processamento, podendo assumir apenas os valores Ø (zero) e 1 (um). Todas as informações contidas num computador, estão na forma binária, ou seja, representados por conjuntos de Bits.

Ex. ØØ111ØØ1 = conj. de 8 Bits = cod. ASCII 39 = Número Decimal 9.

BYTE

O Byte corresponde a um conjunto de 8 Bits, sendo que

todas as informações contidas no computador (nas unidades de memória) ou processadas por ele, o são na forma de Bytes.

Ex: ∅1∅1∅111 = 1 Byte = Cod. ASCII 57 = Caracter W

Obs: Veja no Apêndice "D" a tabela de códigos ASCII

Memória

É o local onde os dados vindos de um dispositivo de entrada (periférico) serão armazenados para futura utilização. O conjunto de células de memória forma um bloco denominado por unidade de memória, onde os dados ou programas são armazenados, em forma binária, para serem processados. Existem vários tipos de memória: a RAM (Randon Access Memory) onde os dados são armazenados temporariamente, permanece sendo válidos, enquanto mantivermos o computador energizado; a ROM (Read Only Memory) onde os dados são armazenados de forma definitiva independendo de o computador estar ou não energizado, e outros tipos aqui não descritos.

Disquete

O Disquete, ou disco flexível, é a forma mais comum nos dias de hoje para guardar grandes quantidades de dados (memória de massa) de forma rápida e relativamente barata.

Quando você for adquirir um Disquete será necessário verificar algumas características:

- Disquete de 8 ou 5 1/4 de polegada.
- Disquete de face simples ou dupla
- Disquete de densidade simples ou dupla.

Os Disquetes de 8 polegadas são mais caros do que os de 5 1/4", devido ao fator da capacidade de armazenamento. Quanto à utilização de Disquetes de face simples ou dupla, isto dependerá do acionador de disco (Drive) ter dupla ou simples cabeça magnética. O custo evidentemente é superior no acionador de dupla cabeça. A densidade simples significa poder gravar 128 Bytes em cada setor (setor será elucidado logo adiante), sendo esta técnica padrão IBM, enquanto dupla densidade significa 256, 512 ou até mesmo 1024 Bytes por setor.

A superfície do Disquete é formada por uma série de círculos concêntricos denominados por trilhas, que por sua vez são subdivididas em setores, nos quais serão armazenados os dados. A trilha mais externa é denominada de trilha zero "Ø".

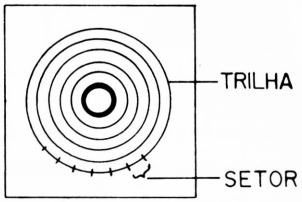


Fig. I.2 - Representação interna do disquete.

O CP/M original utiliza o Disquete de 8" com 76 trilhas, onde cada trilha é subdividida em 26 setores, e cada setor terá 128 Bytes.

Aspecto Físico

O Disquete é feito de um material magnético, idêntico aos da fita K-7 (Mylar). No apêndice "C" mostramos os cuidados a serem tomados no manuseio dos Disquetes.

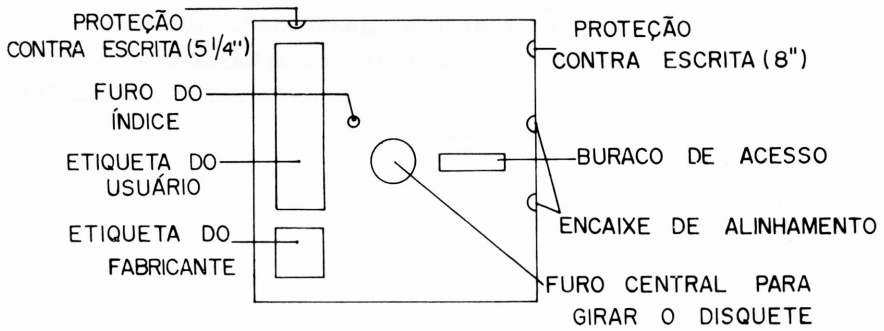


Fig. I.3 - Desenho de um Disquete típico.

Com um Disquete em mãos, podemos verificar que há uma superfície magnética protegida por um papelão, onde apresentou-se 03 furos, o buraco central, buraco do índice e o rasgo de acesso. Há também o entalhe lateral que é a proteção contra escrita. O buraco central é por onde o mecanismos do Drive pren

de-se ao Disquete para que este gire dentro da capa. O buraco do índice serve como indicador para o Drive. É através deste furo, que o Drive reconhece, via foto transistor, o início do Disquete (trilha Ø), ou seja, local onde deverá ficar o diretório do disco. Pelo rasgo de acesso é que a cabeça leitora do Drive entra em contato com o Disquete, a cabeça magnética move-se para frente e para trás, comunicando-se trilha a trilha. Pelo entalhe lateral pode-se proteger o disco contra a leitura.

CPU

A unidade central de processamento é o coração de um computador. No CP/M normalmente utiliza-se o microprocessador 8Ø8Ø, atualmente o Z8Ø e em alguns casos o 8Ø85. A CPU é a responsável pelo controle do computador, executando as ordens emanadas do sistema operacional. Todo o processamento acontece na CPU.

Impressora

Equipamento responsável pelo armazenamento dos programas em papel (HARDCOPY). Muito útil na impressão de relatórios, programas, etc, sendo uma importante ferramenta no desenvolvimento de Software. No CP/M a impressora é reconhecida como uma T.T.Y porque na época do desenvolvimento do CP/M as T.T.Y eram mais comuns, tanto que os consoles (vídeo) eram também raros.

1. OS SISTEMAS OPERACIONAIS

Existem sistemas operacionais espalhados pelo mercado da informática, como por exemplo o MP/M, CP/NET, UNIX, DOS e etc.

Todo o programa instalado num computador para gerenciar os arquivos de discos, controlar as atividades dos componentes deste computador e permitir a operação de programas transientes no mesmo pode ser chamado de sistema operacional.

Quando ligamos um computador sem um programa qualquer, este equipamento não tem nenhum valor. A partir do instante que instalamos um programa para executar uma tarefa específica, nós teremos um computador dedicado, mas no momento que instala-se um sistema operacional este computador tem agora múltiplas aplicações, ora na área comercial com programa afins, ora na área científica, técnica e até mesmo em desenvolvimento de outros computadores.

O sistema operacional se constitui num Software preparado para fazer todo o controle do computador. Normalmente ele se localiza em disco, e ao ligarmos o computador há um pequeno programa residente em EPROM que tem a função de transferir o sistema para a memória principal e daí passar o controle do computador para o mesmo.

Os sistemas operacionais mais comuns são o MP/M, CP/NET, CP/M-86, UNIX, TURBO-DOS, DOS, etc.

Quando ligamos o computador, o programa residente em EPROM, chamado carregador ou BOOTSTRAP se encarrega de transferir o sistema operacional que está no disco para a memória e transfere também o controle para o mesmo. A partir deste instante tudo o que for digitado no controle, como por exemplo comando, chamado de algum programa, edição de um programa, etc., está sobre controle deste Software chamado sistema operacional. Os sistemas operacionais são normalmente divididos em algumas partes, como por exemplo:

1 - Rotinas de comunicação com o usuário
2 - Rotinas de acesso e de controle de disco
3 - Rotinas de acesso à impressora
4 - Rotinas para formação dos comandos internos

Estas rotinas e muitas outras fazem uma função distinta a interligação entre elas forma o sistema.

O CP/M é um dos primeiros sistemas operacionais criado para microcomputadores, veremos mais adiante as partes que o compõe com suas respectivas funções.

O CP/M-86 foi desenvolvido para trabalhar com um microprocessador de 16 Bits, o 8086 da INTEL, enquanto que o CP/M-80 foi desenvolvido para trabalhar originalmente com 8080 da mesma empresa.

O sistema MP/M é chamado normalmente de CP/M para multiusuário ou seja, um sistema onde vários operadores trabalham no computador simultaneamente. O MP/M utiliza o microprocessador Z80. Outro sistema operacional é o CP/NET que permite a um computador usar os recursos de outro computador ou seja, utilizar sua impressora, Drives etc.

Para se utilizar o CP/NET é necessário que pelo menos um dos computadores tenha o MP/M, enquanto os outros devem usar o CP/M-80.

O sistema operacional UNIX vem mostrando no mercado que veio para ficar por muito tempo, ele tende a substituir o CP/M-80 e com muitas vantagens.

Foi desenvolvido para apoiar um ambiente de múltiplas tarefas e múltiplos usuários, de maneira que cada usuário tenha acesso integral aos recursos de um computador central em tempo compartilhado.

As suas características básicas são:

- Alto grau de portabilidade
- Independência de dispositivos
- A capacidade de iniciar processos assíncronos
- Linguagem de comando versátil para uma interface dinâmica e flexível com o usuário.
- Sistema que suporta redirecionamento de entrada e saída definido pelo usuário.
- Sistema de arquivos com organização baseada em estruturas hierárquicas.

A maioria dos sistemas operacionais nasceram do CP/M-80, desde o CP/M-80 1.3 até o UNIX. (O termo 1.3 significa o primeiro CP/M, o atual é o 2.2).

Na figura 1.1 temos um pequeno gráfico da evolução dos sistemas operacionais.

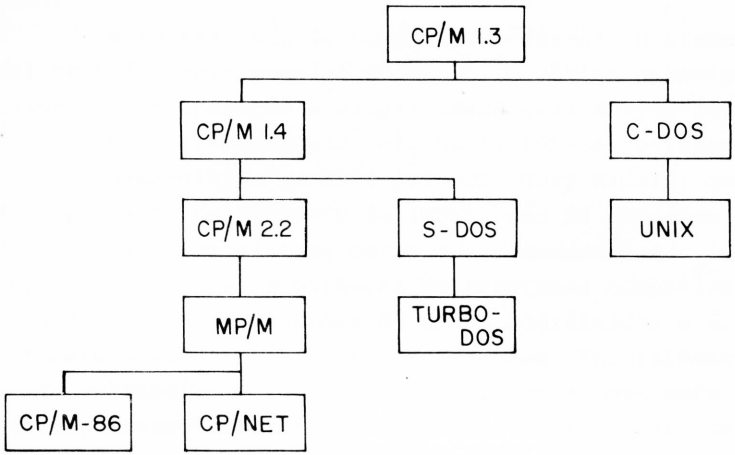


Fig. 1.1 - Diagrama dos sistema operacionais a partir do CP/M.

Um sistema operacional muito popular e de grande aplicação e importância é o DOS (Disk Operation System), ou seja, sistema operacional baseado em disco. Muito usado em microcomputadores hoje no Brasil, desenvolvido pela RADIO SHACK Americana, é usado nos computadores da linha TRS-8ϕ, tendo seu processo de operação muito parecido com o CP/M-8ϕ. O que torna o DOS de uma certa forma mais prático que o CP/M na hora de operar é o fato dele ser mais transparente ao usuário, ou seja, há mais mensagens no vídeo, maior quantidade de comandos através de uma função chamada HELP para auxiliar o usuário na manipulação dos comandos.

O objetivo deste livro é mostrar o funcionamento do sistema operacional CP/M, mas vamos dedicar uma atenção aos outros sistemas operacionais existentes no mercado, facilitando assim a compreensão do objetivo básico deste livro que é o CP/M.

Nos capítulos seguintes iremos mostrar em alguns comandos, informações, detalhes sobre o CP/M-86 e o MP/M que por serem muito populares é a base de onde nasceram os mais famosos e poderosos sistemas operacionais da atualidade.

2. O CP/M-80

Em se tratando de computador pessoal, o sistema operacional CP/M é o mais popular do mundo. O motivo principal é a facilidade na sua operação e simplicidade para adaptá-lo a qualquer computador (respeitando alguns critérios, evidentemente).

Desenvolvido em 1973 pelo DR. Gary Kidall, que na época era consultor de Software da INTEL, não se esperava que o CP/M se tornasse um sistema quase que Standard para microcomputadores, com milhares e milhares de programas compatíveis a este sistema e centenas e centenas de CP/Ms modificados e alterados para os mais diversos tipos de computadores. Inicialmente muitas empresas desenvolviam seus próprios sistemas, mas para redução de custos, muitos adotaram o CP/M como sistema para seus computadores.

Este sistema pode ser usado em qualquer computador que tenha como microprocessador 8Ø8Ø, 8Ø85 ou Z8Ø, 64K de memória RAM (alguns CP/Ms estão preparados para 16K ou 48K), uma unidade de disco de 5 1/4 ou 8 polegadas pelo menos (ideal seria duas no mínimo) e alguns complementos como impressora, modem, gravador de EPROM e etc.

Um dos primeiros programas criados para o CP/M foi o CBASIC e o seu predecessor o EBASIC. Desde então tem-se a cada dia desenvolvido novos programas para CP/M.

Um fato interessante aconteceu em 1981, quando foram criados equipamentos com o CP/M como opção, por fabricantes como IBM, HP, xerox e etc.

Com isso o número de usuários passou a marca de um quarto de milhão neste ano. Nos dias de hoje, o TRS-8Ø, Apple, PC, etc. têm placas adaptadoras para transformar seus computadores em máquinas compatíveis com o CP/M.

Provavelmente um usuário iniciante em posse de uma máquina com CP/M poderá não saber se seu computador está ou não preparado com placas, adaptações e etc. para "rodar" CP/M, então devemos verificar o seguinte:

- Linha TRS-8Ø

  Se sua máquina for um TRS-8Ø ou similar nacional, es

te deverá ter uma placa que está colocada sobre o microprocessador Z8Ø, dentro do computador, evidentemente, e nesta placa há o lugar do microprocessador, bem como 16K de RAM para completar com os 48K internos os 64K de RAM necessários à utilização do CP/M.

- Linha Apple

Se sua máquina for um Apple ou similar nacional, deverá ter uma placa com o Z8Ø num dos Slots internos. Esta placa contém além do Z8Ø algumas pastilhas TTL para controle. Juntamente com a placa do Z8Ø há uma placa exclusivamente no SLOT Ø contendo 16K de RAM. Existem algumas versões do Apple com o Z8Ø na própria placa principal.

Os dois computadores mais comuns no mercado da linha TRS-8Ø e Apple, existindo outras marcas, mas que não são compatíveis com CP/M.

Se você tem algum computador que não seja os acima descritos mas foi informado que o mesmo "roda" CP/M, deve verificar no manual do usuário do seu computador como operar o CP/M instalado. Note sempre que um computador construído inicialmente para trabalhar com um sistema operacional que não seja o CP/M, mas aceita o CP/M, ele deve normalmente ter uma placa de adaptação ou algo parecido.

No caso do TRS-8Ø e o Apple apenas com o simples fato de suas placas adaptadoras estarem instaladas e receberem o disco com CP/M, ao ligar o computador reconhece no disco o CP/M e irá trabalhar com este sistema.

Se desligar o computador e inserir o disco com o sistema operacional próprio do computador e tornar a ligá-lo, a partir daí passará a aceitar o sistema próprio.

Muitas modificações foram feitas no sistema original criado por Kindall, mas apesar de sua simplicidade, existe muito ainda para se aprender sobre o CP/M-8Ø.

A partir do instante que você obter um CP/M-8Ø para adaptar ao seu computador, receberá junto com um Disquete os seguintes manuais:

- An Introduction to CP/M Features and Facilities
- CP/M 2.2 User's Guide.

- Ed-A Context Editor for the CP/M Disk System
- CP/M Assembler (ASM)
- CP/M Dynamic Debugging Tool (DDT)
- CP/M 2.2 Alteration
- CP/M Interface Guide

Os manuais acima são originais da Digital Research, proprietária do CP/M-8Ø, sendo que alguns fabricantes de Software ao invés de fornecer estes manuais, desenvolveram os seus.

Deve-se lembrar que os manuais originais são complicados e dirigidos aos leitores com profundos conhecimentos técnicos. Tentamos assim mostrar neste livro uma forma simples e prática de utilizar e modificar o CP/M-8Ø.

Antes de entrarmos em detalhes mais profundos vamos abordar alguns termos técnicos importantes.

Ao Ligar o Computador

Quando você inserir um disco com CP/M no seu computador muitas coisas irão acontecer, após ligar o mesmo. Uma partida à frio significa carregar o CP/M que está no disco para a memória RAM do seu computador. Esta função é realizada por um programa que está em ROM que será desativada quando o CP/M estiver rodando. Alguns computadores têm uma partida à frio um pouco diferente, ou seja, a ROM não é desligada, mas fica no fim do mapeamento de memória (normalmente nos dois últimos K de memória). Como acontece esta carga, analisaremos no capítulo 7.

Após o computador carregar o CP/M na memória, o controle do sistema passa para o CP/M, ou seja, o programa CP/M começa a funcionar. Ele prepara convenientemente os periféricos, acerta uma série de dados, na seguinte ordem:

- O CP/M é carregado no memória RAM.
- O controle é transferido para o CP/M.
- O CP/M prepara rotinas de inicialização
- O CP/M apresenta-se na tela
- Aparecem os caracteres A> no vídeo
- CP/M aguardando comandos

Note que para o CP/M entrar no ar, aparentemente é sim

ples, sendo evidente para o operador que a inicialização também é simples, mas o que acontece internamente é relativamente complexo.

Em alguns computadores a partida à frio acontece automaticamente, bastando inserir o disco e ligar o computador para carregar o CP/M. Em alguns computadores, após ligarmos e inserirmos o disco, é necessário digitar-se através do teclado um código de inicialização.

Para computadores com partida automática a seqüência de ligar é a seguinte:

- Ligar os periféricos
- Inserir o Disquete
- Ligar o computador
- Aguardar o CP/M apresentar-se

Para computadores com partida através do teclado, temos a seqüência:

- Ligar os periféricos
- Ligar o computador
- Inserir o Disquete
- Digitar a instrução de inicialização
- Aguardar o CP/M apresentar-se

Desligando o Computador

O processo de desligar será o oposto ao de ligar iniciando-se pela remoção dos Disquetes dos Drives desligando-se os periféricos e por fim o computador. Se você tiver algum acionador de disco rígido, (Disco Winchester) desligue-o primeiro, pois qualquer ruído ou falha no sistema pode acarretar graves conseqüências como perda de informações no disco.

Não esqueça que antes de desligar qualquer parte do sistema verificar se o CP/M está no processo de aguardar um comando, e nunca desligue o sistema se você estiver rodando algum programa, pois alguns programas só fazem a atualização no disco, no momento em que você deseja sair do mesmo.

Partes Básicas do CP/M

O CP/M é constituído de três partes distintas a saber:

CCP

O módulo CCP interpreta os comando do CP/M, sendo esta etapa a responsável pelo controle do console, somente alguns comandos são reconhecidos. (CCP Processador de Comandos do Console).

BDOS

Este é a parte central do CP/M, responsável pelo controle e gerenciamento dos arquivos dos Drives Não é acessível por comandos do console. Somente por Software (comandos de registros do microprocessador) pode-se acessá-lo. No capítulo sobre informações técnicas é que teremos detalhes importantes sobre o mesmo (BDOS = Sistema Operacional Básico em Disco).

BIOS

É a parte responsável pela interface entre o CP/M propriamente dito e o computador. O CCP e BDOS são sempre os mesmos, mas o BIOS é alterado de acordo com o computador utilizado. Quando se adaptar um microcomputador ao CP/M torna-se necessário alterar o BIOS adequadamente ao computador em questão. Na figura 2.1 temos o desenho em bloco, básico, de como fica um sistema com CP/M. A memória fica subdividida em cinco partes distintas.

O BIOS e BDOS formam o FDOS, ou seja, etapa de interface com os periféricos e o CP/M propriamente dito. O CCP tem o controle para os comandos básicos (a rotina de acesso ao console está no BIOS). A página zero é uma importante ferramenta do CP/M onde são gerados alguns dados importantes e também os "Jumps" para chamar os comandos no BDOS.

A área denominada TPA é a região da memória reservada ao usuário.

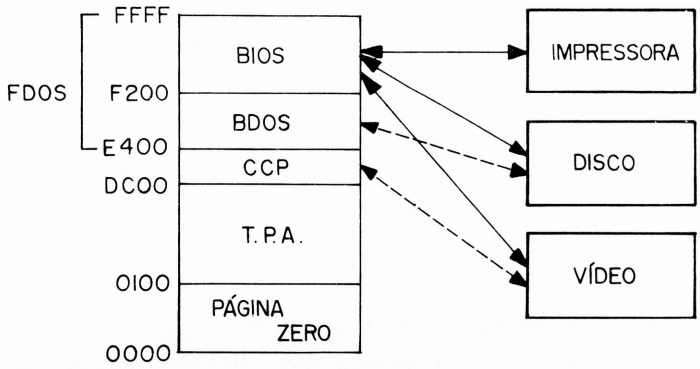


Fig. 2.1 - Esquema básico do CP/M na memória e sua inter-relação com os periféricos.

3. COMANDOS INTERNOS DO CP/M-80

No capítulo anterior nós mostramos como ligar o computador, vamos agora começar a manusear o CP/M.

Existe no CP/M atual (versão 2.2) sete comandos básicos a saber:

- Dir, Era, Type, User, Ren, Save, D:

Cada comando deste, funciona como uma instrução para o CP/M. Sempre que o mesmo apresentar no vídeo o sinal "A>" significa que está pronto para receber comandos.

Os comandos internos se localizam "dentro" do CP/M, ou seja, quando carregamos o CP/M do disco para memória estes comandos são levados juntos. Vamos estudar no próximo capítulo os comandos que chamaríamos de externos, porque apesar de serem comandos propriamente ditos, eles estão armazenados em arquivos.

Os comandos internos são rápidos, porque o CP/M não consulta o disco, enquanto os comandos externos são procurados nos arquivos do disco.

Além dos comandos internos anteriormente descritos, temos também os comandos de interpretação de linhas de edição, são eles:

^C ^E ^H ^J ^M ^P ^R ^S ^U ^X

Alguns comandos que são novos e pertencem à versão 2.2, no apêndice "C" você poderá verificar as diferenças entre as versões do CP/M.

Entrando com os Comandos

Antes de iniciarmos devemos nos informar sobre uma convenção que será utilizada neste livro:

- O termo <CR> significa acionar a tecla de Carriage Return, que em alguns computadores são chamados de RETURN, RET,<CR>, ENTER ou mesmo RETORNO.
- Quando houver uma entrada do usuário no computador, esta será sublinhada.

Exemplo de entrada de um comando:

DIR <CR>

Exemplo de entrada de um programa qualquer:

A> LOAD <CR>

O símbolo "^" significa CONTROL (Ex. ^X = control X). O control algumas vezes é abreviado como CTRL ou CTL.

No exemplo abaixo mostramos o que você digita e o que o computador responde:

```
A>DIR LOAD.* <CR>

A:LOAD.ASM      LOAD.BAK        LOAD.HEX        LOAD.PRN
A:LOAD.COM      LOAD.SYM
```

Antes de iniciarmos a análise dos comandos internos do CP/M será necessário conhecermos a convenção adotada para arquivos em CP/M.

O CP/M atual (versão 2.2) tem a capacidade de acessar até 16 unidades de disco. Para identificar para qual unidade de disco o sistema está voltado o CP/M, coloca no vídeo as letras A> até P>, respectivamente disco 1 até 16. Para trocar o direcionamento de um disco para outro deve-se digitar a letra correspondente ao Drive seguido de ":" e <CR>.

Ex:

```
                    dirigindo para o Drive A
Direcionando
para o Drive B      B> A:<CR>
Direcionando p/     A>
o Drive A                                   tela do vídeo
```

Quando ligamos o computador o CP/M acessará o Drive "A", a menos que se especifique outras unidades de disco.

Referência à Diferentes Arquivos de mesmo Nome

Muitas vezes deseja-se verificar a existência ou não de um determinado arquivo no disco, onde sabemos apenas o tipo

de arquivo, mas não o nome.

Em outros casos sabemos o nome do arquivo, mas não o tipo. O CP/M é dotado de um atributo que é chamado em português de referências de arquivo ambíguas, onde o asterisco (*) pode ser o substituto para o nome do arquivo, o tipo ou ambos. O ponto de interrogação (?), pode ser usado para substituir uma letra do nome ou tipo de arquivo.

Exemplos de algumas referências ambíguas:

*. ASM = Todos os arquivos que terminarem com ASM
Teste.* = Todos os arquivos "Teste" com qualquer extensão
. = Todos os arquivos

Os exemplos acima trocam o nome ou o tipo do arquivo. Isto significa que um determinado comando pode estar atuando por exemplo, em todos os arquivos que chamam-se Teste ou todos arquivos do tipo ASM e assim por diante.

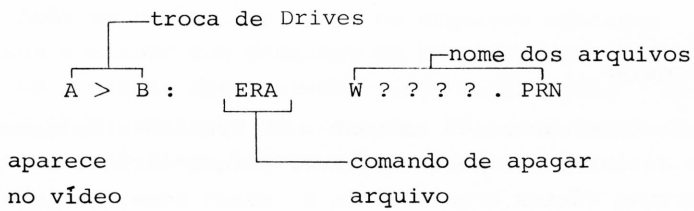
Como exemplo, caso você deseje apagar do disco todos os arquivos que tem o tipo SYM, então digita-se:

ERA *. SYM <CR>

Exemplos de aplicação para o ponto de interrogação:

TESTE?. ASM = Todos os arquivos que começam com a palavra TESTE e tem mais uma letra qualquer é do tipo ASM.
???????.??? = Tem o mesmo efeito que *.*

Pode acontecer de você estar com o sistema dirigido para o Drive A e deseja apagar todos os arquivos do Drive B que tenha o nome iniciando com a letra W mais quatro letras do tipo PRN, então teremos:



As Formas dos Arquivos

Nomes

Cada arquivo registrado no disco tem um nome e o mesmo deve respeitar alguns parâmetros, entre eles:

- Usar sempre letra maiúscula, apesar que se você usar letra minúscula o CP/M converte para maiúscula.
- Não usar os caracteres .,;:=?*[ ]( ) / ou TAB porque o CP/M utiliza estes caracteres de forma especial.
- Não usar arquivo em branco. O CP/M não reconhece arquivo em branco ou seja sucessivo teclar na barra de espaço e depois <CR>.
- Usar no máximo oito caracteres para o nome do arquivo, em seguida o ponto mais três caracteres para informar o tipo de arquivo, conforme mostraremos logo à frente.

Nomes de arquivos válidos:

NOMEAR
TESTE
OK /
3456

Nomes não válidos:

ESTE NOME É COMPRIDO DEMAIS	- MUITOS CARACTERES
ESTE,ARQ	ILEGAL
(ARQUIVO)	ILEGAL
ARQUIVO ^S	CARACTER DE CONTROLE
* ARQ	ILEGAL

Tipos de Arquivo

Em CP/M não se dá um nome a um arquivo simplesmente, é necessário declarar o tipo de arquivo, porque durante o processamento teremos vários arquivos com o mesmo nome, mas com informações diferentes. Por exemplo, TESTE. COM será um arquivo com os dados em hexadecimal e estes dados na realidade formam um pro

grama executável. No entanto, o arquivo TESTE . SYM é formado apenas de símbolos utilizados no programa original do qual foi criado o TESTE. COM, e este programa original chama-se TESTE. ASM.

O ponto entre o nome do arquivo e o tipo do mesmo serve para separar as duas partes distintas, por isso não deve ser usado no nome do arquivo.

Abaixo temos uma pequena lista dos tipos de arquivos mais comuns:

. ASM = Arquivo de fonte em Assembler (CP/M-80)
. A86 = Arquivo de fonte em Assembler (CP/M-86)
. BAK = Arquivo de reserva (BACK - UP)
. BAS = Arquivo de fonte em BASIC
. CAL = Arquivo de dados em SUPERCALC
. CMD = Programa transiente exęcutável diretamente (CP/M-86)
. COB = Arquivo fonte em COBOL
. COM = Programa transiente executável diretamente (CP/M-8Ø)
. DOC = Arquivo documento (texto)
. HEX = Arquivo objeto no formato HEX (INTEL)
. LIB = Arquivo biblioteca
. MAC = Arquivo de programa fonte em macro Assembler
. PAS = Arquivo fonte em PASCAL
. PRN = Arquivo para listagem em Assembler
. REL= = Arquivo de programa relocável
. SYM = Tabela de símbolos
. TXT = Arquivo documentos (Texto)
. $$$ = Arquivo temporário

Descrição dos Comandos Internos

Após as convenções sobre os arquivos adotados para o CP/M vamos analisar com detalhes os comandos internos.

Os comandos apresentados nesta seção estão localizados dentro do CP/M, portanto são comandos rápidos, devido ao fato de o CP/M não ir ao disco buscar este comando.

DIR

Mostra o diretório do disco. Este comando mostra ao usuário o nome dos arquivos do disco corrente quando digitado:

A> DIR

Na versão 1.3 aparece um nome abaixo do outro, mas na versão 2.2 aparece quatro nomes em cada linha.

Quando for digitado DIR D: será listado no vídeo o conteúdo de arquivos do Drive D.

Pode-se usar as referências ambíguas (* e ?) para processar grupos de arquivos.

Mensagens de erro

Se você digitar o comando com nome errado, o CP/M mostrará o que você digitou com o sinal de interrogação.

Por exemplo:

A> DIV <CR>
DIV ?

Se aparecer a mensagem:

No File ou File Not Found - significa que o CP/M não encontrou no Drive especificado o arquivo com o nome discriminado pelo usuário.

BDOS ERR ON D: - (Onde D: identifica o Drive). O CP/M não encontrou Disquete no Drive especificado. Você deve verificar se há Disquete no Drive se está colocado do lado certo ou se está mal formatado.

ERA

Apaga o nome do arquivo do diretório. Este comando é usado acompanhado do nome ou nomes dos arquivos e seu respectivo tipo para serem apagados do diretório.

Ex: A> ERA TESTE.*

O exemplo acima apaga todos os arquivos TESTE de qualquer tipo.

É importante lembrar que ao operar-se com Disquete, deve-se estar sempre atento, pois pode-se por descuido apagar arquivos errados, e às vezes o erro pode causar sérias conseqüências, portanto tome muito cuidado antes de apagar um arquivo do

disco.

No CP/M se você digitar ERA *.* significa que você quer apagar o disco todo, como isto não é uma operação muito comum, o computador escreverá no vídeo:

ERASE ALL?

Quer dizer apagar tudo ? se você digitar (Y) de Yes, então o disco será totalmente apagado. As mensagens de erro são as mesmas do comando DIR.

REN

Trocar o nome de um arquivo. Podemos dar um novo nome a um determinado arquivo, portanto deve-se digitar o nome antigo e completo do arquivo e o novo nome. Não se pode usar os caracteres ambíguos (* e ?) para alterar o nome de grupos de arquivos de uma só vez. Normalmente quando o computador recebe um comando de equivalência, o novo nome deverá ficar do lado esquerdo, enquanto o nome antigo do lado direito. Se for incluído num nome deve-se aplicá-lo nos dois, e se for aplicado nos dois nomes será necessário que seja o mesmo Drive. Se não for colocado o código (letra) do Drive, o Drive corrente é usado.

Ex: REN D : ARQNOV TIP = D: ARQ VELH TIP <CR>

As mensagens de erro são as mesmas do comando DIR. Há uma mensagem nova que pode aparecer que é a FILE EXISTS, acontece quando você tenta criar um nome já presente no Disquete.

TYPE

Mostra no vídeo (lista) os dados de um arquivo com os caracteres em código ASCII. Este comando permite visualizar no monitor de vídeo, o conteúdo de um arquivo com caracteres ASCII. Os arquivos do tipo HEX, ASM, MAC, BAS, etc. funcionam para o comando TYPE, mas os comandos COM, REL e outros que não estão em caracteres ASCII, pois são instruções em Hexadecimal, para o computador, não é possível visualizar. A forma de usar o comando é a seguinte:

TYPE D: Arquivo TYP <CR>

Mostrará o conteúdo do arquivo especificado. Este comando está presente em todas as versões do CP/M. As mensagens de erro são as mesmas do DIR.

D:

Troca da unidade de disco. Num sistema com dois ou mais Drives troca-se o Drive corrente pressionando a letra correspondente ao mesmo seguido de ":" e <CR>.

Quando se faz referência à qualquer arquivo do Drive corrente não é necessário especificar, mas para Drive diferente do corrente, você precisa especificar o Drive.
Se aparecer a mensagem de erro:

BDOS ERROR D: SELECT - significa que está se tentando acessar um Drive que não existe ou o CP/M não consegue acessar o Drive por algum problema (Drive desligado ou inexistente).

USER:

Troca o número do usuário. No CP/M mais atualizado (versão 2.2), no CP/M-86 e no MP/M existe um comando que permite dividir o disco em 16 subdivisões chamados de usuário Ø até usuário 15.

Quando você ligar o CP/M ele assume usuário Ø (USER Ø) e a partir daí você pode escolher o usuário que desejar. Esta área é imaginária, porque na realidade o nº do usuário está associado ao nome do arquivo no diretório.

A forma de usá-lo é a seguinte:

USER ## <CR> - Onde ## é um número entre Ø e 15.

Quando você usar um comando ERA *.* o CP/M apaga os arquivos somente do usuário corrente.

Mensagens de erro:

FILENAME ? ou NO FILE - Esta mensagem aparece quando tenta-se acessar um programa de um usuário dentro de outro.

? ou ? - Você esqueceu de especificar o número ou usou número maior que 15.

SAVE

Salvar o conteúdo da memória em DISCO. Com este comando pode-se criar um arquivo e nele gravar um programa ou dados residentes na memória.

O início da memória a ser gravado deverá ser a partir do endereço ØlØØ H. É necessário especificar no comando o número de páginas a serem gravadas (blocos de 256 caracteres). Escolha o nome com cuidado porque o Save apagará do diretório algum nome igual antes de proceder à operação.

Este comando tem maior aplicação para os programadores que trabalham em Assembler. O programa DDT (será analisado no próximo capítulo) permite a depuração de um programa e através do comando Save, podemos então gravá-lo.

A forma de digitá-lo é a seguinte:

SAVE D: Nome Arq TIP <CR> - Onde ## é o número de blocos (256 caracteres) que se deseja gravar.

D: - É o Drive a ser gravado

Lembre-se que antes de gravar o programa, de verificar o tamanho do mesmo, divida em páginas de 256 caracteres. Se resultar num número não inteiro coloque o número imediatamente superior, e só então proceda à operação.

Mensagem de erro

DISK FULL: - Acontece se o diretório estiver lotado ou se o tamanho do programa não couber no espaço disponível.

Comandos Usados na Edição de Linha

Estes comandos são utilizados na edição de uma linha, ou seja, quando digita-se o nome de um arquivo, ou em um dos comandos já mostrados, etc. A finalidade seria apagar um caracter, interromper uma listagem, apagar uma linha inteira, etc.

Todos estes comandos são constituídos de duas teclas pressionadas simultaneamente correspondendo a pressionar a tecla CONTROL, em conjunto com a tecla do comando desejado.

CONTROL J

É o mesmo que LINE - FEED, e esta tecla está normalmente ao lado do<CR> (Carriage Return).

CONTROL M

É o mesmo que CR, ENTER ou RETURN em certos teclados.

CONTROL U OU X

Este comando é usado para cancelar um comando que você acabou de digitar. Imagine que você digitou algo indevido, então, poderá cancelar com um dos comandos. Na primeira versão (1.3) use o CONTROL - U, nas atuais use o CONTROL - X.

CONTROL R

Com este comando o CP/M repete a linha que você acabou de digitar. A existência deste comando está aliada ao fato de algumas versões antigas terminais exibirem o caracter cancelado. Imagine que você errou várias vezes ao digitar um comando, então o vídeo ficaria assim:

DIQQR TESTE,,. COMMOC *

Evidente que seria difícil entender o que realmente foi digitado, por isso, ao pressionarmos CONTROL - R teremos:

DIR TESTE .*

CONTROL S

Fixa o conteúdo Display do console. Se você está visualizando o conteúdo de um arquivo e deseja paralisar a amostragem do vídeo, basta pressionar CONTROL-S. Para recomeçar, digite outro CONTROL-S.

Alguns programas bem elaborados só colocam em uso no vídeo os caracteres que a tela suporta, e aguarda em <CR> para prosseguir.

CONTROL H

Permite apagar o caracter à esquerda do cursor. (Apenas na versão CP/M 2.2).

DELETE, DEL OU RUBOUT

Igual ao CONTROL-H só que ele repete o caracter cance

lado, estando esta repetição aliada ao fato de no início da utilização do CP/M o console ser uma TELETYPE ou similar, os quais imprimiam em papel, não permitindo o retorno do carro.

CONTROL E

Para ser possível digitar um comando longo na linha seguinte, digitando-se CONTROL - E e o cursor move-se para a próxima linha e você continua a digitar o comando. Tuda terminará com o <CR>. O CONTROL - E não é considerado parte da linha de comando.

CONTROL C

Este comando é usado para reiniciar o CP/M. Quando você liga o computador ou pressiona o botão de RESET estará provocando uma partida à frio.

Ao pressionar CONTROL-C estará provocando uma partida à quente. A diferença é que a primeira instala o CP/M na memória destruindo programas ou dados contidos na mesma, sendo que a segunda apenas carregará algumas partes do CP/M na memória e manterá a mesma intacta.

A partida à quente é normalmente usada para duas aplicações básicas:

- Tornar corrente um Disquete ao inserí-lo no Drive
- Interromper um programa transiente voltado ao CP/M.

Sempre que você trocar discos de um determinado Drive digite o CONTROL-C, senão poderá acontecer algumas coisas estranhas no sistema, salvo casos em que o programa que está "RODANDO" e pede a troca para possivelmente retirar ou trocar dados com outro disco.

CONTROL P

Liga ou desliga a impressora. Para deixar a impressora ecoando com o vídeo, digita-se CONTROL-P. Se pressionar novamente desliga-se a mesma.

Este comando exige cuidado porque após digitá-lo, tudo o que aparecer no vídeo, ecoa na impressora, portanto, se a mesma for lenta, a velocidade de transmissão para o vídeo ficará lenta.

Para jogar, por exemplo, o conteúdo de um arquivo em ASCII na impressora deve-se fazer o seguinte:

A> ^P TYPE NOMARQ. TXT <CR>

No exemplo acima digitou-se ^P para habilitar a impressora, o comando TYPE e depois o nome do arquivo. No fim da impressão digita-se novamente ^P.

Se ao digitar ^P, apareceu no vídeo:

PRINTER NOT READ

Ou mensagem parecida, significa que a impressora ou tem problemas ou está mal conectada, ou então, ela não está em ON-LINE, ou seja, habilitada para receber dados.

Abaixo temos uma tabela resumida dos comandos de linha

COMANDO	OPERAÇÃO
^C	Reinício do CP/M-8Ø
^P	Controle da impressora
^X ou ^U	Cancelar linha
^E	Próxima linha para comandos longos
^R	Repete a linha
^S	Paralisa Display de vídeo
^H	Apaga caracter à esquerda
CR ou ^M	Retorno do carro
LF ou ^J	Término de linha
DEL	Apaga o caracter do comando repetindo o caracter.

4. COMANDOS EXTERNOS DO CP/M

No capítulo anterior foram analisados os comandos internos do CP/M, comandos que estão localizados dentro do CP/M, agora vamos trabalhar com os comandos externos, tambem chamados de comandos transientes do CP/M.

Nós já analisamos os Disquetes e foi dito que o CP/M (Sistema Operacional) está localizado no mesmo, na trilha Ø e 1. Os comandos externos também estão gravados em disco, mas em forma de arquivos, e cada arquivo em disco do tipo ".COM", sendo na verdade um programa executável pelo computador. Quando deseja-se "rodar" um programa ou um comando externo (que não deixa de ser um programa) deve-se digitar o nome e pressionar <CR>.
Por exemplo:

A> TESTE <CR>

A seqüência que o CP/M executa quando você usa nome, conforme o exemplo acima é a seguinte:

Verifica se não é um comando interno, se não busca no disco um arquivo que tenha um nome idêntico. Então, você percebe que comando externo e arquivos com programas comuns na realidade são a mesma coisa, a diferença é apenas uma convenção. Também é fato que os comandos externos são funções muito ligadas ao sistema como por exemplo:

STAT, FORMAT, ETC.

Os comandos externos mais comuns são:

- STAT : Comando para fornecer dados importantes sobre um determinado arquivo ou grupo de arquivos.
- FORMAT : Comando de inicializar um disco
- DUMP : Comando para ler um arquivo do tipo "COM".
- COPY : Comando para duplicar o conteúdo de um disco.
- SYSGEN : Comando para gerar um novo CP/M em um disco novo.

Na seqüência vamos analisar cada um em separado com suas respectivas aplicações:

DUMP = DUMP D: NOMARQ.COM <CR>

Para visualizar uma informação no vídeo é necessário que a mesma esteja em ASCII. Os arquivos "COM" estão em HEXA, por isso é necessário transformá-los em ASCII e jogar no vídeo. Esta função de ler o arquivo "COM", transformar os caracteres em ASCII e jogá-los no vídeo é a função executada pelo comando DUMP.

Usando o ^S pode-se interromper a mostragem do arquivo no vídeo e um novo ^S continua a mostrar o arquivo. Durante o processo, ao pressionarmos qualquer tecla a execução do programa DUMP, é interrompida e volta para o sistema (A>).

FORMAT = FORMAT <CR>

Quando você compra um Disquete é necessário inicializá-lo, ou seja, prepará-lo para posteriormente gravar os arquivos. Esta preparação chama-se formatar o disco. Há vários tipos de formatação diferentes, sendo que o padrão CP/M é de 128 Bytes por setor, com 26 setores por trilha, num total de 77 trilhas (tirado do padrão IBM 3740). Digita-se o nome FORMAT <CR> e então aparecerá um MENU perguntando em qual Drive deseja-se formatar o disco. Depois de informar ao sistema qual o Drive aparecerá um menu perguntando que tipo de formatação deseja-se realizar.

Ex:

A = uma face simples densidade (128 Bytes)
B = duas faces simples densidade (128 Bytes)
C = uma face dupla densidade (256 Bytes)
D = duas faces dupla densidade (256 Bytes)

Em seguida o computador perguntará:

Qual a sua opção?

Poderá ter alguns formatos com mais opções, ou também, como é o caso do Apple com CP/M onde o formato não tem opções, há apenas um único tipo de gravação.

Existem várias implementações diferentes do comando FORMAT. Há casos, conforme mostrado anteriormente mas, há casos em que o programa fica mostrando no vídeo setor a setor, trilha a trilha o que está sendo formatado. A única mensagem de erro é a de não ser possível (não conseguir) formatar um determinado setor, então o sistema interrompe o processo e avisa ao usuário.

COPY = COPY <CR>

Este programa também chamado de CÓPIA, DUPLIC, DUPLICAR é usado para duplicar o conteúdo de um disco para outro. Sempre que você adquirir um programa ou vários programas num disco deve-se tirar cópias (BACK-UPs) como medida preventiva contra qualquer imprevisto.

Quando digitar COPY <CR> aparecerá um menu perguntando: duplicar de qual Drive ? Logo respondido a esta pergunta virá um outro menu com as possibilidades de:

F - FORMATAR O DISQUETE DO DRIVE D:
R - RESETAR OS DRIVES
C - CONTINUAR O PROCESSO
Q - SAIR PARA O SISTEMA

Os menus costumam variar de fabricante para fabricante e normalmente as palavras estão em inglês.

Em caso do seu sistema não ter dois ou mais Drives e sim, apenas 1, existe um programa chamado COPY 1, CÓPIA 1, etc, que realiza a função de cópia de um arquivo ou vários arquivos de um disco para outro, usando a mesma unidade de Drive. Para isto torna-se necessário trocar os disquetes quando requisitado pelo sistema.

SYSGEN

Este comando é usado para cópias do sistema operacional de um disco para outro. Quando você usa o comando COPY ou DUPLIC estará fazendo a cópia de um disco para outro, setor a setor. O programa não verifica quais são os arquivos ou onde estão, o tipo e etc., faz simplesmente uma duplicação, sem verificar o que há no mesmo. Quando você quer transportar um arquivo de um disco para outro, existe um programa chamado PIP ou XFER, etc., que faz isto. Este programa faz a cópia de todos e quaisquer arquivos que existam no disco, menos o sistema operacional, devido ao fato de que o sistema operacional está gravado de uma forma especial nas trilhas Ø e 1. A partir daí tem-se a necessidade de um programa que tenha a função de um comando por assim dizer, que executa esta tarefa, e no caso é o Sysgen.

O processo é o seguinte:

a) Você pode transportar o sistema de um disco para outro:

```
A>SYSGEN <CR>

SYSGEN VER. 2.Ø
SOURCE DRIVE NAME (OR RETURN TO SKIP) A
SOURCE ON A THEN TYPE RETURN <CR>
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO SKIP) B
DESTINATION ON B THEN TYPE RETURN <CR>
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <CR>
A>
```

b) Você pode tirar o sistema da memória ou de um arquivo que o contenha e colocá-lo no local do sistema.

A> SYSGEN NOMEAR Q. SYS <CR> - digita-se o SYSGEN + nome do arquivo que contém o CP/M e <CR>.

Em alguns casos mais antigos deve-se colocar o programa que contém o CP/M na memória (usando o DDT que será descrito no próximo capítulo), e posteriormente digita-se simplesmente SYSGEN <CR>.

```
A> SYSGEN <CR>
SYSGEN VER.2.Ø
SOURCE DRIVE NAME (OR RETURN TO SKIP) <CR>
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
DESTINATION ON B THEN TYPE RETURN <CR>
FUNCTION COMPLETE
DESRINATION DRIVE NAME (OR RETURN TO REBOOT) <CR>
A>
```

O comando SYSGEN apresenta pequenas diferenças entre fabricantes, com algumas tentativas, logo você saberá manuseá-lo, mas o princípio de operação é descrito.

STAT - Estatística dos Arquivos

O STAT nos fornece importantes informações estatísticas dos discos e arquivos, bem como sobre todo o sistema. Com este comando podemos ver o espaço livre na memória, tipos de arquivos existentes, espaço ocupado por certos arquivos, determinar atributos e etc.

Para usarmos o STAT precisamos antes conhecer algumas terminologias importantes, tais como:

R/O

Arquivos apenas para leitura - Não é possível alterar ou atualizar um arquivo R/O.

R/W

Arquivo de leitura e escrita - Arquivo livre para escrita (atualização) e leitura.

SYS

Arquivo de sistema - Um arquivo SYS não aparece quando, pedimos o diretório de um disco. Pode-se usá-lo normalmente, mas jamais aparecerá no diretório.

DIR

Arquivo de diretório - Um arquivo DIR aparece em todos os diretórios de todos os usuários.

Um arquivo pode ser R/O ou R/W, SYS ou DIR, não podendo ser ambos.

O STAT pode ser usado de duas formas diferentes, uma na manipulação de arquivos e outra para manipular dispositivos.

Manipulação de Arquivos

STAT <CR>

Mostra o atributo e o espaço livre no disco corrente.

```
A>STAT<CR>
A:R/W,SPACE:56K
OU
A>STAT<CR>
A:R/W,SPACE:56K
B:R/O,SPACE:10ØK

A>STAT B:<CR>
BYTES REMAINING ON
B: 5ØK
```

A STAT D: NOME.EXT <CR>

Mostra o espaço ocupado no arquivo NOME.EXT do Drive especificado. Pode-se usar caracteres ambíguos, com isto será mostrado uma relação de arquivos conforme exemplo.

```
A>STAT TESTE.COM<CR>

RECS    BYTES   EX      ACC     D:FILENAME.TYP

10      2K      1       R/O     A:TESTE.COM

A>STAT TESTE.*<CR>

RECS    BYTES   EX      ACC     D:FILENAMES.TYP

5       1K      1       R/O     A:TESTE.COM

10      2K      1       R/W     A:TESTE.ASM

5       1K      1       R/O     A:TESTE.SYM
```

As terminologias são:

RECS
Número de registros de 128 Bytes usados pelo arquivo.

BYTES
Tamanho do arquivo. Lembrar que 1K significa 1024 Bytes.

EX
Número de extensões ocupadas pelo arquivo no disco.

ACC
Atributo de acesso do arquivo.

D: FILENAME.TYP
Número do Drive e nome do arquivo.

Designando Atributos para os Arquivos

STAT D: NOME.EXT $ATR <CR>

No lugar do ATR coloca-se o atributo desejado. Pode-se usar caracteres ambíguos.

Exemplos:

```
A>STAT B:TESTE.COM $R/O<CR>

TESTE.COM SET TO R/O

A>STAT A:*.* $R/O<CR>

TESTE.COM SET TO R/O

MICRO.ASM SET TO R/O

NOME.COM SET TO R/O
```

STAT D: = R/O <CR>

Determina temporariamente o estado R/O para o Drive D:. Quando houver uma partida à frio ou à quente este estado será cancelado.

STAT D: DSK: <CR>

Mostra como estão montados os dados no disco corrente ou D:

Exemplo:

```
A> STAT DSK: <CR>

   A:DRIVE CHARACTERISTICS
3840:128 BYTES RECORDS CAPACITY
 480:KILOBYTES DRIVE CAPACITY
 128:32 BYTES DIRECTORY ENTRIES
 128:CHECKED DIRECTORY ENTRIES
 128:RECORDS/EXTENT
  16:RECORDS/BLOCK
  52:SECTORS/TRACK
   3:RESERVED TRACKS
```

Estas informações são referentes ao sistema, foram parte do projeto usado no micro que você está operando. As terminologias são as seguintes:

BYTES RECORD CAPACITY

Significa que o sistema de gravação é 128 Bytes por setor.

KILOBYTE DRIVE CAPACITY

Significa a capacidade máxima em Kbytes do disco.

BYTE DIRECTORY ENTRIES

Isto mostra o número máximo de arquivos que se pode criar por disco.

RECORDS/EXTENT

O número máximo de registros em cada entrada no diretório.

RECORD/BLOCK

O menor espaço de disco para alocar um arquivo.

SECTORS/TRACK

Determina o número de setores por trilhas do disco.

RESERVED TRACKS

Número de trilhas reservadas onde não se pode gravar arquivos.

Manipulação com Dispositivos

Podemos também usar o STAT para receber informações sobre os dispositivos lógicos e físicos e alterar as relações entre eles.

No CP/M há uma terminologia que devemos entender que são os dispositivos lógicos e físicos. O CP/M necessita que selecionemos um dispositivo físico para que ele execute a função de um dispositivo lógico.

Há no CP/M quatro dispositivos lógicos:

LST = Impressão; função de listar.
PUN = Envia dados; função perfurar papel.
RDR = Recebe dados; função ler fita de papel.
CON = Console; entrada e saída de informações.

Há no CP/M 12 dispositivos físicos:

TTY = Console lento (Teletype)
CRT = Console rápido (Vídeo)
BAT = Processador de Batch
LPT = Impressora de linha
UL1 = Dispositivo de listagem definido pelo usuário
UP1 = Perfurador nº 1 definido pelo usuário

UP2 = Perfurador nº 2 definido pelo usuário
UR1 = Leitora nº 1 definido pelo usuário
UR2 = Leitora nº 2 definido pelo usuário
UC1 = Console definido pelo usuário
PTR = Leitora de papel
PTP = Perfuradora de papel.

Há uma relação entre os dispositivos lógicos e físicos como segue:

LÓGICO	FÍSICO
RDR	TTY, PTR, UR1, UR2
PUN	TTY, PTP, UP1, UP2
CON	TTY, CRT, BAT, UC1
LST	TTY, CRT, LPT, UL1

Devemos lembrar que sempre que você for digitar a terminologia de um dispositivo lógico ou físico deve-se digitar em seguida ":".

Usando o STAT nos Dispositivos

STAT DEV: <CR>
Mostra como estão realcionados os dispositivos

Exemplo:

```
A>STAT DEV:

CON: IS SER
RDR: IS TTY
PUN: IS TTY
LST: IS PAR
```

STAT VAL: <CR>
Mostra as possíveis designações dos dispositivos físicos e lógicos e um resumo dos comandos existentes no STAT.

```
Temr R/O Disk: d:=R/O
Set Indicator: d:filename.tur $R/O $R/W $SYS $DIR
Disk Status  : DSK: d:DSK:
User Status  : USR:
Iobyte Assinn:
CON: = VDB: SER: BAT: UC1:
RDR: = TTY: PTR: UR1: UR2:
PUN: = TTY: PTP: UP1: UP2:
LST: = PAR: SER: LPT: UL1:
```

STAT LOG: = FIS: <CR>

Este comando permite alterar as relações entre os dispositivos físicos e lógicos especificados.

Exemplo:

```
A > STAT CON:=TTY:.RDR:=UR1
```

A vírgula permite fazer várias designações ao mesmo tempo.

STAT USR: <CR>

Permite saber qual é o usuário corrente e mostra em quais usuários há arquivos.

Exemplo:

```
10)
A>STAT USR:<CR>
ACTIVE USER:1
ACTIVE FILES: 0,1,3,6,7
```

Como podemos ver o STAT é uma ferramenta importante no CP/M. Você deve treinar em algum computador o manuseio com o STAT porque dependendo do fabricante haverá algumas diferenças na sua operação.

5. ASPECTOS TÉCNICOS DO CP/M

Estrutura do CP/M

No início deste livro mostramos as três partes básicas em que se subdivide o CP/M, ou seja, CCP, BDOS e BIOS. CCP é a parte onde se localiza os comandos internos como DIR, ERA, SAVE, etc.

Nesta parte não há muito o que comentar, pois o acesso que temos são apenas a comandos quanto ao BDOS e o BIOS que analisaremos neste capítulo. No que concerne ao BDOS mostraremos as funções, quanto ao BIOS veremos sua concepção e como adequá-lo a cada computador.

Funções do BDOS - BASIC DISK OPERATING SISTEM

Até o presente momento vimos como usar os comandos do CP/M, programas aplicativos, etc. Imagine agora que você deseja construir um programa para rodar no computador, sendo assim, você vai precisar ler o teclado, jogar os dados no vídeo, acessar o disco e imprimir. Para tal é necessário conhecer as funções do BDOS porque são as mesmas que nos permitirão realizar as tarefas acima.

Para usar a tabela do BDOS devemos realizar o seguinte procedimento: No "registrador C" teremos o número da função desejada e no "registrador E" o dado a ser transmitido. Se for executada a função de ler um caracter o comando ficará no "registrador C", conforme dito anteriormente, e o dado estará disponível no "registrador A". Quando você desejar imprimir um String, o endereço do mesmo estará no par de "registrador DE". Em seguida executa-se um CALLØØØ5H, pois neste endereço está um JUMP para o BDOS onde será realizada a função desejada.

Ex:

Vamos supor que deseja-se ler um lado do teclado e jogar na impressora até encontrar o caracter 'Ø', quando então iremos voltar ao CP/M.

```
        ORG     0100H;          INDICADOR DE INICIO DO PROGRAMA
TESTE:  MVI     C,01H;          PREPARA PARA LER O CARACTER
        CALL    0005H;          CHAMA O BDOS
        CPI     '$';            VERIFICA SE CHEGOU NO FIM
        JZ      0000H;          SE FIM RETORNAR PARA O CP/M
        MVI     C,05H;          PREPARA PARA ESCREVER NA INPRESSORA
        MOV     E,A
        CALL    0005H;          CHAMA O BDOS
        JMP     TESTE;          VOLTA AO INICIO
        END
```

Algumas funções do BDOS utiliza um termo chamado FCB cujo significado é FILE CONTROL BLOCK (Bloco de controle de arquivo). Para entender o FCB precisamos conhecer primeiro a página zero que é uma área de memória que vai do endereço ØØØØH até ØØFFH.

Página Zero

Esta área de memória é onde se localiza o JUMP para o BIOS, para dar a partida à quente (End.ØØØØH), o Jump para o BDOS (End. ØØØ5H), identificação dos nomes dos arquivos que serão posteriormente manipulados, etc.

POSIÇÃO DE MEMÓRIA	FUNÇÃO
ØØØØ - ØØØ2	Pulo para partida à quente no BIOS
ØØØ3	IOBYTE
ØØØ4	Número do Drive corrente, número do usuário corrente
ØØØ5 - ØØØ7	Pulo para o BDOS
ØØØ8 - ØØ37	Reservado para interrupção
ØØ38 - ØØ3A	RST7 do DDT
ØØ38 - ØØ3F	Reservado para interrupção da máquina
ØØ4Ø - ØØ4F	Reservado para uso do BIOS
ØØ5Ø	Comando de Drive carregado no CP/M 3
ØØ5l	Endereço de código de passagem para o primeiro FCB por ausência.

ØØ53	Comprimento do código de passagem para o primeiro FCB
ØØ54	Endereço do código de passagem para o segundo FCB
ØØ56	Comprimento do código de passagem para o segundo FCB
ØØ57 - ØØ58	Reservado
ØØ5C - ØØ68	Primeiro bloco de controle de arquivo por ausência (FCB)
ØØ6C - ØØ7F	Segundo bloco de controle de arquivo por ausência (FCB)
ØØ8Ø - ØØFF	Buffer de 128 Bytes de disco por ausência/Buffer de comando

Importante dizer que o CP/M 1.4 e o CP/M 2.2 e o MP/M 1 tem apenas um único FCB, iniciando no endereço ØØ5CH com a área de ØØ7D - ØØ7F reservada para a posição de registro aleatório no CP/M 2.2 e MP/M 1.

TABELA DAS FUNÇÕES DO BDOS

REG. C	FUNÇÃO	ENTRADA	SAÍDA	OBSERVAÇÕES
ØØ	Partida à quente	-	-	Igual ao ^C.FIM do programa
Ø1	Leitura de dados vindos do console	-	A=caracter em ASCII	Retorna ao programa após digitação de um caracter
Ø2	Envia um caracter para o console	E=caracter em ASCII	-	Mostra no vídeo o caracter do registro E.
Ø3	Leitura de dados vindos da fita perfurada	-	A=caracter em ASCII	Retorna do programa o caracter lido pela leitora
Ø4	Envia um caracter para perfuradora	E=caracter em ASCII	-	Envia o conteúdo do registro E para perfuradora
Ø5	Envia um caracter para a impressora	E=caracter em ASCII	-	Envia o conteúdo do registro E para impressora

Ø6	Entrada e saída de dados direto do console	E=FFh para INPUT ou E=caracter em ASCII p/ saída	A=caracter em ASCII quando INPUT	Se Reg. E=FFh então a função é de entrada. Se Reg E=caracter então será enviado ao console.
Ø7	Ler IOBYTE	-	A=IOBYTE	Lê o endereço e coloca no registrador A
Ø8	Altera IOBYTE	E = novo IOBYTE	-	Coloca um novo IOBYTE no sistema
Ø9	Mostra String	DE=endereço do String	-	Mostra uma seqüência até aparecer o $
ØA	Recebe String vindo do teclado	DE = endereço do Buffer	Dados do Buffer	Lê um String vindo do teclado. Sendo que 1º Byte contém o tamanho máximo do String e o 2º Byte quantos Bytes foram realmente digitados
ØB	STATUS do console	-	A=STATUS	Verifica STATUS. Se A=ØØ não há caracter. Se A=FFh há caracteres
ØC	Recebe versão do CP/M	-	HL=versão	Se H=ØØ então é CP/M, se H=Ø1 é MP/M. Se l=ØØ -CP/M 2.0. Se L=Ø1-CP/M 2.1, se L=Ø2- CP/M 2.2
ØD	Usado para informar a troca do disco	-	-	Informa ao CP/M que foi trocado algum Disquete
ØE	Seleciona o disco	E=Ø1h até 10h	A=achou ou não	Seleciona o disco a ser usado. Se A=FF- não achou se A< Ø4 = OK
ØF	Abrir arquivo	DE=endereço do FCB	A=achou ou não	Usado para abrir um arquivo especificado no FCB

10	Fechar arquivo	DE=idem ao acima	A=idem ao acima	Executa a função de fechar o arquivo e atualizar o diretório
11	Pesquisa o primeiro	DE=idem ao acima	A=idem ao acima	Abre o arquivo e preenche o Buffer de disco corrente com 128 Bytes da entrada de diretório do arquivo especificado
12	Pesquisa o próximo	-	A=idem ao acima	Idem ao anterior apenas que procura a partir do último arquivo coincidente
13	Elimina arquivo	DE=idem ao acima	A=idem ao acima	Apaga do diretório o nome do arquivo especificado no FCB
14	Leitura de forma seqüencial	DE=idem ao acima	A=00 então OK se A≠00 houve problema	Tem a função de ler os próximos 128 Bytes e colocar na memória a partir do endereço do DMA
15	Escrita de forma seqüencial	DE=idem ao acima	A=idem ao acima	Tem a função de escrever próximos 128 Bytes do DMA no arquivo especificado no FCB
16	Criar arquivo	DE=idem ao acima	A=código do diretório	Cria um novo arquivo especificado no FCB. O BDOS não verifica se o nome já existe no diretório, você precisa verificar. Quando se cria um arquivo não é necessário abri-lo
17	Troca o nome do arquivo	DE =Endereço do FCB	A=código do diretório	Troca o nome do arquivo referenciado nos primeiros 16 Bytes do FCB pelos 16 Bytes seguintes

18	Retorna com o vetor de conexão	-	HL=DISCO CONECTADO	Os Bits do par HL mostram qual Drive está ativo. O 1º Bit do registrador L indica o Drive A e o 8º Bit do H indica o Drive P. O nível 1 indica ativo e nível Ø indica inativo
19	Disco Corrente	-	A=DISCO corrente	Os números de Ø a 15 são usados para indicar o disco corrente
IA	Indica o endereço do FCB	D=DMA	-	Usado para especificar o início do DMA para transferência de dados entre disco e memória. Normalmente o CP/M utiliza o endereço ØØ8ØH
IB	Endereço de alocação	-	HL=Endereço de alocação	Indica a tabela que é mantida na memória para cada disco conectado
IC	Torna o disco Protegido	-	-	Protege temporariamente o Disquete do Drive corrente
ID	Pega o vetor R/O	-	HL=DISCO R/O	Mostra no par HL qual drive está protegido contra escrita
IE	Determina atributos dos arquivos	DE=End. do FCB	Código do diretório	Determinar os atributos dos arquivos que indicam se este é sistema, diretório, R/O ou R/W
IF	Parâmetros do disco	-	HL=Endereço do DPB	Recolhe o bloco de parâmetro do disco para o Drive corrente. Com estes parâmetros o usuário, pode determinar espaço disponível, trocar características,etc

2Ø	Pega o código do usuário	E=FF	A=usuário corrente	No registro A você terá o nº do usuário corrente
21	Gera o código do usuário	E=código do usuário	A=usuário corrente	Se tiver no REG E=FF no registrador A voltará o código do usuário corrente, ou então fixar no REG. E o número do usuário que se deseja. O comando USER permite Ø a 15 usuários enquanto o BDOS permite Ø a 31
22	LER ALEATÓRIO	DE=END. do FCB	A=código do erro A=Ø não achou, e A>4 = OK	Lê o número do registro aleatório que encontra-se nos Bytes 33, 34 e 35 do FCB indicado no DMA apontado
23	Escreve aleatório	DE=END. do FCB	A=código de erro	Escreve o conteúdo do DMA nos registros aleatórios apontado pelos Bytes 33, 34 e 35 do FCB
24	Computar tamanho do arquivo	DE=END. do FCB	Indicação de RRF	Retorna o tamanho atual do arquivo aleatório nos três Bytes do campo de registro aleatório do FCB
25	Determine Registro Aleatório	DE=END. do FCB	Indicação de RRF	Retorna com o próximo registro aleatório logo após a última leitura dos registros aleatórios e seqüenciais.

Durante o processo de utilizar as funções do BDOS o seu programa deve preservar alguns registros porque eles serão alterados da seguinte maneira:

A = Quaisquer 8 Bits passados de ou para BDOS

B = Não usado no BDOS

C = Número da função do BDOS a ser utilizado
DE = Localização de memória de 16 Bits ou variáveis
HL = Valores de 16 Bits passados de ou para o BDOS e o
L cópia do conteúdo do registrador A.

O CP/M sempre faz uma cópia do registrador H no registrador A, se nada deve ser escrito no A. Isto é usado para reduzir a movimentação de informação entre os registradores A e H.

CCP - Processador de Comandos do Controle

Este módulo tem a função de interpretar os comandos do CP/M. Toda vez que aparece no vídeo o A>, estará neste instante em operação o CCP.

Quando você digita um comando qualquer, este ficará do endereço ØØ8ØH - ØØFFH. Se o CCP não reconhecer o comando, ele irá verificar no diretório do disco, encontrando um programa em que o nome coincida e tenha extensão ".COM", este será colocado na memória a partir do endereço ØØØØH, e daí para a execução do mesmo.

BIOS - O Sistema Básico de Entrada e Saída

Esta parte do CP/M é a responsável entre a ligação do CP/M (BDOS + CCP) com a sua máquina. É no BIOS que estão as rotinas de acesso ao disco, como por exemplo:

- Acesso à trilha zero
- Avanço trilha à trilha/setor à setor
- Ler um setor
- Escrever num setor, etc.

Estão no BIOS as rotinas de acesso à vídeo, impressora, e demais periféricos. Por isso o CP/M é um sistema operacional capaz de rodar em inúmeras máquinas, justamente porque o BDOS e o CCP são um pacote fechado onde todo processamento (controle do sistema) é feito, mas no BIOS está a interação da máquina com o CP/M.

Quando você compra um disco com CP/M para adaptá-lo ao seu computador, ele estará com o BIOS semi-pronto, necessitando ser colocadas as rotinas de interfaceamento.

Quando o seu computador já tem o CP/M instalado é devido ao fato de que o fabricante já preparou o CP/M para esta máquina.

Devemos lembrar que qualquer alteração no BIOS deve ser feita somente por pessoas que tenham profundos conhecimentos em Assembler. Se você for iniciante procure primeiro entender profundamente o CP/M, a linguagem Assembler do 8Ø8Ø ou Z8Ø, e só assim você terá condições de alterar o seu CP/M.

O BIOS do CP/M 2.2 inicia 16ØØ Hexa posições após o CCP e contém uma série de Jumps, chamado tabela de Jumps. Esta tabela deve ser mantida na ordem que se encontra.

Normalmente se você tiver uma listagem do BIOS do seu computador ele deverá começar assim:

```
                    ORG   CCP + 16ØØH    ; INÍCIO DO BIOS

JMP   COLDSTART/      ; partida à frio
JMP   WARMSTART/      ; partida à quente
JMP   CONSTATUS/      ; STATUS do console
JMP   CONINPUT/       ; entrada no console
JMP   CONOUTPUT/      ; saída do console
JMP   LISTOUT/        ; saída para impressora
JMP   PUNCH/          ; saída para perfuradora
JMP   READER/         ; entrada da perfuradora
JMP   HOME/           ; coloca disco na trilha ØØ
JMP   SETDISK/        ; seleciona disco  para uso
JMP   SETTRACK/       ; seleciona trilha
JMP   SETSECTOR/      ; seleciona setor
JMP   SETDMA/         ; determina endereço para DMA
JMP   READISK/        ; ler setor corrente
JMP   WRITEDISK/      ; grava setor corrente
JMP   LISTSTATUS/     ; mostra STATUS dos dispositivos
JMP   SECTORTRAN/     ; translação de setores
```

Se você deseja incorporar novas funções poderá acrescentá-los no final da tabela.

Cada JUMP da tabela anterior "salta" para uma rotina específica. Na próxima tabela mostramos em detalhes o que deve fazer cada rotina.

JUMP	ENTRADA	SAÍDA	FUNÇÃO
COLDSTART	-	C = Ø	A rotina deve executar todas as tarefas de inicialização do sistema.
WARMSTART	-	C = Drive	A rotina deve executar todas as tarefas de inicialização sem alterar os dados da página zero.
CONSTATUS	-	A = Status	A rotina deverá dizer se há caracter no console.
CONINPUT	-	A=Caracter	A rotina deverá ler um caracter do console.
CONOUTPUT	C=Caracter	-	A rotina deverá colocar o caracter do registrador C no console.
LISTOUT	C=Caracter	-	A rotina deverá colocar na impressora o caracter do registrador C.
PUNCH	C=Caracter	-	A rotina deverá colocar no PUNCH o caracter do registrador C.
READER	-	A=Caracter	A rotina deverá ler o caracter do READER e colocá-lo no registrador A.
HOME	-	-	A cabeça do Drive deverá ser retornada à posição da trilha Ø setor Ø.
SETDISK	C = Drive	DHA	A rotina deverá selecionar o Drive indicado pelo registrador C. O HL deverá no final conter o endereço do cabeçalho do disco.

SETTRACK	C = Trilha	-	A rotina deverá colocar o Drive na trilha indicada pelo registrador C.
SETSECTOR	C = Setor	-	A rotina deverá colocar o Drive no setor indicado pelo registrador C.
SETDMA	BC=ENDEREÇO DO DMA	-	O endereço indicado pelo par BC será o endereço usado em qualquer transferência de informação da memória para o disco e vice-versa.
READDISK	-	A = Status	Lê a trilha e setor corrente e coloca os dados a partir do endereço especificado para DMA. Um número ØlH deve retornar no Reg. A se a operação foi bem sucedida.
WRITEDISK	-	A = Status	Idêntico à função acima, apenas será uma escrita e não leitura.
LISTASTATUS	-	A = Status	A rotina deverá verificar ao dispositivo escolhido no IOBYTE se há caracter pronto voltando com FF no registrador A. Se não há caracter volta com ØØ.
SECTORTRAN	BC = Setor lógico DE=Endereço do mapa de setor	HL = Setor Físico	Uma rotina especial usada em sistema que não trabalham em setores com (128 Bytes). O setor de entrada é alterado para refletir o setor real do disco.

O IOBYTE

O IOBYTE é um Byte reservado no endereço ØØØ3H para indicar as designações correntes que associam dispositivos lógicos e físicos.

Os quatro dispositivos lógicos usados no CP/M são:

CON: Dispositivo de CONSOLE
LST: Dispositivo de LISTAGEM
RDR: Dispositivo de LEITOR
PUN: Dispositivo perfurador

A forma que está montado o IOBYTE é a seguinte:

BITS	7 6	5 4	3 2	1 0
FUNÇÃO	LISTAR	PERFURAR	LEITOR	CONSOLE

A interpretação do IOBYTE deverá ser feita da seguinte forma:

FUNÇÃO DO DISPOSITIVO LÓGICO		DISPOSITIVO FÍSICO 00	01	10	11
CONSOLE	CON	TTY	CRT	BAT	UC1
LEITOR	RDR	TTY	PTR	UR1	UR2
PERFURADOR	PUN	TTY	PTP	UP1	UP2
LISTAGEM	LST	TTY	CRT	LPT	UL1

Se você ler o IOBYTE do seu computador (End. ØØØ3H) e obtiver por exemplo 25H, indica que os dispositivos físicos estão associados aos lógicos da seguinte forma:

Um dispositivo CRT está executando a função de console
Um dispositivo PTR está executando a função de leitor
Um dispositivo UP1 está executando a função de perfurador
Um dispositivo TTY está executando a função de listador

Todas as entradas e saídas para dispositivos e console devem ser antes consultados no IOBYTE para saber qual dispositivo está sendo selecionado.

Alteração do BIOS

Lembramos novamente que esta parte é dirigida ao leitor com profundos conhecimentos de linguagem Assembler.

Vamos imaginar que você deseja acrescentar funções ao seu BIOS ou simplesmente alterá-lo em algumas funções. O procedimento mais comum é o seguinte:

1 - Utilize um disco novo recém-formatado e faça uma cópia de seu CP/M com os seguintes programa:

```
ASM.COM
DDT.COM
SYSGEM.COM
MOVCPM.COM
ED.COM
```

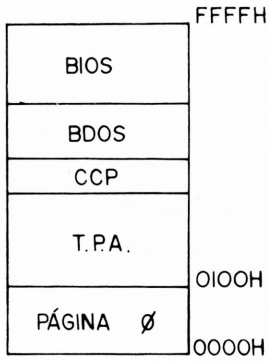
2 - Tendo em mãos uma cópia do BIOS, estude sua estrutura e sub-rotinas procurando familiarizar-se com o mesmo.

3 - Usando o ED.COM edite seu BIOS colocando no mesmo as alterações desejadas.

Importante dizer que toda alteração feita no BIOS original deve ser muito bem sinalizada no campo de comentários, com data, nome de quem fez, etc.

4 - Verifique se o novo BIOS tem um tamanho final que não estoure a memória, ou seja, se você usar um CP/M de 60K e o BIOS for muito grande, talvez você precise usar um CP/M de 54K. Para tal será necessária lançar mão do MOVCPM para criar um com capacidade menor. Isto significa o seguinte:

- O BIOS é a porção que fica na parte superior da memória, conforme demonstrado a seguir:



DIVISÃO DO CP/M

Sendo assim, se ele for ampliado poderá cair fora da área de memória, então você precisa reduzir o seu TPA (trocar de 60K para 54K, por exemplo).

5 - Utilize agora o ASM.COM para a montagem do novo BIOS. Se houver erro, identifique-o e corrija-o.

6 - Para testar o novo BIOS, crie uma imagem do CP/M com o tamanho necessário, usando o MOVCPM. Alguns MOVCPM dão uma mensagem da seguinte maneira:

READY for "SYSGEN" ou SAVE XX CPMYY.COM onde XX é o tamanho em páginas de 256 Bytes e YY é o tamanho do CP/M (60K, 54K, 16K, etc)

7 - Use o DDT para carregar este CP/M recém-criado com MOVCP/M, na memória.

8 - Use o DDT para carregar o novo BIOS alterado por você na correta área da memória. Para você saber onde é esta área, você deverá ver o "ORG" do seu BIOS, ou então verificar o endereço ØØØØ até ØØØ2 usando o DDT da seguinte:

```
-LØ,2

ØØØØ JMP XXXXH
```

O endereço XXXXH é o endereço do segundo JUMP das tarefas, então se você subtrair XXXX - 3 terá o início do BIOS.

9 - Você tem agora uma imagem do CP/M colocado a partir do endereço ØØØØH. Este CP/M contém o tamanho adequado ao

seu novo BIOS e este BIOS está alterado para as suas necessidades. Devemos agora sair o DDT com ^C e salvar este novo CP/M com um SAVE XX CPMYY.COM onde XX é o número de páginas de 256 Bytes e YY é o tamanho do CP/M.

10 - Com o SYSGEN grave este novo CP/M no disco. Em algumas versões você precisa primeiro carregar o programa na memória para depois chamar o SYSGEN, ou seja:

```
A> SYSGEN CPMXX.COM<CR>

OU

A>DDT CPMXX.COM
 C
A>SYSGEN<CR>
```

11 - Seu CP/M está instalado. Agora dê um "Reset" no seu computador e verifique se o CP/M entra e se as funções que você alterou e/ou acrescentou operam corretamente. Em caso negativo volte ao item 1 E recomece outra vez.

Certos fabricantes fornecem um manual de instruções de como alterar o CP/M instalado no computador deles, então é só seguir a orientação do manual que não haverá problemas.

6. PROGRAMAS TRANSIENTES PARA CP/M

Neste capítulo vamos abordar os programas mais comuns, construídos para CP/M. São programas chamados aplicativos, ou seja, desenvolvidos para trabalhar com CP/M, facilitar o manuseio, desenvolvimento e utilização de outros programas para CP/M.

Existe no mercado aproximadamente quatro mil tipos diferentes de programas para CP/M, desde um simples programa de cópias até sofisticadas planilhas eletrônicas como são chamados certos programas. Vamos neste capítulo mostrar alguns dos mais antigos e mais comuns que utilizamos a todo instante quando estamos operando o CP/M.

São eles:

a) ED = Editor de texto.
b) PIP = Manipulador de arquivos.
c) ASM = Assembler para o micro 8Ø8Ø.
d) LOAD = Carregador para arquivo. Com extensão em .HEX
e) DDT = Ferramenta de depuração dinâmica.

ED - Editor de Texto

Este programa é seguramente o mais popular editor de texto do mundo para CP/M.

Para iniciarmos uma seção usando o ED, devemos verificar antes se há espaço no disco para gravação do arquivo após a edição, porque os editores de texto não operam se não houver espaço disponível no disco.

Para iniciar devemos digitar:

```
        Drive          tipo do arquivo
        ↑              ↑
A>  ED  D:  NOMEARQ.  TYP  <CR>
    ↓       ↓
  editor   nome do arquivo
```

Se o arquivo já existe o ED apenas irá abri-lo, mas se for um arquivo novo ele irá criá-lo.

A seguir acham-se descritas algumas abreviações para designar comandos, os quais utilizaremos no texto seguinte:

~ indica que o comando usa sinal de + ou -

n sempre que há necessidade de um número

Sempre que for digitado o <CR>no fim de uma frase o editor coloca o LF (Lene Feed).

O editor tem vários comandos os quais podemos subdividir em comandos de movimentação de dados e comandos de edição.

Movimentação de Dados

#A <CR>

Este comando recolhe o arquivo original e coloca no Buffer de edição. No lugar do símbolo # pode-se colocar um número. O editor assume que deve recolher o número de linhas colocado no lugar do #.

nw <CR>

Com este comando você pode transportar do Buffer de edição para um arquivo temporário (NOMEARQ. $$$) criado pelo ED.

w escreve apenas 1 linha

#w escreve o Buffer inteiro

ow escreve até à metade do Buffer

O <CR>

Este comando funciona como um "Reset" em toda edição até agora desenvolvida. Ou seja, apaga o arquivo temporário, zera todo o Buffer de edição e aponta para a 1ª linha. Antes de executar este comando o ED pergunta "Y/N ?", pois você poderá perder tudo o que foi editado até o presente.

Q <CR>

Sai do editor sem qualquer alteração no arquivo que estava sendo editado, retornando ao CP/M. Antes de retornar o editor pergunta: Q - (Y/N) ? e você responderá com Y ouN, dependendo se deseja sair ou não do editor.

A figura a seguir mostra um resumo da movimentação de dados realizado pelo editor da memória para o disco e vice-versa.

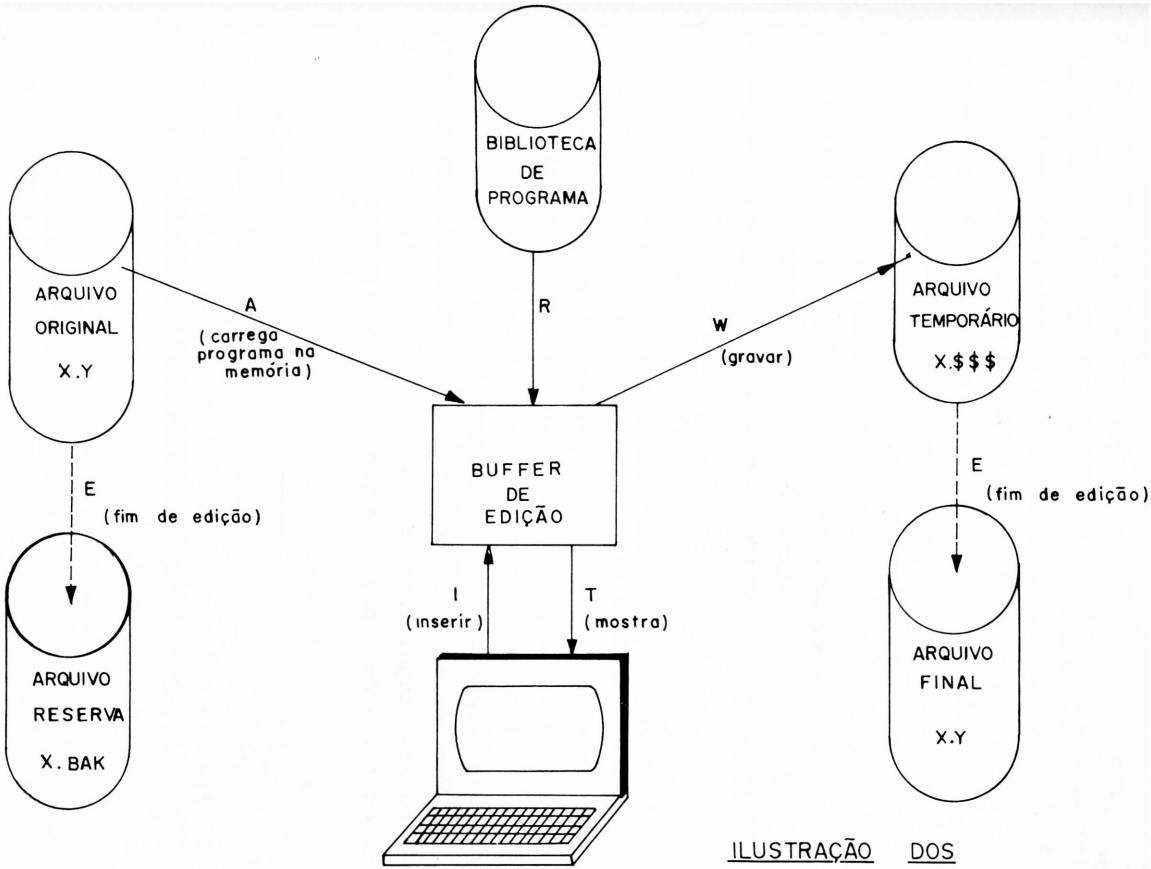


ILUSTRAÇÃO DOS COMANDOS BÁSICOS DO ED.COM

E <CR>

Fim de uma seção de edição. O texto que foi editado é transferido pra o arquivo e a remuneração dos arquivos fica da seguinte forma:

a) O texto restante no Buffer é passado ao arquivo temporário.
b) Todo o texto que resta é passado do arquivo original para o temporário.
c) O original torna-se ".BAK"
d) O arquivo temporário recebe o nome de original.
e) Volta ao sistema A>

Só usamos o comando E quando terminamos uma seção de edição.

H <CR>

Este comando é para salvar um texto que está sendo editado. Ele transfere o texto para o arquivo temporário. Todo o texto do original vai para o temporário e torna o original como ".BAK". O arquivo temporário torna-se original criando assim um novo arquivo temporário.

Você deve usar este comando quando deseja editar um texto ou programa longo, resguardando-se de possíveis problemas na máquina ou falta de energia, o que faria você perder o texto em questão.

R <CR>

Este comando lê o conteúdo de um arquivo auxiliar chamado X$$$$$$.LIB e insere no Buffer de edição a partir do ponto onde está o cursor. Este arquivo auxiliar é destruído quando do uso dos comandos E, Q ou C.

R NOMEARQ <CR>

Idêntico ao comando acima, apenas que o arquivo a ser lido é definido pelo usuário. Será colocado a partir da posição do cursor. Este arquivo auxiliar definido pelo usuário não é destruído pelo EDITOR.

nX

Transfere n linhas do Buffer de edição para o arquivo auxiliar X$$$$$$$.LIB. Usando os comandos X e R você pode transferir blocos de dados dentro de texto em edição.

Comandos quando em Edição

I <CR>

Entra no modo de edição. A partir daí todos os caracteres ASCII são aceitos. Existem alguns comandos dentro da edição, a seguir descritos:

^H - BACKSPACE elimina o caracter à esquerda do cursor
^L - insere um CR/LF
^M - insere um CR/LF
^R - reimprime a linha corrente
^X - elimina a linha corrente
^U - elimina a linha corrente

~B

Move o cursor (apontador de caracteres) para o início e fim do Buffer de edição usando respectivamente + ou - na frente do B.

~nC

Coloca o cursor em mais ou menos "n" caracteres. O CR/LF são contados como 2 caracteres.

nD

Elimina "n" caracteres imediatos antes (-) ou depois (+) do cursor.

I Texto ^Z

Insere um "texto" a partir da posição do cursor.

I Texto <CR>

Insere linha, porque neste caso há um CR/LF no final do texto.

~nK

Elimina linhas do Buffer de edição. As linhas eliminadas são as do Buffer de edição sendo que o Buffer original permanece inalterado, este passará no final a ser Back-UP.

~nP

Move o apontador de caracter uma página ou n páginas para frente ou para trás. Uma página de edição é um número de linhas que varia de CP/M para CP/M, dependendo da versão. Normalmente uma tela cheia corresponde à 24 linhas.

~nT

Para visualizar linhas que estão no Buffer de edição.

~nU

Usado para trocar minúscula por maiúscula. Para iniciar a troca digitar +U e no final digitar -U.

~nV

Com este comando pode-se visualizar os números das linhas. Digita-se -V para eliminar este efeito.

OV

Tem por função mostrar a área livre do Buffer e a área ocupada. O primeiro número é a área usada e o segundo é o máximo permitido. Subtraindo o primeiro do segundo teremos a área disponível.

n:

Este comando move o cursor para a linha número n.

:m

Inicia no cursor e vai até a linha número :m. Este comando é usado em conjunto com outros.

Ex: 80:40::60K, podemos com este exemplo eliminar as linhas 40 até 60, independente da posição do cursor.

~n

Move para frente ou para trás e mostra a linha. Este comando substitue o ~nLT. Uma maneira simples é um <CR> apenas.

Não devemos esquecer que o ED permite que se agrupem comandos em uma linha para ganhar tempo.

LOAD - Carregador para Arquivos com Extensão em .HEX.

A função do comando LOAD é de criar um arquivo .COM a partir de um arquivo .HEX. É importante frisar que este arquivo ."HEX." deverá iniciar no endereço Ø1ØØH. Após criar um programa em Assembly, usando o ED, compilando com o Assembler (ASM) tere mos um arquivo ".HEX", a partir daí é só usar o "LOAD" para criar um programa executável.

Exemplo:

```
A>LOAD B:TESTE<CR>

FIRST ADDRESS Ø1ØØ
LAST ADDRESS  Ø514
BYTES READ    Ø415
RECORDS WRITTEN Ø3
```

As mensagens que aparecem são apenas informativas como por exemplo:

```
FIRST ADDRESS (PRIMEIRO ENDERECO)
LAST  ADDRESS (ULTIMO ENDERECO)
BYTES READ (BYTES LIDOS)
RECORDS WRITTEN (GRAVACOES EXECUTADAS)
```

Mensagens de erro do LOAD

```
DISK READ ERROR
DISK WRITE ERROR
NO MORE DIRECTORY SPACE
CANNOT CLOSE FILE
```

As mensagens anteriores indicam problemas que envolvem o acionador do disco.

INVERTED LOAD ADDRESS

O programa em HEXA está iniciando fora do endereço Ø1ØØH, ou seja um endereço que está acima ou abaixo de Ø1ØØH.

DDT - Ferramenta de Depuração Dinâmica

O DDT é usado para depurar e testar programas em linguagem de máquina.

A forma de carregá-lo é a seguinte:

A> DDT <CR>

A partir deste instante o DDT se instala próximo ao CP/M, mostra a mensagem de apresentação e aguarda comandos. Podemos também carregar o DDT junto com o programa que será testado.

Exemplo: A>DDT D:TESTE.EXT<CR>

Onde D: é o Drive onde está o programa TESTE.EXT. A extensão deverá ser .HEX ou .COM com endereço inicial em Ø1ØØH.

Comandos do DDT

Resumo

A - Montagem de instruções
D - Mostra conteúdo da memória em Hexa.
F - Preenche a memória
G - Executa programa
H - Aritmética Hexadecimal
I - Carrega o FCB com nome do arquivo para carregar
L - Lista em Assembly
M - Move bloco de memória
R - Carrega programa do FCB para memória
S - Substitui o conteúdo da memória
T - Mostra conteúdo da CPU executando uma instrução
U - Executa parcialmente o programa
X - Mostra registros

An

Este comando funciona como um micro-editor de texto, porque após digitar A, endereço inicial e <CR>, poderão então serem digitadas as instruções do 8Ø8Ø.

Exemplo:

```
-A0200<CR>
0200 MOV8,A
0201 ADD C
0202 .<CR>
```

Para sair do comando An digita-se "." e <CR> ou então um caracter qualquer e <CR>.

Dn,m

Este comando permite visualizar um bloco de memória com 192 Bytes por vez, se for digitado apenas D <CR>. A forma que aparecerá no vídeo será semelhante à da figura a seguir:

```
0100,012F

0100 01 BC 0F C3 3D 01 43 4F 50 59 52 49 47 48 54 2D ...=COPYRGHIT
0110 28 43 29 20 31 39 38 30 2C 20 44 49 47 49 54 41(C)1980 DIGITA
0120 4C 20 52 45 53 45 41 52 43 48 20 20 20 20 20 L RESEARCH
```

Podemos "dizer" ao DDT qual o endereço inicial digitando Dn <CR> ou também o endereço inicial e final com o comando Dn,m <CR>.

Fx,y,z

Este comando permite preencher um bloco de memória do endereço x até y com a constante Hexadecimal de 8 Bits,z. Comando muito usado para zerar bloco de memória.

Exemplo:

```
F0100,01FF,00
```

Com este comando vamos colocar ØØ do endereço Ø1ØØ até Ø1FF.

Gx,y <CR>

Este comando permite executar um programa existente na memória. Há quatro métodos para aplicar o comando G. O primeiro seria "G <CR>" onde a execução partiria do endereço especificado no PC. A segunda maneira seria "G,x <CR>" onde x é o endereço de Breakpoint, ou seja, endereço onde o programa foi interrompido. Na terceira forma fixa-se o endereço inicial "x" e o final "y" da seguinte maneira "Gx,y <CR>". Por último é quando você não

coloca Breakpoint, ou seja, o programa irá "rodar" indefinidamente, apenas sendo definido o endereço inicial. A forma será Gx <R.

Hx,u <CR>

Este comando permite fazer a soma e a subtração Hexadecimal dos números x e y. O primeiro número mostrado é a soma e o segundo a subtração.

I NOME.EXT <CR>

Com este comando vamos identificar um ou mais arquivos que se deseja carregar na memória. Para efetuar o transporte do arquivo do disco para a memória será usado o comando R <CR>. Quando digita-se um comando I seguido do(s) nome(s) do(s) arquivo(s), o(s) mesmo(s) ficarão a partir do endereço ØØ5CH, e com o comando R vamos carregar o arquivo especificado do Drive corrente. Não é possível especificar o Drive no comando I, será necessário usar o comando S e alterar o conteúdo do endereço ØØ5C Hexa conforme tabela a seguir:

POSIÇÃO ØØ5C em Hexa	Drive Selecionado
ØØ	Drive corrente
Ø1	Drive B
Ø2	Drive C
Ø3	Drive D
Ø4	Drive E
Ø5	Drive F
Ø6	Drive G
Ø7	Drive H
Ø8	Drive I
Ø9	Drive J
ØA	Drive K
ØB	Drive L
ØC	Drive M
ØD	Drive N
ØE	Drive O
ØF	Drive P

L <CR>

Com este comando podemos ter um "LIST" no vídeo após o último endereço listado, conforme exemplo a seguir:

```
L<CR>
0100 MOV D,B
0101 CALL 0005
0104 ADD B
0105 PUSH H
0106 MOV B,C
0107 MVI H,10
0109 CALL 1000
010C JMP 0100
010F ADC C
0110 NOP
0111 NOP
```

Quando o DDT não encontrar um símbolo em Assembler para o Hexadecimal encontrado, ele apresenta a mensagem ?? = XX onde XX é o valor em Hexa do endereço especificado.

Lxx,yy <CR>

Como outros comandos do DDT pode especificar somente o endereço inicial, final ou ambos. Não se deve esquecer a vírgula.

Mxx,yy,zz <CR>

Este comando é chamado movimento de bloco de memória onde xx é o endereço inicial e yy o endereço final do bloco a ser transferido para o endereço inicial zz.

Rxxxx <CR>

Com este comando efetua-se a carga do arquivo especificado pelo comando I mais o deslocamento xxxx. Se naõ for especificado o endereço xxxx o suposto deslocamento será zero. Se o arquivo tiver extensão .COM será carregado no endereço 0100H se não for definido o deslocamento.

Exemplo:

```
I TESTE.COM<CR>

R<CR>
0100
-
```

Se for digitado o comando I seguido de um arquivo com extensão .HEX ao usar o comando R para ler o arquivo o mesmo será carregado a partir do endereço especificado no arquivo.

Como exemplo imagine o programa abaixo guardado num arquivo chamado TESTE.HEX. Quando for carregar este arquivo na memória o DDT se comportará da seguinte forma:

```
PROGRMA COM 1K DE MEMORIA

A>DDT TESTE.HEX

DDT VERS 2.2
NEXT  PC
04FF  0000
-
```

Sxxxx

Este comando mostra o conteúdo do endereço xxxx permitindo ou não substituir o seu conteúdo e a partir daí desloca para o próximo endereço.

Exemplo:
```
S1000<CR>
1000=BC

1001=FF
```

Então você digita um número qualquer e pressiona <CR>, e o DDT mostrará o próximo endereço:

```
1001=FF
```

Tx <CR>

Este comando é o comando TRACE que tem a função de mostrar a situação dos registradores do 8080 e qual a próxima instrução a ser executada. O número X indica quantas instruções devem ser visualizadas. Um programa acompanhado pelo comando Tx será executado muito lentamente, cerca de 500 vezes mais lento.

O TRACE será visualizado da seguinte maneira.

```
C0Z0M0E010 A=FF B=0123 D=FFFF H=0000 S=0100 P=0101 MVI A,01
```

onde

CØ = indica que o Flag Carry está resetado (estado Ø)
ZØ = indica que o zero está resetado (estado Ø)
MØ = indica que o sinal está resetado (estado Ø)
EØ = indica que a paridade está resetada (estado Ø)
IØ = indica que o Auxiliar Carry está resetado (estado Ø)
A = FF = indica que o registrador A tem FFH
B = Ø123 = indica que o par de registradores BC contém Ø123 H
D = FFFF = indica que o par de registradores DE contém FFFF H
H = ØØØØ = indica que o par de registradores HL contém ØØØØ H
S = Ø1ØØ = indica que o STACK POINTER contém Ø1ØØ H
P = Ø1Ø1 = indica que o CONTADOR DE PROGRAMA contém Ø1Ø1 H
MVI A,Ø1 = indica a próxima instrução a ser usada.

Ux <CR>

Este comando é parecido com o T, apenas ele mostra o conteúdo dos registradores após a execução de um trecho do programa. E como colocar uma interrupção num comando G, precedendo-o com um comando x.

X <CR>

Com este comando você pode examinar e alterar o conteúdo de qualquer registrador da CPU bem como contador, Stack ou Flags.

O DDT é uma ferramenta poderosa para verificar, alterar e fazer funcionar programas próprios para rodar na máquina, porque você pode testar o seu programa de todas as formas. Mas para testar programas de outro tipo de computador como por exemplo programas para rodarem numa máquina de controle de processo, teste e etc, torna-se necessário bloquear partes deste programa, simular certos efeitos que não seria possível executar dentro de seu computador, devido ao fato de o Hardware ser diferente. Por exemplo se o seu programa que está sendo testado irá ter a função de ler um dado de uma porta de I/O com número x e então executar um determinado cálculo com o valor lido, com o DDT pode-se testar o cálculo a ser utilizado mas a leitura do canal de I/O número x deverá ser simulada, porque provavelmente o seu com

putador não terá este canal de I/O, por isso você deve usar o comando X e colocar o dado que você deseja direto no acumulador.

PIP - Programa para Intercâmbio de Periféricos

Este programa é usado para copiar informações de um lugar para outro, dentro do computador. Pode-se copiar um arquivo de um disco para outro, de um disco para impressora, etc. Estes arquivos podem ser ambíguos e o número dos parâmetros variável.

Podemos com o PIP trocar informações entre os dispositivos do sistema.

A forma básica de operá-lo é a seguinte:

```
A> PIP  linha de comando  <CR>                    (1)

      ou então

A> PIP  <CR>                                      (2)
* linha de comando
↑
└──O PIP coloca asterisco
```

A primeira forma (1) é quando se deseja apenas uma única operação. A segunda forma (2) é quando se deseja várias operações com o PIP, portanto o PIP coloca o asterisco e aguarda comandos. Para encerrar na segunda forma digita-se no final ^C.

O comando no PIP deverá ter sempre a configuração:

DESTINO = FONTE <PARÂMETROS>

O destino entende-se quem receberá as informações e fonte quem as fornece. Parâmetros são como serão transmitidas estas informações.

A seguir temos algumas operações básicas com o PIP em arquivos:

- Copiar vários arquivos de um disco para outro
- Copiar um arquivo de um disco para outro
- Copiar um disco inteiro para outro
- Criar um arquivo concatenando vários outros
- Mostrar o conteúdo de um arquivo que está sendo copiado

A seguir temos algumas operações com dispositivos usando o PIP:

- Mandar as informações de um arquivo do disco para um dispositivo
- Passar informações de um dispositivo para outro
- Jogar dados de um arquivo para impressora
- Converter maiúsculas em minúsculas e vice-versa
- Guardar num arquivo os dados vindos de um dispositivo de entrada

Operações com o PIP

Para carregar o PIP na memória digita-se:

PIP <CR>

E quando for retornar ao CP/M ditita-se:

^C

Para copiar um arquivo de um disco para outro digita-se:

PIP D: ARQ1.EXT = D: ARQ2.EXT (P) <CR>

Copia o arquivo ARQ2.EXT especificado no Drive D: para o arquivo ARQ1.EXT no Drive D: com os parâmetros P

Se deseja-se copiar o disco inteiro digita-se:

A> PIP B: = A: *.* <CR>

Será copiado dentro do Drive B todos os arquivos contidos no Drive A.

Pode-se abreviar os comandos quando deseja-se copiar um arquivo de um disco para outro, usando o mesmo nome.

Exemplo:

A> PIP A: B: TESTE.EXT <CR>

Para se interligar dois arquivos e gravá-los num terceiro teremos:

PIP D: ARQ1.EXT = D:ARQ2.EXT (P), D: ARQ3.EXT (Q)<CR>

Será criado um arquivo ARQ1.EXT no Drive D:, que consiste de dois arquivos: o ARQ2.EXT com os parâmetros (P) mais o ARQ3.EXT com o parâmetro (Q).

Dispositivos Especiais do PIP

EOF:

Manda um caracter ^Z ou 1A Hexa para indicar fim de arquivo para o dispositivo destino.

NUL:

Manda 4Ø caracteres nulos (ØØ) para o dispositivo destino. Esta operação era usada antigamente quando havia as fitas perfuradas, porque o envio de 4Ø caracteres em branco na fita facilitava o manuseio.

PRN:

Manda dados para o dispositivo LST. Quando você quer jogar as informações contidas num arquivo para impressora usa-se o comando PRN.

INP: E OUT:

São rotinas de entrada e saída respectivamente criados pelo usuário para o PIP.

Esta rotina especial de entrada deverá ter seu início especificado no endereço 1Ø3, 1Ø4 e 1Ø5 Hexa e o dado deverá estar no endereço 1Ø9 Hexa. A rotina especial de saída deverá ter seu início especificado no endereço 1Ø6, 1Ø7, 1Ø8 Hexa. O PIP terá a informação a ser transmitida contida no registrador C. A área de 1ØAH até 1FFH é livre para colocar as rotinas.

As formas de usar os dispositivos são:

```
PIP D: ARQ.EXT=DISP:(P)<CR>

OU

PIP DISP:=O: ARQ.EXT(P)<CR>

OU

PIP DISP1:=DISP2:(P)<CR>
```

Onde ARQ.EXT é o nome do arquivo e DISP um dispositivo qualquer.

Parâmetros Gerais do PIP

(B)

Parâmetro de transferência do bloco. Usa-se o B para os dispositivos fonte que transferem dados continuamente e que poderiam lotar o Buffer do PIP. Para periféricos lentos deve-se usar o B, mas para operação entre discos não é necessário.

(DX)

Elimina os caracteres depois da coluna x. O número X vai de 1 a 255. Com este parâmetro pode-se eliminar os caracteres após a coluna x.

(E)

Mostra no vídeo (ecoa) o que está sendo cópia de um dispositivo para outro. É um método bom para verificar o que se está copiando.

(F)

Filtra os Form-Feed's do arquivo original. Muito usado para economizar papel, mostrar o arquivo no vídeo ou para uma impressora que não entende o FF (Form-Feed).

(Gx)

Copia arquivos de outros usuários, ou seja o x poderá variar de Ø a 15 dependendo em qual usuário está o arquivo que se deseja copiar.

(H)

Transferir dados de um dispositivo ou disco para outro, no formato Hexa da INTEL.

(I)

Ignora registros nulos na transferência em formato Hexa da INTEL.

(L)

Faz a conversão de maiúscula em minúscula.

(N)

Coloca número crescente em cada linha transferida. A cada <CR> que aparece o PIP incrementa o contador interno colocando um número na linha.

Exemplo:

Sem o (N)	Com o (N)
TESTE NO	1 TESTE NO
PIP VERIFICA	2 PIP VERIFICA
O PARÂMETRO (N)	3 O PARÂMETRO (N)

(O)

Com este parâmetro podemos fazer a transferência de um arquivo não ASCII, ou seja, em código objeto. O parâmetro (O) diz ao PIP para tratar por exemplo o caracter ^Z (IAH) como um caracter qualquer. Sempre que tiver um programa com extensão (.COM) é necessário usar o parâmetro (O).

(PC)

Usado para gerar um Form-Feed após a linha x. O número x deverá estar entre 1 a 255. Se não for digitado o número o PIP supõe-se 60 linhas.

(Q TEXTO ^Z)

Faz a cópia de uma parte do arquivo até encontrar a palavra "Texto".

(R)

Permite que o PIP copie um arquivo do sistema.

(S TEXTO ^Z)

Faz a cópia dos dados de um dispositivo a partir da seqüência "TEXTO".

(Tx)

Gera a tabulação a cada x coluna

(U)

Troca minúscula para maiúscula.

(V)

Verifica se a cópia foi executada com sucesso. O arquivo destino deverá ser o disco, caso contrário o V será ignorado.

(W)

Permite ao PIP escrever num arquivo onde está especificado o atributo R/O. Se tentarmos escrever num arquivo R/O sem usar o parâmetro (W) receberemos a mensagem:

DESTINATION FILE IS R/O

(Z)

Zera o Bit de paridade durante o processo de transferência. Devemos usar o (Z) para enviar o oitavo Bit em zero dos caracteres ASCII.

ASM - Linguagem Assembler

Como já é de nosso conhecimento, o computador trabalha apenas com sinais digitais (Ø e 1), sinais cuja manipulação é difícil e trabalhosa. Para resolver este problema, foi criada uma forma de representação onde as instruções, conjuntos de números binários, são representados por mnemônicos, sendo estes de fácil manipulação.

Como exemplo, tomemos uma das instruções do microprocessador 8Ø8Ø:

ADI XXH

ADI = Mnemônico, cujo correspondente em Hexadecimal é C6H, onde C6H corresponde em binário à |11ØØ |Ø11Ø|
C 6

O que esta instrução realiza é a soma do dado representado em XX com o conteúdo do acumulador.

O programador elabora o programa, lançando mão das instruções na forma de mnemônicos e o programa ASM traduz o mesmo para código objeto. Quando você for verificar um programa que está pronto para "rodar", ou seja, em código objeto, irá verificar que o resumo não está em binário e sim em Hexadecimal, isto

é devido ao fato de que manipular os sinais em 1 e Ø é como já dissemos difícil, portanto os sinais ficarão agrupados conforme exemplo a seguir:

BINÁRIO			HEXA
11Ø1	1ØØ1	=	D9H
D	9		
ØØ11	ØØ1Ø	1Ø1Ø Ø111	= 32A7H
3	2	A 7	

O CP/M foi criado a partir da CPU 8Ø8Ø da INTEL, mas o sistema funcionará perfeitamente com Z8Ø e 8Ø85 porque os mesmos utilizam instruções idênticas às do 8Ø8Ø.

Há vários Assemblers atualmente no mercado como por exemplo:

ASM - analisado neste capítulo
ASM86 - idêntico ao ASM, mas para a CPU 8Ø86
M8Ø - macro Assembler para 8Ø8Ø e Z8Ø
ZASM - Assembler para o Z8Ø
MAC - Assembler da Digital Research parecido com o ASM, só que mais completo.

Linhas de Comando do ASM

Depois que você criar o programa num editor de texto tem agora que converter para código objeto executável. Para tal deve chamar o ASM da seguinte forma:

A> ASM NOME.OPC <CR>

O exemplo dado significa:

ASM - Assembler
NOME - arquivo a ser "Assemblado"
OPC - opções como segue:

1º letra o Drive onde se localiza o arquivo fonte
2º letra o Drive onde ficará o arquivo objeto

3º letra você usará para definir o Drive que receberá o arquivo .PRN

Há duas opções para serem usadas:

X = listar o arquivo PRN no vídeo ao invés de armazenar no disco.

Z = Para evitar a geração dos arquivos Hex e PRN.

Quando você chama o Assembler aparecerá uma mensagem durante o processo:

```
CP/M ASSEMBLER-VER 2.0
```

Se ocorrer erro, então aparecerá a mensagem de erro, caso contrário:

```
XXXX
ZZZH USE FACTOR
AND OF ASSEMBLY
```

Onde XXXX significa, onde inicia o programa enquanto ZZZ é a indicação da área da tabela de símbolos usados.

Arquivos Criados pelo ASM

Quando você criou um arquivo TESTE.ASM com o editor foi automaticamente criado o programa TESTE.BAK pelo editor como segurança. Após aplicar o ASM teremos os arquivos TESTE.PRN e TESTE.HEX.

Um exemplo de .HEX é dado a seguir:

```
:100100000E01CD0500FE24CA00000E055FCD0500DE
:03011000C3000128
:0000000000
```

Um exemplo de .PRN é o dado a seguir:

```
0100                    ORG     0100H;      INDICADOR DE INICIO DO PROGRAMA
0100 0E01       TESTE:  MVI     C,01H;      PREPARA PARA LER O CARACTER
0102 CD0500             CALL    0005H;      CHAMA O BDOS
0105 FE24               CPI     '$';        VERIFICA SE CHEGOU NO FIM
0107 CA0000             JZ      0000H;      SE FIM RETORNAR PARA O CP/M
010A 0E05               MVI     C,05H;      PREPARA PARA ESCREVER NA IMPRESSORA
010C 5F                 MOV     E,A
010D CD0500             CALL    0005H;      CHAMA O BDOS
0110 C30001             JMP     TESTE;      VOLTA AO INICIO
```

O programa que você criou será idêntico à forma do exemplo a seguir:

```
            ORG 100H
INICIO: MVI     C,01H; LER CARACTER DO CONSOLE
        CALL    0005H; CHAMA BDOS
        MVI     C,05H; PREP.P/ ESCREVER NA IMPRESSORA
        MOV     E,A
        CALL    0005H
        JMP     0000H
        END
```

Pode-se verificar que no TESTE.ASM existe apenas instruções em Assembler e no TESTE.HEX as instruções estão em Hexadecimal no formato INTEL. O .PRN nos mostra a função dos dois arquivos acima.

Organização do Programa Fonte

O programa que você vai criar deverá seguir alguns critérios específicos. O "ASM" quando foi compilar um programa deverá encontrá-lo como segue:

LABEL MNEMÔNICO OPERANDO; COMENTÁRIOS

LABEL

O Label é um identificador usado para representar um endereço importante ou (subrotina) que deseja-se acessar. Deve ser construído de 1 a 6 caracteres, sendo que o 1º caracter tem que ser letra e os outros podem ser números ou letras.

Algumas palavras especiais não podem ser usadas, como por exemplo:

OS MNEMÔNICOS DO 8Ø8Ø
Os nomes e diretivas
Os nomes dos registros

MNEMÔNICO

São as instruções do 8Ø8Ø. Neste livro, especialmente neste capítulo, subentende-se que o leitor conheça programa em linguagem Assembler.

OPERANDO

Um operando pode ser uma letra ou um número, sendo que algumas instruções podem ser um ou mais operando.

COMENTÁRIOS

Este campo é opcional, pois é reservado para o programador comentar o que está acontecendo no programa. É importante para a fase de documentação.

Diretivas do Assembler

Existem algumas instruções especiais que podemos incluir no programa que não fazem parte do conjunto normal das instruções do 8Ø8Ø.

Estas instruções servem para gerar algumas situações especiais no programa fonte, contornando dificuldades causadas pelo uso de instruções tradicionais.

As diretivas são divididas em 3 grupos:

Diretivas para abrir áreas de memória

DB - define um Byte (cria uma área Byte por Byte)

DW - define uma palavra (cria uma área de dois Bytes por vez)

DS - definição de área. (Reserva uma área de memória de tamanho especificado).

Diretivas de locação

ORG - Esta diretiva indica o endereço inicial das instruções que a seguem. A forma do comando é:

LABEL ORG EXPRESSÃO; comentários.

END - Esta diretiva indica o fim do programa. A forma é a seguinte:

LABEL END expressão; comentário.

Este comando é opcional.

EQU - Esta diretiva atribui um valor ou uma expressão a um Label qualquer. O formato é:

LABEL EQU expressão; comentário.

Diretivas condicionais

Esta diretivas são usadas quando criamos programas que terão subrotinas que serão interligadas ou não dependendo das condições, conforme exemplo:

```
LABEL IF expressão; comentário
        .
        .
        .
determinações subseqüentes
        .
        .
        .
ENDIF     , comentário
```

O programa irá calcular a expressão após o IF, se a mesma resultar em valor zero, as instruções que estiverem entre o IF e o ENDIF serão desprezadas, se a expressão tiver resultado diferente de zero, então esta rotina será incluída no programa principal.

Mensagens de erro do Assembler

NO FILE - Não foi encontrado o arquivo.

DIRECTORY FULL - Não há espaço no diretório para carregar o programa que está sendo gerado. A solução será destruir arquivos desnecessários do disco.

SOURCE FILE NAME ERROR - Você usou nomes como "*" ou talvez "?".

DISK READ ERROS - Há problema com o disco, pois o Assembler não conseguiu ler o programa.

OUTPUT FILE WRITE ERROR
O disco está protegido contra escrita.

Quando o Assembler estiver "trabalhando" no seu programa para traduzir as instruções que estão em Assembly e passar para código objeto, poderá ser encontrado erros, e a forma de o Assembler indicar será a seguinte:

```
x  yyyy  zzzz  LABEL  MNEMÔNICO operando; comentário
```

O "x" indica o tipo de erro, conforme a tabela de erros, o yyyy mostra em que endereço isto acontece e o zzzz será o que o Assembler colocou em linguagem de máquina. Normalmente o Assembler coloca zero no total ou em parte, porque quando ele encontra algum código (ou mnemônico) desconhecido ele não sabe o que deve colocar.

Tabela de Erros do Assembler

D - Erro no dado, o valor deve ser maior que o recomendado, não cabe na área especificada.

E - Erro de expressão, você provavelmente formulou a expressão erradamente.

L - Erro no Label, você construiu um Label impróprio.

N - Comando não interpretadc - Você usou diretivas que o ASM não conhece, apenas o MAC é capaz de reconhecer.

O - Erro de OVERFLOW - Sua expressão é muito complexa. Necessário dividí-la em partes.

P - Erro de fase - Você usou dois Labels com o mesmo nome.

R - Erro de registrador - Você especificou um registro impróprio ou seja DAD B é possível, mas DAD A não.

S - Erro de sintaxe - Este é o erro mais comum, você escreveu a instrução erradamente. Por exemplo, ao invés de escrever MVI B, você escreveu MUI B.

U - Símbolo sem definição - Você usou um Label ou símbolo sem uma definição prévia. Por exemplo, L X I H,ENDEREÇO - A palavra endereço não foi especificada no começo do programa com uma diretiva EQU.

V - Erro de valores - Você provavelmente esqueceu uma vírgula, ponto ou letra num determinado valor digitado.

Outras Linguagens e Programas

Neste capítulo nós abordamos os programas mais comuns para CP/M, mas devemos lembrar que atualmente existe no mercado milhares de programas e dezenas de linguagens já disponíveis para este sistema operacional.

Na área de editores de texto temos o WordStar que seria o mais conhecido e importante. Para banco de dados existe o DBII. Na parte das linguagens podemos citar entre outras o COBOL, BASIC, FORTRAN C, PASCAL e etc.

Concluindo temos que desde quando foi lançado o PIP, ED, ASM até os dias de hoje, muito se fez e muito ainda se fará para ser usado no sistema operacional CP/M-8Ø.

7. UM COMPUTADOR COM CP/M

Depois de analisarmos o que seja um sistema operacional, o que é o CP/M, quais os seus comandos internos, externos, os programas transientes (aplicativos diretos) e os aspectos técnicos, vamos agora analisar o conjunto, verificar qual a configuração mínima de um computador para usar o CP/M e uma parte prática, ou seja uma "seção" com CP/M.

Configuração Mínima

O sistema tradicional para o CP/M é 64K de RAM, 2 unidades de disco, uma impressora, um vídeo de 80 colunas por 24 linhas e uma CPU com Z80, 8085 ou 8080. Mas devido às dificuldades de cada um, este sistema pode ser ampliado ou reduzido, ou seja:

- RAM, pode-se trabalhar desde 16K até 64K lembrando que certos programas para CP/M só trabalham a partir de 48K. Há versões modernas do CP/M (CP/M 3.0) que trabalham com 128K.
- Disco, o CP/M 2.2 aceita até 16 unidades de disco, mas podemos trabalhar com um mínimo de 1 unidade. Deve-se deixar claro que trabalhando com 1 Drive as dificuldades para cópia, troca de Disquete, etc, torna o trabalho cansativo. Pode se começar com um disco de no mínimo 256K simples/simples e depois evoluir para um sistema de dupla/dupla.

Impressora

Esta unidade pode-se dizer que seria o último dispositivo a ser adquirido, por ser o mais caro e cuja utilização pode ser contornada. Por exemplo, verificar um programa usando o vídeo, não tirar relatórios, etc.

Mas lembrar que apesar de ser o último a ser adquirido, não significa ser supérfluo. A impressora é um dispositivo importante. Pode-se por questão de economia começar com uma impres

sora 8Ø CPS e ir evoluindo até uma impressora veloz.

Vídeo

O sistema usado em CP/M trabalha com vídeo de 8Ø colunas por 24 linhas. Este sistema é padrão para uma infinidade de programas como WordStar, dB II, Cobol, etc. Você pode usar um vídeo com 64 colunas por 16 linhas, mas não poderá usar certos programas porque o vídeo ficará confuso. O que faz um vídeo ser 8Ø por 24, 64 por 16 e etc, é o circuito de deflexão horizontal e a banda passante no canal de vídeo que no televisor normal é 4,5 MHz (64X16) e no vídeo de 8Ø X 24 deverá ser no mínimo 1Ø MHz. Hoje em dia quando você adquirir um monitor de vídeo profissional, este já estará preparado para receber 8Ø colunas, sendo que se for usado um televisor tradicional você deverá adaptá-lo para uma banda maior e diminuir o tempo de deflexão horizontal.

CPU

O CP/M foi originalmente escrito para 8Ø8Ø. Como o 8Ø85 e Z8Ø são compatíveis, então pode-se usar qualquer um dos três microprocessadores.

Operando o CP/M

Vamos à uma "seção" de um computador com CP/M. Deseja-se construir um programa aplicativo qualquer.

Inicialmente ligamos o computador, e aguardamos o CP/M ser carregado do disco para a memória. Digitamos em seguida alguns protocolos que porventura existam no computador (Hora, dia, nome do operador, etc.)

Começamos criando o arquivo com ED.COM

A> ED NOMEARQ. ASM <CR>

Com este comando estamos obstruindo um arquivo caso este já exista, ou então criando e abrindo um arquivo novo, caso este não exista.

A <CR>

Com este comando vamos carregar no Buffer de edição linhas do arquivo em disco (como foi comentado no capítulo sobre o ED.COM).

I <CR>

Com este comando entramos no modo de edição. A partir daí deve-se digitar o programa, conforme foi analisado no capítulo sobre o ED.COM

Vamos assumir o exemplo abaixo:

```
        ORG     0100H;          INDICADOR DE INICIO DO PROGRAMA
TESTE:  MVI     C,01H;          PREPARA PARA LER O CARACTER
        CALL    0005H;          CHAMA O BDOS
        CPI     '$';            VERIFICA SE CHEGOU NO FIM
        JZ      0000H;          SE FIM RETORNAR PARA O CP/M
        MVI     C,05H;          PREPARA PARA ESCREVER NA IMPRESSORA
        MOV     E,A
        CALL    0005H;          CHAMA O BDOS
        JMP     TESTE;          VOLTA AO INICIO
        END
```

O programa acima lê um carater do teclado e joga na impressora.

Após o término da edição digita-se ^Z em seguida digitar E. Agora temos o arquivo criado e editado com uma gravação no disco e um Back-Up.

A MAC NOMEARQ ASM <CR> ou ASM NOMEARQ <CR> ou M80 D: = D: NOMEARQ <CR>

Realiza a função Assembler do programa editado.

O programa MAC é similar ao ASM que foi descrito anteriormente. O programa M80 é um macro Assembler poderoso, largamente usado sendo que o MAC e o ASM criam arquivos com extensão Hex, enquanto o M80 cria também Hex, mas é largamente usado por gerar a extensão "REL". Esta extensão é chamada de arquivo com endereço relativo. Se após a complicação o programa apresentar algum erro, este será mostrado no vídeo. Então deve-se entrar no item A novamente e corrigir os mesmos.

A> LOAD NOMEARQ ou L8Ø NOMEARQ, NOMEARQ / N / E

Passamos agora para a fase de ligar o programa (L8Ø) ou carregar (LOAD). Estes dois programas têm a função de transformar um arquivo Hex em COM (LOAD) e um arquivo REL em COM (L8Ø).

Neste momento temos no disco um programa chamado NOMEARQ.COM pronto para ser executado. Mas, precisamos verificar se não há nenhum problema na estrutura do programa, porque os erros de sintaxe e situações impróprias já foram sanados com o Assembler. Normalmente programas que acabaram de ser desenvolvidos contêm problemas e defeitos, para isso vamos ao item D que é o Debbuging.

A> DDT NOMEARQ.COM <CR>

O programa está colocado do endereço Ø1ØØh até o endereço Ø1ØEh.

Com o comando L<CR> podemos ter uma listagem do programa que está na memória. Com o comando D<CR> teremos uma lista do programa em HEXA. Podemos agora com o comando T<CR> executar o programa passo a passo, e verificar se o mesmo opera conforme o nosso projeto exemplo. Se der algum problema pode-se alterar com o comando S ou A conforme aprendizado do capítulo sobre programa aplicativos para CP/M.

Portanto ao usar os comandos do DDT você estará "debugando", tirando os problemas, ensaiando o seu programa, se durante este processo você for encontrando problemas no programa altere direto na memória, coloque o programa para rodar até o seu completo funcionamento. Procure durante este processo ir assinalando as modificações para depois através do "ED" alterar o programa fonte. Ou até mesmo para que cada alteração seja feita direto no programa fonte e então compilado e ligado para então continuar os testes com o DDT Quando tudo estiver OK teremos o programa pronto para rodar.

A prática com estes programas descritos aqui e outros fará com que você adquira flexibilidade em operá-los.

Se o programa que você está desenvolvendo for usado em um Hardware próprio, ou seja, sem CP/M, então há algumas alterações, o ORG deverá ser no endereço apropriado, o que na maioria das vezes é ØØØØ.

Não pode usar o LOAD, apenas o L8Ø. Neste momento será necessário gravar o programa em PROM ou EPROM. Para isto o seu computador deverá estar equipado para tal.

O motivo da não possibilidade de usar-se o programa LOAD para transformar a extensão HEX em COM é devido ao fato que o LOAD foi construído para transformar apenas programas para rodar dentro do CP/M. O programa L8Ø é um Software largamente usado em compiladores Cobol, Pascal, etc.

A construção de Software não compatíveis com CP/M (ou seja para um Hardware próprio) exige do programador conhecer o Hardware em questão, em que posição da memória irá este programa operar, etc. Em tais programas não é na maioria das vezes possível ser feito o teste no DDT devido ao envolvimento de Hardware, por isso é costume serem testadas apenas as partes que não envolvem I/O e determinados blocos ou endereços de memórias.

A prática e treino com os programas aqui descritos neste livro, levará o leitor a sentir a importância de um sistema operacional num computador, a sua responsabilidade para gerenciar a máquina.

Por mais sofisticados que sejam os sistemas operacionais atualmente, por mais velozes, poderosos e complexos, ainda com relação ao CP/M temos muito a aprender.

APÊNDICE A - RESUMO DOS COMANDOS DO CP/M

Neste apêndice vamos mostrar de forma resumida os comandos do sistema operacional CP/M e de seus programas aplicativos mais comuns.

Se o seu CP/M tem alguma diferença com relação ao que aqui foi apresentado procure escrever ao lado do comando do livro, a forma correta de utilização no seu computador.

Comandos de Edição de Linha

CONTROL C
Faz partida à quente. Reinicialização do CP/M.

CONTROL E
Move o cursor para o início da próxima linha.

CONTROL H
Elimina o último caracter digitado. (CP/M 2.2 o mais recente).

CONTROL J
O mesmo que <CR>.

CONTROL M
O mesmo que <CR>.

CONTROL P
Liga a impressora. Se novamente acionado desliga o mesmo.

CONTROL R
Repete a limha de comando digitada.

CONTROL S
Congela a impressão de dados no vídeo. Acionando qualquer tecla, volta à impressão.

CONTROL U
Cancela a linha de comando atual.

CONTROL X

Mesmo que Control - U.

Comandos Internos do CP/M

DIR D: NOMEARQ.EXT <CR>

Mostra o diretório do disco D: (se ausente assume o Drive corrente) ou nomes que coincidam com o ambíguo (*EXT) ou (NOMEARQ.*).

TYPE D: NOMEARQ.EXT <CR>

Mostra no vídeo o conteúdo de um arquivo em formato ASCII

SAVE N D: NOMEARQ.EXT <CR>

Grava uma área do T.P.A (a partir do end. 1ØØH) com N blocos de 256 Bytes no disco D: com o nome determinado no comando.

USER n <CR>

Troca o usuário do disco para "n", onde n vai de Ø a 15.

D: <CR>

Troca o Drive corrente para D:

REN D: NOVO.EXP = VELHO.EXP <CR>

Troca o nome do arquivo de velho para novo no diretório.

ERA D: NOME.EXT <CR>

Apaga o arquivo "NOME.EXT" do disco D:. Pode-se usar caracter ambíguo (NOME. * ou *.EXT).

Comandos Externos

DUMP D: NOME.COM

Mostra no vídeo, em Hexa, o conteúdo de um arquivo ".COM" (arquivo em programa objeto) residente no Drive D:.

SYSGEN <CR>

Carrega o CP/M de um disco para outro.

LOAD D: NOME <CR>

Converte um arquivo ".HEX" em arquivo executável ".COM."

MOVCPM nn <CR>

Prepara uma nova cópia do CP/M que tem "nn" K de memória; passa o controle para este novo CP/M. Não salva em disco.

MOVCPM nn <CR>

Prepara uma nova cópia do CP/M, mas não passa o controle para este novo CP/M, permitindo assim que usado o SYSGEN ou SAVE que você grave este novo CP/M em disco.

STAT <CR>

Mostra atributos e o espaço livre do disco corrente.

STAT D: <CR>

Mostra atributos e o espaço livre do disco especificado (D:).

STAT D: NOME.EXT <CR>

Mostra atributos e tamanho do(s) arquivo(s) especificado(s). O nome pode ser ambíguo.

STAT DEV: <CR>

Relaciona todos dos dispositivos físicos que estão associados aos quatros dispositivos lógicos.

STAT VAL: <CR>

Relaciona todas as atribuições existentes.

STAT LOG: = FIS:

Atribui ao dispositivo lógico o dispositivo físico.

STAT D: = R/O

Atribui temporariamente o Status R/O para o Drive D:

SUBMIT NOME <CR>

Cria um arquivo $$$.SVB onde está contido comandos existentes no arquivo NOME.EXT. O CP/M executa os comandos do arquivo $$$.SUB até o fim.

ASM NOME.ABC <CR>

Monta o arquivo NOME.ASM que está no Drive A:, grava um arquivo gerado NOME.HEX no Drive B: ou pula se B: é Z:. Grava o arquivo PRN no Drive C:, manda para o vídeo se B: é P: ou pula se B: é Z:.

DDT <CR>

Carrega o DDT e aguarda comandos DDT D: NOME.EXT<CR>. Carrega o DDT depois o arquivo NOME.EXT (onde EXT deve ser .COM ou .HEX) proveniente do Drive D: e aguarda comando.

Axxxx

Entra no modo Assembler, ou seja, permite digitar instruções em Assembly a partir do endereço xxxx em Hexa.

D

Mostra os próximos 192 Bytes da memória em Hexa.

Dxxxx, zzzz

Mostra do endereço xxxx até zzzz em Hexa.

Fxxxx, zzzz, ww

Preenche do endereço xxxx até zzzz com a constante de 8 Bits ww.

G

Executa o programa a partir do endereço fixado no P.C.

Gxxxx

Executa o programa a partir do endereço xxxx

Gxxxx,zzzz

Executa o programa do endereço xxxx até zzzz

Hx, z

Calcula a soma e a subtração em Hexa dos números x e z.

I NOME.EXT

Coloca no FCB o NOME.EXT

I

Lista as próximas 11 linhas em Assembly.

Lxxxx

Mostra em instruções Assembly as próximas 11 linhas a partir do endereço xxxx.

Lxxxx, zzzz

Mostra em Assembly as linhas compreendidas entre os endereços xxxx até zzzz.

Mxxxx, zzzz, wwww

Move o conteúdo da memória do endereço xxxx até zzzz para o bloco iniciado em wwww.

R

Lê um arquivo em que o nome se encontra no F.C.B (usar antes o comando I). Coloca a partir do endereço 1ØØH.

Rxxxx

Idem ao anterior, apenas coloca-se no endereço especificado xxxx.

Sxxxx

Mostra o conteúdo do endereço xxxx permitindo alterá-lo.

Txxxx

Acompanha a execução de xxxx linhas em Hexa do programa.

Uxxxx

Executa instruções do programa, parando e mostrando o conteúdo dos registradores da CPV.

X

Mostra o conteúdo dos registrados internos da CPV.

XR

Mostra o conteúdo de um determinado registrador e permite a sua alteração.

PIP <CR>

Carrega o PIP na memória e aguarda comandos.

PIP comando <CR>

Carrega o PIP, executa o comando e volta ao CP/M.

Comandos do PIP

D: NOVO.EXT = D: VELHO.EXT (P) <CR>

Copia de um arquivo velho para o novo usando o parâmetro P.

D: NOVO.EXT = D: VELHO.EXT (P) D: VELHO2.EXT (P) <CR>

Cria um arquivo NOVO.EXT no Drive D: que consiste de dois arquivos, sendo um o VELHO1.EXT com parâmetro P e VELHO2.EXT com parâmetro P.

D: NOME.EXT = DISP: (P) <CR>

Copia dados do dispositivo DISP para o arquivo especificado.

DISP: = D: NOME.EXT

Copia dados do arquivo e Drive especificado para o dispositivo DISP.

DISP1: = DISP2 (P) <CR>

Copia dados do dispositivo DISP2 para o DISP1.

Parâmetros do PIP

B - Especifica transferência no modo blocado.
Dn - Elimina todos os caracteres após n colunas.
E - Mostra no vídeo os dados durante execução.
F - Remove os Form.Feeds durante execução.
Gn - Direciona o PIP para cópias de um arquivo a partir da área do usuário n.
H - Verifica se o arquivo tem formato Hexa da INTEL.
I - Ignora qualquer registro ØØ: na transferência de arquivo na forma HEXA-INTEL.
L - Transforma maiúscula em minúscula.

N - Coloca número de linha a cada linha que for transferida.
O - Transferência de código objeto.
Pn - Gera paginação a cada n linhas.
Qs^Z - Especifica saída de cópias após encontrar a seqüência "s".
R - Direciona o PIP para copiar a partir de um arquivo do sistema.
Ss^Z - Especifica início da cópia após o encontro da seqüência "s".
Tn - Gera paradas de tabulação a cada "n" colunas.
U - Transforma minúscula em maiúscula.
V - Verifica se a cópia está perfeita.
W - Direciona o PIP para cópias de um arquivo R/O.
Z - Zera o Bit de paridade nos caracteres ASCII.

Dispositivo Fonte do PIP

```
CON: RDR:
TTY: PTR:
CRT: UR1:
UC1: UR2:
NUL: EOF: INP:

A>
```

ED D: NOME.EXT <CR>

Chama o editor que após instalado na memória procura o arquivo NOME.EXT, não encontrando o ED criará o arquivo, abrindo um arquivo temporário D: NOME.$$$ para guardar o texto editado. O Drive D: é opcional, se omitido, supõe-se o Drive corrente.

Linhas de Comando do ED

nA
Coloca n linhas do arquivo original no Buffer de edição.

+/-B
Move o CP para o início ou fim.

+/-nC
Move o CP por "n" caracteres.

+/-nD

Elimina "n" caracteres antes ou após o CP.

E

Fim de edição. Fecha o arquivo e volta ao CP/M.

nFtexto^Z

Acha o texto. Procura a enésima ocorrência do "texto" iniciando na posição do CP.

H

Move para o início do arquivo editado. Termina edição, rebatiza os arquivos e então o arquivo temporário.

I <CR>

Entra no modo de inserção de dados. Para sair usar o ^Z.

I texto ^Z

Insere o texto no Buffer de edição após o CP.

I texto <CR>

Insere linha e o texto com um CRLF no Buffer de edição após a posição do CP.

+/-nK

Elimina "n" linhas após ou antes do CP.

+/-nL

Move o CP para n linhas à frente ou para trás da linha em que se encontra o CP.

O

Retorna ao arquivo original. Esvazia o Buffer de edição, o temporário e ignora os comandos ED anteriores.

+/-nP

Desloca o CP para frente ou para trás uma página, mostrando a próxima página. "nP/ mostra n páginas com pausa entre elas.

Q

Limpa o Buffer temporário e o arquivo de movimentação de bloco, retornando ao CP/M.

+/-nt

Imprime n linhas para frente ou para trás.

+/-U

Translação para maiúscula. Quando houver o comando +U, as entradas alfabéticas no Buffer são passadas de minúsculas para maiúsculas.

OV

Mostra o espaço livre do Buffer de edição.

+/-V

Após o +V mostra o número de linhas existentes. Após o -V os números de linhas são suprimidos.

R <CR>

Lê o arquivo de movimento de blocos. Cópia do arquivo X$$$$$$.LIB para o Buffer de edição após o CP.

nX <CR>

Transfere "n" linhas do Buffer de edição após o CP para arquivo X$$$$$$.LIB.

nZ

Atraso na edição.

n:

Move o CP para a linha número n.

+/-n

Move e apresenta uma linha.

APÊNDICE B - COMPARAÇÃO ENTRE OS CP/Ms E O CP/M-86

Comando Interno

Comando	versão 1.3	versão 1.4	versão 2.2	CP/M-86
DIR	SIM	SIM	Mostra 04 em cada linha	SIM
ERA	SIM	Pergunta All? Após *.*	Limpa os arquivos do usuário corrente	SIM
SAVE	SIM	SIM	Não altera memória	NÃO
USER TYPE	NÃO SIM	NÃO SIM	SIM SIM	SIM SIM

Diferenças nos Discos

ITEM	CP/M 1.3	CP/M 1.4	CP/M 2.2	CP/M-86
Nº de Drive	2	4	16	16
Capacidade máxima dos Drives	1 MByte	1 MByte	16 MByte	16 MByte
Arquivos	64	64	expandível	expandível
Acesso	seqüencial	seqüencial	seqüencial ou aleatório	seqüencial ou aleatório
Localização das características do disco	BDOS	Bloco de parâmetro do disco	BIOS	BIOS

Comandos de Edição após o A>

Comando	versão 1.3	versão 1.4	versão 2.2	CP/M-86
^C	SIM	SIM	SIM	SIM
^E	NÃO	SIM	SIM	SIM
^H ou BACKSPACE	NÃO	NÃO	SIM	SIM
^J	NÃO	NÃO	SIM	SIM
^M ou CR	SIM	SIM	SIM	SIM
^P	SIM	SIM	SIM	SIM
^R	NÃO	SIM	SIM	SIM
^S	SIM	SIM	SIM	SIM
^U	SIM	SIM	SIM	SIM
^X	NÃO	^U	BACKSPACE	SIM
DELETE ou RUBOUT	SIM	SIM	SIM	SIM

APÊNDICE C - MANUSEIO CORRETO DE UM DISQUETE

Os Disquetes são produtos delicados que requerem alguns cuidados para sua conservação. Alguns itens que devem ser evitados:

NUNCA TOQUE NA SUPERFÍCIE DO DISQUETE:

Segure sempre na capa de vinil, mas nunca na superfície magnética, pois qualquer partícula ou gordura de seus dedos podem danificar o disquete.

NUNCA APROXIME CAMPO MAGNÉTICO:

Como a construção do disco envolve propriedades magnéticas, a aproximação de campo magnético pode acarretar a perda total do Disquete.

FAIXA DE TEMPERATURA DE TRABALHO:

Utilize o Disquete na faixa de temperatura entre 10°C até 55°C. Evite ao máximo transportá-lo para ambientes onde aconteça a brusca mudança de temperatura do tipo 35°C numa sala para 10°C em outra.

EVITE TRABALHAR EM AMBIENTES SUJOS:

Nunca usar o computador em ambientes sujos devido ao fato de que o computador e os Disquetes são peças delicadas. Lugar de comer é no refeitório e não sobre o computador. Sujeira no disquete pode destruir os dados contidos nele.

NÃO DOBRAR O DISQUETE:

Não dobrar o Disquete quando for guardá-lo; na capa protetora, inserir com cuidado, pois neste instante pode ocorrer uma dobra acidental. Proteja sempre seu disquete usando a capa de proteção.

OUTROS CUIDADOS IMPORTANTES:

TENHA SEMPRE CÓPIA DOS DISQUETE:

Apesar do Disquete ser uma fonte de reserva de informações de muita confiabilidade, procure ter sempre reservas

(BACK-UP) porque você pode acidentalmente apagar algum arquivo ou até mesmo destruir o disco. O Back-Up deve ficar muito longe do computador, ou seja, guardá-lo em outro departamento da firma ou em sua casa. Ter mais que um BACK-UP é desejável.

NÃO DEIXE O DISQUETE COMPLETAMENTE CHEIO:

Muitos programas usam áreas do disco para guardar dados temporariamente. Se o Disquete estiver cheio ao usá-lo pode-se perder informações.

ROTULAR OS DISQUETES:

Quando você adquirir uma caixa de Disquetes, receberá junto etiquetas para colar nos mesmos. Procure identificá-los corretamente. Use sempre caneta com ponta porosa para escrever nas etiquetas, quando estas já estão colocadas no disco.

APÊNDICE D - OS CÓDIGOS DE CARACTERES ASCII

O ASCII (American Standard Code for Information Interchange) consiste num código composto de 96 caracteres alfanuméricos e 32 caracteres de controle. Todo texto construído em CP/M, gravado em disco, impresso no vídeo ou papel está construído na codificação ASCII. O ASCII utiliza os sete primeiros Bits do Byte, sendo o oitavo utilizado na detecção de paridade.

Existem outras formas de codificação alfanumérica para binária (Ex.: EBCDIC), mas o ASCII é o mais usado.

NUL - NULL	CR - CARRIAGE RETURN
SOH - START OF HEADING	SO - SHIFT OUT
STX - START OF TEXT	SI - SHIFT IN
ETX - END OF TEXT	DLE - DATA LINK ESCAPE
EDT - END OF TRANSMISSION	DC1 - DEVICE CONTROL 1
ENQ - ENQUIRY	DC2 - DEVICE CONTROL 2
ACK - ACKNOWLEDGE	DC3 - DEVICE CONTROL 3
BEL - BELL OR ALARM	DCY - DEVICE CONTROL 4
BS - BACKSPACE	NAK - NEGATIVE ACKNOWLEDGE
HT - HORIZONTAL TABULATION	SYN - SYNCHRONOUS IDLE
LF - LINE FEED	ETB - END OF TRANSMISSION BLOCK
VT - VERTICAL TABULATION	CAN - CANCEL
FF - FORM FEED	EM - END OF MEDIUM
SUB - SUBSTITUTE	RS - RECORD SEPARATOR
ESC - ESCAPE	US - UNIT SEPARATOR
FS - FILE SEPARATOR	SP - SPACE
GS - GROUP SEPARATOR	DEL - DELETE

TABELA DE EQUIVALÊNCIA HEXADECIMAL E ASCII (AMERICAN STANDARD CODE FOR INFORMATION INTER CHANGE).

HEXA	ASCII	HEXA	ASCII	HEXA	ASCII	HEXA	ASCII
00	NUL	20	SP	40	@	60	\
01	SOH	21	!	41	A	61	a
02	STX	22	"	42	B	62	b
03	ETX	23	#	43	C	63	c
04	EOT	24	$	44	D	64	d
05	ENQ	25	%	45	E	65	e
06	ACK	26	&	46	F	66	f
07	BEL	27	'	47	G	67	g
08	BS	28	(	48	H	68	h
09	HT	29	)	49	I	69	i
0A	LF	2A	*	4A	J	6A	j
0B	VT	2B	+	4B	K	6B	k
0C	FF	2C	,	4C	L	6C	l
0D	CR	2D	-	4D	M	6D	m
0E	SO	2E	.	4E	N	6E	n
0F	SI	2F	/	4F	O	6F	o
10	DLE	30	0	50	P	70	p
11	DC1	31	1	51	Q	71	q
12	DC2	32	2	52	R	72	r
13	DC3	33	3	53	S	73	s
14	DC4	34	4	54	T	74	t
15	NAK	35	5	55	U	75	u
16	SYN	36	6	56	V	76	v
17	ETB	37	7	57	W	77	w
18	CAN	38	8	58	X	78	x
19	EM	39	9	59	Y	79	y
1A	SUB	3A	:	5A	Z	7A	z
1B	ESC	3B	;	5B	[	7B	{
1C	FS	3C	<	5C	`	7C	|
1D	GS	3D	=	5D	]	7D	}
1E	RS	3E	>	5E	^	7E	~
1F	US	3F	?	5F	-	7F	DEL

BIBLIOGRAFIA

LIVROS

CARDINALLI, PAULO ROBERTO E CYPRIANO, LUIZ BENEDITO - Microprocesador Z-80 - Hardware - 3º Edição - São Paulo - "LIVROS ÉRICA EDITORA LTDA" - 1983.

FERNANDES/ASHEY - Usando CP/M, Um Guia em Ensino Programado - 1º Edição - São Paulo - "EDITORA CAMPUS" - 1984.

HOGAN, THOM - CP/M-80 - Guia do Usuário - 1º Edição - São Paulo "MC GRAW HILL".

WAITE, MITCHELL AND LAFORE, ROBERT - Soul Of CP/M - 1º Edição - Indianápolis - "EDITORA HOWARD W. SAMS & CO." - U.S.A. - 1983

WAITE, MITCHELL - CP/M BIBLE - The Authonitatine Reference Guide To CP/M - Indianápolis - "EDITORA HOWARD W. SAMS & CO." - U.S.A. 1983

MANUAIS

"CP/M OPERATING SYSTEM" - Manual Digital Research - P. O BOX 579 PACIFIC GROVE - Califórnia - 93950 U.S.A. - 1982.

"O MANUAL DE CP/M INCLUINDO MP/M" - Rodray Zaks - 1º Edição - São Paulo - EDITORA CAMPUS - 1984.

Impressão e acabamento
(com filmes fornecidos):
EDITORA SANTUÁRIO
Fone (0125) 36-2140
APARECIDA - SP

PUBLICAÇÕES EM ELETRÔNICA

Elementos de Eletrônica Digital

Iniciação a Eletrônica Digital, Álgebra de Boole, Minimização de Funções Booleanas, Circuitos Contadores, Decodificadores, Multiplex, Demultiplex, Display, Registradores de Deslocamento, Desenvolvimento de Circuitos Lógicos, Circuitos Somadores/Subtratores e outros. 10.ª Edição, 512 páginas.
Autores: Capuano / Idoeta.

Teoria e Processo de Desenvolvimento em Eletrônica

Estudo e Associação de Bipolos Passivos e Ativos, Circuitos Ressonantes, Aparelhos de Medidas, Válvulas, Semicondutores, Leis de Kirchhoff, Teoremas de Thevenin e Norton, Osciloscópio e outros. 2.ª Edição, 260 páginas.
Autor: **Eng.º Sidnei David**

Microprocessadores 8080 e 8085 - Hardware - Vol. 1

Memórias RAM, ROM, PROM e EPROM, o 8224, 8228, 8080, 8085, 8255 e 8253, suas aplicações e montagem de um microprocessador. 5.ª Edição, 144 páginas.
Autor: **Eng.º Antonio Carlos José Franceschini Visconti**

Microprocessadores 8080 e 8085 - Software - Vol. 2

Estudo das instruções dos microprocessadores 8080 e 8085, Fluxogramas, iniciação a programação e desenvolvimento de programas com a utilização dos microprocessadores 8080 e 8085. 6.ª Edição, 208 páginas.
Autor: **Eng.º Antonio Carlos José Franceschini Visconti**

Amplificador Operacional

Ideal e Real, em componentes discretos, Realimentação, Compensação, Buffer, Somadores, Detetor de Picos, Integrador, Gerador de Sinais, Amplificadores de Audio, Modulador, Sample-Hold, etc.

Possui cálculos e projetos de circuitos e salienta cuidados especiais.
3.ª Edição, 272 páginas.
Autores: **Eng.º Roberto Antonio Lando/Eng.º Serg Rios Alves**

Basic para Computadores Pessoais

Apresentação da Linguagem, com desenvolvimento detalhado das instruções, com uma série de exercícios resolvidos e propostos (com respostas), jogos e programas aplicativos. Linguagem simples e de aceitação pelos aficcionados no ramo dos microcomputadores.
3.ª Edição, 270 páginas.
Autor: **Arsonval Fleury Pereira**

Microprocessador Z-80 - Hardware - Vol. 1

Estudo dos Algoritmos, Arquitetura, Estrutura e Ciclo de Tempo do Microprocessador Z-80, CTC (contador), PIO (porto), Memórias 4801, 4802, 2732 Circuito de Clock, Reset, Teclado, Display e outros circuitos. 3.ª Edição, 186 páginas.
Autores: **Eng.º Luiz Benedito Cypriano**
Eng.º Paulo Roberto Cardinalli

Microprocessador Z-80 Software Vol. 2

Pesquisa do SET de Instruções do microprocessador Z-80. Tipos de Endereçamento. Tipo de Instrução. Fluxo de Dados. Interrupção. Linguagem de Máquina e Assembly. Pseudo-Instrução. Desenvolvimento de Programas.

Este livro também se destina à aplicação de micros pessoais que operam em linguagem de máquina. 3.ª Edição, 334 páginas.
Autor: **Eng.º Luiz Benedito Cypriano.**