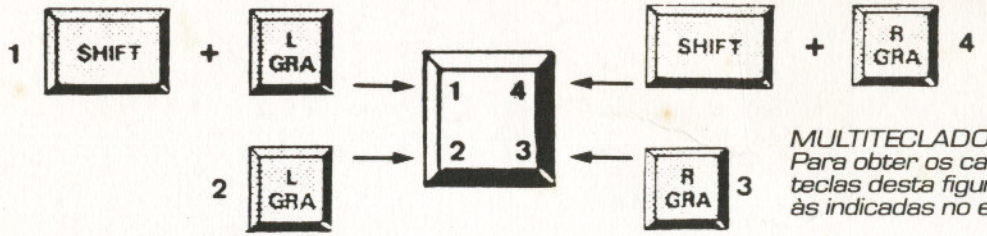
gradiente

EXPERT DD Plus

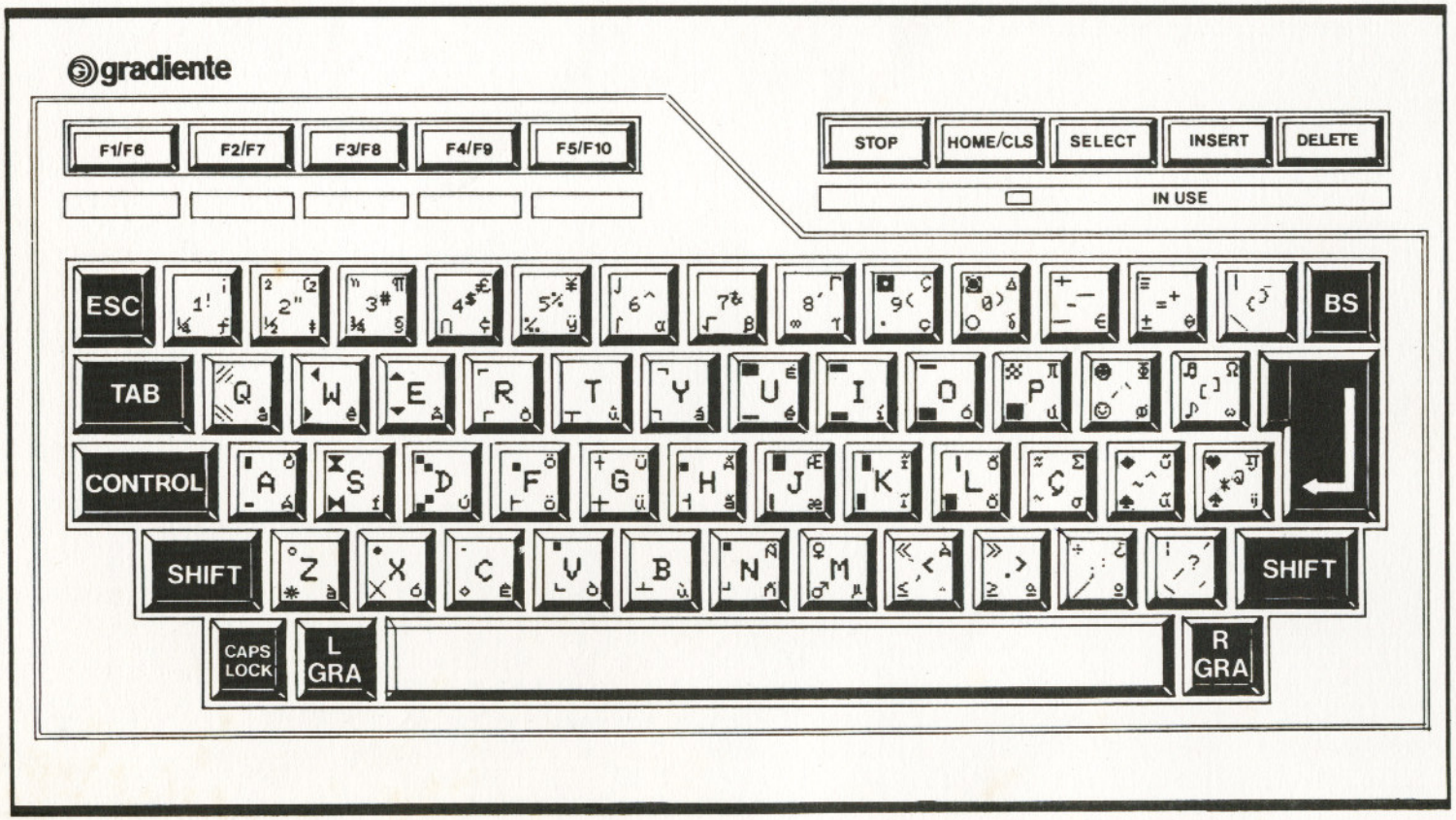
MANUAL DE INSTRUÇÕES

gradiente

MANUAL DE INSTRUÇÕES EXPERT DD Plus



MULTITECLADO DO EXPERT DD PLUS
Para obter os caracteres especiais desenhados nos 4 cantos das teclas desta figura, pressione a tecla desejada simultaneamente às indicadas no esquema ao lado.



Na figura abaixo temos o "mapa" do teclado do Expert DD Plus para auxiliar o leitor na digitação dos caracteres especiais.

EXPERT DD PLUS

Manual de Instruções

gradiente

06-3102-001-000

901-03210-000

EXPEDIENTE:

Coordenação Editorial: Pierluigi Piazzi
Coordenação Didática: Betty Fromer Piazzi
Redação didática: Pierluigi Piazzi
Renato da Silva Oliveira
Luiz Tarcísio de Carvalho Júnior
Matias August Gruber
Editoração: Henrique de Figueredo Luz
Arte: Ana Lúcia Antico
ilustrações: Nicoletti

ALEPH Publicações e Assessoria Pedagógica Ltda.
Av.Dr.Luis Migliano 1110 c/301
CEP 05750 São Paulo–SP
Tel (011) 843-3202
Caixa Postal 20.707 – CEP 01498

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

Piazzi, Pierluigi, 1943-
Expert DD PLUS : manual de instruções / Pierluigi Piazzi. -- São Paulo : Aleph, 1989.

Bibliografia.

1. Expert DD PLUS (Computador) 2. Expert DD PLUS (Computador) - Programação 3. MSX (Computadores) - Programação I. Título.

CDD-001.64
89-0951 -001.642

Índices para catálogo sistemático:

1. Expert DD PLUS : Computadores : Programação : Processamento de dados 001.642
2. Expert DD PLUS : Computadores eletrônicos : Processamento de dados 001.64
3. MSX : Computadores : Programação : Processamento de dados 001.642

SUMÁRIO

NÃO DEIXE DE LER ESTE TEXTO

O Expert DD PLUS é um produto de sofisticada tecnologia, fabricado segundo as normas de qualidade GRADIENTE, obedecendo às especificações do padrão MSX.

Para operá-lo é dispensável que seu usuário conheça programação ou informática.

Existem, porém, informações extremamente úteis contidas neste livro, cujo objetivo é permitir ao usuário do Expert DD PLUS tirar o máximo proveito de seu equipamento.

Veja, a seguir, uma breve descrição deste livro para que você possa trilhar o caminho mais rápido para atender às suas necessidades.

É muito comum, por exemplo, se ver um usuário iniciante gastar horas e horas para aprender como programar seu micro e, ao cabo de muito esforço, perceber que existem no mercado programas prontos que fazem exatamente aquilo que ele queria!

A situação simétrica também pode ocorrer: é a do usuário que quer usar seu micro numa tarefa muito específica e não encontra programas prontos que o atendam. Esse usuário, então, ou utiliza serviços de terceiros, ou aprende a programar para desenvolver seu próprio "software".

Vejamos, então, onde buscar as informações de que você precisa.

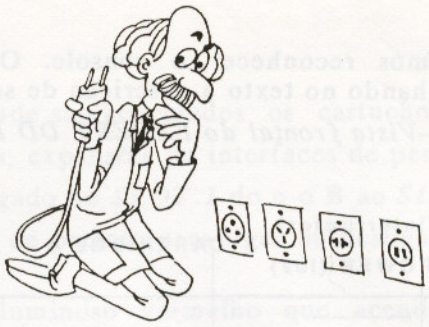
A ORGANIZAÇÃO DESTE LIVRO

Este livro de instruções, que acompanha o seu Expert DD PLUS, aborda as principais informações sobre o universo virtualmente ilimitado de opções que ele oferece.

Nossa intenção é a de fornecer um material introdutório e uma orientação que possa levá-lo a utilizar com o máximo proveito o seu Expert DD PLUS.

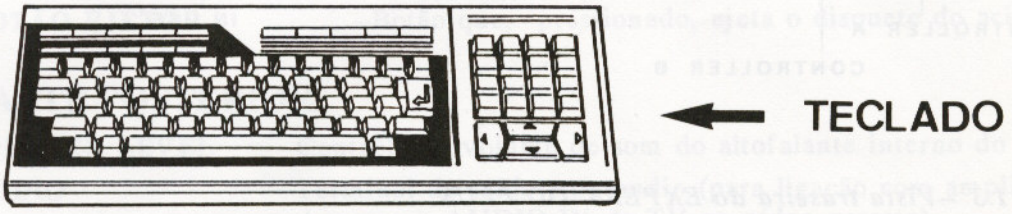
Todas as sugestões no intuito de aprimorar esta obra serão bem-vindas e desde já agradecemos antecipadamente por elas.

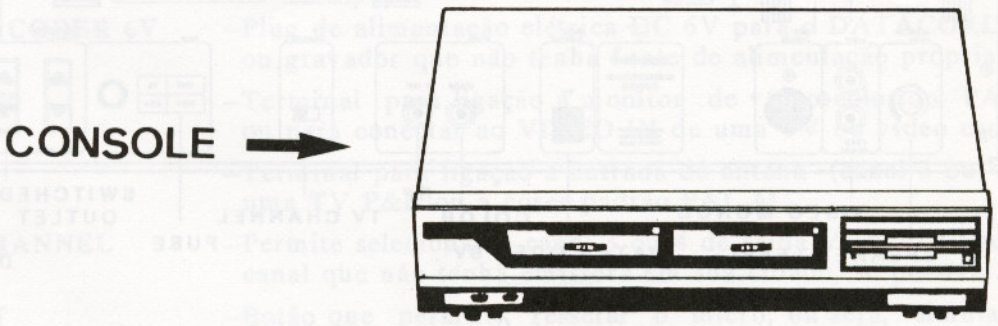
COMO INSTALAR SEU EXPERT DD PLUS 1



Antes de instalar o micro você deve conhecer cada componente e cada uma de suas partes. Vamos identificar a parte principal do seu sistema **EXPERT DD PLUS**: o *teclado* e o *console*.

fig 1.1 –O teclado e o console do EXPERT DD PLUS





O teclado é o principal meio para que você dê ordens ao computador. Ele tem ligado a si um longo cabo que deverá ser conectado ao console. No console é que estão o "*coração*" e o "*cérebro*" do **EXPERT DD PLUS** e, se isso lhe ajudar, você pode considerar o teclado como sendo seus "*ouvidos*" ou seus "*olhos*". Levado essa analogia adiante, o vídeo (T.V. ou MONITOR) seria sua "*boca*".

Vamos reconhecer o console. Observe atentamente as figuras 1.2 e 1.3, acompanhando no texto a descrição de seus contrôles e encaixes.

fig 1.2 –Vista frontal do EXPERT DD PLUS

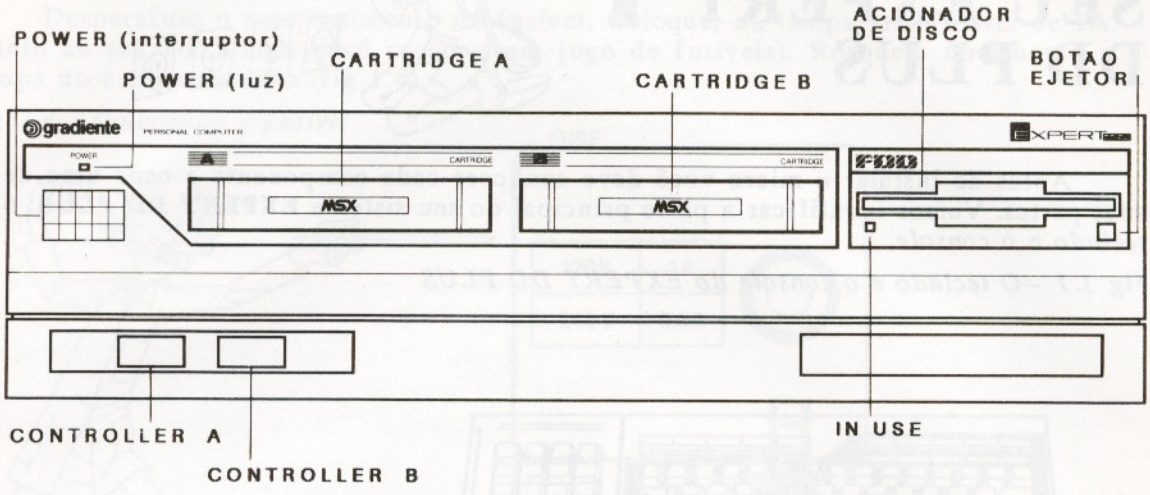
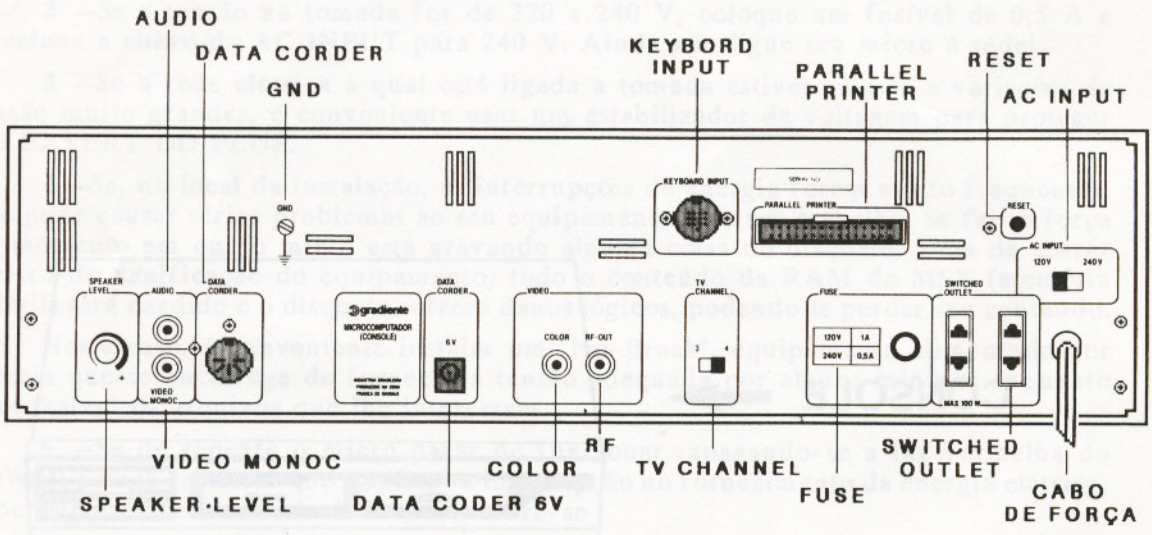


fig 1.3 –Vista traseira do EXPERT DD PLUS



PARTE FRONTAL

CARTRIDGE –Encaixes onde são conectados os cartuchos (memória pré gravada, expansões ou interfaces de periféricos). O A está ligado no *SLOT 1* do e o B ao *SLOT 2*..

POWER (interruptor) –Interruptor de pressão para ligar ou desligar o aparelho.

POWER (lâmpada) –Indicador luminoso vermelho que acende quando o EXPERT DD PLUS está ligado.

CONTROLLER –Encaixe de *JOYSTICKS* (alavancas de contrôle manual) A e B, ou *"mouse"*.

ACIONADOR DE DISCO –Acionador de disco de 3 1/2", face dupla com interface interna.

IN USE –Indicador luminoso que indica que o drive esta sendo acessado pelo computador (leitura ou escrita).

BOTÃO EJETOR –Botão que, pressionado, ejeta o disquete do acionador.

PARTE POSTERIOR

SPEAKER LEVEL –Contrôle de volume de som do altofalante interno do console.

AUDIO –Terminal de saída para audio (para ligação com amplificador de som ou AUDIO IN de TV ou vídeo-cassete).

VIDEO MONOC. –Terminal para ligação a monitor monocromático.

DATA CORDER –Terminal para ligação dos cabos que vão ao gravador cassete.

GND –Encaixe para ligação de fio-terra.

KEYBORD INPUT –Encaixe para conexão do cabo do teclado.

PARALLEL PRINTER –Encaixe para conexão de uma impressora paralela padrão *CENTRONICS*.

DATA CODER 6V –Plug de alimentação elétrica DC 6V para o DATACORDER ou gravador que não tenha fonte de alimentação própria.

COLOR –Terminal para ligação a monitor de vídeo colorido PAL-M ou para conectar ao VIDEO IN de uma TV ou vídeo cassete.

RF –Terminal para ligação à entrada de antena (canal 3 ou 4) de uma TV P&B ou a cores padrão PAL-M.

TV CHANNEL –Permite selecionar o canal 3 ou 4 de saída VHF. Utilize um canal que não tenha emissora em sua cidade, se possível.

RESET –Botão que permite "ressetar" o micro, ou seja, simular um desligamento e religamento sem que seja necessário pressionar POWER.

FUSE –Encaixe para fusível (veja texto a seguir)

SWITCHED OUTLET –Saída para ligação de cabo de força de outros equipamentos de baixa corrente. Ao desligar o micro a força nessas tomadas será automaticamente cortada. O consumo dos equipamentos pode ser, no máximo, de 100 W.

AC INPUT –Seletor de voltagem para rêde elétrica (VERIFIQUE A VOLTAGEM LOCAL!).

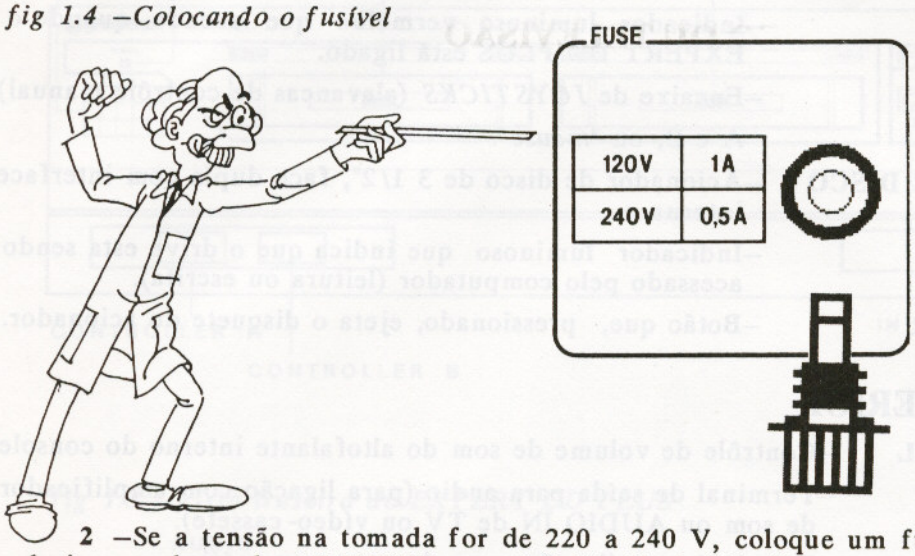
CABO DE FORÇA –Cabo para ligação à rede elétrica (NÃO LIGUE AINDA!)

ESCOLHENDO A TENSÃO DA REDE ELÉTRICA

1 –Verifique a tensão da tomada na qual você vai conectar seu micro. Se ela for de 110 a 127 V, coloque a chave de seleção do **AC INPUT** (veja fig 1.3) na posição 120 V. Ainda não ligue o micro na tomada!

Desparafuse o compartimento do fusível. Coloque, na tampa, um fusível de 1A (junto ao seu DD PLUS você recebeu um jogo de fusíveis). Rosqueie novamente a tampa no compartimento (fig 1.4).

fig 1.4 –Colocando o fusível



2 –Se a tensão na tomada for de 220 a 240 V, coloque um fusível de 0,5 A e selecione a chave do **AC INPUT** para 240 V. Ainda não ligue seu micro à rede!

3 –Se a rede elétrica à qual está ligada a tomada estiver sujeita a variações de tensão muito grandes, é conveniente usar um estabilizador de voltagem para proteger seu EXPERT DD PLUS.

4 –Se, no local da instalação, as interrupções de energia forem muito frequentes, isso pode causar sérios problemas ao seu equipamento e ao seu trabalho. Se faltar força no momento em que o micro está gravando alguma coisa no disquete, além de correr o risco de danificação do equipamento, todo o conteúdo da RAM do **MSX** (memória volátil) será perdido e o disquete sofrerá danos lógicos, podendo se perder seu conteúdo.

Neste caso, é conveniente instalar um "No-Break", equipamento alimentado por bateria que se encarrega de fornecer a tensão adequada por alguns minutos enquanto você "salva" os arquivos que lhe interessam.

5 –Se de repente o micro parar de funcionar, apagando-se a luz vermelha do POWER e você verificar que não houve interrupção no fornecimento da energia elétrica, experimente trocar o fusível. Porém lembre-se

120 V – fusível 1 A

240 V – fusível 0.5 A

Se o fusível queimar novamente, não insista ou, pior ainda, não tente "*ligações diretas*" tipo papel alumínio.

Se o fusível queima, que é sua função, significa que existe alguma anomalia interna, e antes que ocorram consequências graves (queima de componentes ou até princípios de incêndios), rompe-se o filamento do fusível, bloqueando assim a corrente elétrica. Ou seja, seu micro-computador está com um problema interno, e deve obrigatoriamente ser enviado para uma assistência técnica autorizada (anexo ao microcomputador, vem uma lista das assistências técnicas autorizadas Gradiente).

CONECTANDO O TECLADO

1 –Com o micro ainda desligado, instale o console sobre a mesa onde ela vai ficar e ligue o cabo que sai do teclado no conector **KEYBOARD INPUT** que está na parte traseira do console (fig. 1.3). Na cinta metálica do plug deste cabo existe uma marca em baixo-relevo. Ela deve ficar na parte mais alta da conexão.

Não conecte ou desconecte o teclado com o micro ligado.

CONECTANDO O VÍDEO OU TELEVISÃO

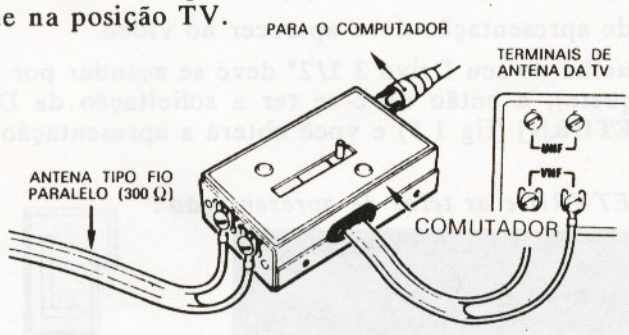
O periférico mais importante no micro-computador é o vídeo, pois é através deste que o usuário passa a receber as informações geradas pelo micro, inclusive, para conferir as mensagens digitadas pelo teclado.

Você pode ligar o seu EXPERT DD PLUS à uma TV P&B ou a cores (padrão PAL–M), a um monitor de vídeo monocromático, ou a um monitor de vídeo colorido.

1) A TV P&B ou a colorida devem ser conectados à saída **RF-OUT**, através do cabo "*RCA*" e o "comutador", fornecidos junto com o micro-computador.

Para isso desconecte o cabo da antena da entrada de VHF da TV, seguindo o manual de instruções de seu televisor. No lugar do cabo de antena coloque os terminais do comutador; conecte o cabo RCA do micro na entrada "COMPUTER" do "comutador", e por fim, conecte os terminais da antena na entrada "TV" do "comutador".

Para que se use o micro, lique a chave do "comutador" na posição COMPUTER, caso contrário, ligue na posição TV.



2) O monitor monocromático apresenta um qualidade de imagem muito superior à das televisões (mesmo sendo monocromático), possuindo um imagem mais estável e definida.

Para se usar o monitor monocromático em seu EXPERT DD PLUS, conecte o cabo tipo RCA, na entrada VÍDEO IN do monitor, e a outra extremidade na saida "VIDEO MONOC." do console de seu micro.

3) O monitor colorido apresenta a melhor imagem colorida do micro. Para usá-lo deve-se conectar o cabo tipo RCA, na entrada VIDEO IN do monitor colorido, e a outra extremidade na saída "COLOR" do console do micro.

A saída COLOR do micro, também pode ser usada conectada na entrada "VIDEO IN" do seu vídeo-cassete, ou Camcorder para gravação das imagens geradas pelo EXPERT DD PLUS.

Na parte traseira do console, o micro possue 2 tomadas de rede (SWITCHED OUTLET), independentes do fusível do micro, mas ligadas à chave POWER. Essas tomadas podem ser usadas para ligar a sua TV, ou monitor e eventuais periféricos, de maneira que pode-se, ao acionar uma única chave (POWER), ligar todo o sistema.

Observe que cada uma dessas tomadas suporta até 100 W de potência, e se colocarmos aparelhos de maior consumo (algumas televisões coloridas), pode-se comprometer a chave do POWER do micro.

CONECTANDO O ÁUDIO

O seu EXPERT DD PLUS possui um alto-falante interno que, para aplicações normais, é o suficiente. Para usuários que estejam interessados em qualidade de áudio, pode-se usar o alto-falante da TV, ou então ligar o áudio a um amplificador (pode ser um VIDEO CASSETE ou CAMCORDER), através de um cabo do tipo RCA, conectando em uma extremidade na entrada "**AUDIO IN**" do amplificador, e a outra extremidade na saída "**AUDIO**" do Expert.

LIGANDO O EXPERT DD PLUS

Depois de todas as instalações mostradas e conferidas nos ítens anteriores pode-se, então, ligar o Expert.

A seguir damos os passos finais que devem ser executados pelo usuário. Caso não se verifique o acionamento do computador confira novamente todos os ítens desse capítulo.

Proceda da seguinte forma:

1) Coloque o "plug" do seu micro na tomada, verificando se a tensão da rede é correta.

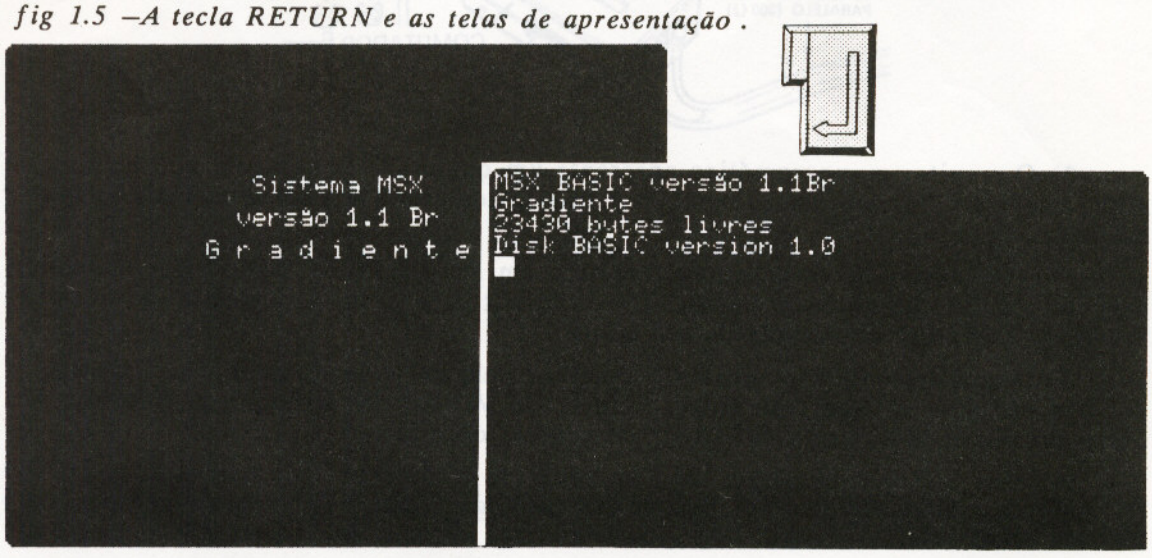
2) Pressione a tecla POWER, e verifique se a luz de POWER também está acesa.

3) Verifique, no teclado, se a luz "IN USE" está acesa. Se não estiver, proceda novamente a instalação do teclado, mas faça isso somente com o aparelho desligado.

4) Uma tela de apresentação deve aparecer no vídeo.

5) A luz vermelha do seu Drive 3 1/2" deve se acender por um momento (ainda não coloque o disquete), e então deve-se ter a solicitação da DATA ATUAL. Por enquanto digite **[RETURN]** (fig 1.5) e você obterá a apresentação do BASIC de disco (DISK-BASIC).

fig 1.5 –A tecla RETURN e as telas de apresentação .



Se todas os ítens forem confirmados, a instalação do seu micro teve sucesso, senão, releia este capítulo, até obter o resultado.

Agora que está tudo bem, desligue o micro, e caso você não tenha noção nenhuma sobre computadores, leia o capítulo 2. Se você não conhece nada sobre a linguagem BASIC, leia o capítulo 3. Se você conhece o BASIC, mas não sabe o que é o MSXDOS e o DISK-BASIC do MSX, leia o capítulo 4 e 5. Se você é um "Expert" no assunto, bastam apenas os dicionários do BASIC MSX e do MSXDOS dos capítulos 6 e 7.

Boa Sorte e Bom Trabalho!

O QUE O EXPERT DD PLUS PODE FAZER POR VOCÊ 2

OS DOIS CAMINHOS DO USUÁRIO

Pronto! Agora seu micro está instalado e pronto para funcionar! Um universo virtualmente infinito de maravilhas se abre à sua frente, porém você precisa de algumas informações adicionais para penetrar neste novo mundo.

Você está um pouco na situação de alguém que comprou um excelente arco mas ainda está sem flechas! Existem, neste caso, duas possibilidades: Você pode comprar flechas prontas ou aprender a fabricá-las.

Se você decidir por fabricá-las, precisa saber quais as técnicas adequadas, que materiais usar e quanto tempo vai gastar nesta tarefa.

Se você for comprar flechas prontas, precisa conhecer qual o melhor modelo para cada finalidade, quem as fabrica e para que aplicação elas foram dimensionadas.

Da mesma forma, para operar seu micro, você precisa de uma "munição" chamada "Software" (programa de computador).

Você pode decidir se tornar um "programador", ou seja, "construir" seu próprio software, ou ser um "usuário de sofware pronto", utilizando programas elaborados por programadores experientes.

Cada um destes caminhos apresenta suas vantagens e desvantagens que enumeraremos na orientação que vem a seguir.

Vejamos qual dessas alternativas você deve seguir.

O PERFIL DO USUÁRIO DE SOFTWARE

Ao optar por este caminho, o fator que mais pesa é o tempo. Elaborar um software por conta própria exige, além de uma certa predisposição natural, uma quantidade de tempo muito grande, não só para incorporar os conhecimentos e as técnicas necessárias, mas também para o próprio desenvolvimento do programa.

Além disso, ao cabo de muito esforço, você pode descobrir que acabou de reinventar a roda! Você pode se sentir na situação de um alpinista que chega ao cume de uma montanha após árdua escalada e se defronta com uma espantada família fazendo pique-nique dominical!

Existem muitos programas prontos que podem desempenhar tarefas interessantes sem necessidade de se perder tempo em elaborá-los!

Vejamos, a seguir, quais os principais tipos de programas que podem "municiar" seu Expert DD PLUS.

TIPOS DE PROGRAMAS

Dependendo da finalidade a que se destinam, os programas são classificados em:

PROFISSIONAIS

Classificamos como profissionais os programas que permitem usar o micro como elemento de automação de atividades empresariais.

São inúmeros mas, dentre eles, podemos destacar algumas grandes categorias:

1) EDITORES DE TEXTO

Fazem o micro substituir, com imensas vantagens, a mais sofisticada máquina de escrever eletrônica.

Permitem a digitação de textos e sua gravação em fita ou disco para uso posterior.

Só para se ter uma idéia, num disquete de seu Expert DD PLUS você pode armazenar algo em tôrno de 100.000 palavras!

Trabalhando na tela do micro, o texto pode ser alterado, trechos podem ser eliminados, outros acrescentados, de uma maneira extremamente simples e rápida.

Imagine, por exemplo, a seguinte situação (muito frequênte na prática): num escritório estão sendo discutidos os termos de um contrato já digitado e as partes interessadas resolvem alterar uma cláusula.

Pelos processos tradicionais o contrato original deveria ser integralmente redatilografado (ou pelo menos a folha em que está a cláusula alterada).

Com um processador de texto basta alterar, na tela do Expert, o parágrafo em questão e, em questão de minutos, tirar outra cópia passada a limpo na impressora.

Imagine o que isto representa em termos de rapidez e eficiência!

Os processadores de texto tem um recurso especial: Busca e Substituição.

Digamos, por exemplo, que sua secretária tenha digitado um texto escrevendo casa com "z" (caza!). Ao perceber o erro, basta ordenar a procura de toda "caza" e sua substituição por "casa" e pronto! O texto ficará integralmente corrigido em segundos!

Além disso, pelo editor, você pode comandar a ativação de recursos especiais da impressora, como letras sublinhadas, negrito, fontes alternativas, etc.

Este livro, por exemplo, foi integralmente digitado no "Editor de Textos Gradiente", processador de texto para MSX.

Quando uma primeira versão do texto é tirada na impressora, ela passa por um processo de revisão. Os erros são corrigidos no texto gravado no Expert e a versão final é rapidamente obtida sem que haja necessidade de se redatilografar todo o texto.

Após cada revisão, os erros vão diminuindo. Antigamente o processo editorial exigia uma nova datilografia do texto corrigido: erros antigos eram eliminados mas novos erros eram introduzidos!

E tem mais: os disquetes contendo os textos digitados no seu Expert, podem ser lidos num IBM-PC, ou compatível, para serem processados, a nível de composição (Desk Top Publishing), de maneira a obter uma arte final.

Quem trabalha com editor de texto, normalmente se pergunta" como é que eu conseguia produzir textos, antes do advento do micro ?"

2) BANCOS DE DADOS

São programas que produzem um "fichário eletrônico inteligente".

Existem programas que fornecem as fichas pré-desenhadas de maneira a que o usuário tenha apenas o trabalho de preenchê-las.

Outros, permitem ao usuário desenhar as suas próprias fichas e determinar o formato dos

relatórios impressos ou etiquetas de endereçamento. Outros ainda, como o DBASE, são verdadeiras linguagens dedicadas ao gerenciamento do banco de dados.

Em um banco de dados gerenciado pelo micro é possível realizar de maneira rápida e eficiente, tarefas que, num fichário tradicional, demandariam muito tempo.

Você pode, por exemplo, pedir a emissão das etiquetas de endereçamento de todos os clientes que façam aniversário na próxima semana. Ou então, levantar quantos fornecedores você tem no estado do Acre, ou ainda descobrir quantos ítens, no seu estoque, estão abaixo de um valor crítico, para desencadear um processo de compra.

Num único disquete do seu Expert DD PLUS você pode ter mais de 3.000 fichas.

Só para exemplificar a grande utilidade e versatilidade de um banco de dados no microcomputador, imagine os fichários de uma biblioteca.

Numa biblioteca não informatizada, cada livro novo exige a datilografia de pelo menos 3 fichas: uma para o arquivo alfabético dos AUTORES, outra para o de TÍTULOS e uma terceira para o de ASSUNTOS.

Usando um banco de dados, digitamos uma única ficha e pedimos ao próprio software para para criar 3 índices em ordem alfabética: AUTOR, TÍTULO e ASSUNTO.

Dependendo do tipo de consulta, seu Expert DD PLUS consultará o índice correspondente para saber em qual das 3 ordens estabelecidas o arquivo deve ser lido!

Tente imaginar estas tarefas sendo desempenhadas por um funcionário, consultando um fichário comum!

3) PLANILHAS ELETRÔNICAS

As planilhas têm a estrutura de uma gigantesca folha de papel dividida em colunas e linhas, com a tela do micro fazendo o papel de uma "janela" que pode correr horizontalmente ou verticalmente sobre esta folha, localizando a área de seu interesse.

Em cada "célula" da planilha (definida pela linha e coluna a que pertence) você pode colocar um texto, um número ou uma fórmula.

Digamos, por exemplo, que você tenha elaborado uma planilha de custos de um produto na qual alguns componentes têm seu preço em dólares.

Se a cotação do dólar mudar, não é preciso calcular de novo o valor destes ítens.

Basta colocar a nova cotação num lugar especial que você reservou para este fim e pedir à planilha para recalcular tudo. Num piscar de olhos você tem os novos valores (em moeda brasileira) atualizados!

Com estes recursos você pode, por exemplo, fazer simulações financeiras, prevendo o que vai acontecer com seus orçamentos se a taxa de inflação mudar, descobrindo dentro de que limites um projeto continua economicamente viável.

Da mesma forma, se você usa sua planilha para fazer uma folha de pagamento e houver alguma alteração na legislação fiscal ou trabalhista, em pouquíssimo tempo ela pode ser reconfigurada, permitindo uma atualização imediata.

Imagine, por exemplo, que você tenha uma confecção de roupas e receba um pedido. Se tiver uma planilha de produção preparada em seu Expert, onde está especificado, para cada modêlo e tamanho, a quantidade de tecido e o tempo de produção, sua tarefa será extremamente simplificada!

Basta introduzir os dados do pedido na planilha e, em questão de segundos, você terá a relação de matéria prima e o tempo necessário para atender o pedido. Pense no tempo e na dor de cabeça que você teria para fazer tudo isso na ponta do lápis!

Como você pode notar, as planilhas eletrônicas são instrumentos indispensáveis para organização de uma empresa e para tomadas de decisões.

E nem só em atividades empresariais as planilhas eletrônicas são úteis. Com todos os recursos que citamos, você ja deve ter deduzido como fica fácil organizar um orçamento doméstico ou até um receituário.

Imagine, por exemplo, que em seu receituário, um dado prato esteja dimensionado para 4 porções e você vai dar um jantar para 6 pessoas. Numa planilha eletrônica, todos os ingredientes serão recalculados num piscar de olhos!

4) PROGRAMAS DE COMUNICAÇÃO

São programas que permitem fazer seu Expert DD PLUS "conversar" com outros computadores, não importando sua marca ou tipo. Com seu Expert ligado a um MODEM, você pode acessar, através de uma linha telefônica comum, serviços como VIDEOTEXTO, CIRANDAO, RENPAC, etc, nos quais você pode consultar os bancos de dados de outros computadores e obter as mais variadas informações.

Você pode obter, em segundos, as cotações da Bolsa, seu saldo bancário, a posição de suas aplicações no Open, a cotação do Dólar no oficial e no paralelo, etc.

Depois de tanto trabalho, você vai querer descansar um pouco: basta acessar, um serviço especial e você terá a crítica dos filmes que estão passando hoje!

5) APLICATIVOS

São programas profissionais dedicados a uma atividade particular.

Dentro desta classificação você pode encontrar "softwares" de controle de estoque, matemática financeira, estatística, agendas, editores gráficos, musicais, etc.

Já existem inúmeras versões destes programas no mercado e muitas em fase de desenvolvimento. Se você necessita de um aplicativo específico, pode optar por encomendar um programa especial a alguma "Software-house".

Se, em sua categoria profissional, existem outras empresas com a mesma necessidade, é usual a formação de um "POOL" de maneira a diluir os custos de desenvolvimento entre vários usuários.

Se você quer ter uma noção mais específica dos vários tipos de programas que existem para MSX, aconselhamos a leitura do "Guia do MSX" onde você achará, inclusive, os endereços e telefones dos produtores e distribuidores.

LAZER

Nesta área, além de uma quantidade enorme de jogos tipo vídeo-game (alguns dos quais sofisticadíssimos!), temos programas que não se destinam à recreação pura e simples, trazendo algum tipo de crescimento nas habilidades do usuário.

Nesta faixa existem vários jogos de inteligência, simuladores de pilotagem de aviões ou automóveis, xadrez e outros jogos de raciocínio.Existem também os chamados "adventures" nos quais o jogador, além de defrontar com situações de ação (tipo vídeo-game), deve responder perguntas, tomar decisões e elaborar um planejamento estratégico para prosseguir o jogo.

Note que muitos usuários consideram um "hobby" extremamente relaxante e divertido programar e criar seus próprios jogos!

EDUCACIONAIS

O microcomputador revelou ser poderosíssimo instrumento na área educacional. Com seus recursos de som e imagem, o MSX é insuperável nesta área.

Existem basicamente 3 tipos de programas educacionais:

1) DE MEMORIZAÇÃO

Nestes programas, o micro repete, com paciência infinita, testes de correlação do tipo *"elementos químicos - seus símbolos", "países - capitais",* taboada, acentuação da língua portuguesa, vocabulário de línguas estrangeiras, etc.

O micro se transforma num professor particular com uma paciência sobre-humana e disponível a qualquer hora.

2) DE SIMULAÇÃO

No microcomputador podemos simular experimentos quase impossíveis de serem realizados em laboratório ou que demandariam recursos absurdos em tempo e dinheiro.

3) DE RACIOCÍNIO

No ensino tradicional, dá-se pouca ênfase ao desenvolvimento do raciocínio, ficando esta tarefa normalmente relegada como sub-produto do ensino da Matemática.

O micro-computador e o MSX em particular, chegaram para revolucionar este conceito, sendo usados para desenvolver as habilidades de raciocínio das crianças, de maneira a produzir jovens mais inteligentes.

O próprio ato de programar o micro é um estímulo muito grande para o desenvolvimento intelectual da criatividade.

Neste contexto a linguagem "LOGO" desenvolvida no M.I.T. por Pappert com base nos estudos de Piaget, tem um papel relevante.

Note que existe uma região limítrofe entre os programas de lazer e os educacionais pois existem "softwares" que, disfarçados de "jogos", acabam gerando um efeito educacional importante.

Vejamos agora, sob que forma todos estes programas podem ser "transportados" para seu Expert DD PLUS:

OS VEÍCULOS DO SOFTWARE

Estes programas são comercializados e divulgados basicamente de 5 formas:

CARTUCHOS

São programas gravados no circuito eletrônico de um cartucho que será conectado num dos slots do micro (no painel frontal do console do seu Expert DD PLUS você tem dois slots: o "cartridge" A e o B). Os cartuchos devem sempre ser colocados ou retirados com o micro desligado.

Alguns, ao serem ligados, começam a executar o programa "tomando conta" do micro, não permitindo a execução de outra tarefa.

Outros, ficam "dormentes" enquanto o micro pode executar outra tarefa e só são ativados quando "chamados" (normalmente através de um comando denominado CALL...).

FITA CASSETE

São programas gravados em fita e precisam ser carregados no micro através de um gravador ou "datacorder". Como o micro deve "ler" os programas sequencialmente, o tempo de carga é muito mais demorado do que no caso dos cartuchos. Costumam apresentar algum problema de carga quando o gravador não está devidamente ajustado ao seu micro.

Como as formas de gravação podem ser variadas, os comandos para ordenar o carregamento podem ser de vários tipos, estando especificados no manual do fornecedor.

DISQUETES

Esta é a forma mais racional e versátil para se carregar um programa no micro. Ele vem gravado num disco que será lido no acionador.

Alguns são carregados automaticamente, não havendo necessidades de comandos especiais.

Outros são carregados a partir de instruções especiais que devem estar especificadas no manual do programa.

A carga é normalmente mais lenta do que nos cartuchos mas é considerávelmente mais rápida que no caso da fita cassete (e muito mais confiável!).

É o sistema que fornece a melhor relação custo/benefício.

LISTAGENS

Quando o programa não e muito extenso, ele pode ser divulgado, em livros e revistas, na forma de listagens. Usando o teclado do micro, o leitor poderá digitá-lo de maneira a carregá-lo na memória do computador.

Posteriormente ele poderá gravar estes programas em fita ou disco.

Apesar de ser um procedimento trabalhoso e demorado tem a vantagem de induzir o aprendizado das técnicas de programação. Neste livro que você está lendo existem exemplos destas listagens nos capítulos subsequentes.

TELEFONE E RÁDIO

Através de uma linha telefônica comum, e com o uso de um periférico denominado MODEM, seu Expert DD PLUS pode receber e enviar dados, "conversando" com outro computador.

Note que o interlocutor de seu MSX pode ser até um computador de grande porte.

Com este sistema você pode receber software transmitido por outro computador. Se você for radio-amador, existem modelos de modem que permitem esta "conversa" à distância sem necessidade de linhas telefônicas.

Um outro processo muito interessante é usado, por exemplo, por algumas emissorasde FM: a transmissão de "software radiofônico"!

Basta sintonizar na estação no horário do programa e, quando o locutor avisa, gravar os ruídos dos códigos enviados. Posteriormente o ouvinte pode mandar seu MSX "ler" esta fita, carregando o programa transmitido na memória do micro.

O PERFIL DO PROGRAMADOR

Existem dois tipos de programadores na área de microinformática: os amadores e os profissionais.

Os programadores amadores são "hobbistas" que extraem prazer do ato de criar coisas novas em seu micro. Acham, por exemplo, muito mais agradável criar um programa interessante do que passar horas jogando xadrez.

Apesar do carácter amadorístico desta atividade, ela pode gerar uma remuneração profissional quando estes trabalhos são aproveitados por alguma "Software-house" ou publicados em livros e revistas.

Além disso, a atividade de programar, faz "cócegas no cérebro", promovendo o desenvolvimento intelectual de quem a exerce e causando um grande prazer no momento em que se vê o trabalho realizado.

O programador profissional, por outro lado, é alguém que se dedica integralmente a esta atividade e dela retira a totalidade ou a parcela mais substancial de sua remuneração.

Como o ensino profissionalizante de computação, no Brasil, é ainda muito incipiente, o que se nota, é uma grande quantidade de amadores, em sua maioria auto-didatas, que migram para a área profissional, ao perceberem que tem vocação e aptidões para esta tarefa.

Neste contexto, os micros pessoais e em particular o Expert, estão desempenhando um papel importantíssimo na formação destes profissionais.

Através de um MSX eles tem oportunidade de acessar, a baixo custo, as mais variadas e modernas linguagens de programação, num processo de atualização constante.

AS LINGUAGENS DE PROGRAMAÇÃO

A rigor, a única linguagem que seu micro entende é a Linguagem de Máquina, uma sequência de códigos binários que, ao ser introduzidos no microprocessador (Z80 no caso do MSX) o obrigam a executar diversas tarefas.

Esta linguagem, porém, é extremamente abstrata e exige um grande esforço (de raciocínio e conhecimentos) por parte do programador.

Consequentemente, para o programador iniciante, é interessante começar pelo aprendizado de linguagens de "alto-nível". Estas linguagens têm uma estrutura mais próxima da linguagem humana, sendo algumas praticamente coloquiais.

Existem muitas linguagens de alto nível desenvolvidas com esta finalidade, cada uma com particularidades que a indicam como mais adequada para a realização de uma determinada tarefa.

Dentre as linguagens disponíveis para MSX, podemos citar:

BASIC: especialmente desenvolvida para principiantes, é de fácil aprendizado e é extremamente versátil.

FORTRAN: uma das primeiras linguagens desenvolvidas para computadores, especialmente para finalidades científicas.

PASCAL: linguagem moderna, para programação estruturada, muito usada por profissionais em desenvolvimento de sistemas.

COBOL: concebida para aplicações comerciais. Apesar de obsoleta, é ainda muito utilizada para manutenção de sistemas desenvolvidos há mais tempo.

C: ferramenta muito útil no desenvolvimento de sistemas mais complexos. Permite a incorporação de rotinas desenvolvidas em Linguagem de Máquina.

LISP: muito usada em sistemas de "inteligência artificial".

LOGO: sub-conjunto da LISP, é muito usada em aplicações educacionais. Foi especialmente desenvolvida para introduzir crianças no mundo da informática. Apesar deste aspecto "introdutório" é uma linguagem extremamente poderosa e a versão do LOGO para MSX permite o desenvolvimento de programas até profissionais.

DBASE: é uma linguagem especialmente desenvolvida para criação e gerenciamento de Bancos de Dados. Pode ser usada tanto no modo interativo (comando a comando) quanto no modo programado.

Normalmente estas linguagens são carregadas no micro, como programas, podendo ser divididas em duas grandes categorias:

1) LINGUAGENS INTERPRETADAS

São linguagens em que cada instrução é lida no texto de um "Programa-Fonte", traduzida para o código de máquina e imediatamente executada.

Têm a vantagem de serem altamente interativas, ou seja, se durante a execução ocorrer algo de errado, o programa pode ser interrompido e o texto do "Programa- Fonte" pode ser alterado e corrigido.

A grande desvantagem está no tempo de execução pois, toda vez que o programa é executado, o tempo de tradução é sempre consumido.

2) LINGUAGENS COMPILADAS

Neste caso o texto do "programa-fonte" é integralmente traduzido para produzir um "programa-objeto" em Código de Máquina.

São muito menos interativas e a fase de depuração de erros é muito mais trabalhosa (a cada revisão é necessário recompilar tudo). Em compensação, uma vez depurado, o programa-objeto roda com alta velocidade, pois todo o tempo de "tradução" é excluído.

Note que, enquanto o Programa-Objeto é específico para cada microprocessador (um programa em L.M. para Z80 não roda, por exemplo num IBM-PC que usa outro microprocessador), o Programa-Fonte pode mais facilmente ser transportado de uma máquina para outra.

Você pode, por exemplo, escrever um Programa-Fonte no seu Expert, compilá-lo e checar se funciona. Posteriormente, você pode pegar o texto do Programa-Fonte, passá-lo para um IBM-PC (que consegue ler textos a partir dos disquetes de MSX) e recompilá-lo usando um compilador específico para o microprocessador desta máquina.

Se você não usar particularidades do MSX (como "sprites" ou rotinas de som sofisticadas),

seu programa rodará tranquilamente na outra máquina!

Com estas possibilidades, ao invés de comprar, para uso em programação, 4 IBM-PC, o usuário pode adquirir apenas 1 e substituir os outros por MSX a um custo absurdamente menor!

A LINGUAGEM DO EXPERT DD PLUS

A rigor o Expert DD PLUS não tem uma linguagem própria (a não ser a L.M. do seu Z80) podendo ser configurado para usar qualquer uma das linguagens citadas.

Seus projetistas, porém, decidiram incorporar à máquina uma linguagem residente (uma espécie de cartucho já montado em seu interior) e escolheram o MSX-BASIC.

Desta forma, ao ser ligado, o Expert já se oferece para o uso do interpretador BASIC residente.

O MSX-BASIC do seu Expert DD PLUS é extremamente poderoso e cheio de recursos, permitindo a elaboração de programas sofisticados mesmo para usuários inexperientes.

Nos capítulos subsequentes você terá uma primeira noção de como utilizar esta linguagem, tanto na sua versão simples (sem DISK– BASIC-Cap.3), quanto na versão mais completa (com DISK BASIC-Cap.5).

Como a finalidade deste livro não é a de transformar seu leitor num "programador", o dicionário de BASIC contido no capítulo 6 apresenta-se de forma resumida, contendo apenas as informações essenciais.

Se você desejar se aprofundar mais nesta linguagem, aconselhamos a leitura do apendice E (Bibliografia Recomendada) para que você possa se orientar sobre a melhor sequência de aprendizado, tanto no BASIC, quanto em outras linguagens.

De qualquer forma, é bom frisar que, para utilizar seu Expert DD PLUS, não é indispensável aprender a programar. Existe uma quantidade enorme de softwares desenvolvidos para o padrão MSX e, sabendo escolher,você certamente encontrará os que atendem suas necessidades.

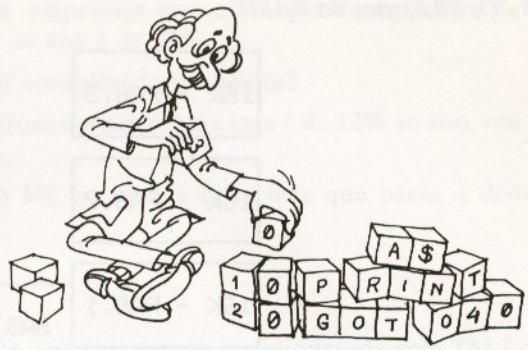
Os dois caminhos estão abertos à sua frente: ambos igualmente promissores.

Se você optar pela programação, leia atentamente os capítulos 3, 4 e 5 deste livro e utilize os capítulos 6 e 7 para consultas. Não deixe de ler o apêndice E (bibliografia aconselhada) para saber como prosseguir.

Se você optou por ser um usuário de software pronto, pode pular o capítulo 3 mas não deixe de ler o começo do 4 e do 5, onde você aprenderá a manusear os disquetes que "alimentarão" seu Expert DD PLUS.

Escolha o caminho que melhor se ajusta às suas necessidades e seja bem vindo ao maravilhoso mundo da informática!

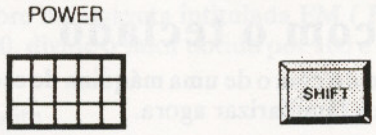
LIGANDO O MICRO SEM DISCO 3



As memórias do Expert

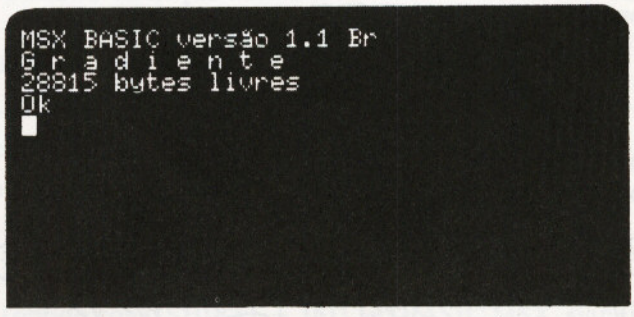
Certifique-se que seu micro esteja convenientemente instalado (reveja o capítulo 1 em caso de dúvida) e ligue-o mantendo a tecla SHIFT pressionada (fig.3.1).

fig.3.1- Tecla do power e do SHIFT



Desta forma ele vai ignorar a existência do acionador de disco e o BASIC disponível estará na sua versão mais simples. Deverá aparecer a mensagem da figura 3.2.

fig.3.2- Tela de Apresentação



Uma das primeiras perguntas que surgem na maioria dos usuários é "Eu comprei um MSX com 80 Kbytes de memória e aqui estão aparecendo menos de 29! Será que meu micro está com defeito?".

Calma! Não existe defeito nenhum: A memória de 80 kbytes está dividida da seguinte maneira: 16 kbytes são para o vídeo e os outros 64 kbytes de memória RAM estão distribuidos em 4 páginas (de 0 a 3) de 16 kbytes cada uma.

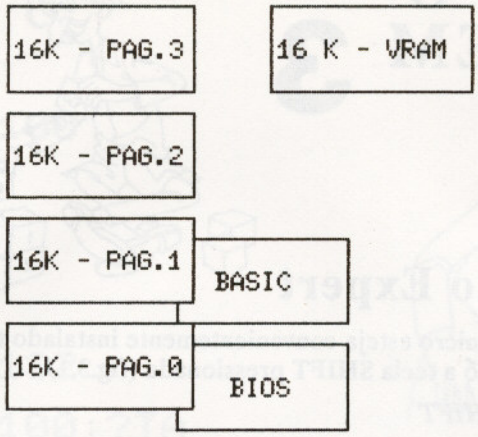
Para que o seu micro possa funcionar em BASIC, uma parte da memória RAM (páginas 0 e 1) não pode ser acessada; pois nesse lugar é colocada a ROM do micro, que nada mais é do que um programa muito grande e complexo que interpreta a linguagem BASIC.

"Sim - você dirá- mas isso me dá direito a 32 kbytes e não menos de 29!"

Você aprenderá mais tarde que todo programa, necessita de variáveis para poder armazenar dados; e, como o interpretador BASIC contido na ROM do micro é um programa, nada mais natural do que ele consumir um pouco da memória RAM para suas próprias variáveis.

De qualquer forma, não se preocupe: 28 Kbytes são mais do que suficientes para as finalidades a que nos propomos agora. .

fig.3.3 - Os 80 Kbytes do BASIC



Familiarizando-se com o teclado

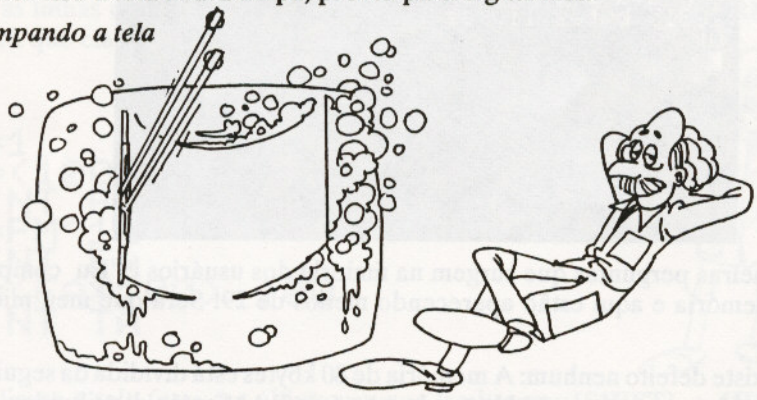
O teclado do MSX é muito parecido com o de uma máquina de escrever, acrescido de algumas teclas especiais com as quais vamos nos familiarizar agora.

A partir deste momento toda vez que escrevemos "aperte as teclas **[A] + [B]**" isto significará: pressione a primeira (no caso A) e SEM ABANDONÁ-LA, pressione também a segunda (no caso B).

Experimente digitar **[SHIFT] + [HOME/CLS]** (ou seja aperte com um dedo a tecla **[SHIFT]** e, mantendo-a pressionada, aperte **[HOME/CLS]**.

Ao fazer isso a tela ficará limpa, pronta para digitarmos.

fig.3.4 — Limpando a tela



Agora que já aprendemos a limpar a tela, vamos começar a escrever alguma coisa.

Pressione a tecla **[A]** uma única vez: você verá um "a" minúsculo surgir na tela e o quadradinho luminoso (cursor) se deslocará de uma posição para direita.

fig.3.5

a

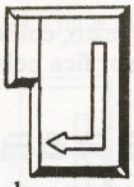
Pressione a tecla **[A]** e mantenha-a pressionada algum tempo: isso ativará o repetidor automático de teclado e um monte de "aaa" surgirão na tela (fig. 3.6).

fig.3.6

aaaaaaaaaaaaaaa

Pressione a tecla do **[RETURN]** (fig. 3.7)

(fig.3.7)

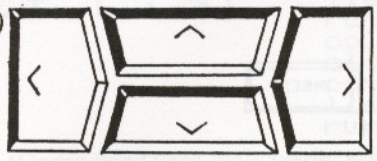


E na tela deverá aparecer uma mensagem de erro pois, para o BASIC, "aaaaaa..." Não significa nada !(fig. 3.8).

```
aaaaaaaaaaaaaa
Syntax error
Ok
```

Vamos agora aprender a apagar: usando as setas (fig 3.9)

(fig.3.9)



Leve o cursor até o "n" da mensagem "Syntax error" (figura 3.10).

(fig.3.10)

```
aaaaaaaaaaaaaa
Syntax error
Ok
```

Pressione a tecla **[BS]** (Back Space) uma única vez: o "y" deve desaparecer (fig.3.11).

fig.3.11

```
aaaaaaaaaaaaaa
Sntax error
Ok
```

O efeito desta tecla é fazer o cursor retroceder de uma posição apagando o que encontra pelo caminho.

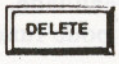
Se você apertasse novamente o **[BS]**, quem deveria sumir agora seria o "S".

Mas, ao invés disso, aperte uma vez **[DELETE]**.

Quem desapareceu foi o "n" (fig.3.12)

fig. 3.12

```
aaaaaaaaaaaaaa
Sax error
Ok
```



O efeito do DELETE é o de manter o cursor no local, apagar o que está em baixo dele e "puxar" para esquerda o resto da linha.

Percebeu a diferença entre o **[BS]** e o **[DELETE]** ?

Vamos, agora, reestabelecer a mensagem "Syntax error". Pressione o **[Y]** uma vez. O "y" substitui o "n" e cursor foi parar em cima do "a" (fig. 3.13)

fig. 3.13

```
aaaaaaaaaaaaaa
Syax error
Ok
```

Para não destruir o resto do texto, coloque o cursor no modo " inserção" pressionando uma única vez a tecla **[INSERT]**. O cursor fica pela metade (fig.3.14)

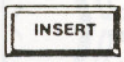
fig. 3.14

```
aaaaaaaaaaaaaa
Syax error
Ok
```

Digite agora **[N]** e **[T]** de maneira o reestabelecer a mensagem original. Note que agora, com o cursor pela metade (modo "inserção") as novas letras digitadas não apagam as seguintes, mas se "inserem" no texto, deslocando o resto para direita. Aperte mais uma vez a tecla **[INSERT]** e veja o cursor voltar ao normal (fig. 3.15)

fig. 3.15

```
aaaaaaaaaaaaaa
Syntax error
Ok
```



Leve agora o cursor até a primeira linha livre, usando as setas, e aperte uma vez a tecla **[CAPS LOCK]**.

A seguir digite, na sequencia **[C],[L],[S]** (fig.3.16).

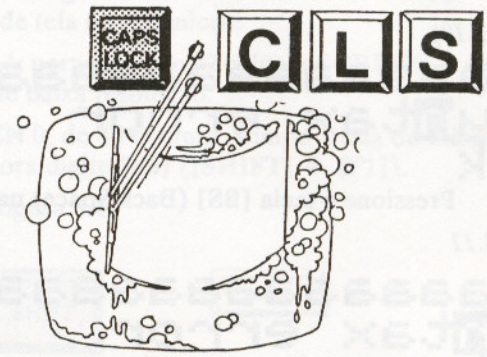


fig.3.16

```
aaaaaaaaaaaaaa
Syntax error
Ok
CLS
```

A tecla **[CAPS LOCK]** funciona como a "trava de maiúsculos" da máquina de escrever. Note que ela age apenas sobre as letras (se você quiser o ponto de exclamação (!) terá que digitar **[SHIFT]** + **[1]**). Para "destravar", basta apertar **[CAPS LOCK]** mais uma vez.

Por enquanto, porém, deixe travado. Agora aperte o **[RETURN]** (a tecla da figura 3.7) .

Ao invés de aparecer "Syntax error" como aconteceu antes, a tela foi apagada e apareceu um "Ok". Isso porque a palavra CLS, para o BASIC, tem significado e ele conseguiu executar a ordem. Ela significa "LIMPE A TELA" (CLear Screen) e foi isso que ele fez.

Você deve ter notado, até agora, que aparecem dizeres no rodapé da tela: eles correspondem às teclas de função.

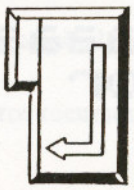
No teclado tem 5, mas na realidade, são 10. Se você apertar **[SHIFT]**, verá outras 5 mensagens: a tecla F1,por exemplo, pode funcionar como F6 se for pressionado em conjunto co **[SHIFT]** :

[F6] = [SHIFT] + [F1]

Se você digitar KEY OFF e, a seguir, **[RETURN]**, verá o rodapé desaparecer (fig.3.17).

(fig.3.17)

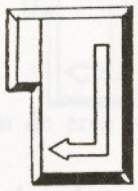
```
Ok
KEY OFF
```



Se quiser que o rodapé reapareça, digite KEY ON e **[RETURN]** (fig. 3.18).

(fig. 3.18)

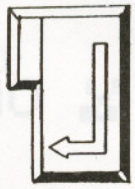
```
Ok
KEY OFF
Ok
KEY ON
```



Se você quiser ver o conteúdo das teclas, basta digitar KEY LIST e **[RETURN]** (fig. 3.19)

fig.3.19

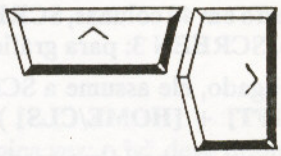
```
KEY LIST
color
auto
goto
list
run
color 15,1,1
cload"
cont
list
run
```



Como você verá mais tarde, estas teclas podem ser reprogramadas com o conteúdo que você quiser. Mas, por enquanto, vamos continuar a exploração do teclado: leve o cursor até o "I" do KEY LIST, usando as setas (fig. 3.20)

fig. 3.20

```
KEY ON
Ok
KEY LIST
color
```



Pressione, agora, **[CONTROL] + [E]**. Como você pode observar na fig. 3.21 o efeito dessa combinação de teclas é o de apagar toda linha abaixo e à direita do cursor.

fig. 3.21

```
KEY ON
Ok
KEY L
color
```

Leve agora o cursor até a linha do KEY OFF, com as setas, colocando-o em cima do O, como indica a figura 3.22.

fig. 3.22

```
Ok
KEY OFF
Ok
KEY ON
```

Pressione as teclas **[CONTROL] + [U]**: a linha inteira desaparecerá (fig. 3.23).

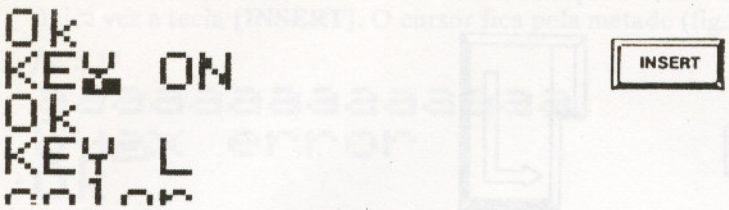
fig. 3.23

```
Ok

Ok
KEY ON
```

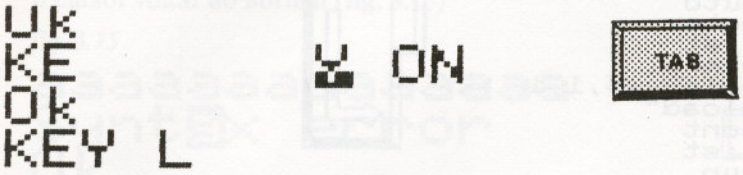
Existem outras combinações da tecla **[CONTROL]** que veremos mais tarde. Por enquanto vamos nos limitar em analisar o efeito de mais uma tecla. Coloque o cursor sobre o "Y" do KEY ON e pressione **[INSERT]** para colocar o cursor no modo "inserção" (pela metade) como mostrado na figura 3.24 .

fig. 3.24



Pressione a tecla **[TAB]**: como você pode notar, ela funciona como um TABulador que desloca o cursor nas posições múltiplas de 8 na tela.(fig. 3.25).

fig. 3.25



USANDO A SCREEN 0

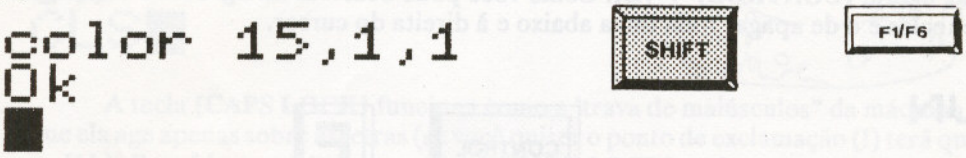
O padrão MSX-1 impõe a existência de 4 tipos de tela no seu micro:

SCREEN 0: para texto em 40 colunas, SCREEN 1: para texto em 32 colunas SCREEN 2: para gráficos de alta resolução, SCREEN 3: para gráficos de baixa resolução.

Quando o micro é ligado, ele assume a SCREEN 0, de 40 colunas. Limpe a tela do micro (Lembra ? Só digitar **[SHIFT]** + **[HOME/CLS]**). Agora digite **[F6]** (**[SHIFT]** + **[F1]**).

Na tela deve aparecer o comando mostrado na figura 3.26

fig. 3.26



Em alguns televisores o tamanho da imagem "transborda" um pouco os limites da tela, fazendo com que as primeiras letras não apareçam.

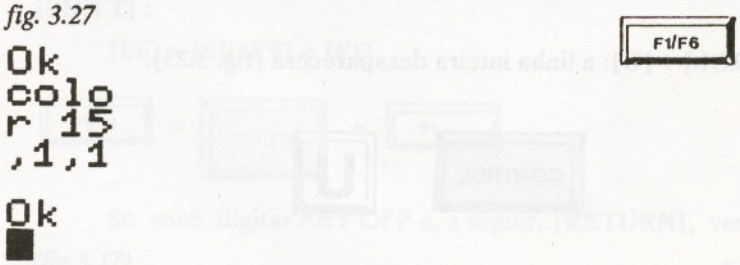
Se este é seu caso, você pode re-ajustar a largura do SCREEN 0 usando um valor menor que o inicial de 40 colunas.

Digite, por exemplo:

WIDTH 4 (e **[RETURN]**)

Obviamente estamos exagerando pois dificilmente uma tela de apenas 4 colunas terá alguma utilidade. Agora pressione novamente **[F6]** para ver como fica o comando que ela ativa (fig. 3.27)

fig. 3.27



Agora comande WIDTH 40 para reestabelecer a largura original ou WIDTH 39 ou WIDTH 38 se sua TV está "transbordando" um pouco.

Pronto! Vamos agora aprender alguma coisa sobre outros recursos do BASIC para perceber a enorme utilidade desta linguagem.

Digamos, por exemplo, que você queira emprestar uma certa quantia e acabou de ler, na Constituição, que a máxima taxa de juros reais ao ano é de 12%.

Qual deve ser a taxa mensal para não ser considerado um agiota?

A resposta mais imediata que muitas pessoas dariam é " se a taxa é de 12% ao ano, vou cobrar 1% ao mês!"

Vamos checar? Se você está cobrando 1% ao mes, a cada mês que passa a dívida fica multiplicado por 1.01 (100% + 1%).

Digite então o comando direto

TM = 1 (e **[RETURN]**)

Neste momento o micro abre uma gaveta em sua memória, etiquetada com TM (de taxa mensal) e nela armazena o valor 1

A seguir digite:

FM = (100 + TM)/100 (e **[RETURN]**)

Com isso o micro abre uma gaveta intitulada FM (Fator Mensal) vai buscar o conteúdo da gaveta TM, soma-o com 100, divide o valor obtido por 100 e armazena o resultado (que no caso deve ser 1.01) na gaveta FM.

Vamos conferir? Digite

PRINT FM (e **[RETURN]**)

Com isso você ordenou ao micro para imprimir (PRINT) na tela o valor de FM. Note que você obteve 1.01 e não 1,01 pois o BASIC MSX usa a notação ANGLO-SAXÔNICA (ponto e não vírgula para separar o inteiro da fração decimal) .

Vamos agora calcular o fator anual deste juro: Se, a cada mês que passa, a dívida fica multiplicada por 1.01, ao cabo de um ano a multiplicaremos por

1.01 x 1.01 x 1.01x...(12 vêzes)

Isso equivale a se calcular

(1.01) elevado a 12

Digite agora

FA= FM ^ 12 (e **[RETURN]**)

O simbolo ^ (obtido com **[SHIFT]** + **[6]**), no BASIC MSX, significa "Elevado a ..." (não confunda com o acento circunflexo, que está na mesma tecla do til).

Se você está curioso para descobrir quanto deu, digite

? FA (e **[RETURN]**)

O sinal de interrogação "?" É uma forma abreviada do "PRINT"

Você obteve

1.126825030132

Para saber a taxa anual (TA) Basta digitar

TA = (FA*100)- 100 : ?TA (e **[RETURN]**)

Se você errar alguma coisa na digitação, lembre-se do **[DELETE]**, **[BS]** e **[INSERT]**.

Nesta última linha aprendemos mais três coisas: Que você pode digitar mais de um comando por linha desde que os separe por ":", que o sinal de multiplicação é "*" e que você corre o risco de ser preso por agiotagem pois a taxa de 1% ao mês corresponde a 12,68% ao ano!

A tela do seu micro deve ter ficado como mostra a figura 3.28.

fig. 3.28

```
Ok
TM=1
Ok
FM=(100+TM)/100
Ok
PRINT FM
 1.01
Ok
FA=FM^12
Ok
?FA
 1.126825030132
Ok
TA=(FA*100)-100:?TA
 12.682503O132
Ok
```

Agora você perguntará "Tudo bem, então qual deve ser a taxa mensal para não ultrapassar os 12% ao ano ?".

Antes de responder, vamos aprender mais alguma coisa: Até agora você usou o micro no modo "Comando Direto" ou seja cada instrução foi imediatamente executada após sua digitação.

Vamos aprender agora a trabalhar no modo programado, ou seja, fazendo uma "LISTA" de instruções que serão depois executadas de uma vez.

Digite as linhas mostradas na figura 3.29 não esquecendo de digitar **[RETURN]** no final de cada uma, para que ela seja armazenada na memória do micro.

fig. 3.29

```
10 TM=1
20 FM=(100+TM)/100
30 PRINT FM
40 FA=FM^12
50 PRINT FA
60 TA=(FA*100)-100
70 PRINT TA
```

Agora limpe a tela (esta é a última vez que lembramos:**[SHIFT]** + **[HOME/CLS]** e digite **[F9]** (**[SHIFT]** + **[F4]**). Esta tecla comanda um "list" (Liste o programa) e ele vai reaparecer na tela. Compare com a figura 3.29: se estiver faltando alguma linha, você esqueceu o **[RETURN]**.

Digite a linha faltante e, dessa vez, não esqueça o **[RETURN]**.

Vamos agora "correr" o programa ou seja vamos fazer o interpretador ler linha a linha, traduzir em L.M. e executar. Basta digitar **[F5]** para que apareceram os 3 resultados obtidos anteriormente: FM, FA e TA (fig. 3.30).

fig. 3.30

```
run
 1.01
 1.126825030132
 12.682503O132
Ok
```

Vamos agora indroduzir uma pequena melhoria em nosso programa: leve o cursor até a linha 30, na posição indicada na figura 3.31, e aperte o [INSERT]

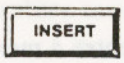


fig. 3.31

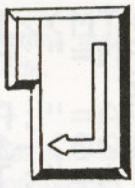
```
10 TM=1
20 FM=(100+TM)/100
30 PRINT FM
40 FA=FM^12
50 PRINT FA
```

Digite agora os caracteres indicados na figura 3.32 e aperte [RETURN]. Note que, para que a linha, assim alterada, seja atualizada na memória, é indispensável apertar [RETURN].

Para tanto, não se preocupe com a posição do cursor, basta que ele esteja sobre a linha em questão

fig. 3.32

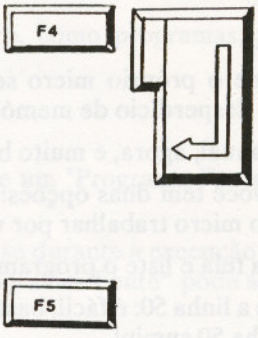
```
10 TM=1
20 FM=(100+TM)/100
30 PRINT "FM=";FM
40 FA=FM^12
```



Faça, com a mesma técnica, as alterações indicadas nas linhas 50 e 70 da figura 3.33. Limpe a tela, liste o programa ([F4] e [RETURN]) e rode-o ([F5]).

fig. 3.33

```
10 TM=1
20 FM=(100+TM)/100
30 PRINT "FM=";FM
40 FA=FM^12
50 PRINT "FA=";FA
60 TA=(FA*100)-100
70 PRINT "TA=";TA
Ok
run
FM= 1.01
FA= 1.126825030132
TA= 12.6825030132
```



Note como ficou mais simples entender o significado dos números fornecidos pelo programa. Como você deve ter percebido, ao comandar PRINT TA, o micro escreve na tela o valor armazenado na "gaveta" TA. Ao comandar PRINT "TA" ele imprime a mensagem que está entre aspas, sem se preocupar com seu significado.

Imagine que você escreva, durante um concerto sinfônico, um bilhete para um vizinho tagarela com os dizeres:

Fale sussurando

Ele passará a sussurar. Agora, se você escrever:

Fale "sussurando"

Corre o risco de ouvi-lo dizer, alto e bom tom:

SUSSURANDO!

Pois uma pessoa que, durante um concerto, tem a cara de pau de tagarelar, normalmente tem Q.I. de rabanete!

O ";" que aparece entre a mensagem entre aspas e o nome da variável é um separador que indica ao comando PRINT: "O que vem depois do ";" deve ser impresso a seguir e não na outra linha".

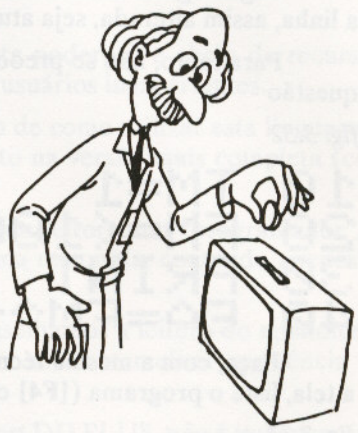
"Sim -você dirá- tudo isso é muito interessante mas ainda não sei quanto vou cobrar de juros por mês".

Vamos então tentar outro valor. Leve o cursor até a linha 10 e, no lugar de 1 digite 0.5 (1/2%). Não esqueça o **[RETURN]** para inserir a alteração na memória.

Limpe a tela, tecle **[F9]** para listar novamente o programa e **[F5]** para rodá-lo. Você deve obter uma tela como a figura 3.34

fig. 3.34

```
list
10  TM=.5
20  FM=(100+TM)/100
30  PRINT  "FM=";FM
40  FA=FM^12
50  PRINT  "FA=";FA
60  TA=(FA*100)-100
70  PRINT  "TA=";TA
Ok
run
FM=  1.005
FA=  1.061677811864 5
TA=  6.16778118645
Ok
```



Note que o próprio micro se encarregou de mudar 0.5 para .5 (ele considera o "zero à esquerda" um desperdício de memória).

A taxa anual, agora, é muito baixa, menor que 6.17% ao ano.

Agora você tem duas opções: repetir o processo até descobrir o valor mais adequado para TM, ou fazer o micro trabalhar por você.

Limpe a tela e liste o programa (não vamos mais dizer como fazer isso!).

Elimine a linha 50: é fácil, basta digitar 50 e **[RETURN]**. Limpe a tela e liste o programa para ver como a linha 50 sumiu!

Agora coloque uma vírgula no final da linha 30. Ela indica: o próximo PRINT deverá ser na mesma linha, mas a partir da coluna 16. Rode o programa assim alterado e obterá a tela da fig.3.35

fig. 3.35

```
10  TM=.5
20  FM=(100+TM)/100
30  PRINT  "FM=";FM,
40  FA=FM^12
60  TA=(FA*100)-100
70  PRINT  "TA=";TA
Ok
run
FM=  1.005          TA=  6.16778118645
```

Limpe a tela e liste o programa. Vamos alterar a linha 10 para :

10 FOR TM = .5 TO 1 STEP .1

Para tanto não precisa levar o cursor até a linha 10: Basta redigitar lá em baixo e não esquecer o **[RETURN]** ao terminar a linha. A antiga linha 10 será substituiuda pela nova.

Da mesma forma, acrescente a linha 80:

80 NEXT TM (e**[RETURN]**!)

A tela deve ficar como mostra a figura 3.36

fig. 3.36

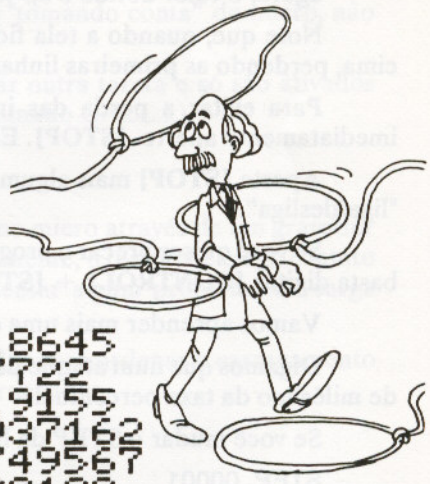
```
list
10  TM=.5
20  FM=(100+TM)/100
30  PRINT  "FM=";FM,
40  FA=FM^12
60  TA=(FA*100)-100
70  PRINT  "TA=";TA
Ok
10  FOR  TM=.5  TO  1  STEP  .1
80  NEXT  TM
```

Você pode achar que o micro vai se confundir pois está tudo fora de ordem. Que nada!

Ele é muito esperto (ou Expert!): Limpe a tela e liste o programa para ver como ele já se encarregou de organizar tudo! Rode o programa e você obtera a figura 3.37

fig.3.37

```
list
10  FOR  TM=.5  TO  1  STEP  .1
20  FM=(100+TM)/100
30  PRINT  "FM=";FM,
40  FA=FM^12
60  TA=(FA*100)-100
70  PRINT  "TA=";TA
80  NEXT  TM
Ok
run
FM=  1.005        TA=  6.16778118645
FM=  1.006        TA=  7.44241677722
FM=  1.007        TA=  8.73106619155
FM=  1.008        TA=  10.0338693716
FM=  1.009        TA=  11.3509674956
FM=  1.01         TA=  12.6825030132
Ok
```



Agora vamos entender o que estas duas linhas adicionais fizeram: Existe uma estrutura de programação denominada "laço" (ou "LOOP" em "informatiquês") que começa com um "FOR" e termina com "NEXT". A linha 10 diz: comece com um valor de TM = .5 (FOR TM = .5) e efetue todas as instruções até achar o NEXT TM (que no nosso caso está na linha 80).

Feito isso acrescente .1 ao valor anterior de TM (STEP. 1) e repita todo o processo até TM valer 1 (TO 1). Se o STEP for omitido, o acréscimo será de 1 em 1. Note que você pode usar um STEP negativo, desde que o valor até inicial seja maior que o final.

Como você notou na fig. 3.37, o valor de TM que estamos procurando deve estar entre .9 e 1.

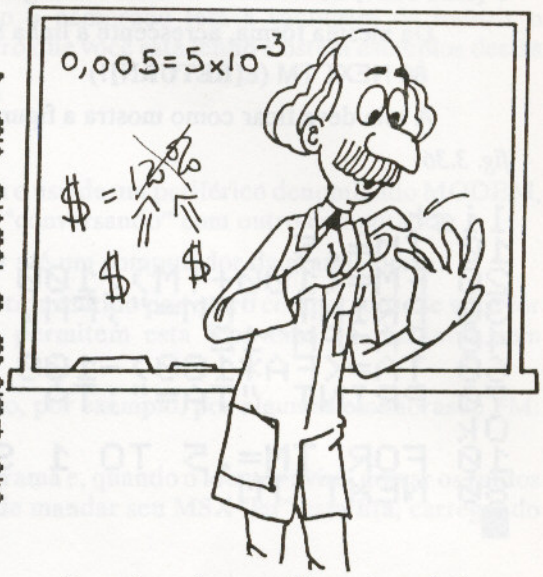
Vamos novamente alterar a linha 10:

10 FOR TM = .9 TO 1 STEP .005 (e **[RETURN]**!)

Limpe a tela, liste o programa (note que o parâmetro do STEP passou para notação científica; 0,005 = 5x10E-3) e rode-o. Você deve obter a tela da figura 3.38

fig. 3.38

```
run
FM= 1.009       TA= 11.35096749567
FM= 1.00905     TA= 11.41720019391
FM= 1.0091      TA= 11.48346900328
FM= 1.00915     TA= 11.54977394165
FM= 1.0092      TA= 11.6161150269
FM= 1.00925     TA= 11.68249227699
FM= 1.0093      TA= 11.74890570985
FM= 1.00935     TA= 11.81535534338
FM= 1.0094      TA= 11.88184119552
FM= 1.00945     TA= 11.9483632842
FM= 1.0095      TA= 12.01492162743
FM= 1.00955     TA= 12.08151162431
FM= 1.0096      TA= 12.14814714923
FM= 1.00965     TA= 12.21481436377
FM= 1.0097      TA= 12.28151790472
FM= 1.00975     TA= 12.34825779008
FM= 1.0098      TA= 12.41503403784
FM= 1.00985     TA= 12.48184666601
FM= 1.0099      TA= 12.54869569262
FM= 1.00995     TA= 12.61558113566
FM= 1.01        TA= 12.68250301 32
Ok
```



O valor procurado deve ser então TM = 0.945% ao mês, maior valor que dá uma taxa de juros menor que 12% ao ano. Se você é um pão-duro que não quer perder sequer uma migalha de juros, mantendo-se nos estritos limites da lei, deve alterar a linha 10 para

10 FOR TM = .945 TO .950 STEP .0001

agora, o valor obtido é 0,9488% ao mês!

Note que, quando a tela fica cheia, ela começa a sofrer um "SCROLL", ou seja roda para cima, perdendo as primeiras linhas.

Para evitar a perda das informações iniciais, rode novamente o programa com **[F5]** e imediatamente aperte **[STOP]**. Ele congela.

Aperte **[STOP]** mais algumas vezes e você perceberá que ele funciona como um interruptor "liga/desliga".

Se você quiser brecar o programa definitivamente para obter novamente o controle do cursor, basta digitar **[CONTROL]** + **[STOP]** durante a execução. Experimente!

Vamos aprender mais uma coisa útil:

Digamos que num ataque de avidez descontrolada você não queira perder nem um centésimo de milésimo da taxa percentual.

Se você mudar o STEP da linha 10 para

STEP. 00001

Você terá uma lista de valores muito mais detalhada e extensa. Experimente, porém fique pronto com o dedo sobre o **[STOP]** senão o valor que lhe interessa será engulido pelo "SCROLL"!

Além da tela ficar "bagunçada" (pois alguns valores de TM ultrapassam a coluna 16), fica difícil encontrar o valor correto.

Vamos então aprender mais uma instrução muito útil. Limpe a tela e liste o programa acrescente a linha 75:

75 IF TA > = 12 THEN END (não esqueça o **[RETURN]**!)

Rode o programa e deixe a tela correr.

A instrução significa: "Se TA (IF TA) for maior ou igual (> =) a 12, então (THEN) encerre a execução (END)".

A tela final deve ter o aspecto da figura 3.39 e nela você descobre que a melhor taxa de juro mensal é 0,94887%.

fig. 3.39

```
TA= 11.99854491172
FM= 1.0094878
TA= 11.99867804685
FM= 1.0094879
TA= 11.9988111822
FM= 1.009488   TA= 11.99894431767
FM= 1.0094881
TA= 11.99907745328
FM= 1.0094882
TA= 11.99921058903
FM= 1.0094883
TA= 11.99934372498
FM= 1.0094884
TA= 11.99947686102
FM= 1.0094885
TA= 11.99960999725
FM= 1.0094886
TA= 11.99974313358
FM= 1.0094887
TA= 11.99987627004
FM= 1.0094888
TA= 12.00000940664
Ok
```

Agora vamos contar um segredo: toda essa trabalheira foi para que você aprendesse um monte de coisas sobre o BASIC. Na realidade o problema proposto poderia ter sido resolvido numa única linha, em modo comando direto! Experimente digitar

? 1.12 ^ (1/12) (e **[RETURN]**!)

A resposta será 1.0094887929345, ou seja, a taxa de juro mensal procurada é de:

0,94887929345 % !

De qualquer forma, o programa que digitamos pode ter outra utilidade (além da didática): Dada uma taxa de juros mensal, ele permite descobrir o juro acumulado anualmente. Para não ter que alterar o programa toda vez que quisermos mudar o valor de TM, liste o programa e altere-o até ter o aspecto da listagem da fig. 3.40

fig. 3.40

```
list
10  INPUT"TAXA MENSAL=";TM
20  FM=(100+TM)/100
30  PRINT:PRINT "TM=";TM;"%",
40  FA=FM^12
60  TA=(FA*100)-100
70  PRINT  "TA=";
75  PRINT USING"######.###";TA;
77  PRINT"%":PRINT
80  GOTO 10
Ok
```

Nesta altura você deve ser capaz de entender a maioria das alterações. Temos apenas 3 novidades:

O INPUT é um comando que, após apresentar a mensagem entre aspas, exibe um ponto de interrogação, congela a execução e fica esperando a entrada de um valor pelo teclado.

Rode o programa e, após o aparecimento da mensagem e do ponto de interrogação, digite, por exemplo, 4 (e **[RETURN]**).

Você deve obter a tela da figura 3.41

fig. 3.41

```
60 TA=(FA*100)-100
70 PRINT "TA=";
75 PRINT USING"#####.###";TA;
77 PRINT"%":PRINT
80 GOTO 10
Ok
run
TAXA MENSAL=? 4

TM= 4 %          TA=    60.103%

TAXA MENSAL=? ■
```

O computador, além de dar a resposta (60.103%) volta a repetir a pergunta, pronto para calcular outro valor. Isso é devido à linha 80 (GOTO 10) que, após o cálculo, manda a execução voltar ao início. Essa é a segunda novidade.

A terceira novidade está na linha 75, o:

PRINT USING "#####.###" ;...,

Este comando estabelece uma "máscara" para os dados numéricos de maneira a alinhá-los segundo um padrão. No caso a máscara impõe um máximo de 5 casas antes do ponto decimal e 3 depois.

Agora pode desligar o micro e descansar um pouco, antes de passar ao próximo ítem.

USANDO A SCREEN 1

Você poderia se perguntar: " Se tenho a SCREEN 0 que tem 40 colunas e eu posso até configurá-la para 32 usando WIDTH 32, para que preciso de uma SCREEN específica para 32 colunas ?".

A resposta é simples: ligue seu micro (mantendo o **[SHIFT]** apertado, lembra ?) e limpe a tela.

Agora pressione a combinação de teclas **[LGRA]** + **[M]** para obter o símbolo de masculino da biologia. Afaste o cursor com a seta para direita e observe atentamente a tela. O símbolo aparece cortado!

Agora comande (no modo direto):

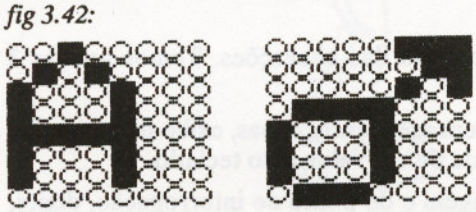
SCREEN 1 (e **[RETURN]**)

Novamente digite **[LGRA]** + **[M]** : agora você obterá o símbolo completo.

Isso ocorre porque, apesar de todos os caracteres do MSX serem definidos uma matriz de 8x8 pontos (pixels, em "informatiquês"), a SCREEN 0 mostra apenas 6 na horizontal, enquanto que a SCREEN 1 mostra os 8.

Para as letras, a SCREEN0 não resulta em cortes porque todas elas tem uma largura máxima de 5 pixels (+ 1 para espaçamento) enquanto que outros caracteres podem ocupar todos aos 8 pixels na horizontal (fig. 3.42).

fig 3.42:



No MSX temos 256 caracteres diferentes.

Para vê-los todos,vamos digitar um curto programinha. Obviamente usaremos a SCREEN 1 para poder ver todos os caracteres integralmente. Digite o programa de figura 3.43

fig.3.43

```
100 N=0:SCREEN 1
110 FOR I=6212 TO 6692 STEP 32
120 FOR K=0 TO 15
130 VPOKE I+K,N
140 N=N+1
150 NEXT K,I
```

Agora rode o programa. Pronto! Na SCREEN 1 você tem em 16 linhas e 16 colunas seus 256 caracteres do MSX (fig. 3.44)

fig. 3.44

Antes de continuar, dê umas estudadinha em tudo que você tem a disposição: símbolos matemáticos, caracteres semi-gráficos, naipes do baralho, até um pouco do alfabeto grego!

Não se preocupe, por enquanto, em entender o programinha da figura 3.40. Basta saber que ele escreve diretamente na memória de vídeo (VPOKE) os códigos (N) dos caracteres de 0 a 255. O último, inclusive, é o caracter do cursor.

Passeie com o cursor (usando as setas) pela tabela e fique observando o último caractere!

Vamos agora aprender mais alguns truques: limpe a tela e digite

KEY 1, "SCREEN 0:LIST "+CHR$ (13) (e **[RETURN]**)

Olhe o rodapé da tela: sabe o que aconteceu ?

Você reprogramou o conteúdo de tecla **[F1]**.. O CHR$ (13) que aparece no fim, corresponde ao **[RETURN]**.

Aperte **[F1]**: o que aconteceu ? A sequência de ordens contidos na tecla **[F1]** foi executada, inclusive o **[RETURN]**!

Aperte **[F5]** para rodar o programa e, ao terminar, **[F1]**. CÔmodo, não ?

Para achar um caracter em nossa tabelinha, podemos usar a técnica da batalha naval, numerando colunas e linhas. Como temos 16 linhas x 16 colunas podemos usar numeração hexadecimal (não se preocupe se você não conhece esta numeração, o MSX a conhece muito bem!).

Acrescente algumas linhas ao programa de fig. 3.43 de maneira a ficar como na fig. 3.45

fig. 3.45

```
100 N=0:SCREEN 1
110 FOR I=6212 TO 6692 STEP 32
120 FOR K=0 TO 15
130 VPOKE I+K,N
140 N=N+1
150 NEXT K,I
160 LOCATE 2,0
170 A$="0123456789ABCDEF"
180 PRINT A$
190 PRINT
200 FOR I=1 TO 16
210 PRINT MID$(A$,I,1)
220 NEXT I
230 PRINT
```

Antes de rodá-lo, vamos a uma rápida explicação:

O LOCATE 2,0 de linha 160, localiza o cursor na coluna 2, linha 0.

A$ é uma variável "string" ou alfa-numérica.

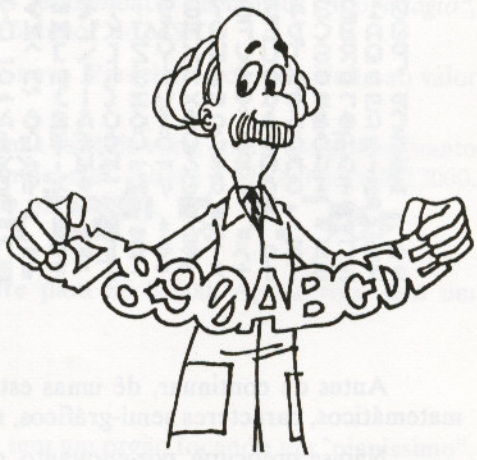
Quando o nome da variável é seguido pelo cifrão ($) a "gaveta" correspondente na memória não armazena dados númericos mas sim caracteres (no caso os caracteres do dígitos hexadecimais).

O laço de 200 a 220 vai "fatiando" o conteúdo de A$ com a função MID$ e imprimindo os caracteres nela contidos, 1 por 1.

Rode o programa e você obterá a tela da figura 3.46

fig. 3.46

```
 0 1 2 3 4 5 6 7 8 9 A B C D E F
0
1
2
3 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
4 @ A B C D E F G H I J K L M N O
5 P Q R S T U V W X Y Z [ \ ] ^ _
6 ` a b c d e f g h i j k l m n o
7 p q r s t u v w x y z { | } ~
8 Ç ü é â ä à å ç ê ë è ï î ì Ä Å
9 É æ Æ ô ö ò û ù ÿ Ö Ü ¢ £ ¥ Pt ƒ
A á í ó ú ñ Ñ ª º ¿ ⌐ ½ ¼ ¡ « »
B ã õ Ã Õ ø Ø œ Œ ¾ ◇ ¶ §
C
D
E α ß Γ π Σ σ µ τ Φ Θ Ω δ ∞ ∅ ∈ ∩
F ≡ ± ≥ ≤ ⌠ ⌡ ÷ ≈ ° • · √ ² ■
```



```
Ok
■
```

Digamos agora, que você queira obter o caractere correspondente ao xadrezinho que está na linha D, coluna 7 da nossa tabelinha. Leve (com as setas) o cursor até o começo da linha do OK e digite.

PRINT CHR$ (&HD7) (e **[RETURN]**)

Você obterá o caracter desejado na linha de baixo. Você comandou: "escreva na tela (PRINT) o caractere (CHR$) cujo código, em hexadecimal (&H) é D7".

Vamos obter o ponto de interrogação de ponta cabeça? Este, por exemplo, é um caractere fundamental para escrever textos em espanhol.

Achou? ótimo! Agora é só acabar de digitar e substituir o D7 por A8 (e **[RETURN]**).

Agora vamos tentar descobrir como imprimir o símbolo de masculino que usamos no começo deste ítem: ele está na linha 0, coluna B da tabelinha. Leve o cursor até a linha do PRINT e substitua o A8 por 0B. Aperte o **[RETURN]**.

Epa! Que coisa estranha! Ao invés de produzir o caracter desejado, o cursor pulou lá para cima!

Isso acontece porque os 32 primeiros caracteres do MSX (linha 0 e 1 da tabelinha) são usados, também, como caracteres de controle. Você já viu, por exemplo, que o CHR$(13) corresponde ao

RETURN.

No caso destes 32 caracteres, para avisar ao MSX que queremos um caracter e não uma ação de contrôle, devemos enviar antes o CHR$ (1) e a seguir o código que nos interessa acrescido de 64. vamos tentar isso? Altere a linha do PRINT para

```
PRINT CHR$ (1) + CHR$ (64 + &H0B)
```

e aperte o **[RETURN]**. Viu?

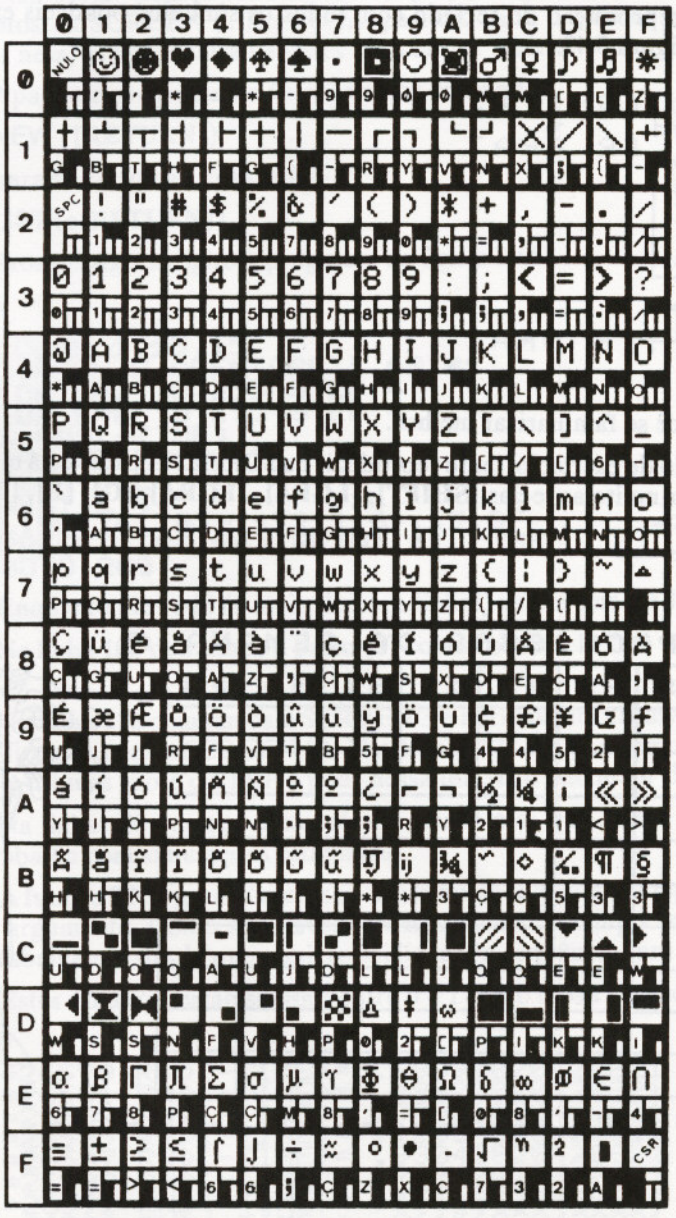
Mude o 0B por 0E: Que tal? Funcionou!

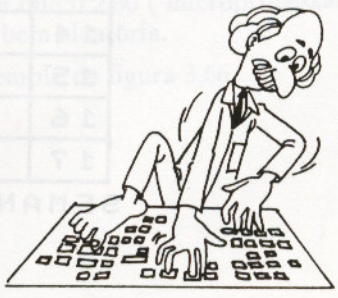
Você vai dizer "Isso é muito interessante mas não é um pouco complicado ?".

Na realidade todos estes caracteres podem ser obtidos através do teclado usando as teclas **[LGRA]**, **[RGRA]** e **[SHIFT]** .

Veja a tabela da figura 3.47, parecida com a que obtivemos na tela até agora.

fig. 3.47





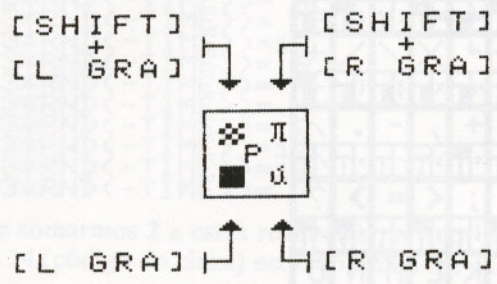
Cada caractere está dentro de um retângulo com o seguinte código:

fig. 3.48

Se você quiser, por exemplo obter o símbolo de libra no teclado, o CHR$ (&H9C), deverá pressionar as tecla **[SHIFT] + [RGRA] + [4]**.

Para consultas rápidas, esta tabela está também impressa na última contra-capa deste livro. Na primeira contra-capa está a reprodução do teclado com todos os símbolos possíveis em cada tecla, obedecendo ao código da figura 3.49.

fig. 3.49



Use aquela com a qual você se familiarizar melhor.

Agora limpe a tela e, a título de exercício, tente reproduzir na SCREEN 1 o que está mostrado na figura 3.50. Dica: você vai usar muito as teclas **[SHIFT]**, **[LGRA]**, **[RGRA]**, **[G]**, **[B]**, **[T]**, **[H]**, **[{]**, **[-]**, **[R]**, **[Y]**, **[V]** e **[N]**.

fig. 3.50

● COMPROMISSOS DA SEMANA ☺

HS	2ª	3ª	4ª	5ª	6ª	SA	DO
08							
09							
10							
11							
12							
14							
15							
16							
17							

SEMANA DE __/__/__ A __/__/__

Após esta árdua tarefa, descanse um pouco e desligue seu MSX para que ele descanse também!

USANDO A SCREEN 2

A tela de desenhos em alta resolução é a SCREEN 2. Ela permite desenhar 256 x 193 = 49152 pontos! Além disso o BASIC MSX tem recursos incríveis para se desenhar nela.

Vamos começar a explora-lá um pouco. Ligue seu MSX com a tecla **[SHIFT]** apertada e digite o programa da fig. 3.51

fig. 3.51

```
100 SCREEN 2:OPEN"GRP:" AS #1
110 PSET (10,10)
120 PRINT #1," (10,10)"
900 GOTO 900
Ok
```

A linha 100 chama a SCREEN 2 e abre (OPEN) um arquivo na tela gráfica (GRP:) como (AS) número(#) 1. Isso permite que,além desenhar na tela gráfica, possamos também escrever nela.

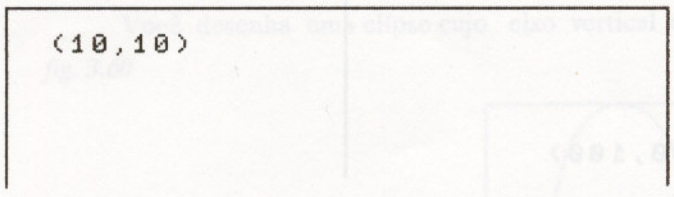
A linha 110 marca um ponto (Point SET) nas coordenandas X = 10, Y = 10. Note que enquanto o X cresce para direita (até 255) o y cresce para baixo (até 191), ao contrário do que acontece nos gráficos cartesianos normais.

A linha 120 escreve no arquivo número 1 que acabamos de abrir (PRINT#1) a mensagem entre aspas (que no nosso caso dá as coordenadas do ponto). Note que a mensagem é escrita a partir do último ponto setado na tela.

A linha 900 faz o papel de um cachorro tentando morder seu próprio rabo. Se ela não existisse, ao terminar a execução o MSX voltaria para a tela de texto, destruindo o desenho. Porisso, para obter sua listagem de volta aperte **[CONTROL]** + **[STOP]** e a seguir **[F4]** (e **[RETURN]**).

Você deve obter a tela da fig. 3.52

fig. 3.52



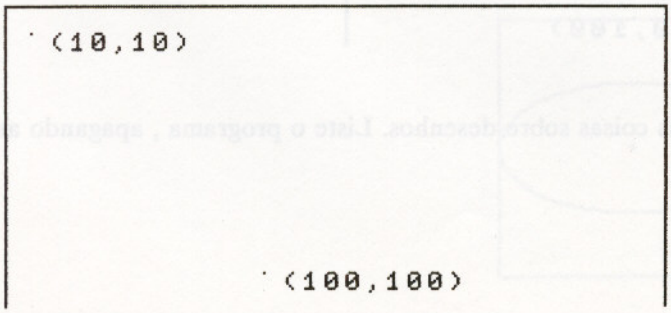
Agora acrescente algumas linhas ao seu programa, para que fique como a listagem da figura 3.53

fig. 3.53

```
100 SCREEN 2:OPEN"GRP:" AS #1
110 PSET (10,10)
120 PRINT #1," (10,10)"
130 PSET (100,100)
140 PRINT #1," (100,100)"
900 GOTO 900
```

Rode o programa: Agora você "plotou" dois pontos (fig. 3.54).

fig. 3.54



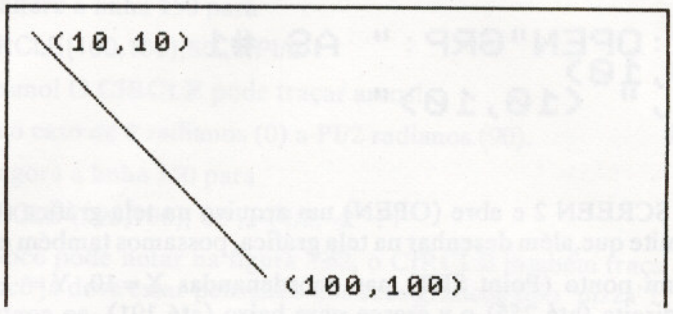
Experimente agora acrescentar a linha

150 LINE (10,10) – (100,100)

e rode o programa . Percebeu o que o LINE faz ? Desenha uma linha que vai do ponto (10,10) ao ponto (100,100).

Você deve obter uma tela como a da figura 3.55.

fig. 3.55



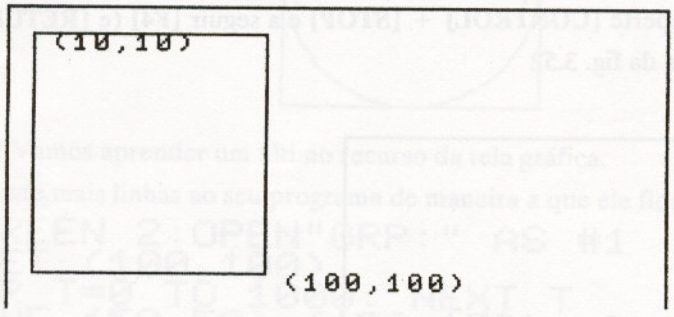
Mas não é só isso que o LINE faz!

Liste o programa e altere a linha 150 para

150 LINE (10,10) – (100,100),, B

Como você deve ter percebido, o B é de Box (caixa). Agora a tela ficou como na figura 3.56

fig. 3.56

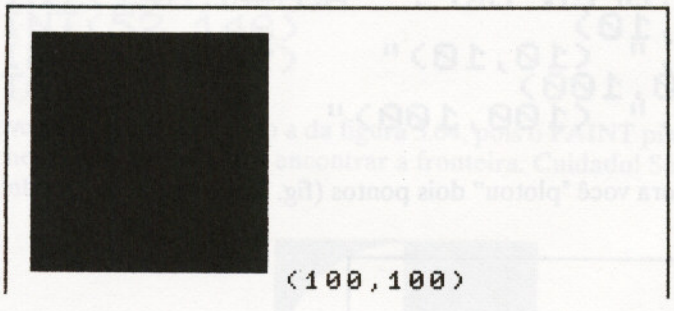


Novamente altere a linha 150 para

150 LINE (10,100) – (100,100),, BF

sendo BF de Box Full (caixa cheia). A tela agora ficará com o aspecto da fig. 3.57

fig. 3.57



Muito bem! Vamos aprender mais coisas sobre desenhos. Liste o programa , apagando as linhas de 100 a 150 comandando

DELETE 110-150 (e[RETURN])

Liste de novo para ver o efeito!

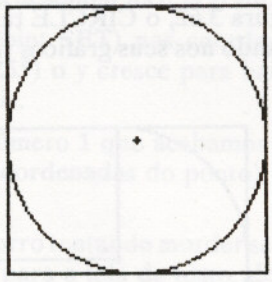
Agora digite as linhas adicionais indicadas na figura 3.58

fig. 3.58

```
100 SCREEN 2:OPEN"GRP:" AS #1
110 PSET (100,100)
120 FOR T=0 TO 1000: NEXT T
130 LINE (50,50)-(150,150),,B
140 FOR T=0 TO 1000: NEXT T
150 CIRCLE (100,100),50
900 GOTO 900
```

As linhas 110 e 130 dispensam comentários. As linhas 120 e 140 são "laços temporizadores",servem apenas para dar um pouco de "suspense". A novidade está na linha 150. Ela diz ao micro: "Desenhe um círculo com centro em (100,100) e raio 50. A tela deve ficar como na figura 3.59

fig. 3.59



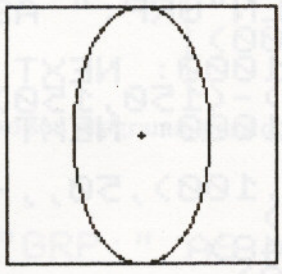
Agora altere a linha 150 para:

150 CIRCLE (100,1000), 50,,,,2

(conte as vírgulas direitinho!)

Você desenha uma elípse cujo eixo vertical é 2 vezes maior que o horizontal (fig. 3.60)

fig. 3.60

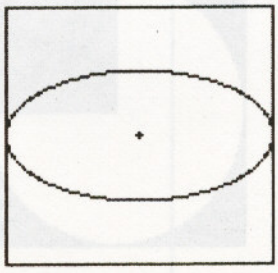


Se você tivesse digitado

150 CIRCLE (100,100), 50,,,,0.5

teria obtido a figura 3.61

fig. 3.61



Vamos ver agora para que serve aquele monte de vírgulas. Como você deve lembrar, uma circuferência completa tem 360° ou 2 * PI radianos. Como o MSX não sabe o quanto vale PI, vamos contar para ele.

Digite a linha

```
145 PI = 4 * ATN (1)
```

Se você não se lembra, o arco tangente (ATN) de 1 é PI/4.

Agora altere a linha 150 para

```
150 CIRCLE(100,100),50,,0,PI/2
```

Isso mesmo! O CIRCLE pode traçar arcos!

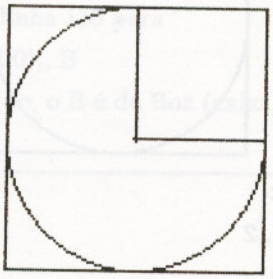
No nosso caso de 0 radianos (0) a PI/2 radianos (90).

Altere agora a linha 150 para

```
150 CIRCLE (100,100), 50 ,, -PI/2, -2* PI
```

Como você pode notar na figura 3.62, o CIRCLE também traça setores (se os arcos forem negativos). Você já deve estar pensando nos seus gráficos tipo "pizza"!

fig. 3.62



Calma ! Vamos aprender um último recurso da tela gráfica.

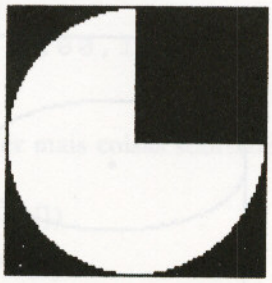
Acrescente mais linhas ao seu programa de maneira a que ele fique como na figura 3.63

```
100 SCREEN 2:OPEN"GRP:" AS #1
110 PSET (100,100)
120 FOR T=0 TO 1000: NEXT T
130 LINE (50,50)-(150,150),,B
140 FOR T=0 TO 1000: NEXT T
145 PI=4*ATN(1)
150 CIRCLE (100,100),50,,-PI/2,-2*PI
160 PAINT(52,52)
170 PAINT(148,148)
180 PAINT(52,148)
190 PAINT(148,52)
900 GOTO 900
```

fig. 3.63

Você deve obter uma tela como a da figura 3.64, pois o PAINT pinta toda região ao redor da coordenandas nele especificadas, até encontrar a fronteira. Cuidado! Se a fronteira for aberta, ele vai "vazar" e pintar a tela toda!

fig. 3.64



Para se certificar disto, comande:

DELETE 110-190 (e [RETURN])

e digite as linhas

110 CIRCLE (127,85), 80,, 4.6,4.5 (não confunda pontos e vírgulas!)

120 PAINT (127,85)

Bonito não? Dá até para perceber como o MSX "raciocina" para pintar uma região!

Vamos descansar um pouco? Não ? Ótimo então passe já para o item seguinte!

USANDO A SCREEN 3

A menos que você seja fabricante de tapetes ou "video-maker", dificilmente encontrara alguma aplicação profissional para a SCREEN 3.

Ela, "plota" pontos em baixa-resolução, ou seja cada pixel da SCREEN 3 é constituido por 4x4 pontos da SCREEN 2. Em contra partida, cada um deles pode ter uma cor diferente e, ao escrever nela, as letras sairão 4 vezes mais largas a altas!

Apague o programa do ítem anterior digitando

NEW ([RETURN])

para se certificar, comande

LIST (e[RETURN])

Pronto! Agora digite o programa da figura 3.65

fig. 3.65

```
100 COLOR 1,15,15
110 SCREEN 3:OPEN"GRP:" AS #1
120 FOR C=0 TO 255 STEP 4
130 FOR L=0 TO 191 STEP 4
140 X=RND(-TIME)
150 CR=INT(X*13)+2
160 PSET(C,L),CR
170 NEXT L,C
180 LINE(76,78)-(172,116),15,BF
190 PRESET (84,86):PRINT #1,"MSX"
200 GOTO 200
```

Enquanto ele roda (é meio demoradinho!) vamos analisá-lo.

Na linha 100 definimos o preto (1) como cor de escrita e o branco (15) como cor de fundo e borda.

Na linha 110 chamamos a SCREEN 3 e abrimos um arquivo na tela para podermos escrever (como fizemos com a SCREEN 2). Nas linhas 120 e 130 abrimos dois laços para varrer todas as linhas e colunas (note o STEP 4 pois, nessa tela, não adianta ir de 1 em 1).

Na linha 140 começa uma coisa esquisita mas que vale a pena aprender, especialmente se você trabalha com simulações estatísticas.

A função RND gera números ao acaso numa sequencia sempre diferente a cada valor negativo de seu argumento. Como a variável TIME armazena o tempo em que o Z80 (microprocessador) ficou trabalhando desde que o micro foi ligado, a sequência será bem aleatória.

Estes números estão sempre entre 0 e 1 como mostra o exemplo da figura 3.66

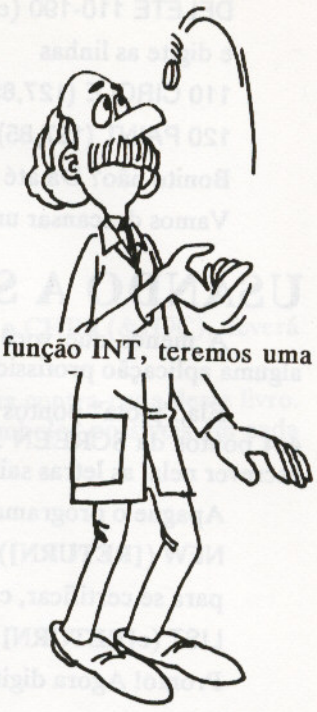
fig. 3.66

```
TIME=26579  RND(-TIME)=  .68890820420821
TIME=26581  RND(-TIME)=  .49928820420821
TIME=26583  RND(-TIME)=  .30966820420821
TIME=26585  RND(-TIME)=  .12004820420821
TIME=26587  RND(-TIME)=  .93042820420821
TIME=26589  RND(-TIME)=  .74080820420821
TIME=26591  RND(-TIME)=  .55118820420821
TIME=26593  RND(-TIME)=  .36156820420821
TIME=26595  RND(-TIME)=  .17194820420821
TIME=26597  RND(-TIME)=  .98232820420821
TIME=26599  RND(-TIME)=  .79270820420821
TIME=26601  RND(-TIME)=  .60308820420821
```

Vamos multiplicar estes números por 13 e obteremos uma sequência como a da figura 3.67

fig.3.67

```
13*RND(-TIME)= 8.955806654706 7
13*RND(-TIME)= 6.490746654706 7
13*RND(-TIME)= 4.025686654706 7
13*RND(-TIME)= 1.560626654706 7
13*RND(-TIME)=12.095566654706 0
13*RND(-TIME)= 9.630506654706 7
13*RND(-TIME)= 7.165446654706 7
13*RND(-TIME)= 4.700386654706 7
13*RND(-TIME)= 2.235326654706 7
13*RND(-TIME)=12.770266654706 0
13*RND(-TIME)=10.305206654706 0
13*RND(-TIME)= 7.840146654706 7
13*RND(-TIME)= 5.375086654706 7
```



Se pegarmos apenas a parte inteira desses números, usando a função INT, teremos uma sequência do tipo mostrado na fig. 3.68

fig. 3.68

```
INT(13*RND(-TIME))= 8
INT(13*RND(-TIME))= 6
INT(13*RND(-TIME))= 4
INT(13*RND(-TIME))= 1
INT(13*RND(-TIME))=12
INT(13*RND(-TIME))= 9
INT(13*RND(-TIME))= 7
INT(13*RND(-TIME))= 4
INT(13*RND(-TIME))= 2
INT(13*RND(-TIME))=12
INT(13*RND(-TIME))=10
INT(13*RND(-TIME))= 7
```

Se somarmos 2 a estes resultados, o valor assim gerado estará sempre entre 2 (códigos do verde) e 14 (código do cinza) estando excluidos o 0 (transparente), 1 (preto) e 15 (branco).

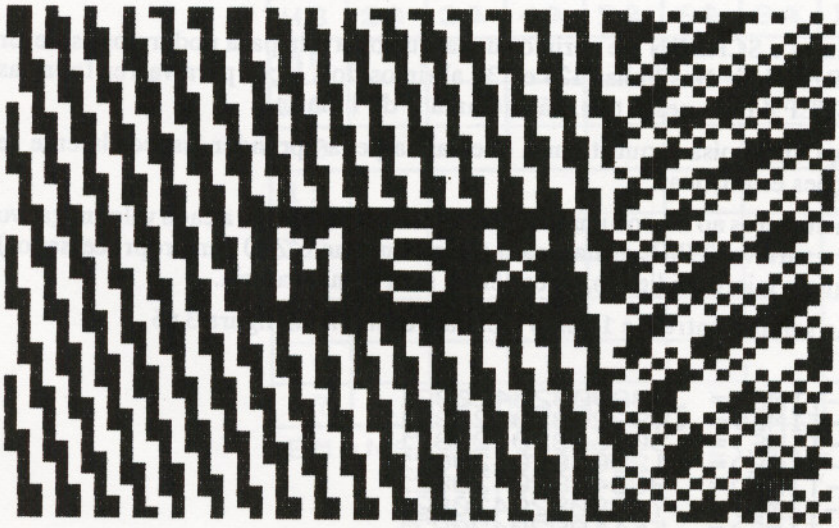
É isso que faz a linha 150.

Na linha 160, "plotamos" pontos na coluna C, linha L, com cor CR, cada vez diferente e ao acaso.

Na linha 170 fechamos os laços (sempre na sequência: "primeiro a fechar...último aberto") e na linha 180 fazemos um retângulo branco (cor 15).

Na linha 190 escrevemos "MSX" dentro do retângulo e a 200 congela a imagem, que deve ser parecida com a da fig. 3.69 .

fig. 3.69



Já imaginou que telas de abertura você pode gerar para seus vídeos? Para gravá-las basta conectar a saída de vídeo do MSX com o "VIDEO-IN" do seu vídeo cassete. Se quiser gravar com som, basta fazer seu MSX tocar uma música (ou gerar um ruído) e conectar a saída de audio com o "AUDIO-IN" do seu vídeo-cassete ou camcorder.

Agora é só gravar!

Você dirá: "Calma! e a música ? Como é que eu faço?"

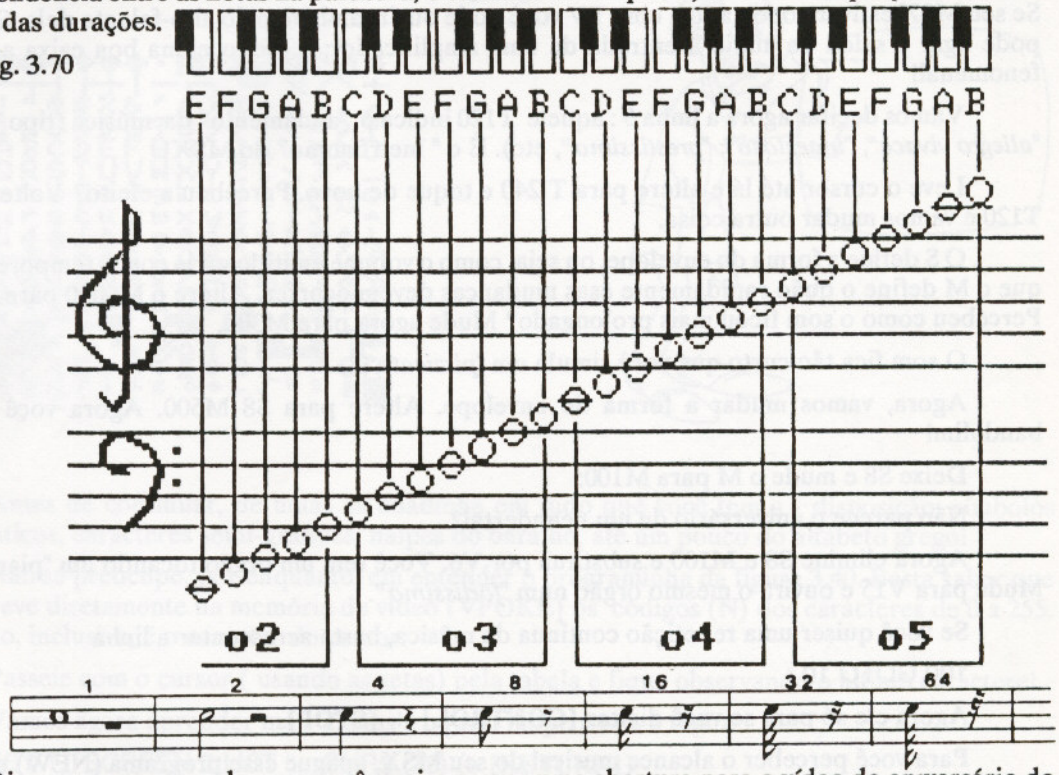
Fácil, é só ler ítem seguinte!

A música no MSX

Fazer música no MSX é bem fácil, mesmo que você não tenha noções muito avançadas de teoria musical. No BASIC MSX existe o comando PLAY que toca a que você quiser e como você quiser, em função dos códigos que ele usa como argumento.

As notas musicais são dadas pela notação Anglo-Germânica: o Do,Re,Mi,Fa,Sol,La,Si são representados por C,D,E,F,G,A,B respectivamente, as pausas por R, a duração por um número após a nota (se seguido por ponto ".", a duração é acrescida da metade). Veja na figura 3.70 a correspondência entre as notas na partitura, o teclados de um piano, as oitavas a que pertecem é os códigos das durações.

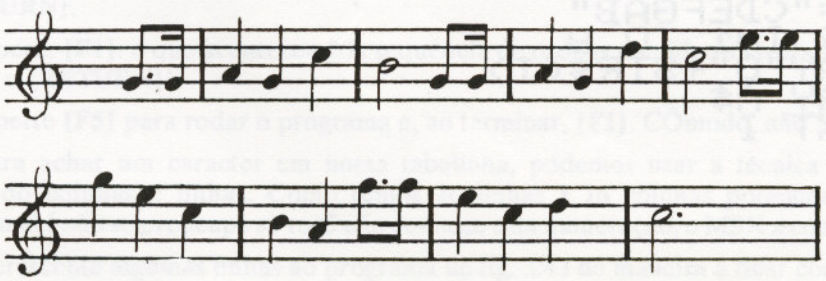
fig. 3.70



Digamos, por exemplo, que você queira gravar uma abertura para o vídeo do aniversário do seu filho. Você já desenhou as velinhas na SCREEN 2, o nome dele, a data, mas só falta a música de fundo.

Você encontrou a partitura do "parabéns a você" (fig. 3.71).

fig. 3.71



Olhando para as figuras 3.70 e 3.71, você pode digitar o programa listado na fig. 3.72.

fig. 3.72

```
5    PLAY"S0 M6000 O4 T120"
10   PLAY"G8.G16"
20   PLAY"A4G4O5C4"
30   PLAY"O4B2G8.G16"
40   PLAY"A4G4O5D4"
50   PLAY"C2E8.E16"
60   PLAY"G4E4C4"
70   PLAY"O4B4A4O5F8.F16"
80   PLAY"E4C4D4"
90   PLAY"C2."
```

Nesta digitação tome muito cuidado para não confundir o "O" (de oitava) com o "0" (zero) e B (nota SI) com o 8 (de duração). Por enquanto, não se preocupe em entender a linha 5.

Rode o programa e ouça (se necessário regule o volume do alto-falante na traseira da CPU). Se seu MSX estiver conectado a uma TV você pode ouvir também pelo alto-falante dela. Se quiser, pode ligar a saída de audio à entrada de uma amplificador: o efeito numa boa caixa acústica é fenomenal!

Vamos decifar agora a linha 5 : aquele T120 indica o "andamento" da música (tipo *"adagio"*, *"allegro vivace"*, *"maestoso"*, *"prestissimo"*, etc). É o "metrônomo" do MSX!

Leve o cursor até lá e altere para T 240 e toque de novo. Percebeu a efeito? Volte ao valor T120 e vamos mudar outra coisa.

O S define a forma do envelope, ou seja, como o volume emitido varia com o tempo, enquanto que o M define o quão rapidamente esas mudanças devem ocorrer. Altere o M6000 para M12000. Percebeu como o som ficou mais prolongado? Mude agora para M300.

O som fica tão curto que você simula um *"pizzicato"* !

Agora, vamos mudar a forma do envelope. Altere para S8 M500. Agora voçê tem um bandolim!

Deixe S8 e mude o M para M100:

Não parece o aniversário de um neandertal?

Agora elimine S8 e M100 e substitua por V6. Você tem um orgão tocando um "pianissimo". Mude para V15 e ouvirá o mesmo orgão num *"fortissimo"*.

Se você quiser uma repetição contínua da música, basta acrescentar a linha

100 GOTO 10

Agora ele só para se você digitar **[CONTROL] + [STOP]**.

Para você perceber o alcance musical do seu MSX, apague esse programa (NEW) e digite o programinha listado na figura 3.73.

fig. 3.73

```
100  PLAY"V15 T120"
110  A$="CDEFGAB"
120  FOR I=1 TO 8
130  PLAY"O"+STR$(I)
140  PLAY A$
150  NEXT I
```

Mesmo com o som do MSX saindo por uma caixa acústica com um bom "Tweeter", existem agudos que quase fogem do alcance do ouvido humano!

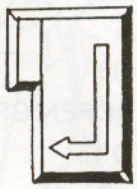
Imagine, agora, que no seu vídeo de viagem existe uma cena de despedida no aereoporto.

Você já desenhou um jato partindo na SCREEN 2 e falta a trilha sonora.

Experimente o programa da figura 3.74 (apagando, obviamente, o anterior).

Antes, porém um truque! Digite:

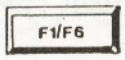
KEY1,"SOUND" (e **[RETURN]**)
KEY2,"FOR L=" (e **[RETURN]**)
KEY3,"NEXT L" (e **[RETURN]**)
AUTO 100 (e **[RETURN]**)

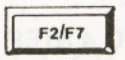


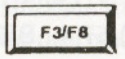
Agora basta apertar **[F1]**. completar com 7,28 (e **[RETURN]**). A linha 110 já aparece esperando um novo **[F1]** (SOUND), etc. nas linhas 190, 200, 220, 240 e 260 você vai usar **[F2]** (FOR L=) e **[F3]** NEXT L- e, ao chegar na linha 290 basta digitar **[CONTROL]** + **[STOP]** para interromper o processo de numeração automática. Limpe a tela, liste o programa, confira tudo e rode-o.

fig. 3.74

```
100 SOUND    7,28
110 SOUND    1,0
120 SOUND    3,0
130 SOUND    6,0
140 SOUND    0,100
150 SOUND    8,13
160 SOUND    9,0
170 SOUND   10,15
180 SOUND   12,255
190 FOR L=1 TO 2000:NEXT L
200 FOR L=100 TO 30 STEP -.04
210 SOUND    0,L
220 NEXT L
230 SOUND    8,0
240 FOR L=2 TO 31 STEP .01
250 SOUND    6,L
260 NEXT L
270 SOUND   10,16
280 SOUND   13,0
```







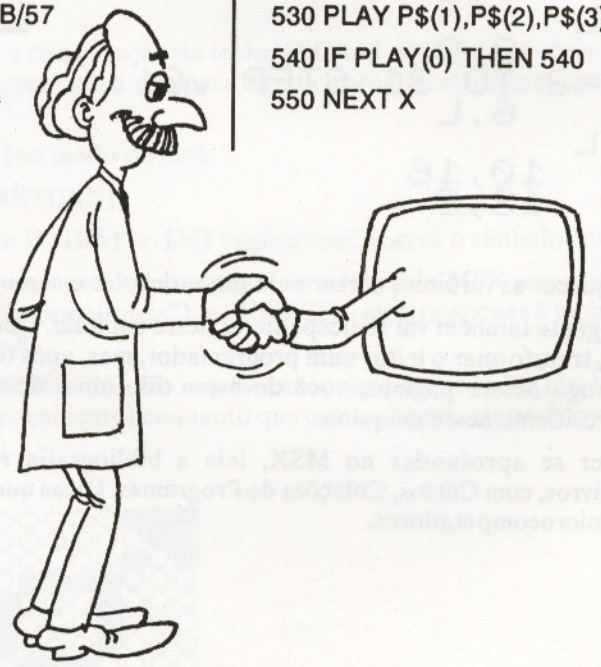
Seu jato vai aquecer as turbinas, correr pela pista, decolar e sumir entre as nuvens!

E, por aqui, a gente também vai se despedindo deste capítulo. Como já foi dito, não é nossa intenção , neste livro, transformar o leitor num programador, mas, após ter trabalhado em conjunto com seu MSX ao longo desses páginas, você deve ter tido uma idéia do imenso potencial da Linguagem BASIC residente neste máquina.

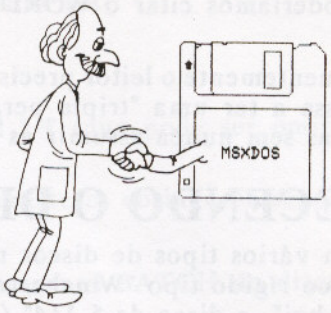
Se você quiser se aprofundar no MSX, leia a bibliografia recomendada onde estão relacionados outros livros, com Cursos, Coleções de Programas, Dicas que podem levá-lo a se tornar um apaixonado por microcomputadores.

Se o seu EXPERT DD PLUS estiver conectado a uma TV a cores, digite o programa a seguir e rode-o, para ajustar as cores, a sintonia e o volume de som. Se você quiser, altere-o para produzir uma imagem de abertura em suas fitas de vídeo.

```
100 REM AJUSTE DA TV
110 COLOR 4,7,5
120 W=128:U=88
130 SCREEN 2,1:OPEN"GRP:"AS#1
140 FOR F=0 TO 15
150 G=F*16:H=G+15
160 PRESET(G+5*(G>0),168)
170 PRINT #1,F
180 LINE(G,176)-(H,191),F,BF
190 NEXT F
200 FOR F=0 TO 256 STEP 16
210 LINE(F,0)-(F,192)
220 NEXT F
230 FOR F=0 TO 192 STEP 16
240 LINE (0,F)-(255,F)
250 NEXT F
260 CIRCLE(W,U),58,7
270 PAINT(W,U),7
280 FOR B=57 TO 4 STEP -8
290 CIRCLE(W,U),57,4,,,57/B
300 CIRCLE(W,U),57,4,,,B/57
310 NEXT B
320 LINE(W,30)-(W,60),4
330 LINE(71,U)-(184,U),4
340 CIRCLE(W,U),20,7
350 PAINT(W,U),7
360 B$=SPACE$(6)+"ALEPH"
370 S$="S0M6000"
380 PLAY S$,S$,S$
390 FOR X=1 TO 89
400 X(1)=X:W=108:U=84
410 X(2)=X+4:Z=150
420 X(3)=X+7:Q=92
430 LINE(W,U)-(Z,Q),7,BF
440 PRESET(110,84)
450 A$=RIGHT$(B$,5)
460 PRINT #1,A$
470 R$=RIGHT$(B$,1)
480 L$=LEFT$(B$,10)
490 B$=R$+L$
500 FOR I=1 TO 3
510 P$(I)="N"+STR$(X(I))
520 NEXT I
530 PLAY P$(1),P$(2),P$(3)
540 IF PLAY(0) THEN 540
550 NEXT X
```



LIGANDO O EXPERT COM DISCO 4



AS "3 FACES" DO MSX

Há alguns anos fez muito sucesso um romance intitulado "As 3 faces de Eva" no qual um psiquiatra relata o comportamento de um caso de esquizofrenia, no qual uma mulher assumia 3 personalidades completamente diferentes.

O seu MSX também tem 3 personalidades, apresentando uma "ESQUIZOFRENIA" extremamente útil. Vamos ver essas "3 Faces" do seu MSX.

1) Quando você liga seu Expert DD PLUS mantendo a tecla **[SHIFT]** apertada, a interface de disco (interna) é ignorada e ele passa a se comportar como um micro que não dispõe deste periférico. Seu BASIC residente funciona numa versão mais simples e você pode ler e gravar programas apenas a partir de uma fita cassete (e cartuchos). Foi este BASIC que nós usamos no capítulo 3, e que deixa 28815 bytes livres.

2) Se você ligar seu Expert DD PLUS sem pressionar nenhuma tecla, e sem colocar o Disco-Mestre no acionador, aparecerá uma mensagem pedindo a data atual. Digitando-se a data ou simplesmente **[RETURN]**, será apresentada uma mensagem informando que agora temos 23440 bytes livres e o micro assume o DISK-BASIC, uma versão mais completa do BASIC que obtemos no caso 1. Neste caso, alguns comandos do BASIC original tem sua sintaxe ligeiramente alteradas e outros são implementados.

Este BASIC será comentado com maiores detalhes no capítulo 5.

A quantidade de bytes livres (23440) diminui em relação ao BASIC original porque a interface de disco reserva, na memória do micro, 2 regiões de memória (denominadas "BUFFERS") que permitem o gerenciamento dos arquivos do disco.

Porque duas regiões? Porque, apesar de você ter apenas um acionador físico, ele se comporta como se estivessem instalados 2 acionadores lógicos (A e B) .

Esta "Sub-Esquizofrenia" permite copiar arquivos de um disco, para outro. O disco original é tratado, por exemplo, como A e o destino como B. Toda vez que houver necessidade de uma troca de discos, o próprio MSX vai solicitá-la com uma mensagem adequada na tela.

Se você ligar seu micro mantendo a tecla **[CONTROL]** pressionada, apenas um "BUFFER" é reservado, deixando 24998 bytes livres.

Você passa a ter mais memória disponível, às vezes indispensável para carregar e usar certos softwares mas, em compensação, para cada acionador físico você passa a ter apenas um drive lógico.

3) Se você ligar seu Expert DD PLUS sem pressionar o **[SHIFT]**, mantendo ou não a tecla **[CONTROL]** pressionada (para reservar 1 ou 2 Buffers) e tiver o Disco-Mestre inserido no acionador, será carregando o MSXDOS.

O MSXDOS, ao ser carregado no micro, o transfigura completamente. Quando o MSXDOS está no comando do micro sua organização interna muda de maneira drástica e seus recursos passam a ser completamente diferentes. Esse sistema é compatível, a nível de comandos, com o CP/M e permite a utilização de uma vasta biblioteca de programas desenvolvidos antes da existência do padrão MSX. Como exemplo desses

programas poderíamos citar o WORDSTAR, o dBASE, o SUPERCALC, o MBASIC, etc.

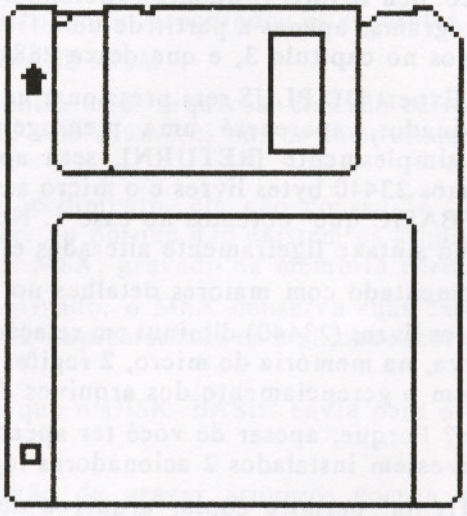
Consequentemente o leitor precisa ter em mente que, dependendo da inicialização, seu MSX passa a ter uma "tripla personalidade", podendo passar facilmente de uma para outra mas sem nunca assumir as três simultaneamente.

CONHECENDO O DISCO DE 3 1/2"

Existem vários tipos de discos magnéticos para computadores, que vão desde o caríssimo disco rígido tipo "Winchester", capaz de armazenar milhões de informações, até o "pé-de-boi", o disco de 5 1/4" (5,25 polegadas de diâmetro), já em uso a vários anos.

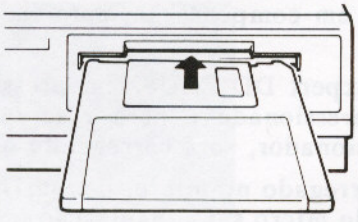
O padrão MSX, ao ser estabelecido, escolheu o tipo de disco que dá a melhor razão custo/benefício em relação à confiabilidade e a quantidade de informação armazenadas. Trata-se do disco de 3 1/2", fruto de uma tecnologia bem mais recente e sofisticada. Ele é acondicionado numa embalagem de plástico rígido e nenhuma parte do material magnético fica exposta quando o disco está fora de uso.

fig. 4.1 - O disco de 3 1/ 2" visto de cima.



Este disco deve sempre ser introduzido no acionador com a face superior voltada para cima. Os "viciados" em discos de 5 1/4", acostumados a virar e desvirar o disco feito panqueca, devem se acostumar a obedecer esta regra básica.

fig. 4.2 - Inserção correta do disco no acionador.



Se você tiver curiosidade, pegue um disco virgem (sem informações gravadas!) e desloque cuidadosamente a placa metálica protetora para a esquerda. Cuidado para evitar poeira e para não tocar o interior com o dedo!

Você verá que a janela da parte metálica, ao deslizar para a esquerda, expõe uma

porção de material magnético em AMBAS AS FACES do disco.

Existem acionadores de face simples onde você tem uma cabeça que grava e lê informações apenas numa das faces do disco, Já no modelo face dupla, existem duas cabeças que gravam e lêem em ambas as faces do disco.

Por isso, existem dois tipos de discos de 3 1/2" (que, entretando, têm o memos aspecto fsico). Os discos de face simples (Single Sided ou SS) têm apenas uma das faces garantida e os discos de face dupla (Double Sided ou DS) têm ambas as faces garantidas.

A grande diferença entre os dois sistemas esta na quantidaade de informações que você pode armazenar num único disco. No de face simples o usuário pode gravar 360 KBytes de informação, enquanto que no de face dupla este número sobe para 720 KBytes.

Consequentemente, como você tem um acionador com duas cabeças, use discos de face dupla (DS) para tirar o máximo de proveito de seu Expert DD PLUS.

CUIDADOS COM O DISCO E COM O DRIVE

Junto a seu Expert DD PLUS você recebeu um disco já gravado. Antes de manipulá-lo, lembre-se que seu disco tem vários inimigos, alguns óbvios e outros invisíveis. Entre eles podemos relacionar:

1 - POEIRA - Não trabalhe em ambiente empoeirado. Procure instalar se computador num ambiente onde o ar esteja razoavelmente limpo. A poeira danifica tanto o disco quanto o acionador.

2 - FUMAÇA - A própria fumaça de um cigarro é inimigo mortal dos discos magnéticos. Não fume perto do computador. Isso fará muito mal aos discos e mais ainda à sua saúde.

3 - CALOR - Não exponha os discos ao calor e ao Sol!

4 - UMIDADE - Evite ambientes úmidos e jamais molhe os discos (ou espirre sobre eles).

5 - PULSOS NA REDE ELÉTRICA - Mesmo que a rede elétrica, à qual seu micro e seu acionador estão conectados, não sofra variações grandes de tensão, existe um risco de dano grave. Se a tomada estiver em paralelo com um circuito de grande indutância (como por exemplo, reatores de lâmpadas fluorescentes), ao desligarmos ou ligarmos outros equipamentos da rede produzimos picos de tensão que podem danificar o micro e o acionador. Para evitar esse inconveniente é bom que o micro esteja ligado a um transformador regulador de voltagem automático, para filtrar esses picos.

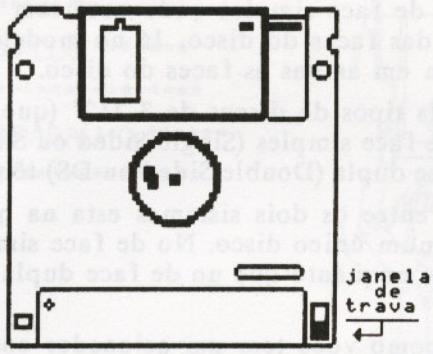
6 - CAMPOS MAGNÉTICOS - Como a gravação das informações no disco se dá por processo magnético, qualquer campo intenso pode danificar o conteúdo do disco. Os campos magnéticos são gerados por ímãs que, às vezes, se encontram em pontas de tesouras ou chaves-de-fenda. Além disso, campos magnéticos são gerados quando uma corrente elétrica passa por um fio condutor. Dentro de aparelhos eletro-eletrônicos, encontram-se transformadores e bobinas (ou também motores elétricos), que irradiam campos capazes de tornarem os dados dos discos ilegíveis. Às vezes uma viagem de metrô ou ônibus elétrico causa danos ao conteúdo de um disquete.

PRIMEIRAS OPERAÇÕES COM DISCO

Antes de começarmos a operar com o disco, existe uma precaução básica extremamente útil: seu disco pode ser "travado" de maneira a não aceitar gravação de dados adicionais. Dessa forma, mesmo que você cometa algum erro na digitação de algum comando, isso realmente não resultará em danos no disco.

Vire seu disco de maneira a olhar sua face inferior (fig. 4.3)

fig. 4.3 - A "trava" contra gravação.



No canto inferior direito você deve ver uma janela de trava que deve estar ABERTA. Estando aberta, o disco pode ser lido, mas nada será gravado nele. Se, por acaso, ela estiver fechada, deslize-a para baixo com a eventual ajuda de um clip de maneira a abrí-la.

Se você for um viciado em disco de 5 1/4", cuidado! A filosofia é exatamente oposta. No disco de 3 1/2" a trava contra gravação é com janela ABERTA!

O Disco-Mestre que você recebeu junto com seu Expert DD PLUS deve ficar sempre travado (janela aberta)!

Vamos finalmente começar as operações com disco.

1 - Certifique-se que o micro esteja desligado.

2 - Conecte todos os periféricos adicionais que você for usar (impressora, modem, etc.). Se você for usar um cartão de 80 colunas, não o ligue por enquanto para poder acompanhar melhor a explicação a seguir.

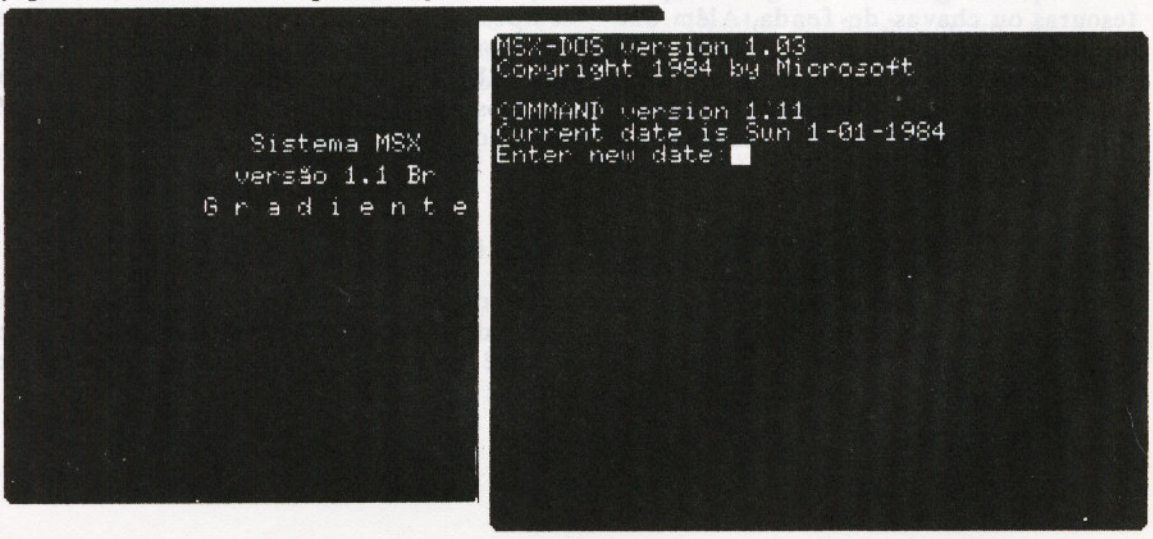
3 - Ligue o vídeo.

4 - Ligue os outros periféricos.

5 - Ligue o micro-computador, sem pressionar nenhuma tecla.

6 - Coloque o disco do sistema (o disco já gravado que vem junto com o equipamento), TRAVADO (janela ABERTA) na frente do acionador com a face superior para cima (fig. 4.2) e empurre-o delicadamente para dentro até sentir que está retido. Dê mais um empurrãozinho, abaixando um pouco o disco até fazer um leve "clic" no mecanismo do acionador. Na tela deverá aparecer a mensagem da Gradiente e a seguir a mensagem do MSXDOS indicando que ele assumiu o comando das operações (fig. 4.4).

fig. 4.4 - As telas de apresentação.



A luz no painel frontal do acionador deverá acender durante um breve intervalo, indicando que o disco está sendo acessado.

Se aparecer a mensagem:

Enter date (D-M-Y):

Significa que o disco não foi bem introduzido e o micro não conseguiu carregar o MSXDOS a partir dele, "escapando" para o DISK BASIC. Pressione a tecla "RESET" localizada na tampa traseira.

Se a tela da figura 4.4 não aparecer, desligue tudo na ordem inversa, verifique as conexões de todos os cabos em seus equipamentos, e tente ligar o equipamento novamente.

Se o problema persistir, seu Disco-Mestre está com problemas. Contacte a Gradiente para substituí-lo. Você não poderá acompanhar o resto deste capítulo mas não fique frustado. Providencie um disquete virgem (DS) e pule para o capítulo 5 para ir aprendendo o DISK BASIC enquanto está sendo providenciada a substituição do seu Disco-Mestre.

Tendo a tela da figura 4.4 no vídeo de seu micro, aperte **[RETURN]** (não se preocupe em atualizar a data por enquanto).

Ao fazer isso, você deverá ver o sinal de PRONTO do MSXDOS seguido do cursor, mostrando que o sistema está esperando a digitação de um comando:

A>

Vamos aprender nosso primeiro comando do MSXDOS: o MODE. Ele é parecido com o WIDTH do BASIC e permite determinar o número de colunas que a tela deve apresentar.

Como em alguns televisores a largura de 40 colunas não permite a visão das colunas laterais digite:

A>mode 39

1 - Não se preocupe com maiúsculas , pois o MSXDOS também reconhece minúsculas.

2 - Não esqueça de deixar um espaço entre o comando MODE e o parâmetro 39. Se você errou algo, é só usar a seta para esquerda (que no MSXDOS faz o papel de BACK SPACE, ou seja, volta apagando).

3 - Após digitar o comando, para que ele seja executado, pressione a tecla **[RETURN]**.

A tela deverá ser apagada e o sinal de PRONTO com o cursor deve se posicionar no canto superior esquerdo.

Se você quiser brincar um pouco, digite:

A>MODE 20

Veja como o cursor se posicionou, mas depois, volte ao MODE 39. Se você é uma pessoa curiosa deve ter experimentado o MODE 1.

Vamos agora ver um dos comandos mais importantes: o DIR. Este comando, ao ser utilizado, faz com que o micro leia o diretório do disco, uma espécie de índice de seu conteúdo. Digite então:

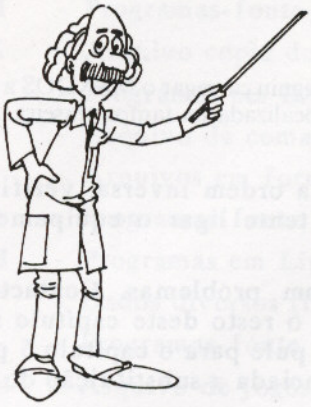
A>dir

Não esqueça de pressionar **[RETURN]** após cada comando! O MSXDOS só os executará após você pressionar essa tecla.

A luz vermelha no painel frontal do acionador deverá acender, mostrando que o disco esta sendo acessado pelo micro. (pressionar o botão de ejeção, neste momento, poderá ter um efeito destruidor no conteúdo do disco!).

A seguir será listado na tela o conteúdo do disco (fig 4.5).

fig 4.5 –Exemplo de um diretório.



```
MSXDOS   SYS     2432  1-01-84
COMMAND  COM     6656  2-09-85  10:10p
BASIC6   TXT    11264  1-06-89   5:56p
SUMARIO  TXT     6144 26-05-89  10:45a
CAP3-1   TXT    19968 24-05-89   2:28p
CAP3-2   TXT    19840 24-05-89   3:09p
CAP5-2   TXT     5248 24-05-89  10:00a
CAP5-1   TXT    15616 24-05-89  11:22a
CAP4-2   TXT    16000 24-05-89   5:38p
CAP4-1   TXT    19968 24-05-89   3:47p
CAP2-1   TXT    14976 24-05-89  12:13p
CAP2-2   TXT     8704 24-05-89  12:41p
CAP1     TXT    10240 30-05-89   4:41p
DDPLUS   TXT     2816 24-04-89  12:23p
CAP7     TXT     8448 25-04-86
BASIC1   TXT    19584  1-06-89  10:24a
BASIC5   TXT    15616  1-06-89   4:52p
BASIC2   TXT    18176  1-06-89  12:24p
BASIC3   TXT    18432  1-06-89  12:18p
BASIC4   TXT    19328  1-06-89   3:58p
INTROD   TXT     3712  1-05-89
SUM1     TXT     6144 26-05-89  10:35a
      22 files      83968 bytes free
```

Na primeira coluna você tem o nome dos arquivos gravados no disco. Na segunda você tem um código de três letras que indica o tipo de arquivo. Maiores detalhes sobre estas três letras (extensão) ainda serão vistas neste capítulo. Na terceira coluna é fornecido o número de bytes de cada arquivo. Por exemplo, o arquivo

MSXDOS.SYS

ocupa 2560 bytes.

Na quarta coluna temos a data em que o arquivo foi gravado e na quinta coluna, o horário. O horário só é disponível em micros com relógio interno (somente algumas interfaces especiais).

Na última linha do diretório é relatado quantos arquivos estão presentes, e quantos bytes livres se encontram no disco.

Quando o diretório é muito longo, às vezes torna-se difícil localizar um arquivo. Para isso você pode lista-lo da seguinte forma (suponha que é desejado o arquivo MSXDOS.SYS):

A>DIR MSXDOS.SYS (e **[RETURN]**!)

Você verá que apenas um arquivo será listado na tela:

MSXDOS SYS 2560 23-08-85

Se você não lembrar o nome do arquivo que quer listar, mas souber que ele é seguido do código COM, por exemplo, bastará digitar:

A>DIR *.COM (e **[RETURN]**!)

Você obterá a listagem de todos os arquivos que têm o código (ou extensão) COM em seu final. O asterisco (*) fez o papel de "coringa": é como se você tivesse dito ao computador "me liste "alguma coisa" seguida de COM".

BACKUP DE DISCO

Vamos aprender agora uma coisa importantíssima: o BACKUP de um disco, ou seja, sua cópia fiel de maneira a não perder informações se algum acidente acontecer com o disco original.

Você já deve ter ouvido falar na LEI DE MURPHY: "Se alguma coisa tem a mais remota chance de dar errado, certamente dará errado!"

Apesar dela ter sido concebida com intuito humorístico, sua validade no mundo da informática é confirmada por milhões de usuários levados às raias do desespero por faltas de energia elétrica, sobrinhos curiosos e distrações fatais.

Por isso o BACKUP de discos que contenham informações importantes é

fundamental.

Inicialmente vamos partir do pressuposto que você não tenha interfaces e acionadores adicionais conectados ao seu Expert DD PLUS.

Coloque o Disco-Mestre no acionador certificando-se de que ele esteja com a janelinha ABERTA (protegido). Ligue o sistema como descrito, de maneira a obter o:

A>

Neste momento o micro está sob o controle do MSXDOS. Se necessário, comande:

A>mode 39 (e **[RETURN]**!)

para visualizar a primeira coluna (somente se ela estiver oculta.)

Pegue agora um disco virgem (não use o Disco-Mestre para essa finalidade). Vamos prepará-lo para que ele possa receber informações do seu MSX.

Retire o Disco-Mestre do acionador (pressionando o botão de ejeção), e destrave o disco virgem (lembre-se, que o Disco-Mestre já esta formatado, e se for re-formatado, perdaderá todo seu conteúdo) FECHANDO a janela de trava. Coloque-o no acionador até obter um "clic" do mecanismo.

Digite agora:

A>FORMAT

Aparecerá a pergunta:

Drive name? (A, B)

E agora você responde com a tecla "A".

A seguir, aprecerá um "menu" de 4 opções (fig. 4.6).

fig 4.6 - Menu de formatação.

1 --> 1 side

2 --> 2 sides

3 --> 1 side, double track

4 --> 2 sides, double track

Como seu acionador é de FACE DUPLA, escolha a opção 4.

Se o disco virgem for de face simples, use a opção 3.

Após digitar a opção adequada, aparecerá a mensagem:

Strike a key when ready

Verifique mais uma vez se o disco virgem está destravado e bem colocado no acionador (lembre-se da LEI DE MURPHY).

Pressione a barra de espaços e você verá a luz frontal do acionador se acender, indicando que o computador está operando o disco (no caso ele está gravando informações!).

Na realidade o disco está recebendo informações de uso INTERNO (números de trilhas e setores, comprimento do setor, etc.), de maneira a, posteriormente, ser reconhecido como disco MSX. Esta é a finalidade da formatação. Isto significa que, nem todos os disquetes gravados por outros tipos de computadores, são identificados pelo MSX (por isso que os discos do APPLE, por exemplo, não podem ser lidos pelo seu Expert DD PLUS). No entanto, a formatação do MSX, é identica à do IBM/PC, e esta característica traz a vantagem de podermos trocar informações entre essas linhas!

Quando a luz do acionador se apagar, aparecerá a mensagem:

Format complete

Na tela, aparece ainda, o sinal de PRONTO do MSXDOS (A>).

Seu disco esta agora "limpo" (não possui dados gravados), e está em condições de

armazenar as informações do MSX (ou do PC!).

Retire-o, então, do acionador, e coloque o Disco-Mestre no acionador (considerando todos cuidados necessários para manipulá-lo), e para ver quais arquivos estão presentes, digite, novamente:

A>DIR

Se o disco não estiver bem colocado, aparecerá a mensagem:

Not ready error reading drive A

Abort, Retry, Ignore?

Neste caso recoloque-o corretamente e tecle "R" (de Retry). ("A" simplesmente abandona o comando e "I" não terá efeito neste comando.)

Após a listagem do diretório, você obtém novamente o "PRONTO", e em seguida digite (respeitando os espaços e a sintaxe):

COPY A:*.* B:/V

Antes de pressionar a tecla **[RETURN]**, vamos entender o tipo de ordem dada ao computador. O que acabamos de comandar pode ser interpretado como:

COPY A: –Copie do drive A

. –Os arquivos com qualquer nome e extensão (as três letras).Veja que o nome e a extensão são separados por ponto "."

B: –Para o drive B

/V –Verifique se a cópia foi perfeita.

A esta altura você deve estar se perguntando: "espera aí: eu só tenho um drive!?!"

Não se preocupe: seu Expert sabe disso! E assim ele usará apenas um acionador, para fazer a cópia, anunciando alternadamente a necessidade da troca dos discos, até que ele termine toda a cópia! Ou seja, é necessário que o usuário faça a troca dos disquetes toda vez que o computador anunciá-la. Após a troca basta apertar uma tecla que, à partir de então, o micro trata o novo disco como pertencente ao outro drive.

Agora digite **[RETURN]**, e na tela você verá os arquivos que estão sendo lidos e carregados na memória do micro. A seguir aparecerá a mensagem:

Insert diskette for drive B:

and strike a key when ready

Isso significa que a interface deixou de considerar o acionador como drive A, e passou a chamá-lo de drive B.

Troque os disquetes, e tecle espaço. A partir de então o drive passa a gravar o disquete B:. Se a cópia estiver encerrada será impressa a mensagem:

N files copied (N = número de arquivos copiados)

Novamente, o sinal de "PRONTO será apresentado.

Porém se a operação de cópia não estiver completa, então serão solicitadas novas trocas.

Agora você tem um BACKUP do seu Disco-Mestre.

Guarde o seu Disco-Mestre num lugar seguro (sem agentes que possam destrui-lo). Não o use mais, a menos em casos inevitáveis. Passe a usar o novo Disco-Mestre que você acaba de gerar.

É aconselhável que o Disco-Mestre que foi fornecido com o micro, não receba nenhuma gravação. A sua existência se justifica apenas para reter o sistema MSXDOS.SYS e o COMMAND.COM (este sistema possibilita acessar programas como o dBASE II, SUPERCALC, TURBO PASCAL, WORDSTAR, FORTRAN IV, etc.).

Para efeito de aprendizagem monte outro disco, onde irá gravar seus arquivos.

No momento você deve ter 3 discos, e para que você os identifique facilmente cole etiquetas com seus respectivos nomes.

Os 3 discos são:

DISCO MESTRE ORIGINAL (TRAVADO E GUARDADO)
DISCO MESTRE COPIADO (TRAVADO E EM USO)
DISCO DE TRABALHO (DESTRAVADO E EM USO)

Aprenda a se organizar e muito tempo e dissabores serão poupados!

AS "TRANSFUSÕES" ENTRE 5 1/4" E 3 1/2"

Quando queremos copiar arquivos de um disco para o outro do mesmo tipo, já vimos que não há problemas. Aliás, já fizemos isto no início desse capítulo ao gerarmos o Disco-Mestre de uso e o disco de trabalho (BACKUP).

Porém se quisermos passar um arquivo de um disco de 5 1/4" para outro de 3 1/2" (ou vice-versa), é indispensável que os dois acionadores estejam conectados simultâneamente no micro.

É necessário, então, se ter uma interface externa (tipo cartucho), conectada a um dos SLOTS do Expert DD PLUS, comandando um acionador de 5 1/4", que vai assumir os drives A/B, devido, à prioridade dos SLOTS A e B sobre o SLOT interno onde se encontra a interface do disco (ou seja o drive de 3 1/2 corresponderá a C/D).

Assim fazendo um dos drives (por exemplo o de 5 1/4") vai operar como A (ou B) e o outro (no caso de 3 1/2") funcionará como C (ou D).

Com o DOS carregado podemos, por exemplo, comandar:

```
A>COPY A:TESTE.TXT C:RASCUNHO.ASC
```

Desta forma, o arquivo TESTE.TXT será lido no drive A (de 5 1/4") e gravado com o nome RASCUNHO.ASC no drive C (de 3 1/2").

Lembre-se que se o sistema for inicializado com a tecla CONTROL pressionada, a cada drive físico corresponderá um drive lógico! (isto é, A corresponde ao 5 1/4" e B ao 3 1/2").

ARQUIVOS EM DISCO

Insira novamente o disco com o MSXDOS no drive e reinicialize o sistema. Vamos agora consultar o "índice" do disco, isto é, o diretório.

Como já vimos, o comando MSXDOS que permite observar o conteúdo do diretório é o DIR. Para usá-lo, basta comandar:

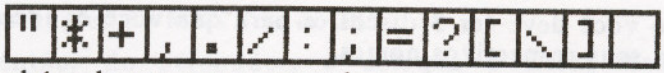
```
A>DIR
```

Ao fazer isso, o MSXDOS apresentará na tela a relação de todos os "arquivos" contidos no disco.

Cada ítem constante nesta lista é um arquivo. Faça analogia com os armários de arquivos em escritórios: Uma gaveta do armário é um disco; uma pasta é um arquivo com um nome (aquela pequena etiqueta que é anexada à pasta); o conteúdo de uma pasta corresponde aos dados do arquivo.

No disco, podemos ter programas em BASIC, Linguagem de Máquina, ou dados de vários tipos. (agendas, cartas, etc.), perfazendo um total de, no máximo, 112 arquivos.

Os arquivos, como antes visto, recebem um nome, e este nome é dividido em 2 partes (nome e extensão) que são separadas por um ponto (.), sendo que a primeira delas, deve ter no máximo 8 caracteres, e a segunda no máximo 3. Esses caracteres podem ser quaisquer caracteres (as letras minúsculas serão convertidas em maiúsculas) exceto os caracteres:



Faz parte também de um nome, quando necessário, o drive (ou acionador) onde se encontra o arquivo. Para indicar o drive desejado, deve-se usar uma letra entre A e F, seguido sempre de dois-pontos (:), e deve ser colocado sempre ANTES do nome do arquivo! Caso for especificado um drive inexistente, será emitida uma mensagem de erro na tela (observe que o MSX suporta até 6 drives instalados !).

Observe que o nome do arquivo é exclusivo para um e só um arquivo. Isso significa que, para que 2 arquivos sejam considerados distintos, deve existir pelo menos um caractere diferente, do seu nome ou de sua extensão.

CORINGAS

O MSXDOS possui dois caracteres, para facilitar a manipulação de blocos de arquivos. Suponha que você queira saber quais os arquivos que comecem com a letra "C", então basta comandar o seguinte:

```
A>DIR  C*.*
```

O caractere asterisco (*), substitui todos os caracteres do nome ou extensão a partir de onde ele ocorre, ou seja, no exemplo acima, somente foi considerado a letra "C" do nome, e o resto, pode ser qualquer combinação de caracteres, e ainda, seguido de qualquer extensão. Cômodo, não ?!?

Agora imagine que você queira procurar um arquivo que tenha como segunda letra o "S", e tenha o comprimento 6, e tenha extensão "SYS"! Basta comandar:

```
A>DIR  ?S????.SYS
```

O caractere interrogação (?), substitue apenas um caractere, somente na posição que ele se encontra (perceba a diferença entre (?) e (*)).

APRESENTAÇÃO NA TELA

Existem duas opções para o comando DIR, para modificar a apresentação na tela. Estas opções devem ser escritas ao final da linha do DIR.

Uma delas é a "/W", que suprime o comprimento, a data, e a hora, e coloca os nomes em 2 ou mais colunas, dependendo do comprimento da linha. Experimente comandar:

```
A>DIR  /W
```

A outra opção é o "/P", que faz com que a listagem do diretório seja interrompida, sempre que a tela do vídeo estiver cheia, esperando o pressionamento de uma tecla para continuar a listagem.

As duas opções podem ser colocadas numa mesma linha. Comande, então, a linha:

```
A>DIR  /W /P
```

Essas opções são muito úteis quando seu disco possui muitos arquivos.

INTERRUPÇÃO DE COMANDOS

Quando uma mensagem é enviada ao vídeo, pode-se pressionar as teclas **[CONTROL] + [S]**, que tem o efeito de congelar a emissão dos dados (observe que só funciona durante uma listagem). Para liberar a listagem basta pressionar uma tecla.

Existe uma outra forma de interromper o processamento, pressionando-se **[CONTROL] + [C]**, que simplesmente aborta um comando, devolvendo o sinal de "PRONTO" do MSXDOS. No MSX, **[CONTROL] + [C]** é igual a **[CONTROL] + [STOP]**.

Experimente executar um comando DIR, e testar essas duas formas de interrupção para conhecer bem a diferença entre ambas.

ECO NA IMPRESSORA

Algumas vezes é interessante, enviar as listagens da tela para a impressora.

Para isso tecle **[CONTROL] + [P]** para ativar o eco da impressora. Toda linha digitada e finalizada pelo **[RETURN]**, será enviada para a impressora (inclusive o PRONTO "A>"), e todas as mensagens enviadas para a tela.

Para desativar o eco da impressora, digite **[CONTROL] + [N]**.

COMO COPIAR OS ARQUIVOS

Até agora você só aprendeu a copiar um disquete inteiro, mas o MSXDOS possibilita que se façam cópias de apenas um arquivo, ou de pequenos blocos, com nomes parecidos.

O comando, como você deve estar desconfiando, é o COPY, que já foi visto neste capítulo. Por exemplo, suponha que você queira copiar um arquivo de nome "TESTE.TXT", do drive A para o drive B (lembre-se que se você tem apenas um drive, o micro emite mensagens de trocas de disco); então basta comandar:

```
A>COPY A:TESTE.TXT  B:
```

Agora imagine que você queira copiar esse mesmo arquivo, só que, durante a cópia mudar o nome para "LIVRO.BAK"; então comande:

```
A>COPY  TESTE.TXT  B:LIVRO.BAK
```

Note que no comando acima não foi especificado o drive de origem **[A]**, e neste caso, é assumido como drive corrente (ou default). O drive default é aquele indicado no "PRONTO" do MSXDOS (é a letra "A").

Se você quiser mudar o drive corrente para B:, basta digitar:

```
A>B:          (e [RETURN])
```

e você obterá o "prompt" do B:

```
B>
```

Anteriormente aprendemos a usar os "coringas", no comando DIR. Estes "coringas" podem ser também usados no comando COPY. Dessa forma é possível copiar um bloco de arquivos. Veja o exemplo:

```
A>COPY  *.BAS  B:
```

O que deve copiar todos os arquivos com extensão "BAS" e

```
A>COPY  A:??.BAK  B:
```

que copia os arquivos de extensão "BAK" com um nome de 2 letras.

Acostume-se a dar nomes próximos para arquivos da mesma categoria, pois quando for desejada uma cópia destes, poucos comandos serão suficientes para executá-la completamente.

TIPOS DE ARQUIVOS

Um disco pode conter vários tipos de informações: textos, planilhas, banco de dados, fontes de linguagens de programação, aplicativos, ferramentas e outros. Isso significa que, deve-se ter algo sistemático para identificar os vários tipos existentes no disco.

É provável que você, no decorrer do capítulo, tenha questionado a razão da existência da "extensão" do nome do arquivo! Como é de se esperar, essa "extensão" tem justamente a finalidade de identificar o tipo do arquivo.

Abaixo relacionamos os vários tipos de extensão e seus respectivos conteúdos:

ASC - arquivos gravados em ASCII;
ASM - Programas-fonte em ASseMbly;
BAK - Arquivo cópia de um existente (BAcK-up);
BAS - Programas em BASic;
BAT - Arquivo de comandos do MSXDOS (BATch);
BIN - Arquivos em formato BINário: programas em linguagem de máquina;
COB - Programas-fonte em COBol;
COM - Programas em Linguagem de Máquina que rodam pelo MSXDOS;
DAT - Dados diversos (DATa);
FOR - Programas-fonte em FORtran;
GAM - Arquivo de jogos (GAMe);
PAS - Programas-fonte em PAScal;
TXT - Textos diversos (TeXT);
SCR - Arquivos de tela gravados pelo DISK-BASIC (SCReen);

Não se assuste com o excesso de terminações. Note, por exemplo que "FOR", "PAS", "TXT", "COB", "ASM" são em formato ASCII ("ASC"), mas como referem-se a assuntos diferentes (linguagens de programação, textos, etc.), devem ser diferenciados.

Habitue-se a usar as terminações. Se você tiver uma biblioteca de algumas centenas de programas, e todos sem terminação, imagine o trabalho que daria identificá-los constantemente! Pense nisso!

Lembre-se não usar os seguintes nomes de dispositivos em nomes de arquivos:

AUX, CON, LST, PRN e NUL.

Com exclusão destes, você pode até inventar suas próprias terminações.

LISTANDO O CONTEÚDO DE ARQUIVOS

O MSXDOS possui um comando destinado a listar arquivos que estejam em formato ASCII. Caso contrário, que alguns arquivos não farão o mínimo sentido para nós (caso de um programa em Linguagem de Máquina).

O comando em questão é o TYPE, e para usá-lo basta especificar o nome do arquivo que se deseja listar. Por exemplo:

```
A>TYPE A:TEXTO.TXT
```

Este comando pode ser interrompido por **[CONTROL] + [S]** e **[CONTROL] + [C]**.

APAGANDO ARQUIVOS DE UM DISCO

Às vezes é necessário apagar algum arquivo, que não é mais utilizado, ou que foi utilizado apenas para testes ou dados provisórios.

Existe um comando que tem a função de apagar um ou mais arquivos. Esse comando só funcionará se o disco estiver destravado (janela fechada).

Este comando possui 2 nomes: ERASE ou DEL, e para usá-lo deve-se especificar o nome do arquivo que deseja-se apagar. Observe que os "coringas" podem ser usados. Veja o exemplo:

```
A>DEL *.BAK
```

ou

```
A>ERASE *.BAK
```

Que devem apagar todos os arquivos com a terminação "BAK".

Simples, não ? Porém tome cuidado para não apagar, inadvertidamente, arquivos importantes.

EDIÇÃO DE LINHAS DE COMANDOS

É muito comum que, durante a digitação de um comando, ocorram erros. À seguir mostraremos os recursos de edição de linha do MSXDOS:

Para testar os recusos de edição digite:

A>DIR /P /W (e **[RETURN]**!)

E então use os recursos listados abaixo:

〉 - Copia um caractere a direita.

〈 - Apaga um caractere à esquerda.

﹀ - Copia toda a linha.

﹀ - Apaga toda a linha.

[DELETE] - Apaga com o caractere sob o cursor e desloca o restante para a esquerda.

[INSERT] - Entra e sai do modo de inserção.

[HOME] ou **[CONTROL]**+**[K]** - Copia a linha em que está o cursor para a área de edição.

[SELECT] +**[TECLA]** - Copia todos os caracteres para a tela até achar o caractere correspondente à tecla pressionada.

[CLS] +**[TECLA]** - Copia todos os caracteres a partir da primeira ocorrência do caractere correspondente a tecla pressionada.

É necessário que cada um desses recursos seja testados frente ao seu Expert DD PLUS, um a um, para que você vá acostumando desde já com essas facilidades.

AUTO-EXECUÇÃO DE COMANDOS E OS ARQUIVOS "BATCH"

Inicialmente vamos aprender o que é um arquivo BATCH e como montá-lo.

Um arquivo BATCH deve obrigatoriamente ter a extensão "BAT", e a sua função é executar uma sequência de comandos MSXDOS, previamente gravado (ideal para tarefas repetidas). Essa extensão indica ao sistema que se trata de um arquivo BATCH; existe outra extensão reservada pelo MSXDOS que é "COM" mas que não pode ser usada em arquivos BATCH (essa segunda é usada com programas em Linguagem de Máquina).

Dois comandos podem, também, ser usados em um arquivo BATCH: o REM que mostra um comentário, e o PAUSE que mostra um comentário e paralisa a execução até que seja pressionada uma tecla.

Para que um arquivo BATCH seja executado toda vez que se liga o micro, seu nome deve ser AUTOEXEC.BAT, sendo que apenas um arquivo pode ter esse nome.

Agora vamos criar um arquivo autoexecutável do tipo BATCH. Pegue o seu disquete de trabalho (não use o Disco-Mestre fornecido com o Expert DD PLUS), insira no acionador e digite o programa fa figura 4.7:

```
A>COPY CON A:AUTOEXEC.BAT
REM ********************************
REM ******* Expert DD PLUS *******
REM ********************************
REM ****** DISCO DE TRABALHO ******
REM ********************************
REM
PAUSE
DIR /W
DATE
^Z
```

fig. 4.7 - Exemplo de arquivo BATCH

Para terminar o texto digite **[CONTROL]+[Z]**, e você verá o micro gravando no seu disquete de trabalho, esses comandos. Para executa-lo digite:

```
A>AUTOEXEC        (.BAT não é necessário)
```

ou então dê um "*RESET*" no micro e veja como ele é executado imediatamente. Esse arquivo só terá utilidade se os arquivos MSXDOS.SYS e COMMAND.COM estiverem presentes no disco.

MAIS RECURSOS DO COMANDO COPY

O comando COPY possui outros recursos, um pouco mais complexos, e um deles foi usado no exemplo anterior (AUTOEXEC.BAT). Você pode copiar arquivos para um outro dispositivo. Os nomes dos dispositivos estão listados abaixo, e não podem ser usados em nomes de arquivos:

NUL - Este é o chamado dispositivo nulo, que não têm efeito nenhum, usado para testes específicos.

AUX - Este é o dispositivo auxiliar, e pode ser qualquer periférico, no entanto deve ser configurado por um software específico.

CON - É o console do sistema: como entrada temos o teclado e, como saída, o vídeo.

PRN ou LST - É a impressora; e deve ser usado apenas como dispositivo de saída.

A:, B:, C:, D:, E:, F: são os 6 drives do sistema.

Como só existem dois SLOTS externos, então só podemos conectar 2 interface externas, com 2 drives cada uma.

Consequentemente podemos ter 5 acionadores ligados simultaneamente no seu Expert DD PLUS. Supondo que todos eles sejam 3 1/2" DS você terá 3.52 MEGABYTES de memória externa acessível via disco!

O comando COPY permite enviar dados entre um dispositivo e outro, podendo-se também concatenar os arquivos. Vamos gerar antes 2 arquivos em seu disco de trabalho.

```
A>COPY  CON  A:CLS.BAT
MODE  39
^Z
A>COPY  CON  A:BEEP.BAT
^G^G^G
^Z
```

Agora estão presentes dois arquivos BATCH no seu diretório. Para uní-los basta comandar:

A>COPY CLS.BAT + BEEP.BAT = CLSBEEP.BAT

Agora liste o arquivo com:

A>TYPE A:CLSBEEP.BAT

O sinal "+" faz a concatenação e o sinal "=" (que pode ser omitido), indica o destino da concatenação.

Você pode concatenar arquivos usando caracteres coringa da seguinte forma:

A>COPY A:*.BAT = BATCHS.BAT

O que gera a concatenação de todos os arquivos BATCH do disco com o nome BATCHS.BAT!

Por enquanto só concatenamos arquivos do tipo ASCII, e nesses o ^Z indica fim de arquivo, e não é copiado. Para programas em Linguagem de Máquina ou Binários devemos usar a opção /B. Suponha que você queira concatenar os arquivos MSXDOS.SYS e COMMAND.COM, então basta comandar:

COPY /B MSXDOS.SYS + COMMAND.COM = LIXO.COM

Esse comando LIXO gerado agora não terá muita utilidade, portanto apague-o com o comando ERASE.

Para que o comando volte a funcionar como antes, use a opção /A para desligar o /B.

Existe ainda, por fim, a opção /V (usada no exemplo do BACKUP) que ativa a verificação da escrita, ou seja, após toda gravação, é feita uma verificação da validade dos dados.

Existe outra forma de ativar ou desativar a verificação, com efeito permanente:

A>VERIFY ON

para ligar e

A>VERIFY OFF

para desligar a verificação.

INDO PARA O DISK-BASIC

Se você estiver no MSXDOS e deseja passar para o DISK-BASIC, basta comandar:

A>BASIC

E o MSXDOS se desliga, e o controle é assumido pelo DISK-BASIC.

Se você digitar:

A>BASIC PROGRAMA.BAS

onde PROGRAMA.BAS é, por exemplo, o nome de um programa em BASIC gravado no disquete, o Expert abandona o MSXDOS, assume o DISK-BASIC e executa o programa citado. Para passar do BASIC para o MSXDOS use:

CALL SYSTEM

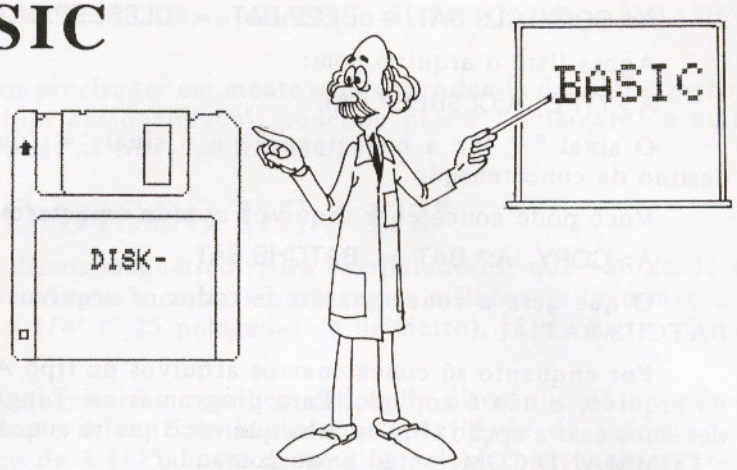
ou

_SYSTEM

Mas isso já é outro capítulo!!!

O DISK-BASIC

5



COMO ACESSAR O DISK-BASIC

Quando você liga o micro com a interface de disco ativa (sem apertar **[SHIFT]** ou **[CONTROL]**) a primeira coisa que o sistema faz é acionar o drive para ver se há um disco nele. Existindo disco, serão procurados os seguintes arquivos:

MSXDOS .SYS e COMMAND .COM.

Quando o Expert encontra esses arquivos, eles são carregados e reconfiguram o MSX deixando-o compatível com inúmeros outros computadores (e perdendo algumas características do MSX).

Se, ao longo desse procedimento, ele não conseguir carregar o MSXDOS, ou porque não há disco no drive, ou porque não existem os arquivos acima citados, então será ativado o DISK –BASIC MSX, gravado na memória permante da interface.

Com o DISK-BASIC ativado, o MSX conserva suas características peculiares e sua linguagem BASIC original, implementada com alguns recursos adicionais para poder operar com discos.

A primeira mensagem que o DISK-BASIC envia para o usuário é:

Enter date (D-M-Y):

Se você não faz questão de gravar arquivos com a data atual pode digitar simplesmente **[RETURN]**. Se você está trabalhando profissionalmente com seu MSX e vai gravar bancos de dados que sofrem atualizações periódicas, é conveniente criar o hábito de sempre digitar a data atual, usando 2 dígitos para o Dia, 2 digitos para o Mês, e 2 para o Ano, separados por um traço (sinal de menos). A seguir digitando **[RETURN]**, você terá a tela da figura 5.1:

fig. 5.1 - Tela de apresentação do DISK-BASIC.

```
MSX BASIC versão 1.1Br
Gradiente
23430 bytes livres
Disk BASIC version 1.0
```

A outra maneira de se acessar o DISK-BASIC, como já vimos, é a partir do MSXDOS.

Se você já está com o computador ligado e este carregou o MSXDOS, basta comandar:

A>BASIC (e [RETURN])

Dessa forma você obterá a mesma tela apresentada na figura 5.1.

A primeira coisa que vamos aprender agora é como ler o diretório do disco estando em BASIC.

Experimente comandar:

FILES (e [RETURN])

ou

files (e [RETURN])

Você obterá na tela a lista dos arquivos contidos no disco. Essa lista é menos completa que a produzida pelo comando DIR do MSXDOS, pois além de não ser fornecida a data de gravação (e eventualmente o horário), não diz quantos bytes cada arquivo ocupa.

O comando FILES aceita ser seguido por nome de arquivos de maneira a listar seletivamente alguns arquivos. Valem, também, os mesmos "coringas" usados no MSXDOS (* e ?).

Lembre-se porém, que o nome do arquivo (e eventualmente do dispositivo) deve estar ENTRE ASPAS.

Se, por exemplo, você comandar:

FILES "B:*.BAS"

isso significará que ele listará, a partir do drive B (se houver um só acionador será solicitada uma troca de discos), todos os arquivos com a terminação ".BAS".

Se você tiver uma impressora conectada ao micro, comande:

LFILES (e [RETURN])

Você obterá a listagem dos arquivos contidos no disco na impressora. Com relação aos coringas (* e ?) o LFILES se comporta como o FILES.

A PREPARAÇÃO DO DISCO PARA USO

A partir do DISK –BASIC você também pode formatar discos virgens. Para isso basta comandar:

CALL FORMAT

ou

_ FORMAT

Você terá o mesmo menu que aparecia na formatação pelo MSXDOS (fig. 4.6).

Para o acionador do seu Expert DDPLUS você deve usar o opção de 80 trilhas - face dupla (4).

Se, porém, você quiser preparar um disco para alguém que tem um acionador de 3 1/2" face simples, use a opção 80 trilhas - face simples (3).

CÓPIA E BACKUP

O DISK-BASIC tem um comando COPY que funciona de maneira análoga ao do MSXDOS. Existe, porém, uma diferença importante: a sintaxe é ligeiramente diferente. Os nomes dos arquivos (e eventualmente dos drives) devem estar entre aspas.. Além disso, deve haver um separador "TO" entre o arquivo origem e o arquivo destino. Se

você quiser copiar, por exemplo um arquivo de nome "TESTE.TXT" presente no drive A para o drive B, renomeando-o para "RASCUNHO.ASC", você deverá comandar:

```
COPY "A:TESTE.TXT" TO "B:RASCUNHO.ASC"
```

Neste caso o sistema lerá o arquivo de nome TESTE.TXT no drive A e gravará como "RASCUNHO.ASC" no drive B.

Cópias de arquivos muito extensos ou de muito arquivos, porém, são muito mais rápidas pelo MSXDOS do que pelo DISK-BASIC. No MSXDOS as solicitações para troca de discos (A/B) são muito menos frequentes.

Se você chegou ao DISK-BASIC a partir do MSXDOS, basta comandar

```
CALL SYSTEM
```

ou

```
_SYSTEM
```

com um disco que tenha os arquivos MSXDOS.SYS e COMMAND.COM, para retornar ao MSXDOS.

Mas, cuidado! Ao retornar ao MSXDOS, um programa em BASIC que esteja na memória RAM do MSX é perdido.

Para evitar esta perda é conveniente, então, que você leia o ítem seguinte.

GRAVAÇÃO E LEITURA DE PROGRAMAS

Quando você tem um programa em BASIC na memória do computador e quer salvá-lo em disco, basta usar o comando SAVE, seguido pelo nome. Após o nome, de no máximo 8 caracteres, coloque a extensão ".BAS" para classificá-lo como programa BASIC.

Se você quiser chamar o seu programa BASIC, por exemplo, de "ALEPH" e quer gravá-lo no drive B, basta comandar:

```
SAVE "B:ALEPH.BAS"
```

Se você omitir o nome do drive, o programa será gravado no drive corrente.

O programa será compactado, e gravado no disquete, de maneira a ocupar o menor espaço possível.

Em alguns casos (leitura através de um editor de textos, execução de um comando MERGE, etc.) é necessário que o programa seja gravado no modo 'não compactado' (ou ASCII), e para isso deve-se usar a opção ",A". Então comande:

```
SAVE "ALEPH.ASC",A
```

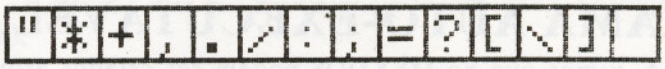
O programa é, então, gravado no drive corrente (A), sendo que, como foi especificada a opção ",A", será gravado no formato ASCII (igual ao mostrado na tela ou impressora!), possibilitando várias outras aplicações. Essa opção tem a desvantagem de ocupar mais espaço e ser mais demorada. A extensão ".ASC" deve ser colocada para indicar o tipo de formato da gravação.

Os 8 caracteres do nome devem obrigatóriamente ter seus códigos entre 33 e 126. Para visualizá-los, rode o programa da figura 5.1:

fig 5.1 - Os caracteres permitidos para nomes de arquivos.

```
10 FOR I=33 TO 126
20 PRINT CHR$(I);
30 NEXT I
RUN
```

Dos caracteres listados na tela, convém observar que as letras minúsculas são transformadas automaticamente em maiúsculas. Além disso, por serem usados com outras funções , você deve excluir da lista os caracteres:



Os nomes dos dispositivos aceitos pelo sistema também não devem ser usados como nomes de arquivos são os seguintes:

AUX - Auxiliar

CON - Teclado e Vídeo

LST - Impressora

PRN - Impressora

NUL - Nulo

Para recuperar um programa em BASIC gravado em disco, basta comandar:

```
LOAD "B:ALEPH.BAS"
```

ou

```
LOAD "ALEPH.ASC"
```

Se o nome do drive for omitido, então será assumido o drive corrente. O comando LOAD trata de identificar o tipo e, se esse for incompatível, será emitida uma mensagem de erro. Veja que essa verificação de formato independe do tipo de extensão (a extensão só tem utilidade prática para o usuário).

Se for colocada a opção ",R" o programa será executado automaticamente, por exemplo:

```
LOAD "ALEPH.BAS",R
```

que é também equivalente a:

```
RUN "ALEPH.BAS"
```

O comando RUN seguido de um nome de arquivo, funciona como o LOAD com a opção R.

Se você tem um programa na memória do micro e outro no disquete com formato ASCII (opção A do SAVE) e nome "EXPERT.ASC", e deseja juntá-los, então comande:

```
MERGE "EXPERT.ASC"
```

E a seguir comande:

```
LIST
```

para listar os programas. Você deve verificar que os programas foram unidos na memória, desde que seus números de linha não conflitem. Se os números de linha estiverem conflitando então serão assumidas as linhas que se encontram gravadas no disquete, perdendo-se assim, as que estavam residentes na memória. Existe outra aplicação para o formato de gravação ASCII. Se você gravar um arquivo neste formato, ele poderá ser lido por um IBM-PC. A surpresa se torna ainda maior quando você percebe que um programa BASIC gravado no IBM-PC em ASCII, pode ser lido e rodado num MSX!

As duas versões BASIC são bastante parecidas, mas, em tese, o MSX-BASIC é um pouco mais completo, sendo que a adaptação do IBM para o MSX é mais fácil do que o caminho inverso.

Se você estiver trabalhando com compiladores, então, a utilidade dos programas em ASCII se torna fundamental.

Veja a respeito no capítulo 2.

O PROGRAMA AUTO-EXECUTÁVEL

No capítulo 4, vimos que no MSXDOS podem ser criados arquivos de comandos auto-executáveis, os arquivos BATCH de nome AUTOEXEC.BAT.

No DISK-BASIC temos algo muito parecido, porém com mais possibilidades. Podemos criar no disco um programa BASIC que seja automaticamente executado após a inicialização do sistema.

Formate um disco virgem e destravado (cuidado para não formatar o Disco-Mestre por engano), com o comando:

```
CALL FORMAT
```

Escolha a opção conveniente (normalmente a 4ª opção é a ideal).

Após a formatação, devemos gerar um arquivo com o nome AUTOEXEC.BAS.

Não grave o MSXDOS.SYS nesse disco para podermos trabalhar apenas com o DISK-BASIC. Digite o programa da figura 5.2:

fig. 5.2 - Programa AUTOEXEC.BAS

```
10 SCREEN 0:WIDTH 39:KEY OFF
20 PRINT:PRINT"ESTE DISCO PERTENCE"
30 PRINT"A XXXXXXXXXXX"
40 PRINT"E CONTEM OS ARQUIVOS:"
50 FILES
60 END
```

NA LINHA 30, NO LUGAR DE XXXXXXXXXXX, escreva seu nome e grave o programa no disco com o comando:

```
SAVE "AUTOEXEC.BAS"
```

Para certificar-se que o arquivo foi realmente gerado no disco, use o comando:

```
FILES
```

Se tudo correu bem, experimente ressetar novamente o sistema, pressionando o botão que se encontra na parte posterior do gabinete do micro, ou pressionando simultaneamente as teclas **[CTRL]+[SHIFT]+[STOP]** sem retirar o disco do acionador.

Após a inicialização do sistema, o DISK-BASIC assume o controle do micro, e executa automaticamente o programa que acabamos de digitar.

TIPOS DE ARQUIVOS

Uma das grandes vantagens dos discos sobre as fitas magnéticas é a velocidade e a confiabilidade das gravações e leituras dos programas.

Além dos programas podem ser gravados outros tipos de dados, como por exemplo, sua relação de discos ou livros, e suas despesas.

Esses arquivos podem ser de dois tipos: sequencial ou randômico. Este último é também chamado de aleatório. O que distingue os dois tipos de arquivos é a forma de armazenamento. Por exemplo, imagine uma lista de nomes e telefones:

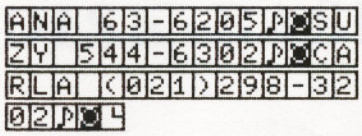
```
ANA     63-6205
SUZY    544-6302
CARLA   (021) 298-3202
```

Note que tanto o nome como o telefone podem ter comprimentos diferentes.

Num arquivo SEQUENCIAL esses dados estariam armazenados como mostra a

figura 5.3:

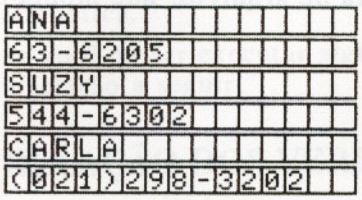
fig 5.3 - Lista em arquivo sequencial.



Note que, entre os dados existe um separador, formado normalmente pelos caracteres de códigos ASCII 13 (CR =Carriage Return) e 10 (Line Feed), para indicar ao DISK-BASIC que alí termina uma informação e começa outra, e no final temos uma que indica o final de arquivo (SUBstitute=caractere 26). O formato descrito acima é o formato ASCII.

Já num arquivo randômico, essa lista seria armazenada como na figura 5.4:

fig 5.4 - Lista em arquivo aleatório.



O arquivo randômico (ou aleatório) não necessita de separadores e do indicador de fim de arquivo, porém as informações devem ser de tamanho fixo, limitados a um comprimento pré-determinado.

Cada informação é camada de REGISTRO (ou ficha) e pode conter vários CAMPOS. Espaços em branco são inseridos nas posições que não forem preenchidas.

Cada tipo de arquivo tem suas vantagens e desvantagens, e com o uso, você aprenderá a utilizá-los adequadamente.

Como regra geral, podemos dizer que os arquivos de acesso sequencial (que consomem menos memória), devem ser usados quando as informações não têm necessidade de serem acessadas individualmente, enquanto que os de acesso randômico se prestam justamente para esses casos, facilitando buscas e seleções.

ARQUIVOS DE ACESSO SEQUENCIAL

Um arquivo de acesso sequencial pode ser aberto de três diferentes modos:

1 - Como saída de dados para arquivo novo (OUTPUT). Nesse caso monta um arquivo e, se ele já existir, seus dados serão renovados.

2 - Como saída de dados (APPEND) para um arquivo já existente. Neste caso ele acrescentará novos dados no final do arquivo anterior.

3 - Como leitura de dados (INPUT). Nesse caso ele os lê sequencialmente.

A estrutura de um programa para trabalhar com arquivos sequenciais deve ser a seguinte:

- Abrir o arquivo (comando OPEN);
- Colocar ou ler os dados, conforme o tipo de escolha de OPEN, sequencialmente;
- Fechar o arquivo após o término dos dados (comando CLOSE).

Para que você compreenda melhor essa estrutura, passemos o exemplo dos nomes e telefones para um programa, gravando-o então num disco de trabalho já formatado

(figura 5.5):

fig 5.5 – Criação de arquivo sequencial.

```
10 REM --- Abrir o arquivo ---
20 OPEN "A:LISTEL.DAT" FOR OUTPUT AS #1
30 REM --- Acessar arquivo: Leitura ---
40 PRINT #1,"ANA 63-6205"
50 PRINT #1,"SUZY 544-6302"
60 PRINT #1,"CARLA (021) 298-3202"
70 REM --- Fechar Arquivo ---
80 CLOSE #1
```

As linhas 10, 30 e 70 são comentários que sinalizam o início de cada bloco. As linhas 40, 50 e 60 gravam os dados no arquivo.

Observe que essa é a estrutura básica de um programa de arquivos, que monta o arquivo LISTEL.DAT com os seus respectivos dados.

Salve o programa com:

```
SAVE "SEQOUT.BAS
```

e limpe a memória com:

```
NEW
```

Digite agora o programa da figura 5.6.

fig. 5.6 - Leitura de arquivo sequencial

```
10 REM --- Abrir o Arquivo ---
20 OPEN "A:LISTEL.DAT" FOR INPUT AS #1
30 REM --- Lê os dados ---
40 INPUT #1,A$:PRINT A$
50 INPUT #1,B$:PRINT B$
60 INPUT #1,C$:PRINT C$
70 REM --- Fecha o Arquivo ---
80 CLOSE #1
```

que ilustra a leitura de arquivos sequenciais.

A linha 20 se encarrega de abrir o arquivo LISTEL.DAT em modo de leitura (INPUT) e define para esse arquivo o número 1.

As linhas 40, 50 e 60 lêem o arquivo número 1 e armazenam os dados em A$, B$ e C$, respectivamente. Os comandos PRINT tem a função de emitir para a tela o conteúdo de cada variável.

A linha 80 simplesmente fecha o arquivo de número 1.

Observe que o INPUT no comando OPEN não é o mesmo INPUT das linhas 40 a 60. (o primeiro define o modo de acesso, e o segundo executa a leitura dos dados).

Salve o programa com:

```
SAVE"SEQINP.BAS"
```

e limpe a memória do micro com NEW. Digite agora o programa da figura 5.7.

fig. 5.7 – Complementando arquivos.

```
10 REM --- Abertura do Arquivo ---
20 OPEN "A:LISTEL.DAT" FOR APPEND AS #1
30 REM --- Adicionando Dados ---
40 PRINT #1,"SHEILA 491-8766"
50 PRINT #1,"CAMILA 495-0203"
60 PRINT #1,"DALVA (031) 66-4536"
70 REM --- Fechar arquivo ---
80 CLOSE #1:END
```

que adiciona informações ao arquivo LISTEL.DAT.

Rode-o e depois grave-o, para referências futuras, como:

```
SAVE "SEQAPP.BAS"
```

A seguir, limpe a memória de seu Expert com NEW.

Para verificar o conteúdo atualizado do arquivo LISTEL.DAT, use o programa da figura 5.8:

fig. 5.8 - Lendo um arquivo sequencial.

```
100 INPUT "QUAL ARQUIVO ";N$
110 OPEN N$ FOR INPUT AS #1
120 IF EOF(1) = -1 THEN 160
130 LINE INPUT #1,A$
140 X=X+1:PRINT X;" -- ";A$
150 GOTO 120
160 PRINT,,"FIM DO ARQUIVO"
170 CLOSE #1:END
```

Rode este programa e, quando ele solicitar:

```
QUAL ARQUIVO?
```

digite:

```
LISTEL.DAT
```

ou o nome de outro arquivo sequencial que você tenha eventualmente aberto por conta própria.

O programa acima usa alguns comandos novos, como por exemplo o EOF que retorna o valor –1 quando for encontrado o fim-de-arquivo; LINE INPUT # que funciona de maneira análoga ao INPUT #. Na linha 140, X armazena qual elemento esta sendo lido. Quando o programa é executado a variável X é iniciada com 0 (atente para essas técnicas de programação).

O "IF" faz o teste de validade da função EOF, saltando para 160, caso o fim do arquivo for encontrado.

Com os exemplos anteriores, acreditamos que as noções básicas sobre arquivos sequenciais tenham sido aprendidas.

Grave este último programa-exemplo com:

```
SAVE "LESEQ.BAS"
```

No seu disquete você terá, então, os seguintes arquivos:

SEQOUT.BAS	(fig. 5.5)
SEQINP.BAS	(fig. 5.6)
SEQAPP.BAS	(fig. 5.7)
LESEQ .BAS	(fig. 5.8)
LISTEL.DAT	(arquivo c/ nomes e tel.)

ARQUIVOS DE ACESSO ALEATÓRIO

Os arquivos de acesso aleatório possuem apenas um modo de abertura, porém ele é mais do que suficiente, uma vez que permite a gravação, leitura e alteração dos dados no arquivo. Os arquivos de acesso aleatório possuem REGISTROS de tamanho fixo, divididos em CAMPOS também com tamanhos fixos! A sequência de uso dos arquivos aleatórios é a seguinte:

- Se o arquivo não existe, criar o arquivo;
- Abrir o arquivo;
- Definir o tamanho dos registros;
- Gravar os campos do arquivo;
- Fechar o arquivo.

Não se assuste com a quantidade de ítens, pois como veremos, o uso dos arquivos randômicos é bem fácil.

Para verificar isso, vamos criar um arquivo randômico e armazenar nele aquela nossa relação de nomes e telefones, só que agora definiremos um campo para o nome e um campo para o número. Digite o programa da figura 5.9.

fig. 5.9 – Criação de arquivo aleatório.

```
10 OPEN "A:LISTEL2.DAT" AS #1 LEN=30
20 FIELD #1, 15 AS A$, 15 AS B$
30 LSET A$="ANA":RSET B$="63-6205"
40 PUT #1,1
50 LSET A$="SUZY":RSET B$="544-6302"
60 PUT #1,2
70 LSET A$="CARLA":RSET B$="(021) 298-3202"
80 PUT #1,3
90 CLOSE #1:END
```

Vamos analisar cada linha deste programa :

A linha 10, cria um arquivo com o nome "LISTEL2.DAT" e registros (fichas) de tamanho 30.

A linha 20 define 2 campos em cada registro; dos 30 bytes de comprimento (definidos no OPEN para cada ficha), 15 serão usados para A$ (nome) e 15 serão usados para B$ (telefone).

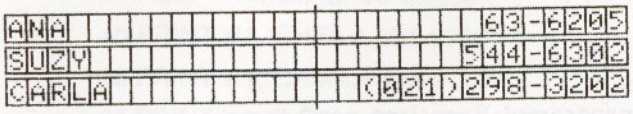
As linhas 30, 50 e 70 alinham o campo do nome à esquerda e o campo do telefone à direita. LSET e RSET funcionam como o LET, com a diferença de alinharem os dados à direita (RSET) ou esquerda (LSET), pois cada campo tem comprimento fixo! Se o comprimento for maior que o definido pelo FIELD, então o excesso é desprezado.

As linhas 40, 60, 80 gravam cada registro no arquivo. Note que no PUT deve ser definido o arquivo (#1) e o número do registro (1,2 e 3).

O comando CLOSE da linha 90, simplesmente fecha o arquivo.

No disquete, porém, o arquivo será armazenado como mostra a figura 5.10.

fig. 5.10 - Dados no Arquivo LISTEL2.DAT.



Note que os dados estão gravados de 30 em 30 bytes sequencialmente, e não existem separadores como no arquivo de acesso sequencial.

Agora altere o programa que ainda está na memória de seu Expert, conforme a listagem da figura 5.11.

fig. 5.11 - Lendo um arquivo aleatório.

```
10 OPEN "A:LISTEL.DAT" AS #1 LEN=30
20 FIELD #1, 15 AS A$, 15 AS B$
30 GET #1,1
40 PRINT "REGISTRO 1: ";A$;B$
50 GET #1,2
60 PRINT "REGISTRO 2: ";A$;B$
70 GET #1,3
80 PRINT "REGISTRO 3: ";A$;B$
90 CLOSE #1:END
```

Nesse programa as linhas que nos interessam são as de número 30, 50 e 70. Nelas, a instrução GET # lê informações do arquivo e as insere nas variáveis A$ e B$. GET # deve ser seguido do número do arquivo e do número do registro.

Veremos agora como acrescentar dados em arquivos aleatórios, e para isso devemos conhecer as funções LOF e LOC.

LOF retorna o número de bytes gravados no arquivo e, ser for dividido pelo comprimento de cada registro, fornece o número de registros gravados!

LOC retorna o número do último arquivo acessado, portanto, após o PUT ou GET, LOC retorna o número desse registro gravado ou lido!

Apague do programa anterior com:

```
DELETE 30-90
```

dê um LIST e acrescente as linhas indicadas na figura 5.12. Note que as linhas 10 e 20, que definem a estrutura dos registros, permanecem inalteradas!

fig. 5.12 - Acrescentando dados em arquivos randômicos

```
10 OPEN "LISTEL 2. DAT" AS # 1 LEN = 30
20 FIELD # 1, 15 AS A$, 15 AS B$
30 PRINT "TAMANHO ANTERIOR:"; LOF (1); "BYTES"
40 PRINT "REGISTROS GRAVADOS:"; LOF (1)/30
50 LSET A$ = "SHEILA": RSET B$ = "491-8766"
60 PUT # 1, LOF (1)/30 + 1
70 LSET A$ = "CAMILA": RSET B$ = "495-0203"
```

```
80 PUT #1,LOF(1)/30+1
90 LSET A$="DALVA":RSET B$ ="(031) 66-4536"
100 PUT #1,LOF(1)/30+1
110 PRINT "ULTIMO REGISTRO:";LOC(1)
120 CLOSE:END
```

Desta vez não vamos analisar o programa! Faça isso você mesmo e não prossiga enquanto o processo não estiver bem claro.

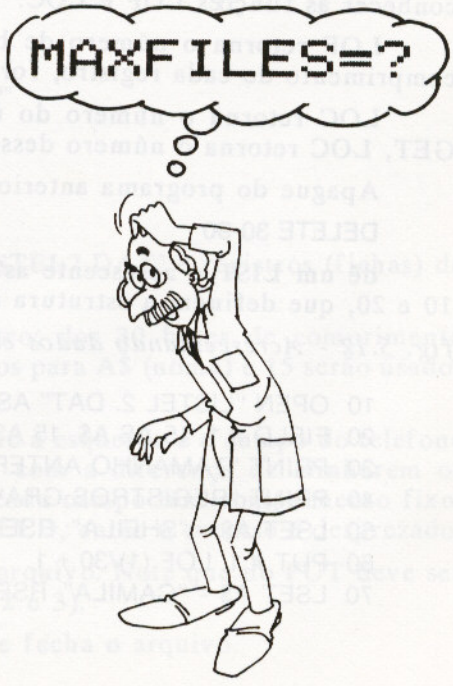
Os arquivos de acesso aleatório possibilitam também o uso de números, transformando-os em "string". Mas existem 6 funções (MKD$, MKI$, MKS$, CVD, CVI, CVS) mais econômicas para uso de números, que não serão vistas nesse capítulo. Para maiores detalhes veja o capítulo 6 e o programa-exemplo 3 do apêndice A.

O USO DE VÁRIOS ARQUIVOS AO MESMO TEMPO

Você pode usar até 6 arquivos em disco simultaneamente. Para isso, entretanto, é necessário avisar ao DISK – BASIC quantos arquivos serão abertos através do comando MAXFILES. Analise o programa da figura 5.13. Ele abre dois arquivos em disco e um arquivo na impressora. Depois os dados lidos no disco são enviados para a impressora.

fig 5.13 - Usando vários arquivos simultaneamente.

```
10 MAXFILES = 3
20 OPEN "A:LISTEL.DAT" FOR INPUT AS #1
30 OPEN "A:LISTEL2.DAT" AS #2 LEN=30
40 FIELD #2,15 AS A$,15 AS B$
50 OPEN "LPT:"FOR OUTPUT AS #3
60 PRINT #3,"DADOS DO LISTEL.DAT:"
70 PRINT #3,"--------------------"
80 IF EOF(1)=-1 THEN 120
90 LINE INPUT #1,C$
100 PRINT #3,C$
110 GOTO 80
120 PRINT #3,""
130 PRINT #3,"DADOS DO LISTEL2.DAT:"
140 PRINT #3,"---------------------"
150 FOR F = 1 TO LOF (2)/30
160 GET #2,F
170 PRINT #3,A$;"-";B$
180 NEXT F
190 CLOSE:END
```



DICIONÁRIO DE BASIC E DISK-BASIC

6

Neste dicionário são listados, em ordem alfabética, os comandos, funções, operadores e variáveis do BASIC e DISK-BASIC. Se você estiver buscando algum verbete já conhecido, para lembrar sua sintaxe ou particularidades, basta procurá-lo pela ordem alfabética.

Se você quiser executar alguma tarefa e quiser saber quais os recursos disponíveis, veja a relação a seguir:

RELAÇÃO DOS VERBETES POR TIPO DE USO:

01- DEFINIÇÃO DE TIPO OU MUDANÇA DE TIPO DE VARIÁVEL:
CDBL, CINT, CSNG, DEFINT, DEFSNG, DEFDBL, DEFSTR, STR$.

02- MANIPULAÇÃO DE VARIÁVEIS:
DATA, DEF FN, DIM, ERASE, INPUT, LET, READ, RESTORE, SWAP.

03- FUNÇÕES MATEMÁTICAS COM ARGUMENTO E RESULTADO NUMÉRICO:
ABS, ATN, COS, EXP, FIX, INT, LOG, RND, SGN, SIN, SQR, TAN.

04- FUNÇÕES COM ARGUMENTO E RESULTADO STRING:
INPUT$, INSTR, LEFT$, LINE INPUT, MID$, RIGHT$, SPACE$, STRING$.

05- FUNÇÕES COM ARGUMENTO NUMÉRICO E RESULTADO STRING:
ASC, BIN$, HEX$, OCT$.

06- FUNÇÕES COM ARGUMENTO STRING E RESULTADO NUMÉRICO:
LEN, VAL.

07- FUNÇÕES COM ARGUMENTO CARACTERE:
CHR$, INKEY$.

08- TELAS EM GERAL:
CLS, COLOR, SCREEN.

09- TELAS DE TEXTO:
CSRLIN, LOCATE, POS, PRINT, PRINT USING, SPC, TAB, WIDTH.

10- TELAS GRÁFICAS:

CIRCLE, DRAW, LINE, PAINT, POINT, PRESET, PSET.

11- MANIPULAÇÃO DE SPRITES:

PUT SPRITE, SPRITE$.

12- EDIÇÃO E EXCECUÇÃO DE PROGRAMAS:

AUTO, CONT, DELETE, END, LIST, NEW, REM, RENUM, RUN, STOP, TROFF, TRON.

13- ARQUIVOS:

BLOAD, BSAVE, CALL FORMAT, CLOAD, CSAVE, CLOSE, COPY, CVD, CVI, CVS, DSKF, DSKI$, DSKO$, EOF, FIELD, FILES, GET, INPUT#, INPUT$, KILL, LINE INPUT#, LOAD, LOC, LOF, LSET, MAX FILES, MERGE, MKD$, MKI$, MKS$, MOTOR, NAME, OPEN, PRINT#, PUT, RSET, SAVE, VARPTR, VERIFY.

14- IMPRESSORA:

LLIST, LPOS, LPRINT, LPRINT USING.

15- TECLAS DE FUNÇÃO:

KEY, KEY LIST, KEY OFF, KEY ON.

16- SETAS, JOYSTICK, PADDLE:

PAD, PDL, STICK, STRIG.

17- SONS:

BEEP, PLAY, SOUND.

18- LAÇOS, DESVIOS, DECISÕES E SUB-ROTINAS:

FOR, GOSUB, GOTO, IF, ON..GOTO, ON..GOSUB.

19- ERROS:

ERL, ERR, ERROR, ON ERROR GOTO.

20- INTERRUPÇÕES:

INTERVAL ON (OFF, STOP), KEY ON (OFF, STOP), ON INTERVAL, ON KEY, ON SPRITE, ON STOP, ON STRIG, SPRITE ON (OFF, STOP), STOP ON (OFF, STOP), STRIG ON (OFF, STOP), WAIT.

21- RELACIONADOS COM A ARQUITETURA MSX:

BASE, CALL, CLEAR, DEFUSR, FRE, INP, OUT, PEEK, POKE, RESUME, TIME, USR, VARPTR, VDP, VPEEK, VPOKE.

SIMBOLOGIA DO DICIONÁRIO:

Na sintaxe de cada verbete, usamos a seguinte convenção:

[ ] As construções entre colchetes, são opcionais.

| Apenas uma das duas construções separadas pela barra vertical pode ser utilizada.

{ } Pelo menos uma das costruções entre chaves deve ser utilizada.

Além disso, para orientação do leitor, colocamos em cada verbete 3 ícones com o seguinte significado:

–Elementar, pode ser usado por principiantes.

–Avançado. Para maiores detalhes consulte a bibliografia aconselhada.

–Comando.

–Função.

–Operador.

–Variável especial do BASIC MSX.

–Indicador de tipo de variável.

–Estrutura que pode ser usada no BASIC sem disco.

–Estrutura específica do DISK-BASIC.

–Estrutura do BASIC sem disco cuja sintaxe é alterada pelo DISK-BASIC.

Os dispositivos a que se refere este dicionário são:

A:...F:	– Drive A (...até F).
CAS:	– Gravador cassete.
CRT:	– Tela de texto.
GRP:	– Tela gráfica.
LPT:	– Impressora.
COM:	– Interface RS-232C

Os nomes de arquivos podem ter até 6 caracteres para o dispositivo CAS: (sendo as maiúsculas diferentes das minúsculas) e 8 caracteres, seguidos de uma terminação de 3 (separados por um ponto) para A: até F: .

Neste caso as minúsculas são transformadas em maiúsculas.

Obs: Os dois pontos que seguem o nome do dispositivo, são parte integrante do mesmo.

ABS(expressão)

Fornece o valor absoluto de um número ou expressão.

Exemplo: A=–3:PRINT A,ABS(A)

Como resultado deverá aparecer o número – 3 e ao lado seu valor absoluto.

AS

Veja **NAME, OPEN**

ASC(string)

Fornece o código **ASCII** do 1º caractere da string.

Exemplo: PRINT ASC("MSX")

A resposta deve ser 77 (código do caractere M)

AND

Operador que faz o "e" lógico *bit a bit* de dois números. Retorna um *bit* 1 somente se ambos os *bits* comparados forem iguais a 1.

Exemplo: PRINT BIN$(&B00101101 AND &B10110001)

Resulta 00100001 em binário.

ATN(expressão)

Fornece o valor do arco (em radianos) da expressão.

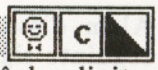
Exemplo: PI=4*ATN(1):PRINT PI

A resposta deve ser 3.1415926535898 pois **ATN(1)** é igual a **PI/4**.

ATTR$

Função não implementada.

AUTO[nº inicial da linha][,incremento]



Numera automaticamente as linhas do programa em edição. Para interrompê-lo, digitar [CONTROL] + [STOP]. Se aparecer um "*" ao lado do nº da linha isto significa que já existe uma linha com este número.

Exemplo: AUTO 100,5

Deve gerar linhas com numeração 100, 105, 110, 115, 120, etc.

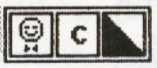
BASE(expressão)[=expressão]

Permite ler ou redefinir o endereço inicial das tabelas do processador de vídeo.

Exemplo: FOR I=0 TO 19:PRINT BASE(I):NEXT I

Fornece os endereços (na *VRAM*) das tabelas utilizadas nas várias telas.

BEEP



Gera um sinal sonoro. Seu efeito é o mesmo de PRINT CHR$(7).

Exemplo: FOR I=1 TO 20:BEEP:FOR K=0 TO 300:NEXT K,I

Gera uma sucessão de 20 sinais sonoros.

BIN$(expressão)

Converte o valor da expressão numa string em notação binária.

Exemplo: FOR I=0 TO 22:PRINT I,BIN$(I):NEXT I

Imprime na tela os números de 0 a 22 em notação decimal e binária.

BLOAD"{[dispositivo][nome arq]}"[,R|,S][,deslocamento]

Carrega arquivos em formato binário para uma região da memória.

dispositivo —Veja no início deste capítulo.

nome arq —Veja no início deste capítulo.

,R —Ativa a execução automática.

,S —Transfere o bloco para a VRAM (só a partir de disco).

deslocamento —Transfere o bloco deslocando os endereços definidos no **BSAVE**.

Exemplo: BLOAD"A:TELA-4.SR2",S,-300

Carrega, a partir do disco A o bloco "TELA-4.SR2" na VRAM subtraindo 300 dos endereços originalmente definidos no **BSAVE**.

BSAVE"{[dispositivo][nome arq]}",inicio,fim[,execução][,S]

Grava, num dispositivo, um arquivo de formato binário.

dispositivo —Veja início deste capítulo.

nome arq —Veja no início deste capítulo

inicio —Marca o endereço inicial do bloco binário

fim —Marca o endereco final do bloco binário

execução —Marca o endereço de execução para automatização na opção R do comando **BLOAD**. Caso não definido é assumido o início do bloco.

,S —Grava o bloco a partir de endereços da VRAM

Exemplo: BSAVE"CAS:JOGO",20000,30000,21000

Grava no K-7 o bloco de memória, com o nome JOGO, e endereço 21000 para execução.

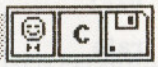
CALL nome do comando [,argumentos]

Executa um comando contido num cartucho. Pode ser usado o "_" como abreviação para "CALL".

Exemplo: CALL EDITOR

Chama o programa de cartucho "EDITOR".

CALL FORMAT



Este comando chama um "*procedimento*" do sistema de disco que serve para inicializar (formatar) um disquete.

Sempre que um disquete for *virgem*, o usuário deve formatá-lo, para que o sistema o adapte a seus padrões de leitura e escrita.

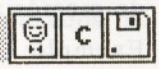
O tipo de formatação depende do acionador de disco e da interface. No **MSX** temos os seguintes tipos:

1) 40 trilhas face simples 180 K.
2) 40 trilhas face dupla 360 K.
3) 80 trilhas face simples 360 K.
4) 80 trilhas face dupla 720 K.

No seu "**EXPERT DD PLUS**", você tem 80 trilhas, face dupla.

Este procedimento só se encontra no **DISK-BASIC**.

CALL SYSTEM

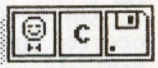


Chama o procedimento da interface de disco, para execução do **MSXDOS** a partir do **DISK-BASIC**. Isto só é possível se o **MSXDOS** esteve presente durante a partida do micro.

Exemplo: CALL SYSTEM

Entra no **MSXDOS**. Apresentando o "*prompt*" (A>).

CALL VERIFY ON/OFF



Chama procedimento da interface de disco que ativa ou desativa a verificação de escrita. Se for escolhida a opção "ON", então a verificação é ativada, senão (OFF), ela é desativada.

Esta opção é aconselhada, para garantir a correta gravação dos dados.

Este comando só é obtido a partir do **DISK-BASIC**.

Exemplo: CALL VERIFY ON

CDBL (expressão)

Converte a expressão em um dado numérico de precisão "dupla".

Exemplo: C=CDBL(12.34!):PRINT C

CHR$ (expressão)

Função inversa a "**ASC**". Fornece o caractere correspondente ao valor da expressão na faixa de 0 a 255.

Exemplo: FOR F=128 TO 255:PRINT F;CHR$(F),:FOR G=1 TO 200:NEXT G,F

Os códigos e caracteres de 128 a 255 serão impressos na tela.

CINT (expressão)

Converte a expressão em um número inteiro, desprezando as casas à direita do ponto decimal, na faixa de – 32768 a +32767.

Exemplo: A=3.2:B=3.8:PRINT CINT(A);CINT(B);–CINT(–A);–CINT(–B)

Veja os resultados e compare com os do exemplo do **FIX**.

CIRCLE [STEP] (var1,var2),raio [,cor][,início][,fim][,proporção]

Traça na tela gráfica, círculos, elípses e setores.

var1 –Expressão que define o valor da abscissa do centro.

var2 –Expressão que define o valor da ordenada do centro.

raio –Expressão entre – 32768 e 32767 que indica o raio.

cor –Indica a cor, entre 0 e 15, do arco traçado.

início –Início do arco, em *radianos*. Se negativo traça um setor.

fim –Final do arco, em *radianos*. Se negativo traça um setor.

proporção –Relação entre absissas e ordenadas; se diferente de um, forma elípses.

Exemplo:10 SCREEN 2:CIRCLE(128,96),20:CIRCLE(128,96),20,,,,.2
20 GOTO 20

Desenha um círculo e uma elípse no centro da tela.

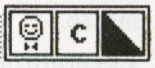
CLEAR (área de string,RAMTOP)

Inicializa todas variáveis, arquivos e define áreas de armazenamento de *strings* e topo de memória **RAM**.

Exemplo: CLEAR 600,50000

Inicializa variáveis e arquivos, define 600 bytes para string's e define 50000 como último endereço disponível para o **BASIC MSX**.

CLOAD[?] ["nome arq"]



Transfere dados em **BASIC MSX** gravados em cassete, para o computador. Se a opção "?" for usada, faz a verificação do arquivo proveniente da fita com o da memória. Se ocorrer uma falha, a mensagem "Verify error" será impressa. Se for definido o "nome arq" ele aguarda a ocorrência do arquivo na fita, e em seguida carrega-o para a memória.

Exemplo: CLOAD?

Compara a primeira ocorrência na fita com o programa da memória.

CLOSE [#][numarq1][,numarq2][,. . .

Fecha um ou mais arquivos abertos pelo "**OPEN**". Se for especificado o número do arquivo, então apenas o citado será fechado, senão, todos.

Exemplo: CLOSE #1

Fecha apenas o arquivo 1. Veja "**OPEN**".

CLS

Limpa toda a tela, e coloca o cursor no canto esquerdo da tela.

Exemplo: CLS:FOR I=0 TO 21:PRINT"GRADIENTE":NEXT I:CLS

CMD

Comando não implementado. Pode ser implementado pelo usuário com bons conhecimentos em programação, para comandos próprios.

COLOR [frente],[fundo],[borda]

Especifica as cores da tela, onde "frente" define a cor dos caracteres, "fundo" a cor de fundo e, "borda", a cor da borda, segundo a tabela:

Exemplo: COLOR 12,15,6

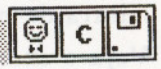
Define frente verde escuro, fundo branco, e borda vermelha escura.

0– Transparente	4– Azul Escuro	8– Vermelho	12– Verde escuro
1– Preto	5– Azul Claro	9 – Verm. claro	13– Magenta
2– Verde	6– Verm.escuro	10– Ouro	14– Cinza
3– Verde claro	7– Ciano	11– Amarelo	15– Branco

CONT

Continua a execução de um programa, a partir da linha seguinte àquela onde o programa foi interrompido pelo **[CONTROL] + [STOP]**.

COPY "[F:] nome1" TO "{[D:] [nome2]}"



Comando usado para a cópia de arquivos em disquete, onde:

F: –Drive onde se encontra o arquivo fonte (se for omitido assume o corrente);

nome1 –Nome do arquivo fonte. Pode se usar as opções "*" ou "?" (veja observação a seguir);

D: –Drive de Destino (se for omitido assume corrente);

nome2 –Nome novo assumido durante a cópia (se omitido, mantém nome1). Se o arquivo já estiver presente no disquete, o arquivo anterior será perdido, e automaticamente renovado. Se o disquete estiver protegido, a gravação não será feita e a respectiva mensagem será exposta na tela.

OBS: –Os nomes podem ser substituídos pelos caracteres "*" e "?", onde "*" substitui um nome e "?" substitui uma letra de modo a poder se copiar blocos de arquivos.

Exemplo: COPY "A:TEST*.B?T" TO "B:"

Faz a cópia de todos os arquivos que comecem com "TEST...." com extensão "B_T" (BAT,BBT,BCT,...,BZT).

Observe que se você tiver somente um acionador instalado, o sistema de disco se encarrega de expor mensagens de troca dos disquetes. Para cópias extensas dê preferência ao "COPY" do **MSXDOS** (veja Cap.7).

COS(expressão)

Calcula o valor do cosseno de um arco em radianos, fornecido pela expressão.

Exemplo: PI=4*ATN(1):PRINT COS(PI/3)

Imprime o valor do cosseno de PI/3 radianos (60 graus).

CSAVE "(nom arq)" [,transmissão]

Transfere programas em **BASIC MSX** para o cassete com um nome de 6 letras (nom arq) e tipo de transmissão 1 ou 2 (1200/2400 bauds).

Exemplo: CSAVE "PROG1",2

Grava o programa da memória com o nome "PROG1" e velocidade de transmissão de 2400 bauds (bits p/ segundo).

OBS: No nome, as minúsculas são consideradas diferentes das maiúsculas.

CSNG (expressão)

Converte o valor da expressão num número em precisão simples.

Exemplo: A!=CSNG(1.234567891):PRINT A!

CSRLIN

Fornece o número da linha onde se encontra o cursor.

Exemplo: PRINT CSRLIN,CSRLIN

CVD(var)

Função que converte variáveis do tipo "string" (var) em precisão dupla, para uso na leitura de números gravados em arquivos de acesso *aleatório* (operação inversa de **MKD$**).

Veja o exemplo em **CVS**.

CVI(var)

Função que converte variáveis do tipo "string" (var) em variável inteira, para uso na leitura de números gravados em arquivos de acesso aleatório (inverso do **MKI$**).

Veja o exemplo de **MKD$**.

CVS(var)

Função que converte variáveis (var) do tipo "string" em variável de precisão simples, para uso na leitura de números gravados em arquivos de acesso aleatório (inverso do **MKS$**).

```
Exemplo:10 OPEN "A:TESTE.DAT" AS #1 LEN=14
        20 FIELD #1,8 AS D$,4 AS S$,2 AS I$:GET #1
        30 D=CVD(D$):S=CVS(S$):I=CVI(I$):PRINT D;S;I
        40 CLOSE #1:END
```

O programa lê um arquivo *aleatório*, com os registros de variáveis de precisão dupla, simples e inteiros, e imprime-os na tela.

DATA dado [,dado2,. . .]

Define uma lista de dados numéricos ou alfa-numéricos separados um a um por vírgula, para serem lidos posteriormente pelo "**READ**".

```
Exemplo:10 FOR F=1 TO 10:READ A$,A:PRINT A$,A:NEXT F
        20 DATA Fe,55.8,Cl,35.5,O,16.0,P,31.0
```

Lê e imprime os dados das linhas "**DATA's**". (Veja comando **READ** e **RESTORE**).

OBS: se um dos caracteres a ser lido for a própria vírgula, ela deve ser colocada entre aspas.

DEF

Veja **DEF FN, DEF INT, DEF SNG, DEF DBL, DEF USR.**

DEF FN nome [(parâmetros)] = expressão

Define uma função "nome" criada pelo usuário. Até 9 parâmetros podem ser passados, para serem usadas como variáveis para calcular a expressão. Não pode ser usado como comando direto, e ao chamar a função, o nome deverá ser precedido por **FN**.

```
Exemplo:10 DEF FN E(X,Y)=X+2*Y
        20 PRINT FN E(-3,2)
```

Obtém o resultado 1

DEF INT/SNG/DBL/STR caractere [-caracter] [,caracter] [,. . .]

Define o tipo de variável correspondente à primeira letra do nome. INT corresponde a inteiras, SNG a precisão simples, DBL precisão dupla e STR a alfanuméricas (string).

```
Exemplo: DEFSTR A-D:DEFDBL E,X-Z:DEFINT F-L:DEFSNG M
```

Variáveis com nome iniciando com letras entre A e D, são alfanuméricas; E e entre X e Z, precisão dupla; entre F e L, inteiras; e M do tipo precisão simples.

DEF USR[X]=endereço

Define o ponto de entrada de uma rotina em *linguagem de máquina*, onde X (entre 0 e 9) corresponde o número da entrada, e o endereço um inteiro. Para chamar a rotina em L.M. veja "**USR**".

Exemplo: DEFUSR0=0:DEFUSR8=40000

Define entrada 0 com endereço 0 e, entrada 8 com endereço 40000.

DELETE {[num1] [- num2]}[.]

Apaga linhas de um programa em BASIC: somente 1 linha (num1); bloco de linhas (num1-num2); do início ate linha (-num2); ou última linha inserida (opção ".").

DIM nome1 (indice1 [,indice2,...]) [,nome2....]

Define variável "nome" como tipo matriz, sendo que a sua dimensão depende do número de "índices" especificados. O valor do índice especifica o número de elementos relativos à matriz.

Exemplo: DIM A$(6),X(3,4,5)

Define uma matriz alfanumérica de 6 elementos e uma matriz numérica de dimensão 3 com 3x4x5 elementos.

DRAW "sub-comandos"

Desenha na tela gráfica figuras criadas pelo usuário, seguindo as ordens especificadas em "sub-comando", em ordem sequencial, de maneira que cada comando desenha a partir do último ponto marcado na tela.

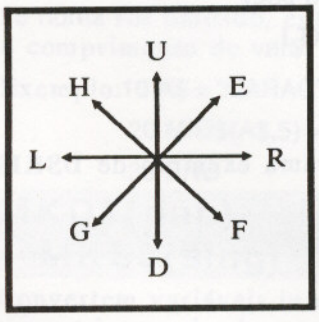
O comando **DRAW** permite vários tipos de sub-comandos gráficos, listados a seguir, com suas respectivas definições:

Sx –x define a escala do desenho (default=4);

Ax –x de 0 a 3: define a orientação dos eixos (default=0);

Cx –x de 0 a 15: define a cor do traço (veja **COLOR**);

Ux –Move o ponto-cursor para cima, desenhando. Além de U existem outros 7 sub-comandos que desenham nas direções indicadas na figura:



B – Quando colocado antes de um sub-comando, desloca o ponto-cursor sem traçar nada;

N – Quando colocado antes de um sub-comando, não atualiza as coordenadas (mantem as anteriores);

Mx,y – Move o ponto-cursor para as coordenadas (x,y) da tela (não é afetado pelo "S");

M±x,±y – Move o ponto-cursor para as coordenadas X0±x e Y0± y, onde X0 e Y0 são as coordenadas anteriores (é afetado pelo "S");

**X A$;** – Permite inserir o conteúdo de A$ na lista de sub-comandos.

=V; – Permite inserir o valor da variável V na string de sub-comandos.

Exemplo:
```
10 SCREEN 2:FOR I=0 TO 4:PSET(160+I,90+I)
20 FOR S=4 TO 28 STEP 4:A=(S/4)MOD 4
30 DRAW "S=S;A=A;U4F2E2D4BR3R3E1H1L2H1E1R3BR3F4BU4G4BE8":NEXT S,I
40 GOTO 40
```

Analise cada parte do programa acima, e observe a flexibilidade do **DRAW**.

Para interromper use **[CONTROL]+[STOP]**.

DSKF(arg)

Função que retorna o espaço disponível no disco.

Se "arg" for 0, é assumido o drive corrente, 1 drive A, 2 drive B e assim sucessivamente, até o máximo de 6.

Exemplo: PRINT DSKF(1)

Indica quantos KBytes estão disponíveis no drive A.

DSKI$ (num1,num2)

Função que armazena num buffer o setor "num2" lido no drive "num1".

"Num1" pode variar de 0 para drive default, 1 para o drive A e assim sucessivamente, até 6,para o drive F.

"Num2" indica o setor lógico, que varia de 0 a 719 para 80 trilhas/face simples, e de 0 a 1439 para 80 trilhas/face dupla.

Para localizar o endereço inicial (EN) do buffer onde se encontra o setor de 512 bytes, proceda da seguinte maneira:

```
EN = PEEK(&HF351) + 256 * PEEK(&HF352)
```

Exemplo:

```
10 A$ = DSKI$(0,0):EN = PEEK(&HF351) + 256 * PEEK(&HF352)
20 FOR F=0 TO 511 STEP 8:FOR G=0 TO 7
30 A=PEEK(EN+F+G):IF A<32 THEN PRINT CHR$(1);CHR$(A+64);ELSE PRINT
CHR$(A);
40 NEXT G,F:END
```

Mostra o conteúdo do setor 0 (setor de especificação).

DSKO$ num1,num2

Comando do **DISK- BASIC** que grava o setor lógico "num2" no drive "num1".

Esta operação só ocorrerá com sucesso se o disquete estiver sem proteção de gravação.

Veja DSKI$ para detalhes do "num1", "num2" e sobre o endereço a partir de onde o sistema de disco copia os dados.

Exemplo:

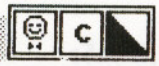
```
10 A$=DSKI$(0,0):EN=PEEK(&HF351)+256*PEEK(&HF352)
20 FOR I=EN+500 TO EN+508:READ A:POKE I,A:NEXT I
30 DSKO$ 0,0
100 DATA 71,82,65,68,73,69,78,84,69
```

Execute este programa e, a seguir, digite e execute o programa exemplo de **DSKI$**. Veja o resultado obtido no final do setor 0.

ELSE

Veja "**IF**".

END



Termina a execução de um programa, fechando todos os arquivos e retorna ao modo de edição. Costuma ser colocado no final do programa, ou para separar a rotina principal das sub-rotinas.

EOF

Função que sinaliza o fim de um arquivo: caso este for detectado, a função retorna o valor –1, senão retorna 0.

Veja o exemplo de **INPUT#**.

EQV

Operador lógico que testa dois argumentos bit a bit. Se os bits forem iguais gera um bit 1, caso contrário gera um bit 0.

Exemplo: PRINT RIGHT$("0000000"+BIN$(&B01010101 EQV &B11000011),8)

O resultado será 01101001. O exemplo acima ilustra ainda as técnicas obtidas pelo **RIGHT$**, formatando a saída para 8 dígitos.

ERASE nome1 [,nome2. . .]

Apaga da memória, a variável "nome" do tipo matriz.

```
Exemplo:10 DIM A(6):FOR I=1 TO 6:A(I)=7-I:NEXT I
        20 FOR I=1 TO 6:PRINT A(I):NEXT I:ERASE A
        30 FOR I=1 TO 6:PRINT A(I):NEXT I:END
```

ERL / ERR

Variáveis que fornecem o número da linha em que ocorreu erro, e o número do erro respectivamente. **ERL** fornece 65535, se o erro ocorreu em um comando direto.

Exemplo: 10 LINHA ERRADA

Ao executar o programa (**RUN**), ocorrerá um erro. Digite então:

PRINT ERL;ERR

ERR

Veja **ERL**.

ERROR número

Permite que o usuário simule um erro, e defina o "número" deste.

```
Exemplo:10 ON ERROR GOTO 1000:PRINT "DIGITE '1'"
        20 A$=INPUT$:IF A$<>"1" THEN ERROR 230
        30 PRINT "OBRIGADO":END
        1000 PRINT "EU DISSE '1'":RESUME 10
```

OBS: Veja também RESUME

EXP (argumento)

Função que fornece o valor do exponencial natural do "argumento", ou seja, o *número de Euler* "e" (2.71828) elevado ao argumento.

Exemplo: PRINT EXP(1)

Que fornece como resultado o *número de Euler*.

FIELD [#] numarq,bytes AS var1$ [,bytes2 AS var2$] [bytes3. . .]

Usado em arquivos de acesso aleatório. Define os campos de um registro, somente para variáveis do tipo string.

numarq –Número do arquivo aberto pelo **OPEN** (verifique!).

bytes –Define o comprimento do campo. A soma destes não pode exceder o valor declarado pelo **LEN** no comando **OPEN**.

var1$ –Nome da variável (tipo *string*) a receber o registro. Esta variável deve ser exclusiva para este fim.

FIELD –É obrigatório num arquivo aleatório antes do **GET** (leitura) e o **PUT** (escrita).

Veja a linha 116 do exemplo 3 no apêndice A.

FILES ["nom. ext"] / LFILES ["nom. ext"]

Comando usado para listar na tela, todos os arquivos de um disquete.

Pode-se listar apenas um, se ele existir, especificando-se o nome e a extensão (nom.ext). Os caracteres "*" e "?" podem ser usados, onde "*" substitui palavra, e "?" letra.

LFILES funciona identicamente a **FILES**, sendo que os dados são enviados para a impressora ao invés do vídeo.

Exemplo: FILES "*.BIN"

Lista todos os arquivos com extensão BIN.

FIX (argumento)

Função que fornece a parte inteira do argumento, desprezando os dígitos após o ponto.

Exemplo: PRINT FIX(3.8);FIX(3.2);FIX(-3.2);FIX(-3.8)

Veja os resultados e compare com os dos exemplos do "**INT**" e "**CINT**".

FN função

Executa uma função definida pelo usuário. Não pode ser usado no modo direto.

Veja também **DEF FN**.

Exemplo:10 DEF FN COT(X) = 1/TAN(X)

20 PI = 4*ATN(1):X = PI/2

30 PRINT "COT(PI/2) = "; FN COT (X)

FOR var=ini TO fim [STEP passo] / NEXT [var [,var2]]

Este dois comandos formam uma das estruturas básicas da programação. "Var" é uma variável usada como contadora, que varia de "ini" até "fim" com um incremento de "passo" (não necessariamente inteiro), repetindo assim o bloco entre "**FOR**" e "**NEXT**". O "**NEXT**" pode apontar mais de uma variável. O valor "*default*" de "**STEP**" é 1.

Exemplo:10 FOR I = 1 TO 2 STEP .1

20 PRINT "EXP(";I;") = ";EXP(I)

30 FOR J = 0 TO 500:NEXT J,I:END

Ilustra a técnica do **FOR/NEXT**, expondo os valores da função no intervalo 1 e 2. Observe como facilita a criação de tabelas e gráficos.

FORMAT

Veja **CALL FORMAT**

FPOS

Função não implementada.

FRE (argumento)

Função que retorna a área de memória livre (em bytes), se o argumento for um número. Se o argumento for do tipo "*string*", fornece a área de "*string*" livre, em bytes.

Exemplo: PRINT FRE(0);FRE("")

GET [#] numarq [,numreg]

Lê um registro de um arquivo de acesso aleatório já aberto.

Numarq determina o número do arquivo aberto em uso. Numreg define o número do registro a ser lido.

O registro lido é transferido para as variáveis string definidas pelo **FIELD**.

Veja o programa-exemplo nº3 do apêndice A.

GO TO linha

Veja **GOTO**.

GOSUB linha1 / RETURN [linha2]

Transfere a execução do programa para "linha1", e executa então a "sub-rotina" até o ponto que encontrar o comando "**RETURN**", retornando assim de onde havia partido. Se "linha2" for especificada, então a execução reinicia-se na linha indicada.

```
Exemplo:10 PRINT "INICIO":GOSUB 500
        20 PRINT "MEIO":GOSUB 500
        30 PRINT "FIM":END
        500 PRINT "*** SUBROTINA ***"
        510 RETURN
```

O uso de sub-rotinas é adequada quando a mesma tarefa deve ser executada diversas vezes, evitando assim redigitar todo o trecho novamente.

GOTO linha

Comando que transfere a execução para a "linha" especificada.

```
Exemplo:10 PRINT"GRADIENTE - ";
        20 GOTO 10
```

Este programa só é interrompido com **[CONTROL]+[STOP]**.

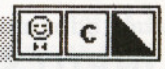
HEX$(argumento)

Converte o "argumento", em uma "*string*" contendo o valor deste, em notação hexadecimal (base 16). É útil quando se trabalha em "*Assembly*".

```
Exemplo: FOR F=0 TO 16:PRINT F;"decimal - ";HEX$(F);" hexadecimal":NEXT F
```

Mostra uma pequena tabela de conversão decimal-hexadecimal.

IF condição THEN comando1 [ELSE comando2]



Permite a verificação de situações divergentes. A "condição" é verificada e se for verdadeira executa "comando1", se for falsa executa "comando2"; se **ELSE** for omitido executa linha seguinte.

```
Exemplo:10 PRINT"Tecle '7'":A$=INPUT$(1)
        20 IF A$="7" THEN PRINT "CORRETO" ELSE PRINT"ERRADO!":GOTO 10
        30 END
```

Observe que esta estrutura é muito importante na tomada de decisões.

INKEY$

Função que retorna o caracter da tecla pressionada. Se não houver tecla pressionada retorna "*string*" nula.

```
Exemplo:10 A$=INKEY$
        20 PRINT A$;:GOTO 10
```

Simula uma máquina de escrever. Pare com **[CONTROL]+[STOP]**.

IMP

Operador lógico de implicação. Comparando dois *bits*, retorna 0 apenas se somente o segundo *bit* for 0. Caso contrário retorna 1.

```
Exemplo: PRINT RIGHT$("0000000"+BIN$((&B11110000)IMP(&B11000011)),8)
```

Deve retornar: 11001111

INP (argumento)

Lê a porta da via de acesso especificada no argumento. Este comando só tem utilidade para aqueles que estão bem familiarizados com o "*hardware*" do **MSX**.

Exemplo: PRINT BIN$(INP(&HA8))

Imprime a atual configuração dos "*slots*".

INPUT

Veja as variações do **INPUT**: **INPUT**, **INPUT#** e **INPUT$**.

INPUT ["mensagem";] var [,var2. . .]

Imprime (na tela) a mensagem, se for especificada, e lê um valor dado pelo teclado e copia na variável "var". Se forem especificados mais variáveis, os seus respectivos valores devem ser separados por vírgula. **[RETURN]** termina a inserção.

Comando ideal para facilitar a entrada de dados pelo usuário, sem que esse, conheça necessariamente a estrutura do programa.

```
Exemplo:10 INPUT "Entre nome e número ";A$,N
        20 PRINT "Nome: ";A$,"Número:";N
        30 PRINT:GOTO 10
```

Lê uma variável do tipo "string" e outra do tipo numérico e as imprime no vídeo.

INPUT # número,lista

Lê dados de um arquivo especificado por "número", associando cada dado às variáveis especificadas na "lista".

```
Exemplo:10 OPEN "CAS:TESTE" FOR INPUT AS #1
        20 IF EOF< >0 THEN 100
        30 INPUT #1,A$,N
        40 GOTO 20
        100 CLOSE #1:END
```

Lê todos nomes (A$) e números (N) de um arquivo em cassete.

INPUT$(argumento) [,# arq]

Lê um número de caracteres pelo teclado, especificado pelo "argumento". Se for especificado o número do arquivo "arq", então a leitura é feita através do arquivo.

Exemplo: A$=INPUT$(10):PRINT A$

INSTR([N,] string1,string2)

Localiza a posição (1º caracter) de "string2" em "string1"; se não for encontrado retorna 0. Se "N" especificado, então inicia a procura a partir do n-ésimo caracter da "string1".

Exemplo: PRINT INSTR("GRADIENTE & ALEPH","ENTE")

Retorna o valor 6.Experimente outras "strings".

INT (argumento)

Função que converte o argumento em um número inteiro, arredondando sempre para MENOS. Difere de **CINT** e **FIX**.

Exemplo: PRINT INT(3.8);INT(3.2);INT(-3.8);INT(-3.2)

Compare os resultados com os obtidos nos exemplos de **CINT** e **FIX**.

INTERVAL ON/ OFF/ STOP

Comando que habilita, desabilita ou adia (respectivamente) a interrupção feita pelo temporizador interno.

Veja o exemplo no comando "**ON INTERVAL GOSUB**"

IPL

Comando não implementado no **BASIC MSX** padrão.

KEY

Veja as variações de **KEY: KEY, KEY LIST, KEY ON/ OFF** e **KEY (n) ON/ OFF/ STOP.**

KEY tecla,string

Associa a "string" à "tecla" de função (**F1** a **F10**).

Exemplo: KEY 1,"GRADIENTE":KEY 2,"ALEPH"

A seguir tecle **[F1]** e **[F2]** e observe o resultado.

KEY LIST

Lista o conteúdo associado à cada uma das 10 teclas de função.

Exemplo: KEY LIST

KEY ON/ OFF

Ativa e desativa (respectivamente) a visualização na parte inferior do vídeo dos conteúdos das teclas de função.

Exemplo: KEY OFF

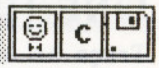
KEY (num) ON/ OFF/ STOP

Comando que ativa, desativa ou adia (respectivamente) a interrupção via teclas de função.

"Num" indica a qual tecla de função deve-se atribuir a interrupção.

Veja o exemplo em **ON KEY GOSUB.**

KILL "[D:] nom. ext"



Comando que apaga um arquivo do diretório do disco, com o nome "nom.ext". "Nom" deve conter até 8 letras, "ext" até 3 letras.

Os caracteres "*" e "?" podem ser usados, sendo que "*" substitui uma palavra e "?" uma letra

"D:" indica o drive onde se encontra o arquivo (**A..F**). Se for omitido, indica o drive corrente.

```
Exemplo:KILL"TESTE.TXT"
        KILL"*.*"
```

O primeiro apaga o arquivo TESTE.TXT e o segundo todos os arquivos do disquete.

Comando semelhante ao **DEL** do **MSXDOS.**

LEFT$ (string,n)

Fornece como resultado os n primeiros caracteres da "*string*".

Exemplo: A$=LEFT$("MICROCOMPUTADOR",5):PRINT A$

O resultado é: MICRO. Veja também **MID$** e **RIGHT$.**

LEN

Veja as variações em: **LEN**(string) e **OPEN.**

LEN (string)

Função que retorna o comprimento da "string" em questão.

Exemplo: PRINT LEN("MICROCOMPUTADOR")

Retorna 15, pois MICROCOMPUTADOR tem 15 letras.

[LET] variável=expressão

Atribui o valor da expressão a uma variável. **LET** pode ser omitido no **BASIC MSX**. Deve ser usado quando quer se gerar um programa mais "universal", cujo texto possa ser lido por outros micros.

O **LET** pode ser considerado o comando básico para qualquer sistema. Através dele pode-se iniciar constantes; atualizar cálculos de fórmulas matemáticas; aceita funções matemáticas, lógicas e outras; trabalha com variáveis numéricas ou cadeias de caracteres (*string*), entre outros.

```
Exemplo:10 A=3:LET B=A*A:PRINT A;" ^ 2 =";B
        20 PRINT "4 x";A;"= ";:A=A+A+A+A:PRINT A
        30 PRINT "4 x";A;"= ";:A=4*A:PRINT A
        40 B$="EDI":C$="ALE":LET B$=B$+"TORA "+C$+"PH":PRINT B$
```

Aqueles que se iniciam em programação, devem procurar conhecer bem as potencialidades deste comando. Observe que dentro do mesmo comando deve-se ter só um tipo de valor (numérico ou alfanumérico) –para eventuais conversões veja **STR$** e **VAL**.

LFILES

Veja **FILES**.

LINE

Veja as variações em: **LINE**, **LINE INPUT** e **LINE INPUT#**.

LINE [[STEP] (X0,Y0)]-(X1,Y1) [,cor] [,B][,BF]

Traça no modo gráfico, uma linha entre os pontos de coordenadas (X0,Y0) e (X1,Y1), com cor especificada. Se **B** for especificado desenha um quadrado; **BF** desenha um quadrado com o interior pintado. A opção **STEP** faz o comando assumir o último ponto desenhado, como o primeiro par de coordenadas do comando.

```
Exemplo:10 SCREEN 2:LINE (10,10)-(245,182),15,BF
        20 LINE (50,50)-(205,142),,B:LINE STEP (0,0),6
        30 GOTO 30
```

Mostra os 3 tipos de opções do comando **LINE**.

LINE INPUT ["mensagem"] variável

Comando semelhante ao **INPUT** (veja **INPUT**), com a diferença de não imprimir o ponto de interrogação no ínicio da entrada e só aceitar variáveis alfanuméricas.

```
Exemplo: LINE INPUT A$:PRINT "COMPRIMENTO:";LEN(A$)
```

Lê uma "*string*" e imprime o seu comprimento.

LINE INPUT #num,variável

Lê uma "*string*" do arquivo numerado por "num", e atribui o conteúdo a uma variável de mesmo tipo.

```
Exemplo:10 OPEN "CAS:TESTE" FOR INPUT AS #1
        20 IF EOF THEN 100
        30 LINE INPUT #1,A$:PRINT "NOME: ";A$:GOTO 20
        100 CLOSE #1:END
```

Lê todos os nomes e imprime-os na tela.

LIST [{[num1] –[num2]}] / LLIST [{[num1] –[num2]}]

Lista o programa entre as linhas num1 e num2. Sem opção, lista todo o texto. As outras opções são:

–num2 –Lista do início até num2;

num1 – –Lista à partir de num1;

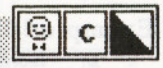
–Lista a última linha inserida, alterada, ou executada.

LLIST é semelhante ao **LIST**, sendo que a listagem é enviada a impressora ao invés da tela.

Exemplo: LIST 10-30

Lista as linhas entre 10 e 30 inclusive, se existirem.

LLIST



Veja **LIST**.

LOAD "disp nome" [,R] –(forma genérica)

Comando que carrega um programa em **BASIC MSX** do dispositivo (disp) para a memória do micro, que tenha o mesmo nome (nome), se especificado.

Este comando tem comportamento diverso dependendo da configuração do micro:

Para um sistema baseado em cassete ele deve ter o seguinte formato:

LOAD "CAS:[nome]" [,R]

"Nome" pode ser omitido, e neste caso carrega o primeiro arquivo encontrado; senão aguarda até a ocorrência do arquivo com o mesmo nome, e só então os dados são transferidos para a memória do micro. O nome deve ter no máximo 6 letras, sendo que maiúsculas e minúsculas são consideradas letras diferentes.

O formato de gravação na fita é o **ASCII**.

Exemplo: LOAD "CAS:TESTE",R

Para um sistema baseado em disco, deve ter o seguinte formato:

LOAD "[D:] nome" [,R]

"D:" pode ser omitido, e neste caso assume que o drive em questão seja o drive corrente; senão sera especificado com uma letra entre **A** e **F**, indicando a partir de que drive deve ser lido o arquivo.

O "nome", por sua vez, deve sempre ser especificado, e este é procurado no diretório e se for encontrado é verificado se é do mesmo formato (existe o binário usado com **BLOAD**), e só então é carregado para a memória do micro.

O "nome" deve ser constituído de duas partes: "nom" (nome) e "ext" (extensão), onde "nom" tem até 8 letras, e "ext" tem até 3 letras ("ext" pode ser omitido). Estas duas partes devem ser separados por ponto ".". Letras maiúsculas ou minúsculas serão convertidas em maiúsculas.

Para ambos os casos a opção "R", faz com que o programa seja executado automaticamente.

Exemplo: LOAD "B:TESTE.BAS"

Carrega o programa TESTE.BAS, se existir no diretório, na memória do micro.

LOC (numarq)

Função que retorna o número do último registro gravado ou lido de um arquivo de acesso aleatório.

Veja o programa-exemplo nº3 do apendice A.

LOCATE num1,num2,num3

Comando que posiciona o cursor na coluna num1 e na linha num2 da tela de texto. "Num1" deve estar entre 0 e 39 na **SCREEN 0** e entre 0 e 31 na **SCREEN 1**.

"Num3" controla a presença ou não do cursor durante a impressão. 1 ativa e 0 desativa. Normalmente é 0 mas alguns programas colocam-no em 1 (caso do **MSXDOS**).

Exemplo: LOCATE 20,10:PRINT "EXPERT DD PLUS"

Coloca o texto PRINT nas coordenadas (20,10) da tela.

LOF (numarq)

Função que retorna o comprimento do arquivo de número "numarq" (desde que tenha sido aberto por **OPEN**) em bytes.

```
Exemplo:10 OPEN "A:TESTE.DAT" AS #1
           20 X=LOF(#1):CLOSE #1
           30 PRINT "COMPRIMENTO=";X: END
```

Imprime na tela o comprimento do arquivo TESTE.DAT.

LOG (argumento)

Função que retorna o logaritmo natural (base e) ou *neperiano* do argumento.

```
Exemplo: X=3:PRINT LOG(X);LOG(X)/LOG(10)
```

Fornece o logaritmo natural e decimal de X.

LPOS (argumento)

Função que, independente do valor do argumento, retorna a posição do cabeçote da impressora.

```
Exemplo:10 A$="":FOR F= 0 TO 60
           20 LPRINT A$;LPOS(1)
           30 A$=A$+">":NEXT F: END
```

LPRINT

Funciona igualmente ao **PRINT**, com a diferença de direcionar a impressão para a impressora ao invés da tela. Veja **PRINT** para sintaxe.

LPRINT USING

Funciona igualmente a **PRINT USING**, com a diferença de direcionar a saída para a impressora ao invés da tela. Veja **PRINT USING**.

LSET var$=expressão string / RSET var$=expressão string

Comando que transfere a expressão string para a variável de saída (var$) definida pelo comando **FIELD**. Os dados são alinhados à esquerda em **LSET** e alinhados à direita em **RSET**.Veja o programa-exemplo 3 no apêndice A.

MAX

Veja a variação **MAXFILES**.

MAXFILES=expressão

A expressão deve estar entre 0 e 15 e limita o número de arquivos que poderão ser abertos por **OPEN**. Se for especificado 0 então nenhum arquivo pode ser aberto, funcionando porém o **SAVE** e o **LOAD**.

```
Exemplo:10 MAXFILES=1:OPEN"GRP:" AS #1
           20 SCREEN 2:PRINT#1,"SCREEN 2"
           30 GOTO 30
```

Este programa escreve um pequeno texto na tela gráfica. Para testar o uso de **MAXFILES** acrescente a linha:

```
15 OPEN "CRT:" AS #2
```

Execute o programa e verifique que ocorreu erro!!

MERGE "disp: nome"

Este comando soma um arquivo gravado no dispositivo (disp), no formato **ASCII**, com o programa da memória do micro. As linhas que tiverem números iguais serão substituídas pelas linhas do programa externo.

Se o sistema for baseado em cassete, então o programa externo deve ter sido gravado com **SAVE** (**CSAVE** ou **BSAVE** têm formato incompatível com **MERGE**).

No **DISK-BASIC** o programa externo deve ser gravado com **SAVE** e a opção **A**.

Para "disp" e "nome" consulte **LOAD** (Atenção: eles variam no cassete ou no disco).

Exemplo: MERGE "cas:TESTE"

MID$

MID$** possue duas variações: **MID$ e **MID$ =**. Veja a seguir.

MID$ (var$,num1 [,num2])

Seleciona um subconjunto de caracteres de uma string.

var$ —Variável do tipo string (ou cadeia de caracteres). Pode ser uma string constante.

num1 —Indica a posição inicial, a partir de onde será extraído o subconjunto.

num2 —Determina o comprimento da nova string. Se for omitido, a *string* resultante será formada por todos os caracteres à partir da posição "num1".

Exemplo: PRINT MID$("GRADIENTE",3,4)

O que resulta: ADIE. Para melhor manipulação de pedaços de string, veja ainda **LEFT$**, **RIGHT$**, **LEN**.

MID$(var$,num1 [,num2]) = var2$

Substitue em var$, a partir da posição num1, num2 caracteres de var2$.

Se num2 for omitido, então todos os caracteres de var2$ serão enxertados, até completar o comprimento de var$.

Exemplo:
```
10 A$ = "CARACTER":B$ = "VELAME"
20 MID$(A$,5) = B$:PRINT A$
30 END
```

MKD$(dup) / MKI$(int) / MKS$(sing)

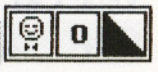
Convertem variáveis inteiras (**MKI$**), precisão simples (**MKS$**) ou dupla (**MKD$**) para tipo string onde devem ser transferidas para a string de saída definida em **FIELD**. Operam para arquivos aleatórios a são inversas à **CVD**, **CVI**, **CVS**.

Veja o exemplo 3 do apêndice A, linhas 230 a 234.

MKI$

Veja **MKD$** acima.

MOD

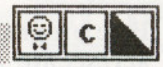


Operador que retorna o resto inteiro da divisão de dois números, variáveis, constantes ou expressões.

Exemplo: PRINT 8 MOD 3;15 MOD 5

Que resulta 2 e 0.

MOTOR ON/OFF

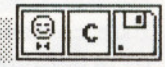


Comando que controla o estado do relé para o motor do cassete, ou seja liga (ON) ou desliga (OFF) o motor do cassete.

Exemplo: MOTOR OFF

Paralisa o gravador, se este estiver ligado.

NAME "[d:] nome1" AS "[d:] nome2"



Comando que renomeia um arquivo do diretório do disco. O arquivo "nome1" passa então a se chamar de "nome2"

"D:" especifica o drive (A-F) onde o arquivo se encontra, se for omitido será assumido o drive corrente.

"*" e "?" podem ser usados para substituir palavra e letra respectivamente.

Exemplo: NAME "TESTE.TXT" AS "PROG1.BAS"

NEW

Reinicia a memória do micro, apagando o programa **BASIC**. Não costuma afetar programas em linguagem de máquina que estiverem na memória.

Exemplo: NEW

Se existir algum programa **BASIC** na memória, este será perdido. Observe que este comando não altera o conteúdo dos periféricos (cassete, Drive, etc.).

NEXT

Veja comando **FOR**.

NOT (expressão lógica)

Uma expressão lógica pode ser constituída por vários testes (=, >, <, **AND**, **OR**, etc.), e retorna apenas dois estados; são estes verdadeiro ou falso. O computador define 0 como falso e -1 como verdadeiro.

O operador lógico **NOT** simplesmente fornece o "*não*" lógico da expressão ou seja "não verdadeiro = falso" e "não falso = verdadeiro".

Isto significa que, se uma expressão lógica retorna um estado, **NOT** o inverte.

Exemplo: PRINT NOT(4>5);NOT(4<5)

Também inverte os "bits" de um número inteiro binário

Exemplo: PRINT BIN$(NOT(&B01100000))

O que resulta 1111111110011111 (os 8 1's que apareceram na frente, se devem ao fato do **MSX-BASIC** manipular os números inteiros em 16 bits).

OCT$(expressão)

Função que retorna uma cadeia de caracteres, contendo a notação octal (base 8) do valor da expressão inteira.

Exemplo: PRINT OCT$(16)

Converte o valor decimal 16 no valor da base octal (Veja **&O**).

OFF

OFF significa "*desligado*", e é encontrado apenas em conjunto com outros comandos. Veja: **CALL VERIFY, KEY, INTERVAL, MOTOR, SPRITE, STOP** e **STRIG**.

ON

ON significa "*ligado*" *ou "em", e é encontrado como comando e em conjunto com outros comandos. Veja:* **CALL VERIFY, KEY, INTERVAL, MOTOR, SPRITE, STOP, STRIG, ON ERROR, ON GOSUB, ON GOTO, ON INTERVAL, ON KEY, ON SPRITE, ON STOP** e **ON STRIG**.

ON ERROR GOTO linha

Desvia o programa para a linha especificada, quando for detectado algum tipo de erro. **ERR** e **ERL** são atualizados com seus valores corretos. Após este desvio pode-se fazer qualquer tipo de manipulação dos erros, e para retornar usar **RESUME.**

```
Exemplo:10 ON ERROR GOTO 100
        20 INPUT A:PRINT "RAIZ DE";A;"=";SQR(A)
        30 GOTO 20
        100 PRINT "NAO EXISTE RAIZ REAL":RESUME
```

Digite e execute o programa acima, e entre um valor negativo.

ON var GOSUB linha1 [,linha2,. . . .]

Salta para a linha indicada por "var". Se var=1 então salta para linha1; se var=2 então executa linha2, e assim por diante.

As sub-rotinas executadas devem terminar com **RETURN** (Consulte **GOSUB**).

```
Exemplo:10 INPUT A:ON A GOSUB 30,40,50
        20 GOTO 10
        30 PRINT "SUB-ROTINA 1":RETURN
        40 PRINT "SUB-ROTINA 2":RETURN
        50 PRINT "SUB-ROTINA 3":RETURN
```

ON var GOTO linha1 [,linha2,. . .]

Salta para a linha1 se var=1, linha2 se var=2, e assim em diante.

```
Exemplo:10 INPUT "Entre: 1-Criança 2-Jovem 3-Adulto";A
        20 PRINT "Você é ";
        30 ON A GOTO 100,200,300
        40 END
        100 PRINT "uma criança":END
        200 PRINT "um jovem":END
        300 PRINT "um adulto":END
```

Pede a entrada de dados de acordo uma tabela, e apresenta a mensagem correspondente.

ON INTERVAL=num GOSUB linha

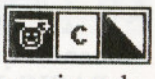
Desvia periodicamente a execução do programa para uma sub-rotina em intervalos de tempo definido por num/60 segundos.

"Linha" indica o início da sub-rotina.

```
Exemplo:10 TIME=0:ON INTERVAL=120 GOSUB 100
        20 PI=4*ATN(1):INTERVAL ON
        30 FOR X=0 TO 2*PI STEP .001
        40 PRINT X;SIN(X):NEXT:END
        100 PRINT TIME\60:RETURN
```

O programa acima imprime os vários arcos e seus respectivos senos e, periodicamente, é interrompido para imprimir o tempo de execução.

ON KEY GOSUB linha1 [,linha2. . .]



Desvia o programa para uma sub-rotina, dependendo da tecla de função pressionada. Para uma tecla ser ativada use **KEY (n) ON.**

Exemplo:10 ON KEY GOSUB 100,200,300:KEY(1) ON:KEY(2) ON:KEY(3) ON

```
20 PRINT"PRESSIONE UMA TECLA DE FUNÇAO"
30 PI=4*ATN(1):FOR X=0 TO 2*PI STEP .001:PRINT X;COS(X):NEXT:END
100 PRINT "FUNÇAO 1":RETURN
200 PRINT "FUNÇAO 2":RETURN
300 PRINT "FUNÇAO 3":RETURN
```

O programa acima mostra uma tabela com o arco e seu cosseno. Se forem pressionadas as teclas de função **[F1]** a **[F3]**, o fluxo interrompe e é iniciada a execução da subrotina correspondente à tecla pressionada, e ao final retorna para o fluxo normal.

ON SPRITE GOSUB linha

Caso dois "*sprites*" se sobreponham, então o programa salta para a sub-rotina especificada por "linha", desde que tenha sido ligado com **SPRITE ON.**

Exemplo:100 SCREEN2,1:BASE(2)=BASE(14):SCREEN 0 :SCREEN 2

```
110 ON SPRITE GOSUB 150
120 FOR X=0 TO 255:FOR S=3TO 6:SPRITE ON
130 PUTSPRITE S,(X,S*20-(S-4.5)*X),15-S,S
140 NEXT S,X:GOTO 120
150 SPRITE OFF:BEEP:RETURN
```

ON STOP GOSUB linha

Desvia o fluxo do programa, quando for detectado o pressionamento de **[CONTROL]** e **[STOP]**. É ativado pelo **STOP ON** (Veja **STOP**).

Exemplo:10 ON STOP GOSUB 100

```
20 STOP ON:PRINT"Pressione CTRL+STOP"
30 GOTO 20
100 STOP OFF
110 PRINT "CTRL+STOP PRESSIONADA!!!" :RETURN
```

ON STRIG GOSUB linha1 [,linha2,. . .]

Desvia o processamento quando detectado o pressionamento da barra de espaço ou os botões do "*Joystick*". "Linha1" indica a barra de espaço; "linha2" botão 1 do *joystick A*; "linha3" botão 1 do *joystick B*; "linha4" e "linha5" para o botão 2 de **A** e **B**, respectivamente.

Deve ser ativado por **STRIG (n) ON** (Consulte **STRIG (n) ON/OFF/STOP**)

Exemplo:10 ON STRIG GOSUB 100,200,300,400,500

```
20 FOR F=1 TO 5:STRIG (F) ON:NEXT
30 PRINT " PRESSIONE ESPAÇO OU BOTÃO DE TIRO":GOTO 30
100 PRINT "BARRA DE ESPAÇO":RETURN
200 PRINT "BOTÃO 1 – A":RETURN
300 PRINT "BOTÃO 1 – B":RETURN
400 PRINT "BOTÃO 2 – A":RETURN
500 PRINT "BOTÃO 2 – B":RETURN
```

OPEN "[disp] [nome]" [FOR modo] AS [#]num [LEN=n]

Comando que abre um arquivo, ou seja, os torna acessível pelos comandos disponíveis no **BASIC MSX**, destinados a manipulação destes.

Para detalhes sobre "disp" e "nome" veja **LOAD**. Ambos determinam onde se encontra

o arquivo (disco ou cassete no **DD·PLUS**).

"Modo" indica o tipo de acesso e pode ser especificado por: **INPUT** para leitura sequencial; **OUTPUT** para escrita sequencial; **APPEND** para escrita sequencial, somando os dados com os de um arquivo já existente (este último só no **DISK-BASIC**). Se "modo" for omitido trata-se de acesso *aleatório*. (Tanto escrita como leitura). Arquivos de acesso aleatório só são disponíveis no **DISK-BASIC**.

"Num" indica qual o número a relacionar com o nome do arquivo, pois os comandos do **BASIC** se referenciam ao arquivo pelo número e não pelo nome deste. O número pode variar de 1 até o limite especificado por **MAXFILES**.

No caso de se especificar um arquivo de acesso aleatório (só no **DISK-BASIC**!), deve-se indicar qual o comprimento de cada registro. Para isso usa-se LEN=n, onde "n" corresponde ao comprimento do registro (no máximo 255).

Exemplo:10 MAXFILES=1:OPEN "CAS:TESTE" FOR OUTPUT AS#1

20 PRINT #1,"GRADIENTE E EDITORA ALEPH"

30 CLOSE #1

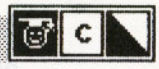
OR

Operador que faz a operação "*ou lógico*" bit a bit de dois números inteiros. Obtém um bit 1 se pelo menos um dos bits operados for igual a 1.

Exemplo: PRINT BIN$(&B10101010 OR &B11110000)

Deve resultar 11111010.

OUT porta, expressão



Comando que envia à porta especificada o valor dos 8 bits menos significativos do resultado da expressão.

A comunicação interna entre os circuitos de som, vídeo e outros dispositivos são feitas por várias portas, e se o programador tem um bom conhecimento, da arquitetura do padrão **MSX**, então, este comando será extremamente útil.

Exemplo:COLOR 15,1,1:SCREEN 0

OUT&H99,&H1F:OUT&H99,&H87

COLOR define branco como frente e preto como fundo, e em seguida os dois **OUT**'s invertem a combinação.

PAD (n)

Retorna o estado do "*touch pad*". Se n estiver entre 0 e 3, refere-se ao terminal A; entre 4 e 7, refere-se ao terminal B.

0 ou 4 –Retorna se o "*touch pad*" foi tocado ou não (0=não – 1=sim);

1 ou 5 –Retorna coordenada **X** do lugar tocado

2 ou 6 –Retorna coordenada **Y** do lugar tocado

3 ou 7 –Estado do interruptor (0=pressionado – 1 não pressionado)

PAINT [STEP] (X,Y) [,cor da pintura] [,cor do limite]

Preenche com uma cor específica, uma área pré-definida. Em **SCREEN 2** a cor que limita a área, e a cor que preencherá a área devem ser iguais. Isto não é necessário na **SCREEN 3**.

"STEP" faz com que a origem do sistema seja à partir do último ponto marcado (origem relativa).

X e Y são as coordenadas do local, a partir de onde será iniciada a pintura.

Exemplo:10 SCREEN 2:CIRCLE (128,96),80,6 : 20 PAINT (128,96),6,6

30 GOTO 30

PDL (n)

Retorna o valor determinado pelo *paddle*; "n" pode estar entre 0 e 12 e os valores obtidos estarão entre 0 e 255.

Se "n" for *impar* então os dados são correspondentes ao terminal A, senão, ao terminal B.

```
Exemplo:10 CLS:C=0
        20 COLOR,C,C
        30 IF PDL(1)=0 THEN 20
        40 C=(C+1)AND15:GOTO 20
```

PEEK (endereço)

Retorna o conteúdo do "endereço" da memória.

O computador armazena todos os dados em uma memória. O seu **Expert DD PLUS** sai de fábrica com 64 KBytes destinados ao uso geral.

O endereço pode variar de 0 a 65535 (0000 a FFFF em *hexa*), e cada posição dessas pode armazenar um dado de 8 bits, que sozinhos ou em conjunto, podem representar algo para o micro; para programadores com vastos conhecimentos (veja a bibliografia recomendada), pode ser uma excelente ferramenta (veja também **POKE, VPEEK, VPOKE**)

```
Exemplo:10 REM EXEMPLO DO PEEK
        20 FOR F=&H8000 TO &H807F
        30 PRINT F;"...";PEEK(F);"... ";CHR$(PEEK(F))
        40 NEXT F
```

Veja o exemplo acima, e repare como o micro armazena os programas **BASIC MSX** na memória. É muito difícil reconhecer o programa ?

PLAY "subcomandos"

Comando que toca sequências de notas e acordes de até 3 notas, através de uma sub-linguagem.

Veja abaixo a relação dos subcomandos para controle de volume tempo duração, pausas, tons, oitavas, etc:

Tn –Determina o andamento da música, e pode variar de 32 a 255 (inicial=T120);

On –n varia de 1 a 8 e determina a oitava em questão. (inicial=O4);

Ln –Determina a duração da nota e n pode variar de 1 a 64. (inicial=L4);

Nn –n pode variar de 0 a 96 e especifica uma nota musical pelo seu número.

A–G –Especifica a nota musical dentro de uma oitava pré-determinada. Se for seguido por um número entre 1 e 64 a sua duração varia conforme em no sub-comando **L**. Se for seguido por "#" ou "+", significa um *sustenido*, se seguido por "– " determina um *bemol*.

Rn –Com n de 1 a 64 e determina uma pausa.

. –Aumenta a duração da nota em 50%.

Vn –Com n de 0 a 15 determina o volume de saída, e aumenta com o valor de n (inicial=V8);

Mn –Determina o período da variação de volume durante a execução da nota, e n pode variar de 0 a 65535;

Sn –Com n entre 0 e 15 determina o formato do envelope (veja **SOUND**).

Veja o exemplo 2 do apêndice A para ilustrar o comando **PLAY**.

PLAY(n)

Função que retorna o estado das filas musicais. Se "n" for 0, testa os três canais; se for 1, testa o canal A; se for 2, testa o canal B; e se for 3, testa o canal C.

Esta função retorna – 1 se o canal estiver ocupado, e 0 se estiver livre.

```
Exemplo:10 PLAY "CDEFGFEDCDEFGFEDCDEFGFEDCDEFGFEDCCCC"
        20 IF PLAY(0) = -1 THEN PRINT"AGUARDE":GOTO 20
```

POINT (X,Y)

Função que retorna o código da cor de um ponto especificado na tela gráfica.

X e Y determinam as coordenadas do ponto a ser verificado.

```
Exemplo:10 SCREEN 2
        20 PSET (128,96),4:C1 = POINT(128,96):C2 = POINT(127,96)
        30 SCREEN 0:PRINT C1;C2
```

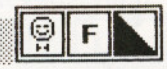
POKE endereço,expressão

Comando que carrega num endereço de memória, os 8 *bits* menos significativos do resultado da expressão.

```
Exemplo: POKE &HF3B1,10:KEYON:SCREEN 0
```

Veja o que acontece com a tela! Para colocá-la na sua condição original redigite a linha-exemplo substituindo 10 por 24.

POS(argumento)



Retorna, para qualquer "argumento", a abcissa X que o cursor ocupa.

```
Exemplo: PRINT "0123456";POS(0)
```

PRESET [STEP] (X,Y) [,cor]

Coloca um ponto colorido nas telas gráficas setadas com **SCREEN 2** ou **SCREEN 3**.

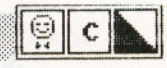
Se for especificada uma cor diferente da cor de fundo, então ele funciona igual a **PSET**; se não for especificada uma cor, então o ponto é preenchido com a cor de fundo, dando a impressão de estar sendo apagado.

X e Y definem a coordenada do ponto na tela.

Veja **PSET** para detalhes sobre **STEP**, e as coordenadas das duas telas gráficas.

```
Exemplo:10 SCREEN 0:LINE (10,10)-(245,182),6,BF
        20 FOR F = 0 TO 90:X = 235*RND(1) + 10:Y = 172*RND(100) + 10
        30 PRESET (X,Y):NEXT F
        40 GOTO 40
```

PRINT expressão



Apresenta na tela os dados listados na expressão

O comando **PRINT** é um dos comandos mais básicos, pois permite que o programador envie mensagens e resultados para a tela.

A expressão pode ser uma constante, uma variável numérica, uma expressão numérica, uma variável string e uma expressão string. Se for uma constante string, ela deve estar entre *aspas*.

O comando **PRINT** possue dois separadores: o ponto-e-vírgula (;), que faz com que a mensagem seguinte (que pode estar inclusive em outro PRINT) seja impressa exatamente a partir do fim da mensagem anterior (dentro do mesmo **PRINT** o (;) pode ser omitido); e a vírgula (,) que divide as linhas em dois campos iguais, e as mensagens são impressas a partir do campo seguinte. O primeiro campo tem 16 colunas.

Veja o exemplo, e procure entender. Se você pretende se tornar um programador, este comando é uma das ferramentas mais úteis.

```
Exemplo:10 PRINT "CONSTANTES STRING E NUMERO";1989
        20 A=2000:A$="ISTO É UMA STRING"
        30 PRINT A$,A,A$,A;A$
        40 PRINT "PODE-SE OMITIR O (;) "A$A"DENTRO DE UM COMANDO PRINT"
```

PRINT USING formato;expressão

Apresenta os dados com um formato específico na tela. O formato deve estar entre *aspas*, e são os seguintes:

"!"	–Apresenta apenas o primeiro caracter de uma *string*;
"\n-espaços\"	–Apresenta n+2 caracteres de uma *string*;
"&"	–Apresenta todos os caracteres de uma *string*;
"#"	–Especifica o formato e o número de dígitos apresentados de um dado numérico;
"+"	–Acrescenta o sinal + ou – antes ou após números conforme forem positivos nulos ou negativos;
"–"	–Acrescenta o sinal – após números negativos;
"**"	–Preenche com asteriscos os espaços ocupados por um dado numérico;
"$$"	–Acrescenta o símbolo $ antes de dados numéricos;
"**$"	–Acrescenta o símbolo $ antes de dados numéricos e preenche com asteriscos os espaços não ocupados;
","	–Acrescenta vírgula a cada três dígitos à esquerda do ponto decimal;
"^ ^ ^ ^"	–Apresenta dados numéricos em ponto flutuante.

```
Exemplo: PRINT USING "###.##",23.4392
```

PRINT #num,expressão

Escreve os dados da expressão no dispositivo (fita, disco, impressora, etc.) aberto por **OPEN**.

O número é o especificado no **OPEN**.

A expressão tem os mesmos parâmetros do **PRINT** convencional. Aliás em computação as telas de texto são vistas como um dispositivo de saída "*default*" (por isso não é necessário abrir o arquivo de impressão de tela). Nas telas gráficas, porém, deve-se abrir o arquivo gráfico (GRP:) para que se possa mandar dados para elas.

```
Exemplo:10 OPEN "GRP:" AS #1
        20 SCREEN 2:CIRCLE (128,96),60
        30 PRESET(105,90):PRINT#1,"Expert"
        40 PRESET(104,90):PRINT#1,"Expert"
        50 GOTO 50
```

PRINT #num,USING"formato";expressão

Este comando é uma ampliação do **PRINT USING**, com a vantagem de poder acessar arquivos externos, abertos pelo **OPEN**.

```
Exemplo:10 SCREEN 2:OPEN"GRP:"AS #1
        20 LINE (20,20)-(235,172),6,BF
        30 PRESET (110,90):PRINT#1,USING"##.##";12.5464
        40 GOTO 40
```

PSET [STEP] (X,Y) [,cor]

Marca um ponto na *tela gráfica*, com a cor especificada. Se a cor não for especificada então será marcado com a cor de frente definido pelo **COLOR**.

X e Y definem as coordenadas do ponto na tela gráfica. Nas telas gráficas **SCREEN 2** e **SCREEN 3** X varia de 0 a 255 e Y de 0 a 191. Na **SCREEN 3** os pontos têm o quádruplo da altura e largura da **SCREEN 2.**

STEP transfere a origem do sistema para a última coordenada impressa, de maneira que podem existir números negativos e positivos. Observe que apos o término do comando o sistema de coordenadas é devolvido à posição original.

O número da cor deve variar entre 0 e 15, e a tabela de correspondencia se encontra listada no comando **COLOR.**

Exemplo:
```
10 FOR A=2 TO 20 :CLOSE
20 SCREEN A MOD 2+2 : OPEN "GRP:" AS #1
30 PRESET (0,0): PRINT #1,"SCREEN "; A MOD 2+2
40 FOR X=0 TO 255 STEP 8 : FOR Y=33 TO 191 STEP 8
50 PSET (X,Y): NEXT Y,X,A
```

PUT [#] num [,reg]

Permite efetuar a gravação de um registro em um arquivo de acesso *aleatório*, que foi anteriormente aberto e definido pelos comandos **OPEN** e **FIELD.**

"num" deve indicar o número do arquivo em questão, e "reg" indica o registro que se quer gravar. Se omitido, será considerado o registro seguinte ao último gravado.

Veja o programa-exemplo 3 do apendice A.

PUT SPRITE camada [[STEP] (X,Y)], [cor], [num]

Comando que permite colocar na tela as figuras móveis definidas pelo usuário.

camada –O MSX possue 32 camadas a disposição de maneira que cada camada pode ter apenas um *sprite*. Este número deve estar entre 0 e 31.

STEP –Esta opção transfere a origem do sistema de coordenadas, para o último ponto plotado. A origem está no canto superior esquerdo da tela.

X –Define o valor da absissa, e deve estar entre 0 e 255.

Y –Define o valor da ordenada, e deve estar entre – 8 a 192.

cor –Define a cor do *sprite*, e deve estar entre 0 e 15. Veja o comando **COLOR** para saber a relação número/cor.

num –Define qual o desenho do *sprite*. Esse desenho é definido com o comando **SPRITE$**, e deve estar entre 0 e 255 se for de tamanho 8x8, ou entre 0 e 63 se for do tamanho 16x16 (veja comando **SCREEN**).

Exemplo:
```
10 SCREEN 1,1:FOR T=1 TO 8:READ A$
20 S$=S$+CHR$(VAL("&H"+A$)):NEXT T
30 SPRITE$(1)=S$:PUTSPRITE 0,(128,N),8,1
40 PRINT " EXPERT DD PLUS";:N=N+1: GOTO 30
50 DATA C7,E3,73,3B,DC,CE,C7,C3
```

Digitando o programa acima, você terá uma idéia da vantagem do uso de sprites.

READ var1 [,var2. . .]

Lê dados sequencialmente armazenados em linhas **DATA's.**

Os dados lidos são colocados na(s) variável(s) listadas no comando **READ.**

Veja o exemplo no comando **PUT SPRITE** ou o comando **DATA.**

Este comando, em conjunto com **RESTORE** e **DATA**, monta um conjunto de comandos, ideais para armazenar dados constantes (No exemplo do verbete **PUT SPRITE** o formato do *sprite* é armazenado em linhas **DATA**).

REM comentário

Este comando é útil para *documentação*, e tem como intenção básica facilitar a descrição interna dos programas. O comando não executa absolutamente nada, e pode ser seguido de um comentário. Este comentário pode conter qualquer tipo de informação.

O comando **REM** pode ser substituído pelo apóstrofo da tecla 8 (').

```
Exemplo:10 REM Este comando não executa nada
        20 ' e pode ser substituído por "'"
```

RENUM [lin1] [,lin2] [,inc]

Renumera as linhas de um programa. "Lin1" é o número inicial da linha, "lin2" indica a partir de que linha deve ser renumerado e "inc" corresponde ao incremento entre linhas.

O **RENUM** renumera inclusive **GOTO, GOSUB** e **RESUME** e outros.

Para ver como funciona, coloque um programa na memória e execute:

```
RENUM 1,,1
```

RESTORE [linha]

Este comando é usado em conjunto com **READ**, e atualiza a partir de que **DATA** devem ser considerados os dados lidos pelo **READ**. Se "linha" for omitido, atualiza a partir da primeira linha **DATA.**

```
Exemplo:10 RESTORE 30:READ A$:PRINT A$
        20 DATA linha de dados 1
        30 DATA linha de dados 2
```

RESUME [linha/ NEXT]

Indica a linha de retorno de uma sub-rotina de erro.

Quando ocorre algum tipo de erro no **BASIC**, este pode ser detectado (veja **ON ERROR GOTO**) e, após o seu tratamento, pode-se retornar, para uma linha do programa. Se "linha" não for especificada, então retorna para a linha onde ocorreu o erro. Se for usada a opção NEXT, então retorna para a linha seguinte à ocorrência do erro.

Veja o exemplo de **ON ERROR GOTO.**

RETURN linha

Veja o comando **GOSUB.**

RIGHT$(string,n)

Função que retorna os n– caracteres pela direita de uma variável string.

```
Exemplo: PRINT RIGHT$("TRABALHO",4)
```

RND(argumento)

Retorna um número *aleatório* maior ou igual a 0, mas menor que 1.

Se o argumento é positivo fornece sempre um valor diferente mas a mesma sequência a cada vez que for iniciado. Se for 0, então retorna o último valor gerado. Se for negativo, retorna o primeiro valor da sequência indicada pelo número.

O modo de uso ideal, é setar uma determinada sequência, através de um número negativo, e os seguintes através de argumentos positivos.

Para números realmente *aleatórios* veja o exemplo:

```
Exemplo:10 X=RND(-TIME):FOR F=1 TO 10
        20 PRINT RND(9999):NEXT F
```

A variável **TIME** varia constantemente, e colocando, então como argumento negativo do **RND**, gera um sequência imprevisível, formando assim sempre números diferentes.

RSET var$=expressão

Veja **LSET**.

RUN [linha | "nomearq"]

Permite executar um programa que está residente na memória.

Se for especificada linha, então inicia a execução a partir da linha indicada. Se for especificado o nome do arquivo (nomearq), carrega o arquivo, e o executa (como no **LOAD** com opção **R**).

Para testá-lo, digite qualquer programa, linha a linha, e comande:

RUN (e **[RETURN]**)

SAVE (veja sintaxe abaixo)

Comando que grava um programa da memória em um dispositivo com um determinado nome.

Este comando tem sintaxe diferente, para as versões disco ou cassete.

Sintaxe da versão cassete:

```
SAVE "cas:[nome]"
```

Na versão cassete é necessário o nome do dispositivo, mas o nome do arquivo é opcional, e este pode ter no máximo 6 letras, sendo as minúsculas diferentes das maiúsculas.

Sintaxe da versão disco:

```
SAVE "[dispositivo] nome" [,A]
```

O dispositivo deve ser um acionador (A até F, se existir) e, se não for especificado, será assumido o drive *corrente*. O nome é obrigatório e deve ter no máximo 8 caracteres, e mais 3 caracteres para a extensão.

A opção "A", faz com que o texto seja gravado em formato **ASCII** e, se não for especificada, grava no formato compactado.

Dentro de um sistema em disco, nada impede usar o cassete, sendo que nesse caso, use a sintaxe para a versão cassete. Note que você pode abrir arquivos em outros dispositivos e "salvar" o programa neles (impressora, tela, etc.).

SCREEN [modo],[sprite],[clic],[bauds],[impressora]

Este comando configura vários recursos do **MSX**:

modo –Seta as várias telas. 0 para textos 40x24; 1 para textos 32x24; 2 para alta-resolução 256x192; e 3 para multicolorido baixa-resolução 64x48.

sprite –Define os tipos de sprite's. 0 para tamanho 8x8 normal; 1 para 8x8 ampliado; 2 para 16x16 normal; 3 para 16x16 ampliado.

clic –Ativa ou não o clic no alto-falante, para as teclas pressionadas. 0 –desliga 1 –liga.

bauds –Seleciona velocidade de transferência de dados para o cassete. 1 para 1200 *bauds*; 2 para 2400 *bauds*.

impressora –Seleciona o tipo de impressora. 0 para o **ABNT**; 1 para impresora **MSX**.

Exemplo:10 A$ = "01234567890123456789012345678901234567890"

```
20 SCREEN 0:PRINT A$:GOSUB 100
30 SCREEN 1:PRINT A$:GOSUB 100
40 SCREEN 2:LINE(9,9)-(99,99),3,B:GOSUB 100
50 SCREEN 3:LINE(9,9)-(99,99),3,B:GOSUB 100
60 END
100 FOR F = 1 TO 2000:NEXT:RETURN
```

SGN (argumento)

Função que retorna o sinal do argumento.

Se o argumento for qualquer número positivo, o resultado é +1. Se for zero, o resultado é 0 e, se for negativo, o resultado é – 1.

Exemplo: PRINT SGN(–4);SGN(0);SGN(200)

SIN (argumento)

Função que retorna o valor do seno do arco, em radianos, fornecido pelo argumento.

Exemplo: PI = 4*ATN(1):PRINT SIN(PI/2)

SOUND registro,expressão

Acessa cada um dos registros do **PSG** (processador de som programável) individualmente.

O **PSG** possue 16 registros, sendo que os registros de 0 a 13 são usados para emissão de sons. Para maiores detalhes, veja a bibliografia aconselhada (Apendice E).

Exemplo: SOUND 6,10:SOUND 8,16:SOUND 12,3:SOUND 13,14:SOUND 7,&B11110111

Você obterá o som de uma locomotiva.

SPACE$ (argumento)

Função que retorna uma *string* com espaços em branco, dependendo do argumento definido.

Exemplo: FOR F = 1 TO 19:A$ = SPACE$(F):PRINT A$;"ALEPH":NEXT F

SPC (argumento)

Função que abre espaços num comando de saída, dependendo do argumento. Este resultado não pode ser inserido em uma variável *string*.

Exemplo: FOR F = 1 TO 19:PRINT SPC(F);F:NEXT F

SPRITE

Veja **SPRITE$, ON SPRITE GOSUB, SPRITE ON/OFF/STOP** e **PUT SPRITE.**

SPRITE ON/OFF/STOP

Comando que habilita, desabilita ou adia a interrupção por sobreposição de *sprites*.

Veja o exemplo de **ON SPRITE GOSUB.**

SPRITE$(numero),string

Define um formato de *sprite*. A string deve ter os caracteres no formato **ASCII**, que atribuídos ao *sprite* definem seu formato. O número do sprite pode variar de 0 a 255 para tamanho 8x8 ou de 0 a 63 para o tamanho 16x16 (Veja **SCREEN**).

Veja o exemplo de **PUT SPRITE.**

SQR(argumento)

Função que retorna o valor da raiz quadrada do argumento.

Exemplo: PRINT SQR(9)

STEP

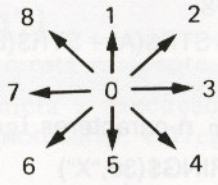
Veja **PSET, PRESET, CIRCLE, FOR/NEXT, PUTSPRITE** e **LINE.**

STICK (N)

Retorna o estado do *joystick* ou das teclas cursoras.

Se N for 0, então trata-se das teclas do cursor. Se for 1 então trata-se do *Joystick A*, se for 2, então trata-se do *joystick B*.

Pode-se obter os seguintes resultados:




```
Exemplo:10 SCREEN 3:X=128:Y=96
        20 PSET(X,Y),6:C=STICK(0)
        30 X=X+(C>5)-((C>1)AND(C<5))
        40 Y=Y+((C=8)OR(C=1)OR(C=2))-((C>3)AND(C<7))
        50 GOTO 20
```

Se você quer ser um programador, observe os *parêntesis lógicos* nas linhas 30 e 40.

STOP

Interrompe a execução de um programa. A execução pode ser prosseguida com **CONT.**

```
Exemplo:10 PRINT "ALO":STOP
        20 PRINT "ESTA LINHA NÃO É EXECUTADA"
```

Veja também **INTERVAL, KEY(n), ON STOP GOSUB, SPRITE, STOP** e **STRIG(n).**

STOP ON/OFF/STOP

Habilita, desabilita ou adia a interrupção mediante o pressionamento de **[CONTROL]+[STOP].**

```
Exemplo:10 ON STOP GOSUB 100:STOP ON
        20 PRINT "TECLE CONTROL+STOP":GOTO 20
        100 STOP OFF:PRINT "PARA TERMINAR PRESSIONE CONTROL+STOP"
        110 GOTO 110
```

STRIG (N)

Função que retorna o estado dos disparadores. Se for detectado o pressionamento do disparador, então retorna o valor – 1, senão 0.

N pode variar de 0 a 4, e indica qual o disparador, que deseja-se ler:

0 –barra de espaços; 1 –disparador 1 do *joystick A*

2 –disparador 1 do *joystick B* 3 –disparador 2 do *joystick A*

4 –disparador 2 do *joystick B*

```
Exemplo:10 CLS:PRINT"PRESSIONE ESPAÇO"
        20 IF STRIG(0)=0 THEN 20 ELSE END
```

STRIG (N) ON/ OFF/ STOP

Comando que habilita, desabilita ou adia a interrupção pelos disparadores. N indica qual o disparador em questão (veja **STRIG**).

Exemplo:10 ON STRIG GOSUB 100:STRIG(0) ON

```
20 PRINT "PRESSIONE ESPAÇO":GOTO 20
100 STRIG(0) OFF:PRINT "****PRESSIONADO****"
110 STRIG(0) ON:RETURN
```

STR$(expressão)

Converte dados numéricos em string.

Exemplo: A=12:B=13:C=A+B:C$=STR$(A)+STR$(B):PRINT C$;C

STRING$(n,caractere)

Esta função retorna uma string com n-caracteres iguais.

Exemplo: PRINT STRING$(20,65),STRING$(30,"X")

SWAP var1,var2

Comando que permuta o conteúdo de duas variáveis.

Exemplo: A=1:B=2:SWAP A,B:PRINT A;B

TAB (n)

Move o cursor n espaços para a direita. "n" não deve exceder 255.

Exemplo: 10 FOR A=1 TO 20:PRINT TAB(A);A:NEXT A

TAN (argumento)

Função que retorna o valor da tangente do arco, em radianos, definido pelo argumento.

Exemplo: PI=4*ATN(1):PRINT TAN(PI/4)

THEN

Veja **IF... THEN ...ELSE.**

TIME

Variável reservada, que retorna o valor do temporizador interno.

Este temporizador é incrementado a cada 1/60 segundos, e ao atingir 65535, volta a 0, e assim repetidamente.

Como toda variável seu conteúdo pode ser alterado (sem o uso do **LET**).

Exemplo:10 CLS: PRINT"PRESSIONE RETURN JÁ!!": TIME =0

```
20A$=INKEY$: IF A$<>CHR$(13)THEN 20
30 PRINT"VOÇÊ DEMOROU";TIME/60;"SEGUNDOS!"
```

TO

Veja **COPY, FOR.**

TROFF

Comando que desliga o "**TRACE**".

Quando o "**TRACE**" está ligado, toda número de linha que estiver sendo executada, é impresso na tela (Ideal para procura de erros).

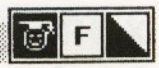
TRON

Permite visualizar o número de linha que esta sendo executado. O inverso de **TROFF.**

USING

Veja **PRINT USING, PRINT# USING.**

USR[n] (argumento)



Executa uma das 10 subrotinas em *linguagem de máquina*.

O número "n" pode estar entre 0 e 9, e indica qual das 10 entradas, está sendo usada.

O argumento é passado para a sub-rotina. Consulte um livro de linguagem de máquina, se desejar informações mais precisas.

A sub-rotina, em **ASSEMBLY**, deve terminar com "RET" (&HC9), para retornar ao **BASIC**.

O comando **DEFUSR[n]** define o endereço de entrada da sub-rotina.

Exemplo: DEFUSR=&H6C:POKE0,USR(0)

Acessa a sub-rotina que executa um SCREEN 0.

VAL(string)

Função que converte uma string em um número. Tem função inversa de **STR$**.

Note que, quaisquer caracteres diferentes de algarismos, serão ignorados!

Exemplo: A$="432+252":PRINT A$;VAL(A$)

VARPTR

Função que retorna o endereço do local onde esta armazenado uma variável.

Exemplo: A=0:PRINT "A = &H";HEX$(VARPTR(A))

VDP(reg) / VDP(reg)=expressão

Variável interna que contém o conteúdo do **VDP** (Processador de vídeo).

Elas podem ser modificadas e lidas como variáveis.

Exemplo: SCREEN0:A=VDP(7):VDP(7)=A XOR 255

Muda a cor da tela.

VPEEK (endereço)

Lê o byte correspondente ao endereço da memória de vídeo (VRAM).

O Expert tem, de fábrica, 80 KBytes de memória, sendo que destes, 16 KBytes se destinam exclusivamente à geração de imagens.

A função **VPEEK** possibilita que acessemos cada um desses 16384 bytes.

Exemplo: PRINT VPEEK(0)

VPOKE endereço,expressão

Coloca em um endereço da memória de vídeo (VRAM), o byte fornecido pela expressão.

Exemplo: 10 CLS:FOR F=0 TO 255:VPOKE F,F:NEXT F

O programa coloca a tabela de caracteres do MSX na **SCREEN 0** (sem o uso do **PRINT**).

WAIT porta,expressão1 [,expressão2]

Aguarda até a ocorrência do valor especificado.

Um **XOR** ("ou" exclusivo) é executado entre os dados da porta de E/S especificada, e o valor da expressão2. Um **AND** ("e") é executado entre os resultados e o valor da expressão1. Se o resultado final for 0, então os dados serão lidos.

Este comando pode ser classificado como avançadíssimo, e de pouco uso em programas.

WIDTH n

Determina o número de colunas que uma tela de texto deve ter.

Para **SCREEN 0** deve ser entre 1 e 40; em **SCREEN 1** entre 1 e 32.

Exemplo: WIDTH 15

O que faz com que a tela seja bastante reduzida.

XOR

Operador que executa um "*ou exclusivo*" bit a bit entre dois números, ou seja, se ambos os bits forem iguais, resulta um bit 0; se forem diferentes, resulta um bit 1.

Exemplo: PRINT BIN$(&B01010000 XOR &B 10111100)

+

Operador de adição, permite efetuar a soma matemática de duas variáveis numéricas ou a concatenação de 2 strings.

Exemplos: PRINT 2+3

Resulta: 5

PRINT "Expert"+" DD PLUS"

Resulta: Expert DD PLUS

–

Operador de subtração, quando usado entre dois números. Pode ser usado para inverter o sinal de um número.

Exemplo: A=4:PRINT –A

Pode ser usado como separador em certos comandos (Veja **LINE, DELETE, LIST, LLIST**).

*

Operador de multiplicação.

Exemplo: A=3:B=2:PRINT A*B

É usado também como "coringa" para nomes de arquivos (Veja **FILES, DELETE** e **COPY**) ou para indicar, na numeração automática, que uma linha de programa já existe (Veja **AUTO**).

/

Operador de divisão.

Exemplo: A=10:B=3:PRINT A/B

Resulta: 3.3333333333333

Veja também "\"

\

Operador de "divisão inteira": fornece o quociente inteiro da divisão de dois números.

Exemplo: A=10:B=3:PRINT A\B

Resulta: 3. Compare com o exemplo do "/".

^

Operador de exponenciação.

Exemplo: PRINT 3^2

Resulta: 9

%

Posto ao final de um número ou variável numérica, indica ser de precisão inteira (ocupado 2 bytes de memória).

!

Posto ao final de um número ou variável numérica, indica ser de precisão simples (ocupando 4 bytes de memória).

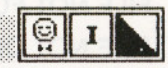
#

Posto ao final de um número ou variável numérica, indica ser de precisão dupla

(ocupando 8 bytes de memória).

É usado também para numerar arquivos (veja **OPEN, FIELD, CLOSE**). Em países anglo-saxônicos o "#" substitue o nosso "Nº".

$



No final do nome de uma variável, mostra ser ela alfa-numérica ("string").

'

Veja **REM.**

_

Veja **CALL.**

?

Veja **PRINT.** Também usado como "coringa" em nomes de arquivos (veja **FILES, COPY, DELETE** e **NAME**)

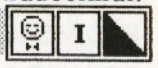
&B

Indica que os algarismos que seguem (são permitidos apenas 0 e 1) representam um número em notação binária.

&H

Indica que os algarismos que seguem (são permitidos apenas 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E e F) representam um número em notação hexadecimal.

&O



Indica que os algarismos que seguem (são permitidos apenas 0,1,2,3,4,5,6,7) representam um número em notação octal.

<, >, <=, >=, = e <>

Correspondem aos operadores relacionais da matemática: Maior que, menor que, maior ou igual, menor ou igual, igual e diferente, respectivamente.

Exemplo: PRINT (4<8);(4>8)

MENSAGENS DE ERRO DO BASIC E DISK-BASIC

BAD DRIVE NAME	O acionador não está conectado ao sistema.
BAD FAT	O disco está com problema no seu mapa de arquivos.
BAD FILE MODE	O arquivo foi aberto de maneira incorreta.
BAD FILE NAME	Usou-se um nome incorreto para o arquivo.
BAD FILE NUMBER	Ou o arquivo não foi aberto ou está fora do MAXFILES.
BAD SECTOR NUMBER	Não existe o setor procurado no disquete.
CAN'T CONTINUE	Impossível executar o CONT.
DEVICE I/O ERROR	Ocorreu um erro de Entrada/Saída em algum dispositivo.
DIRECT STATEMENT IN FILE	O arquivo carregado não é um programa BASIC.
DISK FULL	A capacidade de armazenamento do disquete está saturada.
DISK I/O ERROR	Há um dano físico ou lógico no disquete.
DISK OFF LINE	O disquete não está corretamente inserido no acionador.
DISK WRITE PROTECT	O disquete está protegido contra gravação (janela aberta).
DIVISION BY ZERO	Tentou-se executar uma divisão por zero.
FIELD OVERFLOW	A soma dos FIELDs ultrapassa o especificado pelo LEN.

FILE ALREADY EXISTS	Tentou-se renomear um arquivo com um nome já existente.
FILE ALREADY OPEN	Tentou-se abrir um arquivo sequencial ainda aberto.
FILE NOT FOUND	O arquivo não existe.
FILE NOT OPEN	Tentou-se acessar um arquivo que não foi aberto.
FILE STILL OPEN	Tentou-se abrir um arquivo já aberto.
ILLEGAL DIRECT	O comando usado só funciona no modo programado.
ILLEGAL FUNCTION CALL	Erro no uso de um parâmetro ou argumento.
INPUT PAST END	Tentou-se ler algo de um arquivo que já terminou.
INTERNAL ERROR	Problemas com o funcionamento do micro.
LINE BUFFER OVERFLOW	Uma linha de BASIC possui caracteres demais.
MISSING OPERAND	Argumento errado ou faltante.
NEXT WITHOUT FOR	Há um NEXT sem que antes tenha se especificado o FOR.
NO RESUME	Não foi colocado um RESUME numa rotina de erro.
OUT OF DATA	Tentou-se ler um dado numa sequencia DATA já esgotada.
OUT OF MEMORY	Toda memória disponível ou reservada, esgotou-se.
OUT OF STRING SPACE	Excedeu-se o espaço reservado para strings.
OVERFLOW	A magnitude do número ultrapassa a capacidade do micro.
REDIMENSIONED ARRAY	Dois DIM foram usados para dimensionar a mesma matriz.
RENAME ACROSS DISK	Tentou-se renomear um arquivo de um disco para outro.
RESUME WITHOUT ERROR	Foi encontrado um RESUME sem ocorrer erro.
RETURN WITHOUT GOSUB	Foi encontrado um RETURN fora de uma sub-rotina..
SEQUENTIAL I/O ONLY	Tentou-se usar um arquivo sequencial como randômico.
STRING FORMULA TOO COMPLEX	Uma string é muito longa ou complexa.
STRING TOO LONG	Há uma string com mais de 255 caracteres.
SUBSCRIT OUT OF RANGE	Há um índice maior que o especificado no DIM.
SYNTAX ERROR	A sintaxe do comando está errada.
TOO MANY FILES	Apesar do disquete ainda ter espaço disponível, o diretório já está lotado (máximo de 112 arquivos).
TYPE MISMATCH	Tentou-se atribuir um valor numérico a uma string.
UNDEFINED LINE NUMBER	Um GOTO ou GOSUB envia a execução a uma linha inexistente no programa.
UNDEFINED USER FUNCTION	Tentou-se usar um FN antes do correspondente DEF FN.
UNPRINTABLE ERROR	Erro que não dispõe de mensagem.
VERIFY ERROR	Detectou-se um erro numa rotina de verificação.

DICIONÁRIO DE COMANDOS DO MSXDOS 7

BASIC [nomearq]

Permite sair do MSXDOS para entrar no DISK-BASIC.

nomearq = nome de um arquivo-programa escrito necessariamente em BASIC MSX. Se for especificado, o sistema sai do MSXDOS, vai para o BASIC e executa o programa "nomearq".

Veja o comando CALL SYSTEM para retornar do BASIC para o MSXDOS.

Exemplo:

A>BASIC

Simplesmente sai do MSXDOS e vai para o BASIC, expondo a mensagem de apresentação do BASIC MSX.

A>BASIC PROGRAMA.BAS

Entra no BASIC, carrega e executa o programa com o nome PROGRAMA.BAS.

COPY [disp1] nome1 [disp2] [nome2] [opc]

Permite efetuar cópias de arquivos.

disp1 = dispositivo onde se encontra o arquivo a ser copiado. Pode ser qualquer drive (A:, B:, C:, D:, E: ou F:), ou teclado (CON), ou o dispositivo auxiliar (AUX).

disp2 = dispositivo onde será feita a cópia. Pode ser qualquer drive, ou um dispositivo de saída (por exemplo: PRN, LST, AUX, CON ou NUL).

nome1 = nome do arquivo-fonte a ser copiado. Podem ser utilizados os caracteres coringas (* e ?).

nome2 = nome do arquivo de destino. Se for omitido o arquivo destino tem o mesmo nome do arquivo fonte. Podem ser usados os caracteres coringas (* e ?).

opc = pode ser uma ou mais das seguintes opções:

/V = opção que ativa a verificação da escrita (ideal para cópias seguras).

/B = opção que avisa o MSXDOS que todos os caracteres dos arquivos devem ser copiados, no caso de cópias de arquivos binários.

/A = opção que avisa o MSXDOS para não copiar o caracter de "fim-de-arquivo" (código 26). Usando para cópias de arquivos em ASCII.

Se houver apenas um drive físico conectado e os dispositivos especificados forem os drives A: e B: (origem e destino, respectivamente), a cópia será feita neste único drive, e o MSXDOS se encarrega de expor as mensagens de trocas de disquete, aguardando sempre o pressionamento de uma tecla para prosseguir a cópia. Se durante a inicialização do sistema, for pressionada a tecla [CONTROL], então essa cópia não será possível (é assumido apenas um drive lógico).

Copia de A: para B: todos os arquivos com extensão BAS, verificando se a escrita foi bem sucedida.

```
A>COPY B:TESTE.BAT  A:RASCUNHO.BKP
```

Copia o arquivo "TESTE.BAT" do drive B: para o drive A:, gravando com o nome RASCUNHO.BKP.

```
A>COPY  A:TESTE.TXT  PRN
```

Copia o arquivo TESTE.TXT do drive A: para a impressora conectada.

O comando COPY permite que sejam emendados arquivos, para maiores detalhes veja o capítulo 4.

DATE [dia/ mes/ ano]

Se não for especificado dia/mes/ano então, apresenta a data atual, e aguarda que seja entrada a nova data. Se estiver de acordo, pressione apenas **[RETURN]**.

Se for especificado dia/mes/ano no comando, então a data será atualizada com o novo valor.

O caractere de separação "/" pode ser substituido, também, por " –".

O ano pode ser especificado com 2 ou 4 caracteres. Se for com 2 caracteres (entre 80 e 99) então é subentendido que esteja no intervalo de 1980 a 1999.

Se a data não for válida, então será emitida uma mensagem de erro.

Exemplos:

```
DATE  28-05-1989
```

ou

```
DATE  28/05/89
```

ou simplesmente,

```
DATE
```

DEL [drive] nome

Permite apagar arquivos do diretório do disco.

nome –Indica o nome do arquivo a ser apagado. Podem ser usados os caracteres coringas (* e ?).

drive –Indica o drive onde se encontra o arquivo desejado.

Se drive for omitido, então, será assumido o drive corrente.

Se você comandar DEL *.* , então todos os arquivos serão apagados (neste caso o sistema pede confirmação).

Exemplos:

```
A>DEL  EXEMPLO.TXT
```

Apaga o arquivo EXEMPLO.TXT do disco.

```
A>ERASE  *.BAS
```

Apaga todos os arquivos com extensão BAS. Observe que DEL pode ser substituído por ERASE, com os mesmos princípios e funcionamento.

DIR [drive] [nome] [/ P] [/ W]

Apresenta o diretório do disco, que é constituído por: nome e extensão, comprimento em bytes, data e hora de gravação. A hora só é apresentada, se os micros que gravaram o arquivo, possuirem uma interface com relógio interno.

drive Indica em que acionador se encontra o disco (entre A e F). Se omitido, mostra o drive corrente (ou 'default').

/P Opção que congela a listagem do diretório, toda vez que uma tela estiver repleta. Para prosseguir pressione qualquer tecla.

/W Opção que lista apenas os nomes e extensões dos arquivos, excluindo então o comprimento, data e a hora da gravação do arquivo. É ideal para procura em discos com muitos arquivos.

Exemplos:

A>DIR

lista todos os arquivos; ou

A>DIR *.COM /W

lista todos os arquivos com extensão COM; ou

A>DIR B:JOGO?.GAM

lista os arquivos do acionador B:, com o nome de 5 caracteres iniciando-se por JOGO, seguido por um caractere qualquer (?) e com extensão GAM.

Pode-se interromper a listagem com [CONTROL]+[C], ou paralisá-la momentaneamente com [CONTROL]+[S].

ERASE [drive] nome

Veja o comando **DEL.**

FORMAT

Permite que seja feita a formatação do disco.

No capítulo 4, foi esplicado este procedimento, que permite que o sistema operacional reconheça um disco.

No processo de formatação, são gravadas várias informações, limpando todas as anteriores, permitindo que o sistema operacional localize cada setor do disquete.

O processo de formatação não deve ser interrompido, isto é, não deve-se ejetar o disquete durante a formatação.

Quando **FORMAT** é executado, deve-se escolher o drive onde se encontra o disco, e o tipo de formatação. Como o acionador do DD PLUS é de 3 1/2", deve-se optar pela opção de 80 trilhas face simples ou dupla.

Exemplo: A>FORMAT

MODE numcol

Permite definir o número máximo de colunas que podem existir na tela.

numcol = número de colunas da tela.

O número máximo de colunas, se não estiver instalado um cartão de 80 colunas, é 40.

Exemplo: A>MODE 40

PAUSE [mensagem]

Permite a interrupção de um arquivo do tipo "BATCH".

Durante a execução de um arquivo "BATCH" às vezes, é necessário interromper a execução, ou por uma troca de disquetes, ou para que o usuário possa ler todas as informações da tela. Para essa finalidade pode-se colocar o comando PAUSE, que aguarda até o pressionamento de uma tecla, para prosseguir.

Exemplo: A>PAUSE Coloque disco no drive A

REM [comentário]

Permite apresentar durante a execução de um arquivo "BATCH", um comentário ou mensagem.

Exemplo: A>REM *** Linha de Exemplo ***

REN [drive] nome1 nome2
RENAME [drive] nome1 nome2

Altera o nome de um arquivo de um determinado disco.

drive — Indica em que acionador se encontra o disquete com o arquivo em questão.

nome1 — Indica o nome do arquivo que se encontra no disquete.

nome2 — Indica o novo nome do arquivo. Este nome não deve existir no disco.

Os caracteres coringas podem ser também usados.

Este comando se faz necessário quando for desejada uma reclassificação do arquivo, ou por conveniência, ou por necessidade.

Exemplos:

A>REN A:TESTE.TXT RASCUNHO.ASC

ou

A>RENAME A:TESTE.TXT RASCUNHO.ASC

renomeia o arquivo TESTE.TXT com o nome RASCUNHO.ASC;

A>REN LUXO.* ?I??.*

que renomeia todos os arquivos com o nome LUXO e qualquer extensão, para LIXO e mantendo a mesma extensão.

TIME [h[:m[:s]]]

Apresenta e permite alterar o conteúdo do relógio interno do micro.

h — Hora (0 a 23 ou 1 a 12 seguido de A ou P)

m — Minutos (0 a 59)

s — Segundos (0 a 59)

O comando TIME só funciona em micros que possuam relógio interno. Se não existir sempre apresentará 0:00:00 hora.

TYPE [drive] nome

Apresenta na tela o conteúdo de um arquivo.

drive = indica o drive em que se encontra o arquivo.

nome = indica o nome do arquivo gravado no disco.

Durante a apresentação do arquivo, pode-se interromper com [CONTROL]+[C] ou paralisar momentaneamente com [CONTROL]+[S].

Para que o arquivo contenha algo legível, deve estar gravado no formato ASCII, caso contrário (formato binário), apesar de inofensivo para o micro, aparecerá coisas sem nexo na tela.

Exemplo: A>TYPE TESTE.ASC

VERIFY ON/ OFF

Ativa ou desativa, respectivamente, a verificação da escrita.

Para que as cópias sejam seguras, é aconselhavel que se opte pela verificação ativada.

Exemplo: A>VERIFY ON

MENSAGENS DO MSXDOS

ABORT(A),RETRY(R),IGNORE(I)? — DIgite A se quiser desistir do comando, R se quiser tentar outra vez e I se quiser ignorar o erro.

ARE YOU SURE(Y/N)?	Permite arrependimentos. Digite Y para confirmar e N para cancelar.
BAD COMMAND OR FILE NAME	Tentou-se executar um comando que não existe com a terminação .COM ou .BAT.
BAD FAT, DRIVE X	Problemas no mapa de localização de arquivos.
CONTENT OF DESTINATION LOST BEFORE COPY	O comando COPY foi usado com um arquivo de destino já existente no disco.
DISK ERROR READING (/WRITING) DRIVE	Disquete danificado fisicamente ou logicamente.
FILE CANNOT BE COPIED ONTO ITSELF	Usou-se o comando COPY com mesmo nome e dispositivo para a fonte e o destino.
FILE CREATION ERROR	O MSXDOS não conseguiu criar um arquivo.
FILE NOT FOUND	O arquivo não foi encontrado.
INSERT DISK WITH BATCH FILE AND STRIKE ANY KEY WHEN READY	Ocorre se o disco que contem o arquivo BATCH está sendo executado é retirado do acionador antes
ra.	
INSUFFICIENT DISK SPACE	Esgotou-se a capacidade do disquete.
INVALID DATE/ ENTER NEW DATE	A data foi digitada num formato inválido.
INVALID DRIVE SPECIFICATION	Foi especificado um acionador não conectado ao sistema.
INVALID PARAMETER	Foi usado um parâmetro incorreto junto a um comando (p.ex. MODE 100).
INVALID TIME/ ENTER NEW TIME	A hora foi digitada em formato inválido.
NOT READY	Tentou-se usar um disquete não formatado.
PROGRAM TOO BIG TO FIT IN MEMORY	Tentou-se carregar no micro um programa grande demais para a memória disponível.
RENAME ERROR	Renomeou-se um arquivo com nome já existente.
STRIKE ANY KEY WHEN READY	Pressione qualquer tecla para continuar.
TERMINATE BATCH FILE (Y/N)?	A execução de um arquivo BATCH foi interrompida por um **[CONTROL]+[C]** ou **[CONTROL]+[STOP]** e o sistema quer saber se a interrupção é definitiva (Y) ou deve ser continuada(N).
UNSUPPORTED MEDIA TYPE	A formatação do disquete é incompatível com MSX.
WRITE ERROR	Erro de escrita. Se o erro persistir, mesmo após reformatação, o disquete está com danos físicos.
WRITE PROTECT	O disquete está protegido (janela aberta).

APÊNDICES

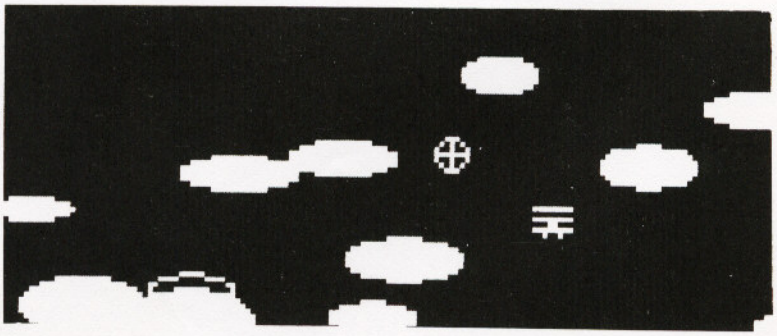
A – EXEMPLOS DE PROGRAMAS

A seguir estamos listando 3 programas-exemplo, escritos em BASIC.

O primeiro permite demonstrar o recurso gráfico do sprites, o segundo demonstra os recursos musicais e o terceiro dá um exemplo de gerenciamento de arquivo aleatório. Se, além de estudá-los, você quiser ver seu efeito, digite-os com atenção, tomando cuidado para não confundir o L minúsculo com o um, o O maíusculo com o zero e o B maiúsculo com o oito.

1 – SPRITES

```
100 REM:##################################
110 REM: MOVIMENTE A MIRA COM AS SETAS
120 REM: E ATIRE COM BARRA DE ESPACOS
130 REM: CADA TIRO  -) -2 PONTOS
140 REM: CADA ACERTO-) +4 PONTOS
150 REM:##################################
160 COLOR 15,0,7
170 SCREEN 2,0
180 X1=64*RND(-TIME)+100
185 Y1=48*RND(99999!)+100:X2=100:Y2=100
190 FOR I= 0 TO 60
200 XN=RND(3)*255:YN=RND(99)*191
210 R=RND(9)*8+6:E=RND(5)*.3+.2
220 CIRCLE (XN,YN),R,,,,E:PAINT(XN,YN)
230 NEXT I:LINE(0,0)-(255,20),1,BF
240 FOR I=1 TO 8
250 READ A$
260 S$=S$+CHR$(VAL("&B"+A$))
270 NEXT I
280 SPRITE$(1)=S$
290 DATA 00000000
300 DATA 11111111
310 DATA 00000000
320 DATA 11111111
330 DATA 00011000
340 DATA 11111111
350 DATA 00100100
360 DATA 00000000
370 PUTSPRITE 11,(X1,Y1),8,1
380 FOR I=1 TO 8
390 READ A$
400 W$=W$+CHR$(VAL("&B"+A$))
410 NEXT I
420 SPRITE$(2)=W$
430 DATA 00111000
440 DATA 01010100
450 DATA 10010010
460 DATA 11111110
470 DATA 10010010
480 DATA 01010100
490 DATA 00111000
500 DATA 00000000
510 RESTORE
520 PUT SPRITE 10,(X2,Y2),2,2
530 SPRITE ON
540 IF STRIG(0) THEN N=N-2:PLAY"S8M8001C16"
550 X1=X1+3*INT(RND(1)+.5)-3*INT(RND(1)+.5)
560 Y1=Y1+3*INT(RND(1)+.5)-3*INT(RND(1)+.5)
570 IF X1<0 OR X1>255 OR Y1<0 OR Y1>192 THEN X1=128:Y1=96
580 A=STICK(0)
590 Y2=Y2+(A=8)+(A=1)+(A=2)-(A=6)-(A=5)-(A=4)
600 X2=X2+(A=8)+(A=7)+(A=6)-(A=2)-(A=3)-(A=4)
610 PUTSPRITE10,(X2,Y2),2,2
620 PUTSPRITE11,(X1,Y1),8,1
630 ON SPRITE GOSUB 680
640 B=B+1
650 IF B>400 THEN 720
660 GOTO 530
670 END
680 IF STRIG(0) THEN BEEP:N=N+4:PSET(X1+4,Y1+4),1
690 SPRITE OFF
700 RETURN
710 LINE(0,0)-(255,20),1,BF
720 OPEN "GRP:"AS #1
730 PSET(10,10),1
740 PRINT#1,"VOCE FEZ";N;"PONTOS!"
750 IF N>0 THEN 800
760 FOR I=Y2 TO 192 STEP 2
770 BEEP:PUTSPRITE 10,(X2,I),2,2
780 COLOR ,,IMOD16:NEXT I
790 GOTO 830
800 FOR I=Y1 TO 192:W=3*(IMOD2)
810 BEEP:PUTSPRITE 11,(X1+W,I),8,1
820 LINE -(X1+3+W,I),1:NEXT I
830 BEEP:FOR I=1 TO 1000:NEXT I
840 RUN
```

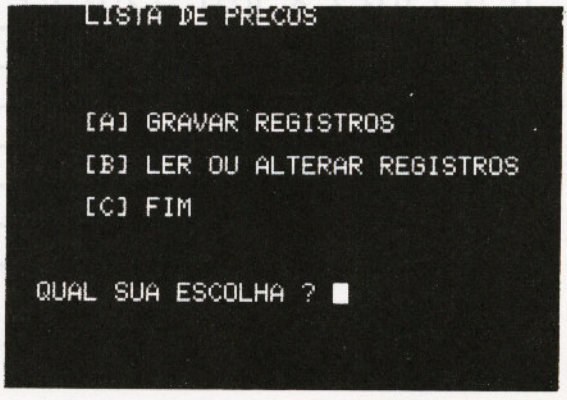


2 – MUSICA

```
10 '*********************************
12 '*       EXEMPLO DO PLAY         *
14 '*********************************
16 '
18 SCREEN 0:KEY OFF
20 PRINT,"THE ENTERTAINER"
22 PRINT "By Scott Joplin"
24 PRINT "1902"
26 PLAY"s0m6000T136","s0m6000T136","s0m6000T136"
28 PLAY"o4d8d#8","r4","r4"
30 FOR I=1 TO 2
32 PLAY"e8o5c4o4e8o5c4o4e8o5c8","o3L4cegb-","riL4"
34 PLAY"c2r8c8d8d#8","faeg","ro4crc"
36 PLAY"e8c8d8e4o4b8o5d4","go4co3gb","rerf"
38 PLAY"c2r4o4d8d#8","o4co3go4cr4","errr"
40 PLAY"e8o5c4o4e8o5c4o4e8o5c8","o3cegb-","rrrr"
42 PLAY"c2r4o4a8g8","faee-","ro4crr"
44 PLAY"f#8a8o5c8e4d8c8o4a8","df#df#","o3rara"
46 PLAY"o5d2r4o4d8d#8","go2gab","rrrr"
48 PLAY"e8o5c4o4e8o5c4o4e8o5c8","o3cegb-","rrrr"
50 PLAY"c2r8c8d8d#8","faeg","o4rcrc"
52 PLAY"e8c8d8e4o4b8o5d4","go4co3gb","rerf"
54 PLAY"c2r4c8d8","o4co3go4cr4","errr"
56 PLAY"e8c8d8e4c8d8c8","cco3b-o4c","rere"
58 PLAY"e8c8d8e4c8d8c8","o3ao4co3b-o4c","rfrf"
60 PLAY"e8c8d8e4o4b8o5d4","o3go4co3gb","rerr"
62 IF I=2 THEN 68
64 PLAY"o5c2r4o4d8d#8","o4co3ged","rrrr"
66 NEXT I
68 PLAY"o5c2.","o4co3gc","o3co2gc"
```

3 – ARQUIVO RANDÔMICO

```
100 REM [ ================================ ]
102 REM [ EXEMPLO DE USO DE CVI,CVS e CVD ]
104 REM [       e MKI$, MKS$, MKD$        ]
106 REM [ ================================ ]
108 REM [      MONTA MENU PRINCIPAL        ]
110 REM [ ================================ ]
112 KEYOFF : CLS : POKE &HFCAB,1
114 OPEN "LISTPRE.DAT" AS #1 LEN=24
116 FIELD #1,10 AS N$,2 AS Q$,4 AS PB$,8 AS PL$
118 LOCATE 5:PRINT"LISTA DE PRECOS"
120 LOCATE 5,5
122 PRINT"[A] GRAVAR REGISTROS"
124 LOCATE 5,7
126 PRINT"[B] LER OU ALTERAR REGISTROS"
128 LOCATE 5,9:PRINT"[C] FIM"
130 LOCATE 2,13
132 PRINT"QUAL SUA ESCOLHA ? ";
134 H$=INPUT$(1)
136 IF H$="A" THEN 200
138 IF H$="B" THEN 300
140 IF H$="C" THEN 406
142 BEEP : BEEP : GOTO 134
200 REM [ ================================ ]
202 REM [      ROTINA DE GRAVACAO          ]
204 REM [ ================================ ]
206 REM [   DIGITE 'FIM' PARA ENCERRAR     ]
208 REM [ ================================ ]
210 CLS
212 R=LOF(1)/24+1
214 PRINT"REGISTRO";R : PRINT
216 INPUT"MERCADORIA ";M$
218 IF M$="FIM" THEN CLS:GOTO 118
220 INPUT"QUANTIDADE ";Q%
222 INPUT"PRECO UNITARIO BRUTO ";PB!
224 INPUT"DESCONTO (%) ";D
226 PL#=(PB!)*((100-D)/100)
228 LSET N$=M$
230 RSET Q$=MKI$(Q%)
232 RSET PB$=MKS$(PB!)
234 RSET PL$=MKD$(PL#)
236 PUT #1,R
238 IF FL=1 THEN FL=0:RETURN
240 R=R+1 : PRINT : GOTO 214
300 REM [ ================================ ]
302 REM [      LEITURA E ALTERACAO         ]
304 REM [ ================================ ]
306 CLS
308 FOR I=1 TO LOF(1)/24
310 GET #1,I
312 Q%=CVI(Q$)
314 PB!=CVS(PB$)
316 PL#=CVD(PL$)
318 PRINT"REGISTRO";I : PRINT
320 PRINT"MERCADORIA       = ";N$
322 PRINT"QUANTIDADE       =";Q%
324 PRINT"PRECO UNIT.BRUTO =";PB!
326 PRINT"PRECO UNIT.LIQ.  =";PL#
328 PRINT:PRINT"QUER ALTERAR ? (S/N) ";
330 H$=INPUT$(1)
332 IF H$="S" THEN:R=LOC(1):CLS:FL=1:GOTO 338
334 IF H$="N" THEN GOTO 340
336 BEEP : BEEP : GOTO 330
338 GOSUB 214
340 PRINT : PRINT
342 NEXT I
344 CLS : GOTO 118
400 REM [ ================================ ]
402 REM [    ENCERRAMENTO DO PROGRAMA      ]
404 REM [ ================================ ]
406 POKE &HFCAB,0 : CLS : END
```



B – CARACTERES DO MSX

A tabela a seguir relaciona os 256 caracteres do MSX com seus códigos em Hexadecimal, sendo o primeiro dígito dado pela linha e o segundo pela coluna.

Para imprimir, por exemplo, o caractere L, basta comandar: PRINT CHR$(&H4C).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		☺	☻	♥	♦	♣	♠	•	◘	○	◙	♂	♀	♪	♫	☼
1	►	◄	↕	‼	¶	§	▬	↨	↑	↓	→	←	∟	↔	▲	▼
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	⌂
8	Ç	ü	é	â	ä	à	¨	ç	ê	ë	è	ï	î	ì	Ä	Å
9	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	ß	ƒ
A	á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐	¬	½	¼	¡	«	»
B	ã	ẽ	ĩ	õ	ũ	ỹ	ʼn	ĳ	Ĳ	ĵ	ķ	Ķ	ļ	Ļ	ŀ	Ŀ
C	ł	Ł	ś	Ś	ŵ	Ŵ	ŷ	Ŷ	ź	Ź	ż	Ż	ž	Ž	ſ	ƀ
D	Ɓ	ƃ	Ƅ	ƅ	Ɔ	Ƈ	ƈ	Ɖ	Ɗ	Ƌ	ƌ	ƍ	Ǝ	Ə	Ɛ	Ƒ
E	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	∅	∈	∩
F	≡	±	≥	≤	⌠	⌡	÷	≈	°	∙	·	√	ⁿ	²	■	

Quando o seu Expert envia caracteres à impressora, pode fazê-lo de duas formas: segundo o padrão MSX (tabela anterior) ou segundo a norma ABNT (tabela abaixo). Os comandos para ativar uma ou outra tabela são:

ABNT - POKE &HF417,0 ou SCREEN,,,,0

MSX - POKE &HF417,1 ou SCREEN,,,,1

MSX	ABNT	
AD	A1	¡
BF	A7	§
A6	AA	ª
AE	AB	«
F8	B0	°
F1	B1	±
FD	B2	²
E6	B5	µ
BE	B6	¶
A7	BA	º
AF	BB	»

MSX	ABNT	
AC	BC	¼
AB	BD	½
BA	BE	¾
A8	BF	¿
8F	C0	À
84	C1	Á
8C	C2	Â
B0	C3	Ã
80	C7	Ç
90	C9	É
8D	CA	Ê

MSX	ABNT	
89	CD	Í
A5	D1	Ñ
8A	D3	Ó
8E	D4	Ô
B4	D5	Õ
8B	DA	Ú
9A	DC	Ü
E1	DF	ß
85	E0	à
A0	E1	á
83	E2	â

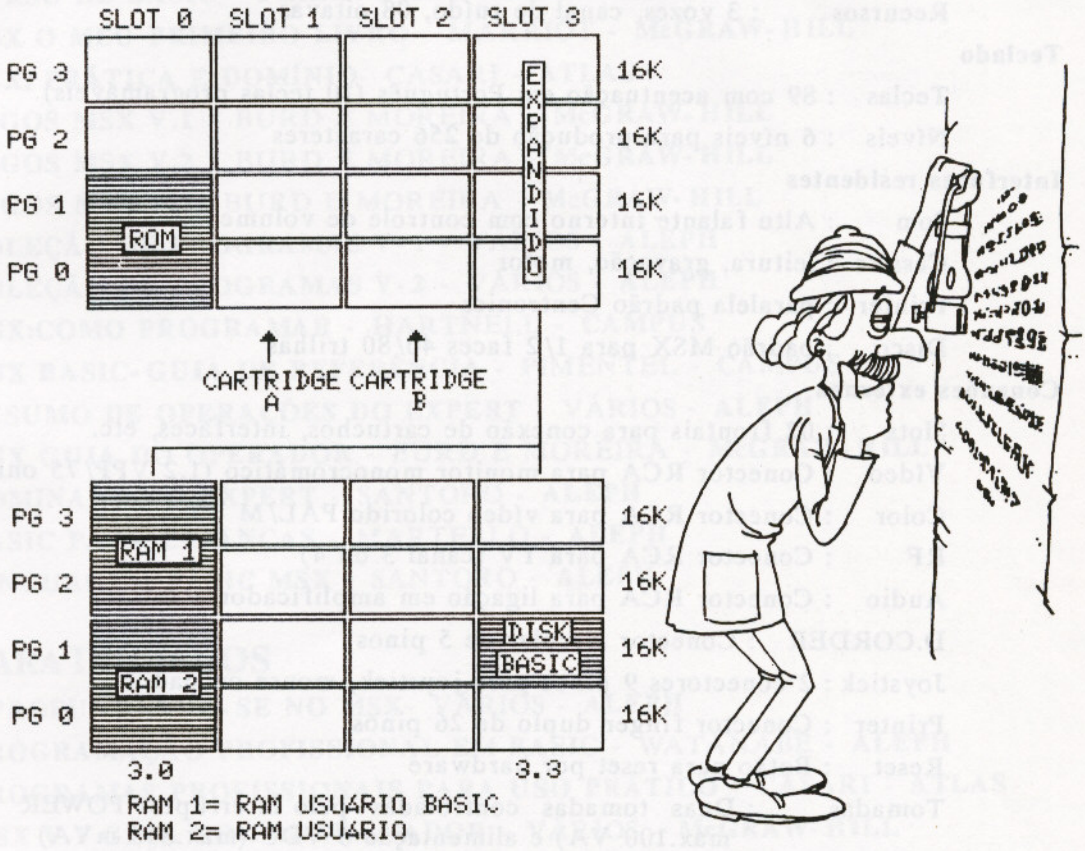
MSX	ABNT	
B1	E3	ã
87	E7	ç
82	E9	é
88	EA	ê
A1	ED	í
A4	F1	ñ
A2	F3	ó
93	F4	ô
B5	F5	õ
A3	FA	ú
81	FC	ü

C – MAPA DA MEMóRIA

O Expert DD PLUS possui 48 Kbytes de memória ROM (pré-gravada de maneira permanente) e 80 Kbytes de RAM (memória volátil para leitura e escrita). O microprocessador de vídeo (VDP) acessa 16 K e os outros 64 K são usados pelo microprocessador principal (Z80).

O Expert DD PLUS tem 4 slots, sendo dois para uso interno (0 e 3) e dois externos (slot 1 = cartridge A ; slot 2 = cartridge B). A prioridade dos slots vai do menor para o maior. Cada slot suporta 64 K e é dividido em 4 páginas de 16 K cada. Os slots externos podem ser expandidos em 4 sub-slots, suportando, portanto, 256 K cada um. No Expert DD PLUS o slot 3 (interno) já está expandido em sub-slots 3.0 e 3.3).

Veja na figura abaixo o esquema destas sub-divisões.



Para o usuário que programa em BASIC, ficam disponíveis as páginas 2 e 3 do sub-slot 3.0. As duas páginas inferiores podem ser acessadas ,em linguagem de máquina, mudando-se a configuração da PPI.

D – ESPECIFICAÇÕES TÉCNICAS

Microprocessador

Z80A (3,58 MHz)

Memória residente

ROM : 32 K (BASIC MSX) +16 K (DISK-BASIC)

RAM : 64 K (usuário) +16 K (vídeo)

Linguagem residente

BASIC MSX, DISK-BASIC, Linguagem de Máquina Z80

Vídeo

VDP : TMS-9128NL
Texto 1 : 32 colunas x 24 linhas
Texto 2 : 40 colunas x 24 linhas
Gráficos: 256 x 192 pontos
Sprites : 32 níveis
Cor : 16 cores

Som

PSG : AY-3-8910-A
Recursos : 3 vozes, canal de ruído, 08 oitavas

Teclado

Teclas : 89 com acentuação em Português (10 teclas programáveis)
Níveis : 6 níveis para produção de 256 caracteres

Interfaces residentes

Som : Alto falante interno com controle de volume
Cassete : leitura, gravação, motor
Printer : paralela padrão Centronics
Disco : padrão MSX para 1/2 faces 40/80 trilhas

Conexões externas

Slots : 02 frontais para conexão de cartuchos, interfaces, etc.
Vídeo : Conector RCA para monitor monocromático (1,2 VPP/75 ohms)
Color : Conector RCA para vídeo colorido PAL/M
RF : Conector RCA para TV (canal 3 ou 4)
Áudio : Conector RCA para ligação em amplificador.
D.CORDER : Conector circular de 5 pinos
Joystick : 2 conectores 9 pinos para joystick, mouse ou paddle
Printer : Conector finger duplo de 26 pinos
Reset : Botão para reset por hardware
Tomadas : Duas tomadas controladas pelo interruptor POWER (carga máx.100 VA) e alimentação 6 VDC (máx.840 mVA)

Alimentação

Tensão : AC 120/240 V, 60 Hz, comutável
Consumo : 30 VA (só CPU), máx.42 VA(com periféricos)

Acessórios

Cabos : 1 cabo RF 75 ohms RCA-RCA
Fusíveis : 1 para 120V e 1 para 240V
Comutador : 1 comutador TV/Computer
Etiquetas : 1 conjunto de etiquetas para as teclas de função
Livro : 1 Manual de instruções

E – BIBLIOGRAFIA ACONSELHADA

LINGUAGEM BASIC

PARA PRINCIPIANTES

MSX GUIA DO USUÁRIO - HOFFMAN - McGRAW-HILL
MSX BASIC...SEM DOR - MARTELLO - CAMPUS
CURSO DE BASIC - PIAZZI E CARVALHO - ALEPH
MSX O MEU PRIMEIRO LIVRO - MARRIOT - McGRAW-HILL
MSX: PRÁTICA E DOMÍNIO- CASARI - ATLAS
JOGOS MSX V.1 - BURD E MOREIRA - McGRAW-HILL
JOGOS MSX V.2 - BURD E MOREIRA - McGRAW-HILL
JOGOS MSX V.3- BURD E MOREIRA - McGRAW-HILL
COLEÇÃO DE PROGRAMAS V-1 - VÁRIOS - ALEPH
COLEÇÃO DE PROGRAMAS V-2 - VÁRIOS - ALEPH
MSX:COMO PROGRAMAR - HARTNELL - CAMPUS
MSX BASIC- GUIA DE REFERÊNCIA - PIMENTEL - CAMPUS
RESUMO DE OPERAÇÕES DO EXPERT - VÁRIOS - ALEPH
MSX GUIA DO OPERADOR - BURD E MOREIRA - McGRAW-HILL
DOMINANDO O EXPERT - SANTORO - ALEPH
BASIC PARA CRIANÇAS - MARTELLO - ALEPH
LINGUAGEM BASIC MSX - SANTORO - ALEPH

PARA INICIADOS

APROFUNDANDO-SE NO MSX- VÁRIOS - ALEPH
PROGRAMAÇÃO PROFISSIONAL EM BASIC - WATANABE - ALEPH
PROGRAMAS PROFISSIONAIS PARA USO PRÁTICO - CASARI - ATLAS
MSX V.1 GUIA DO PROGRAMADOR - VÁRIOS - McGRAW-HILL
MSX V.2 GUIA TÉCNICO DE REFERÊNCIA - VÁRIOS - McGRAW-HILL
100 DICAS PARA MSX - VÁRIOS - ALEPH
+50 DICAS PARA MSX - VÁRIOS - ALEPH

APLICAÇÕES ESPECIAIS

SIMULAÇÕES MSX - BURD - McGRAW-HILL
MSX MÚSICA - BUSSAB - McGRAW-HILL
CURSO DE MÚSICA MSX - BARBIERI E PIAZZI - ALEPH
CIRCUITOS ELETRÔNICOS - FRIEDMANN - ALEPH
DESENHOS BÁSICOS - CARVALHO LIMA - ALEPH
ASTROLOGIA NO MSX- CARVALHO - ALEPH

LINGUAGEM DE MÁQUINA

PARA PRINCIPIANTES

JOGOS DE HABILIDADE - VÁRIOS - ALEPH
LINGUAGEM DE MÁQUINA MSX - FIGUEREDO E ROSSINI - ALEPH
ASSEMBLER PARA O MSX- CARVALHO - McGRAW-HILL
TABELA DE MNEMÔNICOS Z80 - VÁRIOS - ALEPH

PARA INICIADOS

O LIVRO VERMELHO DO MSX - AVALON - McGRAW-HILL
PROGRAMAÇÃO AVANÇADA EM MSX - FIGUEREDO - ALEPH

USOS DO DISK DRIVE

MSX COM DISK DRIVE - CASARI - McGRAW-HILL
DRIVES LEOPARD DE 3 1/2" - VÁRIOS - ALEPH
USANDO O DISK DRIVE NO MSX - PEREIRA - ALEPH
SISTEMA DE DISCO PARA MSX- OLIVEIRA E PEREIRA - ALEPH

DBASE II

DBASE II PLUS INTERATIVO - CASARI - ATLAS
DBASE II PLUS PROGRAMÁVEL- CASARI - ATLAS

LINGUAGEM LOGO

HOTLOGO- PRIMEIROS PASSOS - VÁRIOS - ALEPH

MANUAIS DE SOFTWARE

MSX: USANDO OS MELHORES APLICATIVOS V1 - SEABRA - CAMPUS
MSX: USANDO OS MELHORES APLICATIVOS V2 - SEABRA - CAMPUS
HOTDATA - WATANABE - ALEPH
HOTWORD - WATANABE - ALEPH
HOTPLAN - WATANABE - ALEPH
HOTWORD- HOTDATA- HOTPLAN - SEABRA - CAMPUS

EDITORAS CITADAS

ALEPH

Aleph Publicações e Assessoria Pedagógica Ltda

Av. Dr. Luiz Migliano, 1110 c.301/303 Portal Trade Center

05711 São Paulo SP (011) 843-3202 843-0514

ATLAS

Editora Atlas S.A.

R. Cons. Nébias, 1384 Campos Elísios

C.P. 7185 01203 São Paulo SP (011) 221-9144

CAMPUS

Editora Campus Ltda

R. Barão de Itapagipe, 55 20261 Rio Comprido RJ

(021) 284-8443

McGRAW-HILL

Editora McGraw-Hill Ltda

R. Tabapuã, 1105 Itaim Bibi

04533 São Paulo SP (011) 881-8605 881-8528

NOTAS

NOTAS

NOTAS

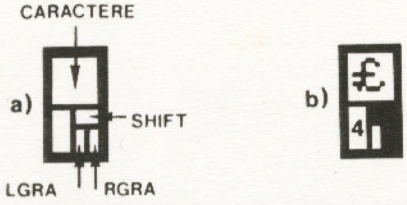
NOTAS

NOTAS

Na figura abaixo, você tem todos os 256 caracteres relacionados com seus códigos hexadecimais.
Por exemplo, se você quiser saber qual o caractere de código hexadecimal 9C, basta procurar na linha 9, coluna C, e você encontrará o símbolo da libra esterlina. Se você comandar

PRINT CHR$ [&H9C]

obterá esse símbolo na tela.
No campo abaixo do caratere, você encontra a combinação de teclas que devem ser pressionadas, simultaneamente, para obtê-lo via teclado [figura [a]].
No caso da libra esterlina, por exemplo, você encontra o correspondente à figura [b]. Isto significa que, para obter o símbolo da libra esterlina você deverá pressionar, simultaneamente as teclas 4+RGRA+SHIFT.



gradiente

Rua Henrique Monteiro, 90 - Pinheiros
CX.P. 30318 - CEP 05423 - São Paulo - SP