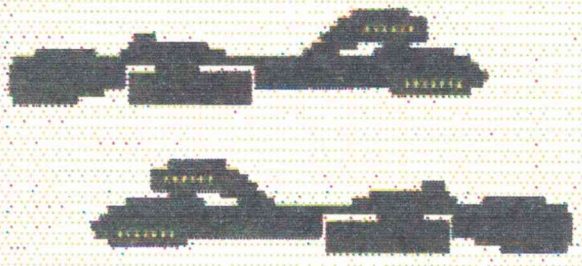
ANO IV - Nº 44 MAIO 1985 - Cr$ 6.300

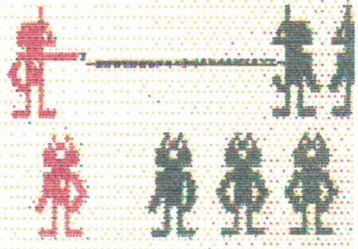
PROLOG
A 5ª REVOLUÇÃO VEM DO SOFTWARE

Micro Sistemas

A PRIMEIRA REVISTA BRASILEIRA DE MICROCOMPUTADORES

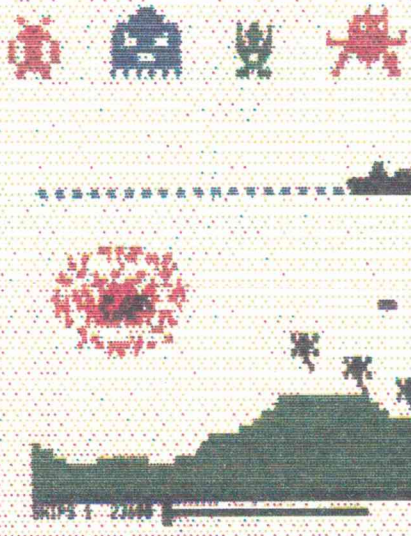


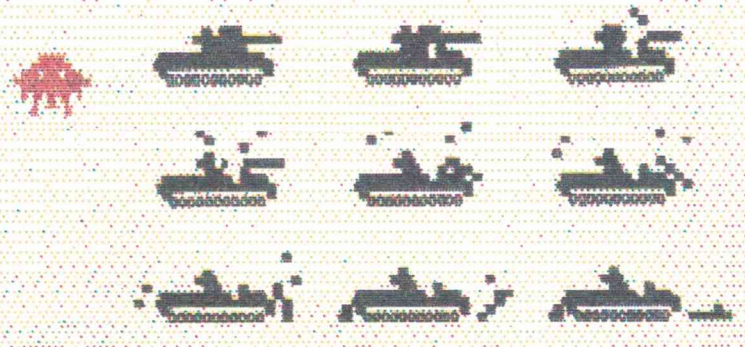
EDITOR DE ANIMAÇÃO EM SINCLAIR



DICAS DE PROGRAMAÇÃO BASIC

A CAPACIDADE GRÁFICA DOS MICROS



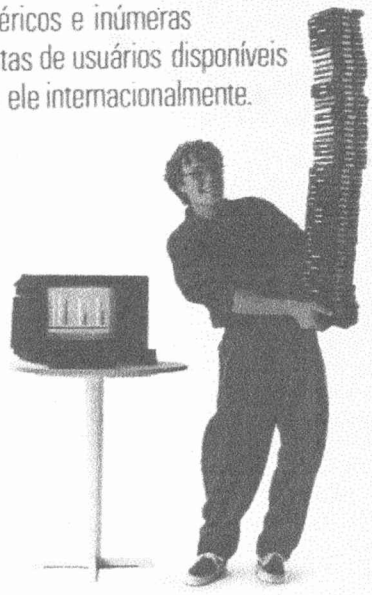


A Microdigital lança no Brasil o micro pessoal de maior sucesso no mundo.

A partir de agora a história dos micros pessoais vai ser contada em duas partes: antes e depois do TK 90X.

O TK 90X é, simplesmente, o único micro pessoal lançado no Brasil que merece a classificação de "software machine": um caso raro de micro que pela sua facilidade de uso, grandes recursos e preço acessível recebeu a atenção dos criadores de programas e periféricos em todo o mundo.

Para você ter uma idéia, existem mais de 2 mil programas, 70 livros, 30 periféricos e inúmeras revistas de usuários disponíveis para ele internacionalmente.



E aqui o TK 90X já sai com mais de 100 programas, enquanto outros estão em fase final de desenvolvimento para lhe dar mais opções para trabalhar, aprender ou se divertir que com qualquer outro micro.

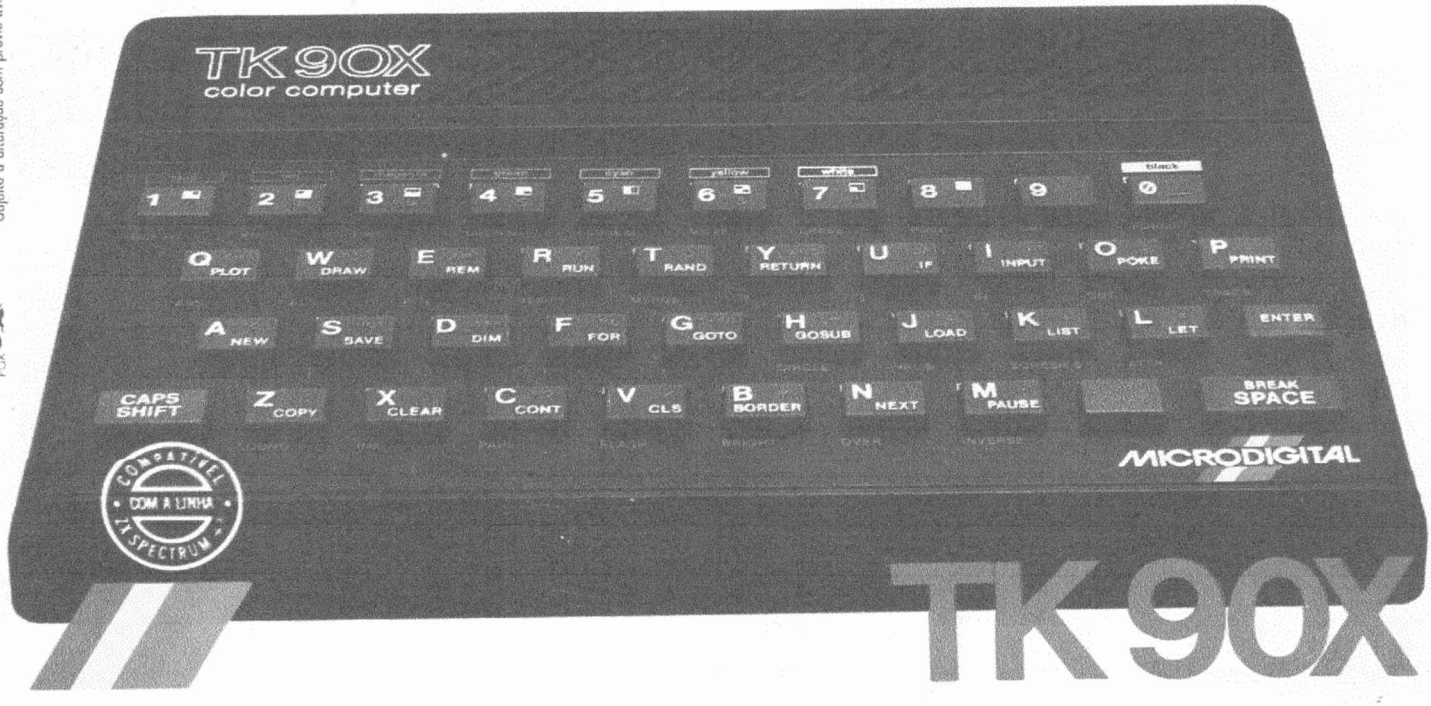
O TK 90X tem duas versões de memória (de 16 ou 48 K), imagem de alta resolução gráfica com 8 cores, carregamento rápido de programas (controlável pelo próprio monitor), som pela TV, letras maiúsculas e minúsculas e ainda uma exclusividade: acentuação em português.

Faça o seu programa: peça já uma demonstração do novo TK 90X.

Preço de lançamento*
16 K - Cr$ 1.499.850 • 48 K - Cr$ 1.749.850

MICRODIGITAL

Chegou o micro cheio de programas.



*Sujeito a alteração sem prévio aviso.

FOX

Filiada à ABICOMP

* ALL RIGHTS RESERVED SINCLAIR RESEARCH LTD

TK 90X, um Spectrum no Brasil

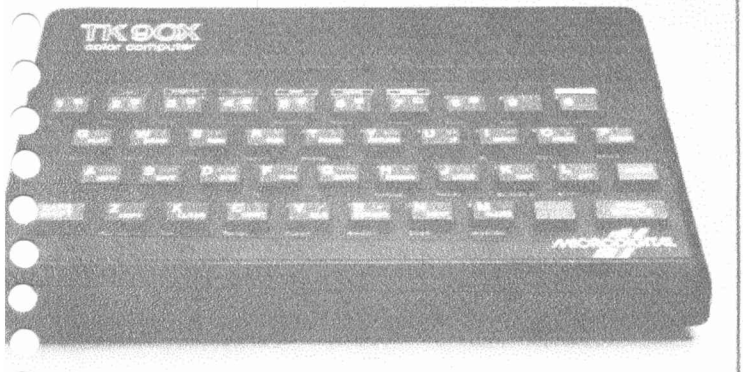
Numa época do ano em que todos os fabricantes fazem sigilo dos equipamentos que pretendem lançar na Feira de Informática, a paulista Microdigital lançou seu torpedo em pleno final de junho. Apresentou à imprensa e aos revendedores, em regime de entrega imediata, o TK 90X Color Computer.

Conforme anunciamos em nossa edição 45, o novo produto é compatível com o ZX Spectrum, da empresa inglesa Sinclair, com a qual, aliás, a mesma Microdigital travou, faz pouco tempo, uma bem-sucedida batalha judicial (veja em MS 46, BITS). Os tiros não pararam, é o que parece. Questionado se não seria *brincar com fogo* lançar outro produto da linha Sinclair, George Kovari, Presidente da empresa brasileira, ameaça bem-humorado: "Estamos lançando um micro internacional, fruto de pesquisa e ainda melhorado. Vamos sedimentar o produto no mercado mundial, e depois *soltar a bomba* em Londres, a um preço competitivo. Quanto à Sinclair, nem quero saber...".

Qualificações para se tornar um sucesso, sem dúvida o TK 90X possui, e experiência também. Colocado no mercado argentino um mês antes de ser lançado no Brasil, o TK 90X, por seus recursos de cor, som, alta resolução e vasta biblioteca de programas, constitui-se uma peça chave na já esboçada estratégia da Microdigital de ganhar o mercado de exportação. Para tal, os manuais de operação do equipamento já são disponíveis em português e espanhol, e a diretoria garante que estão sendo traduzidos para o inglês.

As principais diferenças entre o Spectrum original e o TK 90X, segundo a Microdigital, são o comando TRACE, para detectar erros; saída de som diretamente pela TV; rotina para definição de caracteres pelo usuário; interface para joystick e possibilidade de acentuação em português e espanhol. Com relação à pequena impressora utilizada com o Spectrum, a Microdigital preferiu não se comprometer — esta impressorinha já deu muita dor de cabeça à empresa, quando foi prometida para a linha dos TKs 82/85 —, limitando-se a afirmar que "há possibilidade dela vir a ser fabricada".

Uma especial ênfase tem sido dada, pelo Marketing, às aplicações da área educacional, explorando a disponibilidade da linguagem LOGO nesta máquina. O preço divulgado por ocasião do lançamento era de Cr$ 1,5 milhão para a versão 16 Kb e Cr$ 2 milhões para 48 Kb. Os boatos de que o TK-85 teria a sua produção descontinuada não tiveram eco, embora a própria direção argumente que "não se pode prever a vida de um produto".



TK 90X Color Computer

processador: *Z 80A; 3,58 MHz*
memória: *16 Kb ROM, com interpretador BASIC*
16 ou 48 Kb RAM
memória externa: *gravador cassete; velocidade 1200 bauds*
teclado: *tipo Qwerty; com auto-repeat e beep*
40 teclas contendo funções e comandos do BASIC
maiúsculas e minúsculas; acentuação
vídeo: *modo texto de 24 x 32*
modo gráfico de 256 x 192
cores: *preto, azul, vermelho, lilás, verde, ciano, amarelo e branco*

Tradutor no TRS-80

A software-house SCL, de Belo Horizonte, dispõe de uma linha de programas para a área médica e hospitalar. São eles Estatística Hospitalar, Sistema Integrado de Custos e Controle de Cadastros.

Além destes, a SCL tem um Controle de Fichário para Cartórios de Registros de Imóveis, Folha de Pagamento (com emissão de RAIS e Imposto de Renda), Controle de Estoque e Controle Fiscal. Todos os programas são escritos para a linha TRS-80 e em linguagem de máquina, exceto o Controle Fiscal, escrito em BASIC-80 para rodar sob CP/M.

Disponível para os modelos TRS-80 I e III, há o Dicionário Tradutor que permite ao usuário criar um arquivo de até 4.300 palavras e suas respectivas traduções. O software comporta expressões idiomáticas e possui um sistema próprio para atualizar os arquivos do Backup. A SCL fica na Av. Augusto de Lima, 1646/1005 — Belo Horizonte (MG). Telefone: (031) 335-5120.

Contra a pirataria

A Brasoft, representante da Micropro no Brasil, está fazendo a tradução de telas e manual do processador de textos Wordstar 2000. A versão em português deverá ser lançada na próxima Feira de Informática, em setembro, e os usuários que já possuem o 2000 poderão trocar pela nova versão, cujo preço deverá ser o mesmo que o atual, 135 ORTN.

A Brasoft está lançando também, juntamente com a Datalógica (representante da Ashton-Tate) e a Sacco Computer Store (distribuidora da Lotus em SP), uma campanha contra a pirataria de software. Segundo Aldo Forlin, gerente comercial, o objetivo dessa campanha é conscientizar o usuário para os riscos que ele corre ao comprar uma cópia pirata, tais como a falta de suporte e de assistência técnica.

Novidades da BBS

A Biblioteca Brasileira de Software está lançando, em conjunto com a Hemus Editora Ltda., cartões de referência para micros das linhas Sinclair, TRS-80 e Apple. Estes cartões trazem, entre outras coisas, os mapeamentos de memória; instruções internas; código de caracteres e códigos ASC, e podem ser encontrados nas livrarias e lojas de informática. Além disto, a BBS também está colocando no mercado um software para ligação dos micros da linha Apple ao projeto Cirandão. O programa pode ser encontrado na própria Biblioteca, que fica na Av. Brigadeiro Faria Lima, 1.390/Cj. 82.

Impressoras da Elgin

A Elgin está lançando a impressora MT 250L, voltada para o mercado de OEM, com velocidade de 250 cps para processamento de dados e 50 cps para textos. A nova impressora possui alta definição gráfica, qualidade de carta e apresenta comandos para o ajuste das margens, espaçamento proporcional e movimentação horizontal para modificar o espaço entre os caracteres.

Uma outra novidade é a MT 440D para código de barras, que através de uma seqüência de barras imprime 13 dígitos, caracterizando o produto, seu fabricante e país de origem.

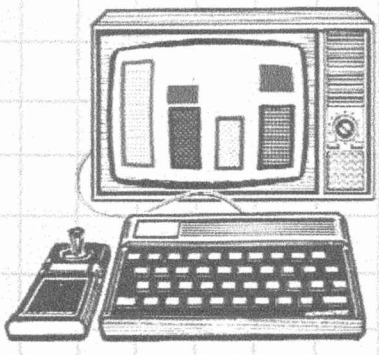
MICROCOMPUTADOR MICRODIGITAL

TK-90X

Color y sonido a través del T.V.

16K y 48 K

EL MICROCOMPUTADOR CON MILES DE PROGRAMAS



GARANTIA 6 MESES

En venta en comercios de microcomputadores, artículos del hogar, electrónica, fotografía y librerías.

SOFTWARE Y PERIFERICOS TOTALMENTE COMPATIBLES CON ZX SPECTRUM +" ®

- Control del volumen del sonido a través del TV (sintetizador operado por BASIC)
- Interface incorporado para joystick
- Mensajes de ejecución y codigo de reportes de errores en castellano.
- TRACE: Comando de seguimiento de programas, permitiendo la rapida corrección de errores de lenguaje.
- UDG: Comando de editor de caracteres especiales definidos por el usuario (acentos, Ñ, etc.).
- Feedback sonoro del teclado
- Fuente de alimentación con interruptor.
- Ameno, facil y completo manual de instrucciones en castellano.

MICRODIGITAL

Importa, distribuye y garantiza:
ARVOC s.a.i.c.f.i.
Avda. DIAZ VELEZ 4149 (1200) Capital Federal Tel.: 981-1980/9212

ZX SPECTRUM +" ALL RIGHTS RESERVED SINCLAIR RESEARCH LTD

CHEGOU O TK95.

O QUE ELE FAZ NÃO É BRINCADEIRA.

O novo TK 95 é o único micro em sua categoria que trabalha como gente grande.

Enquanto outros micros são usados basicamente para brincar com jogos eletrônicos, o TK 95 trabalha com impressora, Video-Texto, Mouse. E ainda permite o uso da exclusiva Light Pen, que desenha direto na tela da Televisão. Quer dizer, enquanto outros micros ainda estão na infância, o TK 95 já indo à escolas, universidades. Ele usa uma linguagem "Logo", para educação, muito superior à dos outros.

Prova disso, é que foi aprovado pelo Ministério de Educação da Inglaterra. Aliás, graças a sua facilidade de uso, é o micro ideal para se iniciar em informática.

O TK 95 faz mais do que os outros, e faz melhor: tem melhor resolução gráfica e oito cores, que podem, inclusive ser combinadas, até 64 cores. E para acabar com qualquer comparação, o TK 95 tem mais de 1000 softwares disponíveis.

Inclusive modernissimos jogos eletrônicos, porque ninguém é de ferro. Nem o computador.

MICRODIGITAL

Hardware

Lançado na última Feira de Informática, o TK95 é hoje o mais sofisticado membro da família ZX Spectrum no País. Acompanhe a equipe do CPD de MS e descubra as novidades que esse micro tem incorporadas em relação ao seu irmão, o TK90X.

O TK95 da Microdigital

A Microdigital Eletrônica Ltda acaba de marcar mais um ponto altamente positivo com o lançamento do microcomputador TK95, versão profissional do TK90X (agora batizado como versão *standard*), complementando sua linha de produtos baseados no inglês ZX Spectrum.

Salvo pequenas modificações na ROM, o novo micro possui todas as características, tanto de hardware quanto de software, já existentes em seu irmão mais velho. As principais diferenças estão em seu bonito gabinete, que incorpora um teclado profissional e um LED indicativo da condição de ligado.

O novo teclado, de muito boa qualidade e ótimo toque, possui teclas específicas de acionamento único para todas as funções que necessitavam do acionamento simultâneo das teclas CAPS SHIFT e SYMBOL SHIFT, aumentando sensivelmente a velocidade de digitação e justificando a denominação de versão profissional. Acompanha o micro um novo manual, único para as duas versões (TK90X e TK95), incluindo três novos apêndices que tratam da utilização da light-pen, *Parallel Printer Interface* e da *Serial Interface* RS-232C.

O único ponto negativo, ao nosso ver, é o fato de não termos sido brindados, ainda desta vez, com um botão ou tecla de RESET. Parece-nos, ainda, que uma apreciável melhoria, no sentido de uma maior profissionalização, seria embutir a fonte de alimentação no gabinete. Reconhecemos, contudo, as razões de economia de produção que devem ter levado a Microdigital a manter uma única fonte para as duas versões, o que, de resto, não chega a comprometer o conjunto.

CARACTERÍSTICAS GERAIS

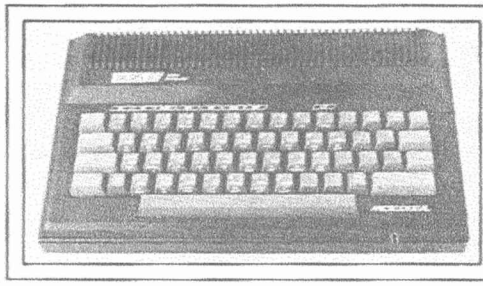
Microprocessador Z80A de 8 bits (clock de 3,58 MHz); 16 Kb de memória ROM; 48 Kb de memória RAM (o TK90X possui uma versão com 16 Kb de memória RAM); teclado profissional padrão QWERTY com teclas auto-repetitivas e sonorizadas; saída de som pelo alto-falante do televisor; entrada e saída para gravador cassete; saída para televisor preto e branco ou a cores (sistema PAL-M); conector de expansão para interface serial, impressora, light-pen, mouse e interface para joystick incorporada; resolução gráfica de 256 x 192 pontos (alta resolução), com 24 linhas x 32 colunas para textos, dispondo de oito cores para ambos.

NOVO TECLADO

O teclado do TK95 segue a mesma disposição do seu irmão TK90X, com exceção de algumas teclas de função única:

TK95	TK90X
<DELETE>	<CAPS SHIFT>+<0>
<CAPS LOCK>	<CAPS SHIFT>+<2>
<EXTENDED MODE>	<CAPS SHIFT>+<SYMBOL SHIFT>
<GRAPHICS>	<CAPS SHIFT>+<9>
<INVERSE VIDEO>	<CAPS SHIFT>+<4>
<TRUE VIDEO>	<CAPS SHIFT>+<3>
<EDIT>	<CAPS SHIFT>+<1>
<' '>	<SYMBOL SHIFT>+<P>
<BREAK>	<CAPS SHIFT>+<SPACE>
<SETAS DE DIREÇÃO>	<CAPS SHIFT>+<5, 6, 7 ou 8>
<;>	<SYMBOL SHIFT>+<O>
<.>	<SYMBOL SHIFT>+<M>
<,>	<SYMBOL SHIFT>+<N>

Foto de Mônica Leme



O TK95 com o novo teclado em destaque.

Embora essa nova aparência sugira profundas modificações na parte de hardware do teclado, sua configuração *standard* permanece camuflada dentro do próprio sistema da ROM, encarregado de realizar a leitura e decodificação do mesmo, ou seja, todas as funções das teclas individuais podem ser realizadas pelo acionamento do conjunto de teclas discriminado no quadro anterior, referentes ao TK90X.

CONCLUSÃO

Devido a algumas modificações feitas na nova ROM do TK95, o micro adquiriu capacidade para executar muitos daqueles programas que não rodavam em seu irmão, o TK90X, porém sua compatibilidade para com o ZX Spectrum ainda não é completa, pois certos programas (como, por exemplo, o LERM-8) não podem ser executados.

Com este lançamento da Microdigital, a comunidade de usuários da linha ZX Spectrum adquire a possibilidade de poder optar, dentre as duas versões de um mesmo equipamento de utilidade e qualidade comprovadas, por aquela que melhor se adapte às suas necessidades e/ou disponibilidades financeiras. Temos a certeza de que o TK95 seguirá a mesma trajetória de sucesso do TK90X. **Análise feita pelo CPD de MS.**

Nome: TK95
Fabricante: Microdigital Eletrônica Ltda.
Endereço: Rua do Bosque, 1234 – Barra Funda, São Paulo-SP CEP 01136.
Telefone: (011) 66-2675 e 67-5528.
Preço: Cz$ 6 mil e 800 (valor em março)

A segunda memória do seu computador: gravador National RQ-2222.

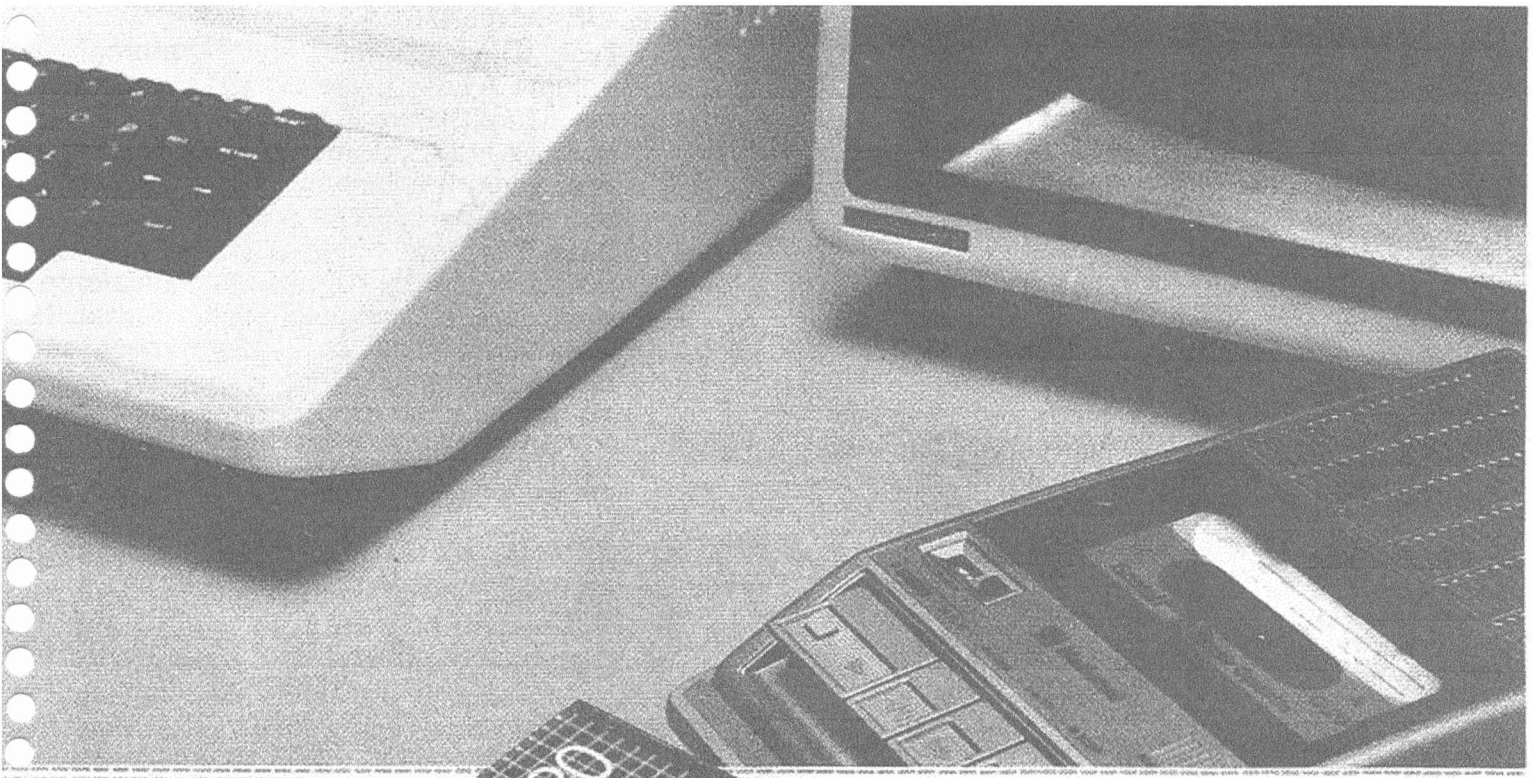
O gravador National RQ-2222 é o preferido pelos usuários de microcomputadores. Ele grava e carrega programas com a mais alta fidelidade e com a maior economia.

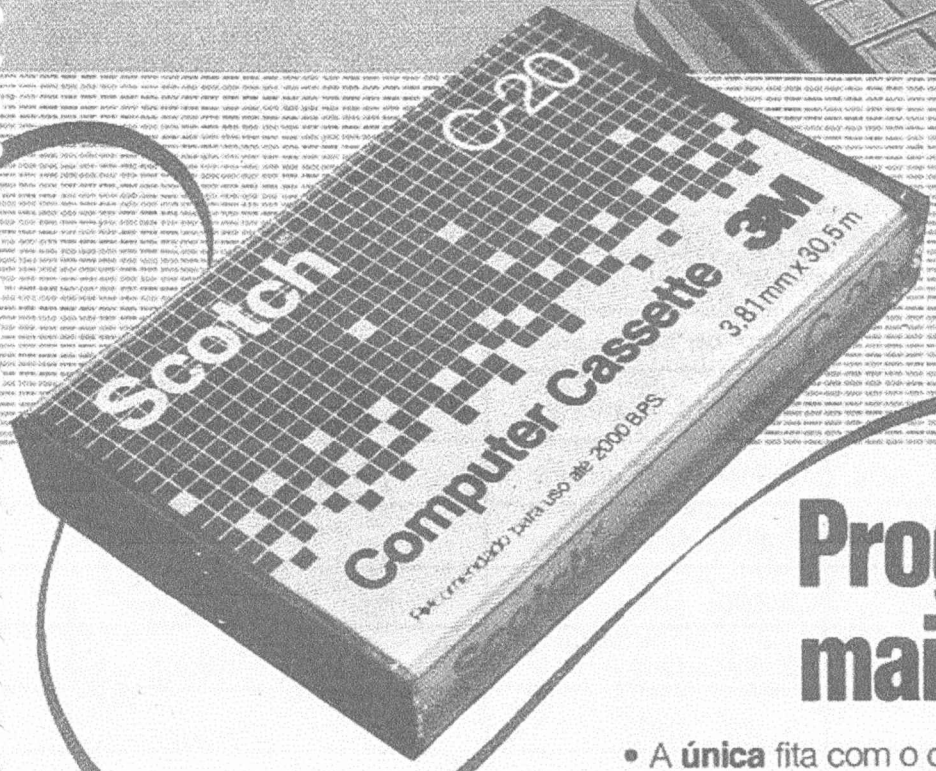
Porque tem um sistema de cabeçote próprio para o uso em micros. O gravador National RQ-2222 tem um contador de fita que facilita a localização do programa a ser utilizado. E tem também um comando único para gravação tipo "Um Toque", muito mais prático. Um gravador que vive na memória do computador merece também viver na sua.

Grave este nome: National RQ-2222.

National

Produzido na Zona Franca de Manaus. Conheça o Amazonas.





NOVA

Programações mais seguras.

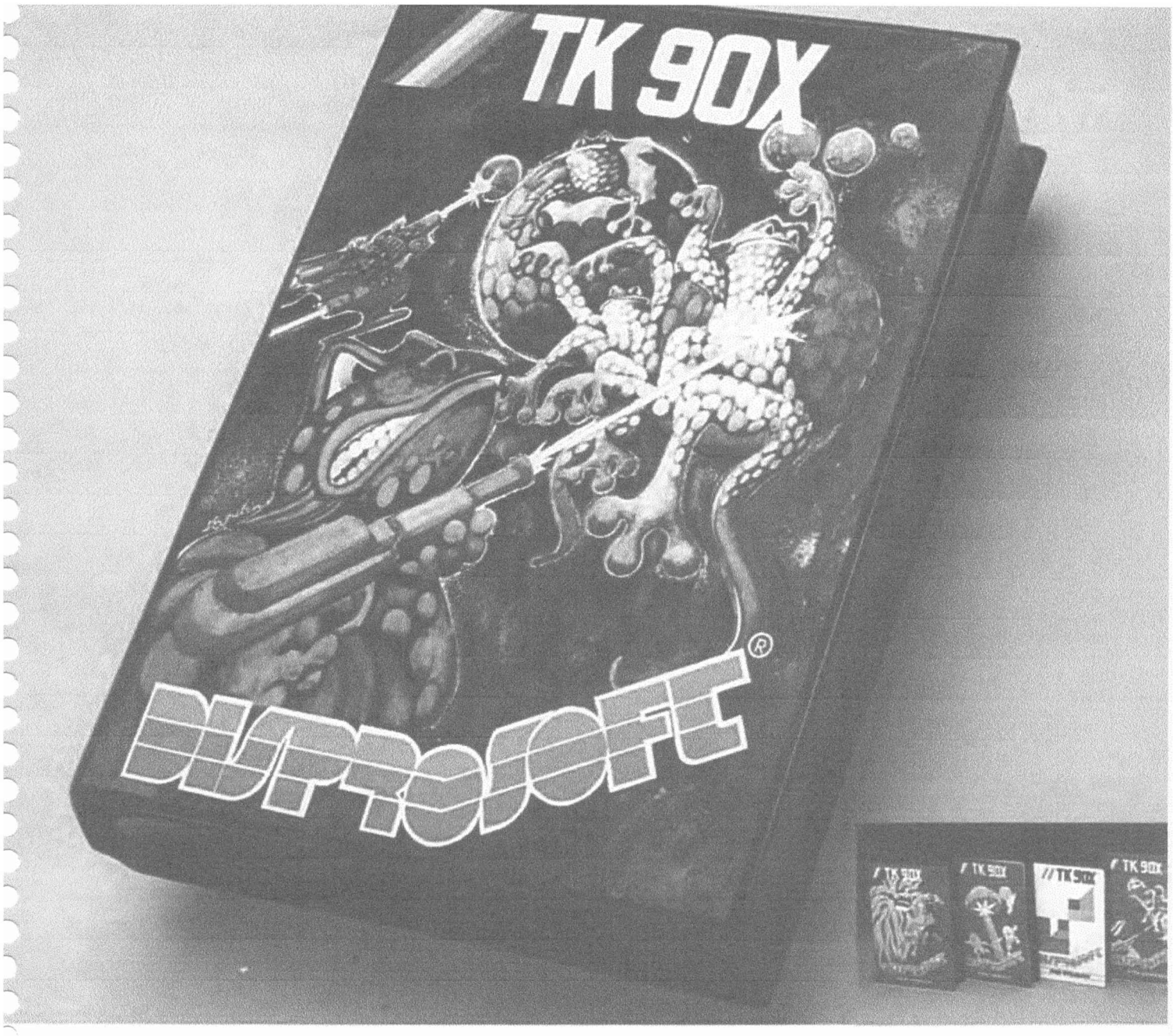
- A **única** fita com o comprimento especialmente dimensionado para 20 minutos.
- Evita a regravação de programas.
- Elimina a ocorrência de **dropouts.**
- Requer um torque mais baixo do gravador.
- Proporciona maiores ganhos nas frequências médias e altas.

Seus programas merecem o Computer Cassette C-20 Scotch!

A fita da mais alta segurança, com a qualidade Scotch.

Produtos Magnéticos CPD/3M
Caixa Postal 123 - Campinas - SP

3M



Programas para TK 90X que falam nossa língua.

Finalmente, uma empresa nacional assumiu uma atitude brasileira: editar programas e jogos para TK 90 X em **português**. A Disprosoft está lançando programas inéditos no Brasil.

São jogos animados, inteligentes, programas com aplicações profissionais, educacionais, comerciais e utilitários. Procure o seu programa, nas melhores lojas.

DISPROSOFT

SEMPRE UM GRANDE PROGRAMA.

CAIXA POSTAL 16441

A Disprosoft está com um apetite enorme para conquistar você. E lançou novos sabores no mercado.
Jogos animados e inteligentes, programas com aplicações comerciais, educacionais, profissionais e utilitárias. São os mais vendidos no Brasil. Refrescantes e deliciosas emoções, em português, para usuários inteligentes de TK-90X, TK-95 e MSX.
Experimente estas verdadeiras vitaminas de micro. Elas estão irresistíveis.

DISPROSOFT®

SEMPRE UM GRANDE PROGRAMA.
TROPIC INFORMÁTICA LTDA.
Cx. Postal 16441 - CEP 02599

LINHA MSX

COMERCIAIS
- Contas a Pagar/Receber
- Matrizes Complexas
- Eletricidade
- Condutibilidade
- Geometria Plana
- Ótica
- Disprocalc (Planilha)

EDUCACIONAIS
- Aprendendo a Contar
- Psyco
- Maior/Menor
- Circo
- Mágico
- Magia
- Kriptos

UTILITÁRIOS
- Editor de Sprites
- Desenhista - Construtor de Desenhos
- Eddy II
- ASMX
- Compilador Basic
- Compositor Musical

CASSINO
- Slot Machine
- 21 - Baralho
- Vídeo Poker
- Strip Poker

JOGOS ANIMADOS
- Futebol - Super Soccer
- XyZolog
- Boxeur
- Ninja
- Hero

JOGOS ANIMADOS
- Turbo AT
- Mr. Gomoku
- Xadrez
- Pill Box
- Mole-mole
- Ye ar Kung Fu II
- Goonies
- Alfa Squadron
- Lode Runner
- Kung Fu
- Elevator Action
- Golf
- Kings Valley

COLLECTION LINE
- Coleção com 12 Programas de Sucesso em embalagem especial.

LINHA TK-90X-95

COMERCIAIS
- Cadastro de Clientes
- Controle de Estoques

CASSINO
- Sinuca Inglesa
- Strip Poker
- 21 Strip

BEST SELLERS
- Sabotador

JOGOS ANIMADOS
- Cosmic Wartoad
- Cauldron
- Zorro
- West Bank
- Cookie
- Desenhista de Jogos
- Pare o Trem
- Caça-fantasma
- Reversi
- Sabre Wulf
- Metabolis
- Monstros
- Horace and the Spider
- Penetrator
- Pegásus
- Piloto de Caça
- Pássaros e Abelhas
- Arcadia
- Campo Minado
- Torre do Inferno
- Monty o Inocente
- Astro Blaster
- Abdactors
- Blind Alley
- Labirinto 3 D
- O Feiticeiro
- Millipede
- Trans-am
- Tele Porting

COLLECTION LINE
- Coleção com 12 Programas de Sucesso em embalagem especial.

À VENDA NOS MAGAZINES E LOJAS ESPECIALIZADAS

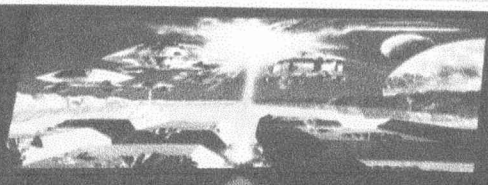
Os melhores programas para você.

PARA TK 2000 COLOR

MICROSOFT

MICROSOFT

Garantia integral



MICROSOFT

MICROSOFT

MICROSOFT

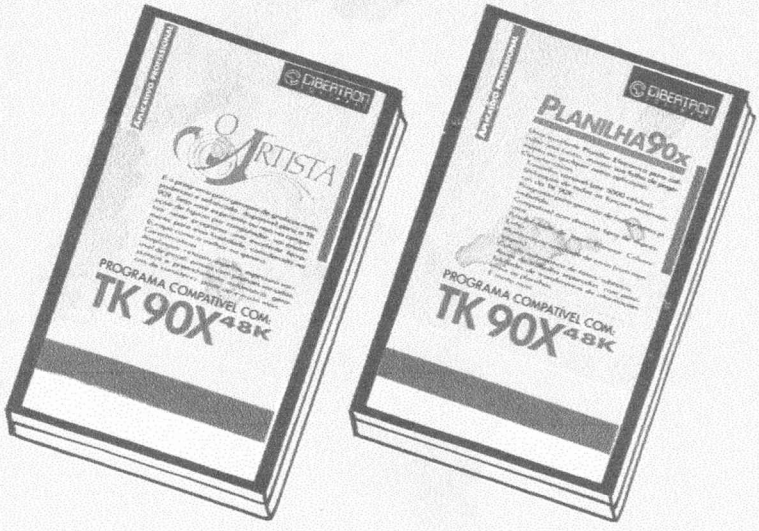
MICROSOFT

A Microsoft tem 120 programas em fitas e disquetes à sua disposição. São sistemas aplicativos para acompanhar e agilizar os negócios de sua empresa. E também jogos eletrônicos para você e sua família se divertirem muito. Todos especiais para TK-83, TK-85, TK-2000, Apple II e compatíveis. E todos com a mesma qualidade dos 100.000 programas já vendidos em todo o Brasil.
Procure o revendedor Microsoft mais próximo (se não encontrar os programas Microsoft escreva para a Caixa Postal 54221 - CEP 01000 - S. Paulo-SP). Você encontrará os melhores programas da sua vida.

MICROSOFT®

Sempre o melhor programa.

TK 90X: MICROCOMPUTADOR...



Para você que quer explorar todo o potencial do seu micro TK 90X como ferramenta de trabalho e como fonte de lazer criativo, a Cibertron tem novidades. "O Artista", "Controle de Estoque", "Banco de Dados"* e "Planilha 90X"*, ao preço de Cz$ 99,90 cada, são os novos programas que a Cibertron está lançando. Aproveite e conheça também os já tradicionais jogos e aplicativos Cibertron para o TK 2000 e TK 85.

...E NÃO APENAS VIDEOGAME.

S O F T W A R E

É C I B E R T R O N

CIBERTRON SOFTWARE

Adquira o Software Cibertron nos revendedores abaixo. Caso você não encontre o programa desejado, peça-o à Cibertron Eletrônica Ltda. Caixa Postal 17005 – CEP 02398 – SP, anexando ao pedido um cheque nominal (remessas em três dias úteis). Solicite, gratuitamente, a nossa lista de programas.

Revendedores autorizados:

amarosom
R Pamplona 982 Tel 288 1900
Al Jaú 661 (esq Pamplona)
Tel 284 9644

FOTOPTICA

MAGNODATA informática ltda.
Tel. (011)255 7653

SANTO ANDRÉ
SHOP audio & vídeo
R Gal Glicério 430 Tel 444 6055

* Disponíveis a partir de 15/05/86.

III inforuso

ENCONTRO MINEIRO DE INFORMÁTICA
FEIRA MINEIRA DE INFORMÁTICA

(Evento oficializado pelo Ministério da Indústria e Comércio através do CDC)

O grande mercado mineiro, com milhares de compradores em potencial estará nos três pisos do MINASCENTRO querendo conhecer o que você tem para mostrar e aderir à qualidade de seus produtos e serviços. Não fique de fora. Tudo mostrado ao vivo às pessoas certas. Reserve o seu stand e venha pronto para vender

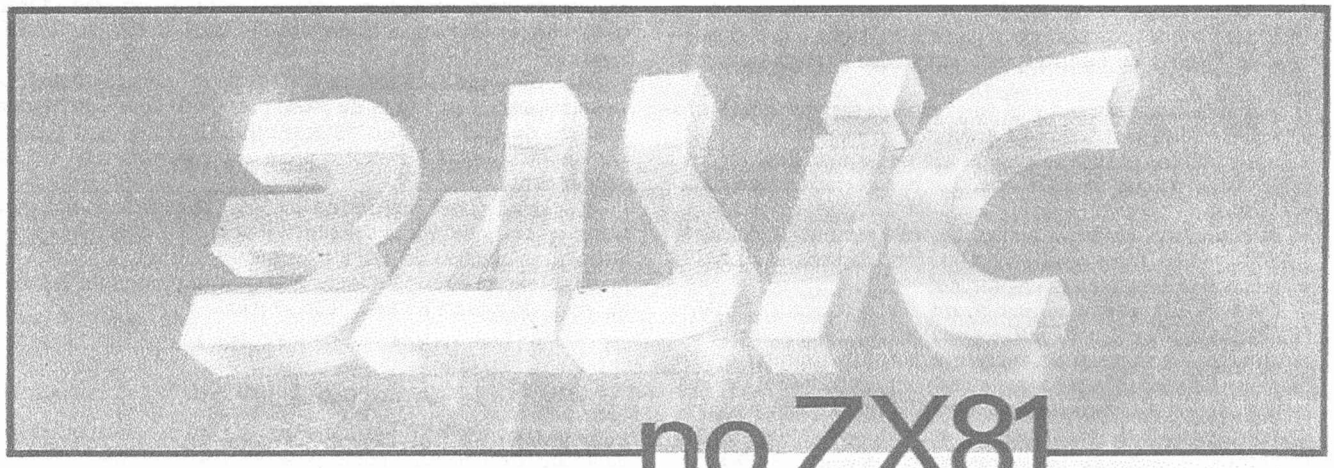
DE 23 A 27 DE JULHO DE 1986 - MINASCENTRO - BELO HORIZONTE

Patrocínio:

SUCESU
sociedade dos usuários de computadores e equipamentos subsidiários · minas gerais

Realização:

NILSO FARIAS
Rua Antônio de Albuquerque, 757 - 3º andar
30000 - Belo Horizonte - MG
Tel.: (031) 225-1944 - Telex (031) 6074 NFCE BR



no ZX81 e ZX Spectrum

Se você é programador de BASIC e deseja saber mais sobre variáveis do sistema, aqui está sua oportunidade.

Claudio de Freitas B. Bittencourt

A linguagem BASIC da linha Sinclair é, ao contrário do que supõem alguns, muito poderosa, e nada fica a dever à dos micros mais sofisticados surgidos na mesma época. É claro que equipamentos lançados recentemente possuem mais recursos, porém nada que, digamos, humilhe os nossos velhos ZX. E não são poucos os pontos em BASIC Sinclair que ainda hoje superam os seus concorrentes, haja vista a simplicidade inigualável no uso da alta resolução e na definição de caracteres. Falo, pois, do Spectrum, é claro.

Assim sendo, foi com satisfação que me prontifiquei a atender o pedido de **MICRO SISTEMAS** para escrever sobre a matéria. Mas depois, diante do papel em branco, embatuquei. Por onde começar? São tantas as coisas. . . E eu não queria de forma alguma redigir um artigo que repetisse simplesmente o manual ou outras publicações, entre tantas, de excelente qualidade que existem sobre esses computadores.

O tempo passando e nenhuma idéia *pintava*. Da hesitação passei ao desânimo e deste, rapidamente, ao pânico. Até que, por fim, emergi das trevas com uma convicção: é impossível, pelo menos para mim, num texto de revista abranger tudo. Porém alguma coisa de útil sempre se pode passar aos leitores.

Resolvi então limitar a matéria dando-lhe uma apresentação didática e explorando ao máximo as dicas e informações, de uso prático, para proveito dos programadores da linguagem BASIC. É claro que eu não inventei nem descobri nada sozinho, está tudo nos livros e, a propósito, apresento uma pequena bibliografia, no final, para aqueles que desejarem ir um pouco mais fundo no tema.

Limitaremos também o assunto ao Spectrum e similares, visto que os seus irmãos mais velhos, o ZX80 e o 81, já foram fartamente abordados em **MS**.

Comecemos por analisar algumas "variáveis do sistema", que ficam naquela área da memória RAM, do endereço 23552 ao 23733. É aí que o sistema (em outras palavras, os programas em linguagem de máquina da ROM) guarda informações de natureza transitória. Esses dados podem ser lidos pelos programas do usuário, e podem inclusive ser alterados, interferindo diretamente no comportamento do micro.

Mudar valores contidos nas variáveis do sistema exige, contudo, um seguro conhecimento do que se está fazendo, sob risco de se provocar efeitos inesperados, dentre eles o temível *crash*.

As variáveis do sistema são identificadas por nomes que só têm valor como referência para nós usuários, não sendo assim reconhecidos pela máquina. As informações armazenadas, contudo, são das mais variadas naturezas e tamanhos. Dessa forma, um único bit é suficiente para o sistema saber se a condição over está ligada ou desligada; atributos são necessariamente armazenados em um byte; e endereços em dois.

Quando a informação for um número de dois bytes (um endereço por exemplo), ela é armazenada de tal maneira que a sua leitura pelo BASIC deve ser feita da seguinte forma:

```
LET C = PEEK E + 256 * PEEK (E+1)
```

onde C é o conteúdo da variável que se quer ler e E, o seu endereço. Para colocar um número C numa variável dessas, devemos fazer:

```
POKE E, C-256 * INT (C/256)
POKE E+1, INT (C/256)
```

As variáveis do sistema que tratam dos atributos não serão abordadas aqui, visto que já o foram em **MS** nº 59 ("Renumerador GGMI"), artigo de nossa modéstia lavra. A despeito da pobreza de estilo do autor, garanto que não se arrependerá o leitor por consultar aquele exemplar.

Como os manuais do Spectrum e do TK90X diferem nos nomes das variáveis, apresentaremos as duas versões, colocando entre parêntesis os desse último. O número ao lado dos nomes é o endereço da variável.

VARIÁVEIS DE LEITURA DO TECLADO

Os primeiros endereços na área das variáveis do sistema são reservados para armazenar informações sobre o teclado. Os oito bytes iniciais (de 23552 a 23559) são uma espécie de arquivo, utilizado pelo sistema para executar a auto-repetição de teclas conforme a conhecemos, ou seja, com um retardo prolongado após a primeira execução, seguido de retardos curtos entre as repetições.

A coisa acontece da seguinte maneira: periodicamente a UCP pára o que está fazendo e passa a executar o que chamamos de "rotina de interrupção". Ela realiza as seguintes tarefas:

- incrementa um contador de três bytes na variável FRAMES;
- verifica se alguma tecla está premida;
- caso positivo, verifica se está ou não na hora de sinalizar, para as outras rotinas da ROM, que houve a entrada de um dado via teclado;
- caso positivo, coloca o código da tecla premida na variável LAST K e seta (toma igual a 1) o bit cinco da variável FLAGS;
- retorna ao programa interrompido.

Duas outras variáveis participam do processo: REPDEL (RPTDLAY) 23561 e REPPER (RPTCCLE) 23562. A primeira armazena o número de vezes que a rotina de interrupção tem que ser chamada para que a auto-repetição inicial de uma tecla premida seja autorizada. Seu valor original é 35. A segunda tem função semelhante, só que aplicada às auto-repetições posteriores à primeira. Seu valor original é cinco. Ambas podem ser mudadas pelo usuário, alterando os tempos de espera do teclado.

A instrução INKEY$ realiza uma leitura independente do teclado, por isso é rápida e não temporizada.

CHARS(PTBLCHR) 23606/7

O conteúdo dessa variável aponta para o conjunto de caracteres do Spectrum e só deve ser alterado se um novo conjunto for definido pelo usuário em alguma posição da RAM. Como cada um dos 96 caracteres é definido por oito bytes, são necessários 768 bytes para abrigar um conjunto completo de caracteres. O endereço a ser colocado em CHARS deve ser o do início do conjunto menos 256, para ser corretamente interpretado pelas rotinas da ROM.

PIP (KCLICK) 23609

O conteúdo de PIP, que originalmente é 0, indica a duração do *bip* emitido pelo teclado, e pode ser alterado à vontade, aumentando a duração do bip.

ERR SP 9 (P ERR) 23613/4

Essa variável, que é uma das mais perigosas de ser mexida, representa uma verdadeira obsessão para os programadores, tudo porque inexiste a instrução ON ERR GOTO no BASIC Sinclair.

ERR SP aponta um endereço stack da máquina, que por sua vez armazena o endereço da ROM para onde deve desviar-se o controle da UCP em caso de erro ou BREAK, em um programa BASIC. Invariavelmente o programa pára com uma mensagem de erro.

É possível fazer uma rotina em linguagem de máquina para tratar os erros de modo diferente, e colocar o seu endereço de entrada na posição do stack apontada por ERR SP. Mas, se a nossa intenção é simplesmente bloquear um programa BASIC, desativando a tecla BREAK, basta fazer:

```
LET r = PEEK 23613 + 256 * PEEK 23614 - 2
POKE 23613,r - 256 * INT (r/256)
POKE 23614, INT (r/256)
```

Com isso, ERR SP passa a apontar para um endereço (ainda no stack da máquina) duas posições abaixo do que apontava antes. Nesse local encontra-se o endereço da rotina da ROM que dá continuidade às instruções de um programa BASIC.

É conveniente, num programa desses, evitar as instruções GOSUB e RETURN, que alteram o conteúdo de ERR SP, podendo na melhor das hipóteses desbloquear o seu programa, e na pior, mandá-lo para o espaço.

MODE (CURSOR) 23617

Essa variável indica o cursor em uso. Os valores assumidos são:

Conteúdo	Cursor
0	L, C, K
1	E
2	G

Um **POKE** pode ser feito para mudar o cursor em um **INPUT**. Veja as diferentes alternativas rodando o programinha abaixo:

```
10 FOR f = 0 TO 255
20 POKE 23617,f
30 INPUT STR$ f;" ";a$
40 NEXT f
```

NEWPPC (LNJMP) 23618/9

Essa variável guarda o número de linhas de programa que contém a próxima instrução a ser executada. Ela pode ser útil ao programador juntamente com a variável que se segue.

NSPPC (INSTRNR) 23620

O conteúdo de NSPPC indica o número de instrução, dentro de uma linha de programa, que deve ser executada.

Poqueando-se primeiro NEWPPC e depois NSPPC, pode-se provocar um desvio, diferente de um GOTO, para o interior de uma linha multifunção. Repare no exemplo abaixo que o fluxo de programa é desviado da linha 30 diretamente para a segunda instrução (PRINT "ok") da linha 200:

```
                                    COMENTÁRIOS:
10 POKE 23618,20          ...       20 = byte menos significativo de 200
20 POKE 23619,0           ...       0 = byte mais significativo de 200
30 POKE 23620,2: STOP ...           2 refere-se à 2ª instrução da linha
40 STOP
200 STOP : PRINT "ok"
```

NXTLIN (NEXEXC) 23637/8

Essa variável armazena o endereço inicial da próxima linha de programa a ser executada. Ela é útil se, por algum motivo, quisermos saber a posição da memória onde se encontra armazenada uma linha de programa. No exemplo abaixo, basta executar o comando direto **GOTO 115** para conhecer o endereço inicial da linha 120:

```
115 PRINT PEEK 23637 + 256 * PEEK 23638 : STOP
120 ...
```

É evidente que se, depois disso, eu apagar, criar ou mexer em qualquer linha anterior a 120, o endereço desta será modificado e a informação obtida perderá o valor.

FLAGS2 (SFLAG2) 23658

Em FLAGS2 cada bit tem função independente. O bit 3, que é o de nosso interesse, indica, se estiver setado (igual a 1), que a condição CAPS LOCK (maiúsculas) está ligada.

Se o amigo leitor é um garimpador de programas publicados nessa revista ou em outras, já deve ter notado que após uma instrução de INPUT costumam aparecer um ou dois **POKEs**. Eles são feitos em FLAGS2 e em MODE (já abordada) para restabelecer as condições de leitura do teclado, eventualmente alteradas pelo usuário.

```
POKE 23617,0          .... restabelece o cursor L
POKE 23658,0          .... desliga as maiúsculas
POKE 23658,8          .... liga as maiúsculas
```

FRAMES (TVCOUNT) 23672/3/4

Essa variável, a única de três bytes, é um contador de tempo, que parte de zero quando o micro é ligado e recebe o incremento de uma unidade a cada vez que a rotina de interrupção é executada. Isso ocorre, no Brasil, 60 vezes por segundo, que é a freqüência da nossa rede elétrica, a despeito do que possa dizer o manual do seu microcomputador.

Podemos usar FRAMES para dispor de um razoável cronômetro em nossos programas BASIC. Primeiro, trate de zerar o contador fazendo:

```
POKE 23672,0 : POKE 23673,0 : POKE 23674,0
```

nessa ordem.

Para ler o conteúdo de FRAMES, faça:

```
LET F = PEEK 23672 + 256 * PEEK  23673 + 65536 * PEEK 23674
```

O tempo total em segundos será dado por:

```
LET S = INT (F/60)
```


ATENÇÃO USUÁRIOS

TK90X • ZX Spectrum • TK95

Promoção

CBI CENTRO BRASILEIRO DE INFORMÁTICA

Micro Sistemas

NOVOS PRODUTOS PARA O SEU MICRO

PREÇO PROMOCIONAL DE LANÇAMENTO

INTERFACE PARA JOYSTICK KEMPSTON CBI-ZX

- Funciona com todos os jogos que possuem opção para Joystick.
- Aceita qualquer tipo de Joystick compatível com Atari, MSX.
- Facílima instalação.
- Com o interface para joystick CBI-ZX você poderá, inclusive operar com dois joysticks ao mesmo tempo (interface CBI-ZX e a interface já incorporada ao TK 90X), com programas que permitem dois jogadores.

PROGRAMADOR DE EPROM'S CBI-90

O programador de EPROM CBI 90 é uma ferramenta de baixo custo e alta performance para profissionais de eletrônica e informática, que possibilita a gravação nos tipos mais populares de EPROMs, inclusive de 128K e 256K Bits.

O CBI 90 permite:

- Fazer Back-up de todas as EPROMs de seus equipamentos a baixo custo (salvar em fita cassete).
- Copiar ou modificar programas, jogos ou dados gravados em EPROM.
- Desenvolver software de base ou aplicativo gravando diretamente em EPROM.

CBI – Rua Teófilo Otoni, 123 - gr. 202/302 - Tel.: 233-1123 - CEP 20090 - Rio de Janeiro - RJ.

E se você preferir minutos e segundos, faça:

```
LET M = INT (F/3600)
LET S = INT (60 * ((F/3600) - M))
```

Alerta, leitor exigente, pois o contador de FRAMES é zerado a cada 77 horas e 40 minutos, que é o tempo para atingir o valor 2↑24-1, maior número de três bytes (24 bits). E, principalmente, cuidado com algumas instruções que desabilitam temporariamente a rotina de interrupção, suspendendo a contagem. São elas: SOUND, SAVE, LOAD, MERGE e VERIFY.

UDG (UDGRAPH) 23675/6

O conteúdo dessa variável aponta para o início do conjunto de 21 caracteres gráficos definidos pelo usuário. Pode-se trabalhar com vários desses conjuntos ao mesmo tempo, desde que o conteúdo da variável seja convenientemente alterado, sempre que necessário.

Parênteses: para definir conjuntos de UDGs fora da área normalmente a eles reservada (acima do endereço 65368), basta mudar o conteúdo dessa variável, para apontar outras posições, por nós escolhidas, e usar as instruções de praxe: UDG 0, UDG 1 ou UDG 2. Aliás, cabe aqui um parêntese dentro do parêntese: nós, felizes usuários brasileiros, podemos usar o UDG 2 (inexistente no Spectrum mas existente no TK90X) para redefinir um completo conjunto de caracteres (aqueles 96 apontados pela variável CHARS, lembra-se?) em substituição ao conjunto normal definido na ROM. Essa é uma tarefa que exige planejamento e paciência.

COORDS (LSTPLOT e COODS) 23677/8

Essa variável armazena as coordenadas cartesianas do último pixel plotado, sendo X em 23677 e Y, em 23678. Pode-se economizar tempo poqueando essa variável antes de um DRAW, em vez de usar a instrução PLOT.

SCR CT (SCRING) 23692

Essa variável é decrementada toda vez que um Scroll automático se realiza. Quando o seu valor cai a um, aparece no vídeo a pergunta "Scroll?" que respondida afirmativamente provoca a continuação do Scroll e atualiza o conteúdo da variável para 22.

Se quisermos um Scroll ininterrupto, devemos impedir que o conteúdo de SCR CT atinja a unidade, efetuando POKEs de qualquer valor nessa variável (quanto maior o valor, mais eficaz).

VARIÁVEIS DELIMITADORAS

Existe um conjunto de variáveis que serve para delimitar as diversas áreas em que se subdivide a memória RAM. Nessas variáveis não podemos efetuar **POKEs**, mas podemos ler o conteúdo para saber, por exemplo, o tamanho de um programa BASIC ou quais os limites da área livre. Essas variáveis estão listadas abaixo, segundo a ordem crescente dos endereços das áreas:

```
CHANS (CHCADD) 23631/2    .... início da área de dados dos canais
PROG (PROGBAS) 23635/6    .... início  do programa BASIC
VARS (VARADD) 23627/8     .... início da área de variáveis de um
                               programa BASIC
ELINE (INADD) 23641/2     .... início da área de edição
WORKSP (WORKPT) 23649/50  .... início do espaço de trabalho
STKBOT (STKEND) 23651/2   .... início da pilha do calculador
STKEND (ADSPFREE)23653/4  .... fim da pilha do calculador, início
                               do espaço livre
RAMTOP (RAMTOP) 23730/1   .... fim da área não protegida
P RAMT (MEMAVLB) 23732/3  .... endereço do último byte da RAM
```

Eis aí tudo que pudemos colocar, de interesse para os programadores da linguagem BASIC, sobre as variáveis do sistema. Mas, tenha o distinto leitor em mente que isso nem de longe se compara com o que se pode obter explorando-as em linguagem de máquina. A diferença está em que, com esta, assume-se de fato o controle da UCP.

Para encerrar, GGMI manda uma bibliografia, pequena porém de peso, para aqueles que desejarem entender um pouco mais do seu equipamento.

Bibliografia

– *Manual de operação* (ruim com ele, pior sem ele);
– **Simson, R. J. e Terrel, T. J.**, *Manual do ZX Spectrum*. Editorial Presença, Lisboa;
– **James, M.** *Guia Avançado para o Spectrum*. Editorial Verbo, Lisboa;
– **Kramer, S.**, *The Spectrum Operating System*. Micro Press, London;
– **Pellier, P.**, *Lenguaje Maquina del Zx Spectrum*. Editorial Gustavo Gili, Barcelona;
– **Webb, D.**, *Advanced Spectrum Machine Language*. Melbourne House;
– **Logan, I. e O'Hara. F.**, *The Complete Spectrum Rom Disassembly*, Melbourne House.

Cláudio de Freitas B. Bittencourt é formado em engenharia metalúrgica e Professor de Pós-Graduação em engenharia nuclear do IME, Instituto Militar de Engenharia, no Rio de Janeiro.

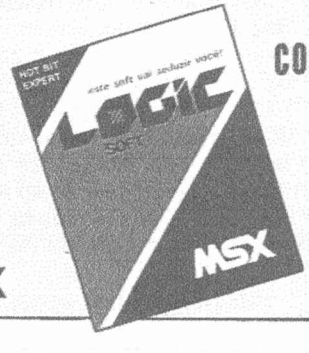
SÉRIE AZUL
MAIS QUE UMA FERRAMENTA
UMA AULA!

A SÉRIE AZUL ensina o novato a trabalhar com a máquina e aprimora o conhecimento dos experts. É a única série de programas abertos com todas as rotinas documentadas no manual.
Use e abuse aprendendo com a Série azul.
TRANS SET – Programa editor de tabela de caracteres. Crie novos caracteres no seu micro.
SPRITE BUG – Anime a tela do seu micro, movimentando suas figuras.
DRAW IT – Super editor do comando DRAW. Elimine o trabalho de codificação e dedique-se à criação.

IMAGE – Editor de desenhos e figuras com recursos de SCROLL, ZOOM, REDUÇÃO, CORTE E COLAGEM (CUT AND PASTE) e outros.
MICRO SIMULADOR – Simula um micro-processador com recursos de programação, edição e visualização gráfica de todas as etapas executadas.
domine seu micro.
DICAS E MACETES I – Utilize as dicas do Denis em seu MSX! Como movtar um jogo, SCROLLS em alta resolução, novas teclas de função, Screen 1 dentro do Screen 2, além de muitas outras novidades.

LOGIC
Temos os melhores títulos para TK-90X e TK-95

MSX (fita) Cz$ 420,00 cada



COPY II PLUS

Preço Cz$ 1.920,00

COPY II PLUS
Faça Back-up de suas fitas-mestres.
Duplique qualquer programa gravado em fita ou cartucho.
Cópias perfeitas para a sua segurança.
O Copy II Plus examina e modifica os parâmetros do HEADER. Acessa vários blocos de uma só vez na memória e coloca Header em qualquer programa. Grava em 3 (três) velocidades diferentes. O Copy II Plus reproduz programas em BASIC ou Linguagem de máquina. É capaz de alterar nomes em fita ou cartucho e adapta-se automaticamente à qualquer máquina MSX do mercado.

Soft Nacional/Autor Denis Ferreira

Solicite pelo correio enviando cheque nominal ou vale postal à
CEDUSOFT EDUCAÇÃO INFORMÁTICA E SISTEMAS LTDA.
CAIXA POSTAL 1210 - CEP 20001 - RJ

Em Abril:
Versão Disco com 50% de desconto aos usuários do COPY II PLUS em Fita.

CEDUSOFT
EDUCAÇÃO INFORMÁTICA & SISTEMAS LTDA.
CAIXA POSTAL 1210 – CEP 20001 – RJ
TEL.: (021) 222-9991

tropical

A tradução pode não ser a melhor alternativa, mas algumas regras podem ajudá-lo bastante na conversão de seus programas entre os diversos...

Dialetos BASIC

Maria Sylvia Marques Abaurre

A grande maioria dos microcomputadores pessoais utiliza a linguagem BASIC. O nosso principal problema é a existência de diferentes *dialetos* variantes do BASIC original, sendo um para cada *família* de microcomputadores (TRS-80, Apple e Sinclair, as mais conhecidas no Brasil). Desta forma, um programa escrito seguindo o *dialeto* Apple fica praticamente inutilizável, a menos que sejam feitas algumas modificações, num computador TRS-80 e vice-versa.

A *tradução* consistiria na adaptação de um programa escrito em um dialeto para outro diferente, desde que respeitadas certas limitações necessárias. Esta *tradução* é uma tarefa um tanto árdua e, por vezes, o programa traduzido é quase que inteiramente diverso do original. Por conseguinte, é um bom passo para quem se inicia na programação de computadores porque dá uma idéia globalizada de como funcionam os comandos e instruções da linguagem BASIC nas diferentes máquinas.

As primeiras tentativas de tradução funcionam, em sua maioria, na base de tentativas/erros e tentativas/acertos como um grande quebra-cabeças, que nem por isso é impossível de ser montado.

Muitos programadores *puristas* argumentam que melhor seria não traduzirmos programas, mas sim criarmos novos. Em parte, concordo com esta alternativa porque não creio que a tradução seja um fim em si mesma, mas um começo para um bom programador. Assim, ao nos entregarmos às traduções, começamos a lidar com o programa como se fosse algo dinâmico, modificando sub-rotinas, alterando partes, inserindo funções e retirando outras consideradas supérfluas. A partir daí, a confecção de programas próprios é quase que imediata.

PRÉ-REQUISITOS PARA A TRADUÇÃO

Quantidade de memória disponível – Este é o principal item que se deve ter em mente antes de começar qualquer tipo de tradução. Seu computador deve dispor da quantidade de memória que a execução do programa exige. A tradução de um programa que consome 32 Kb para utilização num Sinclair cuja capacidade não ultrapassa 16 Kb é completamente inútil.

Um conhecimento geral da linguagem BASIC – Adquirido principalmente através de cursos, da leitura de livros e revistas especializadas e da análise de diferentes programas em dialetos diversos.

Bom senso – Suficiente para não adaptar, por exemplo, um programa musical como os existentes para Apple e Commodore 64 para um Sinclair, que conta com reduzidos recursos sonoros.

Uma tabela de conversão – Para reconhecer as diferentes funções dos comandos e seus equivalentes em diferentes sintaxes. Uma tabela com a reunião dos comandos e instruções mais conhecidas e seus equivalentes é apresentada neste artigo.

BARREIRAS À TRADUÇÃO

Programas em linguagem de máquina – Para se fazer uma tradução neste nível, teríamos que estar de posse dos manuais e livros técnicos de cada um dos micros e, ainda assim, sermos programadores bastante experientes, considerando a diferença existente entre as máquinas.

Peeks e Pokes – Tais traduções só serão possíveis se você tiver os respectivos manuais dos diferentes micros, a não ser que suas funções venham explícitas nos respectivos artigos publicados, que elas sejam dispensáveis ao seu programa ou que você conheça os equivalentes para seu micro.

Exemplo: POKE 53281,1:POKE 53280,0 – mudança de cor das bordas e do fundo da tela do Commodore 64. Ora, caso seu computador não seja colorido, você pode simplesmente eliminar esta linha sem qualquer prejuízo à execução do programa. Se seu micro for colorido, você poderá usar seu próprio código de cores.

FORMATAÇÃO DA TELA

O número de linhas e colunas nos diversos computadores é diferente e você terá que fazer um rearranjo na posição que as palavras, números e/ou desenhos ocuparão na tela, de acordo com o programa. Para isto, é muito útil o uso de tabelas como as que você vai receber de brinde neste artigo.

Vamos agora examinar o tipo de *sintaxe* usada nas diversas famílas, quando »

Tabelas de formatação bidimensional

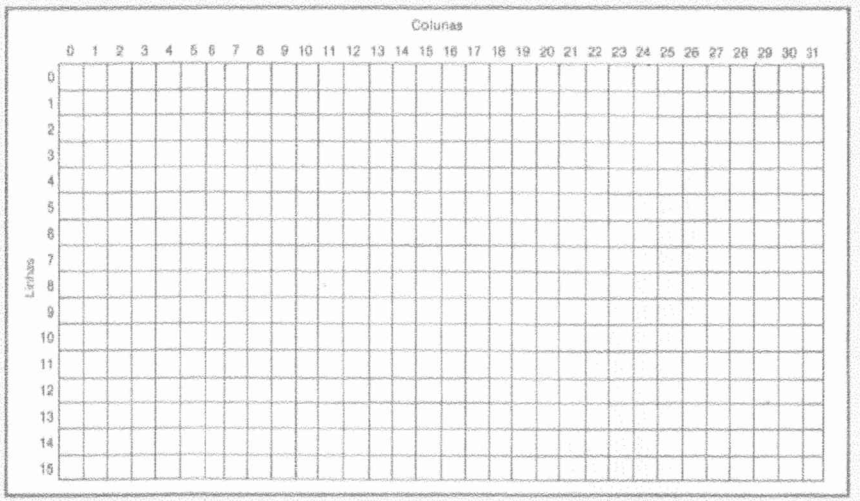


Tabela 1 – Para TRS-80

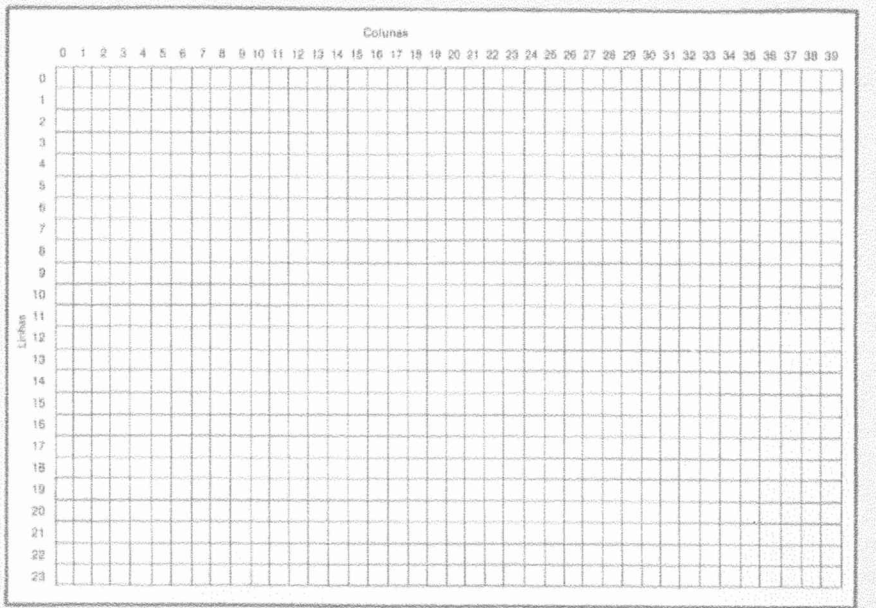


Tabela 2 – Para Apple

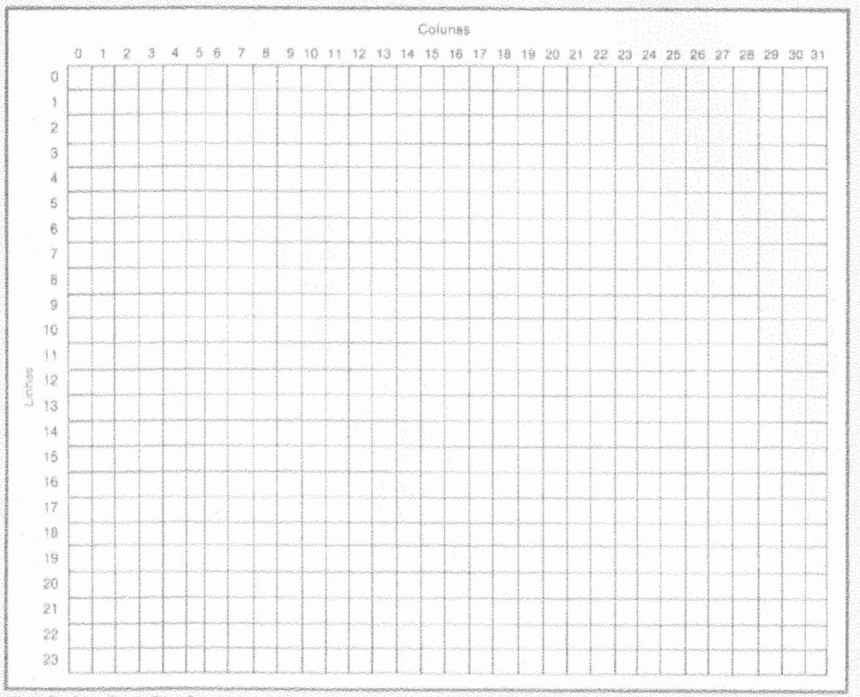


Tabela 3 – Para Sinclar

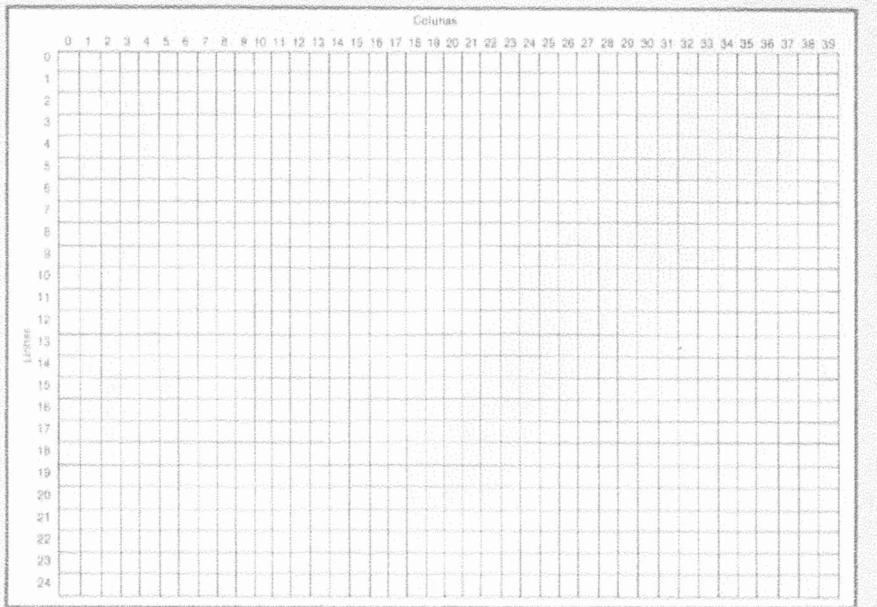


Tabela 4 – Para Commodore 64 e Vic-20

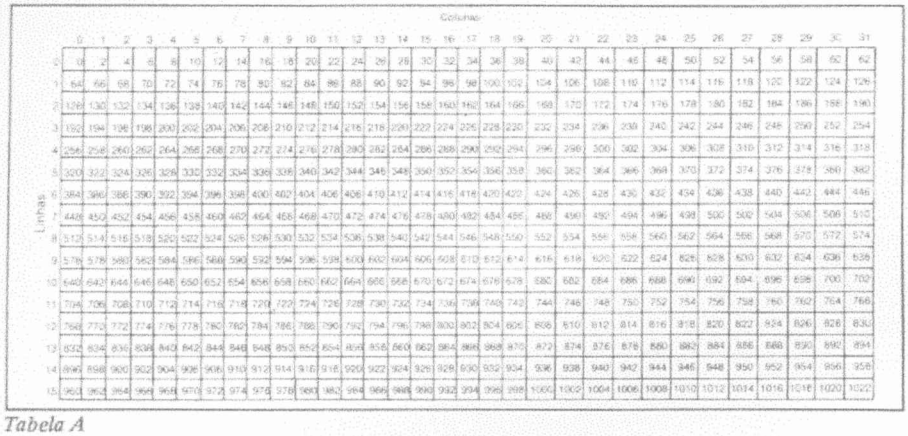


Tabela A

se deseja *imprimir* algo em um determinado local da tela.

A família Sinclair usa principalmente a instrução **print at** e segue a fórmula geral **PRINT AT L, C;**,onde **L** significa linha e **C** coluna (esta convenção será usada até o fim deste artigo). Veja a tabela 3.

Para traduzirmos este comando para um micro TRS-80, devemos procurar o número correspondente ao ponto de interseção entre a linha e a coluna *na tabela A e aplicá-lo na fórmula geral PRINT* @ ——, logo após o símbolo @

Exemplo: **Sinclair** **TRS-80**
PRINT AT 2,7; PRINT@142,

onde 142 é o número encontrado na interseção entre a linha 2 e a coluna 7, quando consultamos a tabela A.

Quando quisermos fazer a mesma tradução do comando acima para um Apple, devemos usar a fórmula geral **H-TAB C+1:V-TAB L+1**. Assim, o **PRINT AT 2,7;** e o **PRINT @142** do TRS-80 se tornariam **H-TAB 8:V-TAB 3:PRINT**, ao usarmos um Apple.

Ao trabalharmos com um Commodore 64 ou com um VIC-20, poderíamos usar a seguinte fórmula geral: **PRINT TAB(C)"[Q] – tantas vezes quantas forem o número de linhas...".** No entanto, esta fórmula só poderá ser usada quando o número de linhas for expresso em algarismos absolutos. Neste caso, [Q] representa a tecla CRSR ↓ . Então, o **PRINT AT 2,7;** (Sinclair) seria convertido em **PRINT TAB(7)"[Q][Q]..",** onde a tecla CRSR ↓ seria pressionada duas vezes no interior das aspas antes da mensagem a ser *printada*, pois o número de linhas é igual a 2.

Quando a tabulação é feita usando-se expressões numéricas, a tradução poderá ser feita através de um pequeno **loop** apresentado em seguida e cuja sintaxe se adapta a quase todas as famílias de microcomputadores.

FOR I = 1 TO L:PRINT:NEXT:PRINT TAB(C)

Neste caso, um **PRINT AT J+K,X-Z;** de um Sinclair se tornaria **FOR I = 1 TO J+K:PRINT:NEXT:PRINT TAB(X-Z)** para as outras famílias de microcomputadores (nos microcomputadores de lógica Sinclair devemos utilizar apenas uma instrução por linha).

Encerrando estas observações sobre tabulação e formatação de telas, convém lembrar que praticamente todos os computadores aceitam a instrução **PRINT TAB(C)**, para locarmos somente a coluna, e a instrução **PRINT**, para saltarmos uma linha em branco.

Para facilitar a tradução como um todo, sugiro que você tente fazer um fluxograma do programa a ser traduzido, para que possa seguir a linha de raciocínio do programador. A repetição deste artifício vai auxiliá-lo bastante no desenvolvimento de traduções ou em suas futuras programações.

Este artigo foi escrito a partir de pequenas observações e experiências acumuladas em cinco meses lidando com um mesmo problema: o de tentar adequar ao meu Commodore 64, programas para computadores Sinclair, Apple e TRS-80. Espero que ele possa ser útil a um grande número de pessoas, já que grande parte dos programas publicados em revistas brasileiras são escritos nesses três dialetos.

Maria Sylvia Marques Abaurre é bióloga, possui cursos de linguagem BASIC e Cobol e atualmente utiliza um Commodore 64.

"MIKROS"

- Microcomputadores Pessoais e Profissionais, Software, Suprimentos e Cursos.
- Financiamento em até 18 meses sem entrada e os preços mais baixos do mercado.
- Atendimento perfeito, profissionais treinados e habilitados para dar a você a certeza de um bom Investimento.

EQUIPAMENTOS
Micros das linhas: TRS-80 • Apple • IBM • Sinclair • TRS-80 Color
Impressoras • Vídeos • Interfaces • Etc.

SUPRIMENTOS
Formulário Contínuo • Disquetes • Fitas • Mesas • Etiquetas • Etc.

SOFTWARE
Nacionais e Importados mais de 2.000 programas e jogos de todas as linhas.

CURSOS
Basic • Basic Avançado e DOS.

Av. Ataufo de Paiva, 566
sobreloja 211 e 202
Rio de Janeiro - R.J.
Tels.: (021) 239-2798 e 511-0599

274-8845
Agora em PABX

Fita Impressora Nacional ou Importada
Ligue 274-8845

Formulário Continuo 1, 2 ou 3 vias
Ligue 274-8845

Aquele Arquivo para diskettes 5.1/4" ou 8"
Ligue 274-8845

Pastas para Listagens 80 e 132 colunas
Ligue 274-8845

Etiquetas Adesivas em Formulário Continuo
Ligue 274-8845

Diskettes 5.1/4" ou 8" (5 anos de garantia)
Ligue 274-8845

Reebobinagem em Nylon e Polietileno
Ligue 274-8845

Nós temos tudo isso, e muito mais...
* *Pronta Entrega*
* *Qualquer Quantidade*
* *Garantia de Qualidade*

Suprimento
MATERIAIS PARA COMPUTADORES

R. VISCONDE DE PIRAJA, 550/202 — 274-8845 — IPANEMA — RIO

Tabela com comandos, instruções e equivalentes

COMANDO	APPLE	TRS-80	SINCLAIR	64/20	SIGNIFICADO
AUTO	ñ tem	AUTO AUTO X AUTO X,Y	ñ tem	ñ tem	Numeração automática a partir da linha de nº X, de Y em Y.
BREAK	CTRL-C ou RST	BREAK(tecla)	BREAK	RUN/STOP (tecla)	Paralisa qualquer tipo de execução.
CALL	CALL(N) N=end da subrotina	USR(N)	USR(N)	USR(N)	Leva o programa para a sub rotina especificada com linguagem de máquina.
CLEAR	CLEAR	CLEAR n n=bytes disponíveis para strings	CLEAR	CLR	Faz todas as arrays numéricas e variáveis=0 e as arrays strings=" ".
CLS	HOME	CLS	CLS	SHIFT+CLR/HOME	Coloca o cursor no extremo superior esquerdo e limpa tela.
CLOAD	LOAD	CLOAD "nome"	LOAD "nome"	LOAD "nome"	Carrega um programa na memória do cassete.
CLOAD?	ñ tem	CLOAD?	ñ tem	VERIFY"nome"	Compara o programa do cassete com o da RAM.
CRL/HOME	ñ tem	ñ tem	ñ tem	CRL/HOME	Coloca o cursor no extremo superior da tela.
COLOR	COLOR=n n=0 a 15	ñ tem	ñ tem	CTRL+nº da tecla da cor	Fornece cor para os próximos símbolos.
COPY	ñ tem	ñ tem	COPY	ñ tem	Copia a tela numa impressora.
CONT	CONT	CONT	CONT	CONT	Prossegue com a execução do programa paralizado por stop.
CSAVE	SAVE	CSAVE "nome"	SAVE "nome"	SAVE "nome"	Grava em cassete o programa da RAM.
DATA	DATA a,b,c	DATA a,b,c	ñ tem	DATA a,b,c	Define constantes a serem lidas por READ.
DEF FN	DEF FN A(x)=exp	DEF FN A(x)=exp	ñ tem	DEF FN A(x)=exp	Define uma função qualquer (exp=expressão).
DEL	DEL A,B	DELETE A,B	ñ tem	nº da linha +RETURN.LIST	Deleta as linhas de A e B.
DIM	DIM A(N) DIM AS[N]	DIM A(N) DIM AS(N)	DIM A(N) DIM AS(N)	DIM A(N) DIM AS(N)	Dimensiona uma matriz.
DRAW	DRAW A AT X,Y	ñ tem	ñ tem	ñ tem	Desenha no ponto X,Y a forma especificada por A.
DEFINT	ñ tem	DEFINT A-C,E,Y	ñ tem	ñ tem	Admite como inteiras todas as variáveis cujas iniciais estejam entre A e C, E ou Y.
DEFSNG	ñ tem	DEFSNG A-C,E,Y	ñ tem	ñ tem	As variáveis cujas iniciais são E ou Y, ou estão entre A e C, são definidas como de precisão simples.
DEFDBL	ñ tem	DEFDBL A-C,E,Y	ñ tem	ñ tem	O mesmo para variáveis de precisão dupla.
DEFSTR	ñ tem	DEFSTR A-C,E,Y	ñ tem	ñ tem	O mesmo para as variáveis strings.
EDIT	EDIT A E A	ñ tem	ñ tem	ñ tem	Para se inserir, deletar ou substituir caracteres na linha A do programa.
END	END	END	STOP	END	Fim de programa.
ELSE	ñ tem	ELSE (instrução ou nº de linha)	ñ tem	ñ tem	Alternativa negativa para desvio no emprego do comando IF.
ERL	ñ tem	ERL	ñ tem	ñ tem	Aponta o nº da linha onde há erro.
(ERR/2)+1	PEEK (222)	(ERR/2)+1	ñ tem	ñ tem	Mostra o código do erro cometido.
ERROR	ñ tem	ERROR (N)	ñ tem	ñ tem	Simula um erro de acordo com o código N.
FAST	SPEED (ver speed)	ñ tem	FAST	ñ tem	Executa mais rapidamente o programa.
FLASH	FLASH	ñ tem	ñ tem	ñ tem	Mostra os próximos caracteres da tela como "piscantes".
FOR I=... TO... step X: NEXT I	Como	o	comando	Inicial	Executa o loop entre os limites com acréscimo de X.
FP	FP	ñ tem	ñ tem	ñ tem	Retorna no interpretador BASIC.
GET	GET AS	GET AS	ñ tem	GET AS	Recebe um comando do tecla do.
GOSUB	GOSUB N	GOSUB N	GOSUB N	GOSUB N	Desvia o programa para a subrotina da linha N.
GOTO	GOTO N	GOTO N	GOTO N	GOTO N	Desvia o programa para a linha N.
GET#	ñ tem	ñ tem	ñ tem	GET # nº do arq. (variável)	Recebe ou lê um comando de cada vez de um arquivo.
GR	GR	ñ tem	ñ tem	ñ tem	Liga a baixa resolução.
HCOLOR	HCOLOR=n	ñ tem	ñ tem	CTRL+tecla com a cor desejada	Cor para os símbolos de alta resolução. (n de 0 a 7).
HGR	HGR HGR 2	ñ tem	ñ tem	ñ tem	Liga a alta resolução (pág. 1). Idem (pag. 2).
HIMEM	HIMEM:X	ñ tem	ñ tem	ñ tem	Onde X será o mais alto endereço para os programas em BASIC.
HLIN	HLIN X,Y AT A	ñ tem	ñ tem	ñ tem	Traça uma linha da coluna X a Y sobre a linha A.
HPLOT	HPLOT X,Y HPLOT X1,Y1 HPLOT X2,Y2	ñ tem	ñ tem	ñ tem	Traça retas ou pontos em alta resolução.
HTAB	HTAB n	PRINT TAB(N)	PRINT TAB(N)	PRINT TAB(N)	Posiciona o cursor na coluna N.
IF.. THEN GOTO...	IF.. THEN GOTO...	IF.. THEN GOTO...	IF.. THEN GOTO...	IF.. THEN GOTO...	Executa ou não comandos, dependendo ou não da afirmativa proposta.
IN#	IN n#	ñ tem	ñ tem	ñ tem	Indica que os dados de entrada virão do periférico ligado a n.
INPUT	INPUT	INPUT	INPUT	INPUT	Lê dados provenientes de algum dispositivo de entrada.
INPUT #	ñ tem	ñ tem	ñ tem	INPUT #nº arq., variáveis	Recupera dados de um arquivo (em disco ou cassete).

COMANDO	APPLE	TRS-80	SINCLAIR	64/20	SIGNIFICADO
INVERSE	INVERSE	ñ tem	ñ tem	CTRL+tecla 9 (REV ON)	Apresenta caracteres em inverso na tecla.
INKEY$	X=PEEK (-16384)	INKEY$ A$=INKEY$	INKEY$ A$=INKEY$	GET A$:IF A$=" " THEN (nº desta linha)	Lê um caracter no teclado, sem parar o programa.
LET	LET A=B	LET A=B	LET A=B	LET A=B (opcional)	Atribui um valor a uma variável.
LIST	LIST	LIST	LIST	LIST	Lista o programa da RAM ou parte dele.
LPRINT	PR#1 PRINT	LPRINT A,B; XY	LPRINT A,B; XY	ñ tem	Envia dados para a impressora.
LOMEM	LOMEM:X	ñ tem	ñ tem	ñ tem	Onde X será o endereço mais baixo para programas BASIC.
NEW	NEW	NEW	NEW	NEW	Apaga o programa BASIC da RAM.
NEXT	NEXT I	NEXT I	NEXT I	NEXT I	Complementa o comando FOR.
NORMAL	NORMAL	ñ tem	ñ tem	REV OFF+CTRL . P/desligar o (REV ON)	Desliga FLASH e INVERSE. Retorna à tela normal.
NOTRACE	NOTRACE	ñ tem	ñ tem	ñ tem	Desligar TRACE (ver TRACE)
ONERR GOTO N GOSUB N	ONERR GOTO N GOSUB N	ON ERROR GOTO N GOSUB N	ñ tem	ñ tem	Desliga o processamento para a linha N se houver erro.
ON..GOTO/ GOSUB A,B	ON..GOTO/ GOSUB A,B	ON..GOTO/ GOSUB A,B	GOTO N*A GOSUB N*A	ON..GOTO/ GOSUB A,B	Dependendo de valor, desvia o processamento para A,B.
PAUSE	ñ tem	ñ tem	PAUSE n	ñ tem	Paraliza o programa em (n/60)segundos.
PLOT	PLOT X,Y	SET (X,Y)	PLOT X,Y	ñ tem	Desenha um ponto em baixa resolução nas coordenadas X e Y.
POINT	ñ tem	POINT (X,Y)	ñ tem	ñ tem	Confere se o ponto das coordenadas X,Y está "ligado".
POKE	POKE END,N	POKE END,N	POKE END,N	POKE END,N	Altera o byte do endereço fornecido para o valor decimal N.
POP	POP	ñ tem	ñ tem	ñ tem	Indica que o programa deve voltar no penúltimo GOSUB quando encontrar RETURN.
PR#	PR#A	ñ tem	ñ tem	ñ tem	Enviar os dados de saída ao periférico de código N.
PRINT	PRINT ou ?	PRINT	PRINT	PRINT ou ?	Imprime na tela.
PRINT AT	HTAB X-1:VTAB Y+1:PRINT	PRINT AT X,Y	PRINT AT X, Y,	PRINT (leia artigo)	Exibe os dados na coluna Y e na linha X.
PRINT#	ñ tem	PRINT#-1,X,Y	ñ tem	PRINT nº arq., lista de variáveis	Grava os dados em cassete.
PRINT USING	ñ tem	PRINT USING CS;D,E	ñ tem	ñ tem	Especifica o formato dos dados D e E através de uma string CS.
RAND	ñ tem	RANDOM N	RAND N	ñ tem	Gera números aleatórios a partir de N.
READ	READ AS ou A	READ AS ou A	ñ tem	READ AS ou A	Lê os dados do comando DATA.
RECALL	RECALL A(X1,X2,..)	ñ tem	ñ tem	ñ tem	Carrega uma tabela para o cassete.
REM	REM	REM ou apóstrofo	REM	REM	Indica que o resto da linha é para comentários do programador.
RESTORE	RESTORE	RESTORE	ñ tem	RESTORE	Recoloca a leitura no início da lista de dados de DATA.
RETURN	RETURN	RETURN	RETURN	RETURN	Volta à primeira linha posterior ao último GOSUB executado.
ROT	ROT=n	ñ tem	ñ tem	ñ tem	Rotação de uma tabela para exibição. (n entre 0 e 63)
RSET	ñ tem	RESET (X,Y)	UNPLOT (X, Y)	ñ tem	Apaga o ponto das coordenadas X,Y.
RUN	RUN	RUN	RUN	RUN	Executa o programa que está na RAM.
RUN/STOP+ RESTORE	NORMAL	ñ tem	ñ tem	ñ tem	Restitui à tela sua cor normal.
SCALE	SCALE=n (0<n≤225) O=maior ampliação	ñ tem	ñ tem	ñ tem	Indica o fator de multiplicação para o tamanho da tabela a ser exibida.
SCROLL	ñ tem	ñ tem	SCROLL	ñ tem	Movimenta a tela uma posição para a inserção de uma nova linha.
SHLOAD	SHLOAD	ñ tem	ñ tem	ñ tem	Carrega uma tabela no cassete.
SPEED	SPEED=n (0<n≤255)	ñ tem	ñ tem	ñ tem	Velocidade no envio de caracteres para a tela.
STOP	STOP	STOP	ñ tem	STOP	Paraliza o programa.
SLOW	SPEED=n	ñ tem	SLOW	ñ tem	Executa o programa na modalidade lenta.
STORE	STORE A(X1, X2..)	ñ tem	ñ tem	ñ tem	Grava uma matriz em cassete.
SYS	ñ tem	ñ tem	ñ tem	SYS loc. de memória	Manda o controle para um programa em linguagem de máquina.
TEXT	TEXT	ñ tem	ñ tem	ñ tem	Seleciona o modo de texto.
TIME$	ñ tem	TIME$	ñ tem	TIME (só hora)	Fornece data e hora.
TRACE	TRACE	TRON	ñ tem	ñ tem	Acompanha a sequência de instruções realizadas pelo micro.
TROFF	NOTRACE	TROFF	ñ tem	ñ tem	Desliga o comando TRON.
VLIN	VLIN X,Y AT N	ñ tem	ñ tem	ñ tem	Desenha uma coluna vertical entre as linhas X e Y, sobre a coluna N.
WAIT	ñ tem	ñ tem	ñ tem	WAIT loc. de memória,X,Y	Paraliza o programa até que certos padrões sejam reconhecidos.
XDRAW	XDRAW A AT X,Y	ñ tem	ñ tem	ñ tem	Desenha a forma A no ponto X,Y usando cor complementar à última usada.

Todo programador sabe como é importante (e difícil, no Brasil) conhecer o sistema operacional de sua máquina. Se você tem um compatível com o ZX Spectrum, aí vai um presentão: a 1.ª parte de um artigo contendo...

O mapa da ROM do TK90X

Álvaro Ferreira de Freitas Borja

Espero que este mapa da ROM venha ajudar aos usuários tupiniquins do ZX Spectrum, e porque não dizer, aos usuários do seu similar nacional, o TK90X, visto que infelizmente os fabricantes brasileiros (com raras exceções) ainda não se convenceram que o usuário precisa de mais informações sobre o equipamento que adquire, pois sem elas fica muito difícil (ou impossível) o desenvolvimento de algo mais complexo (ou sério, como preferirem chamar).

Quem já teve a oportunidade de rodar os programas importados, se é que algum programa (dos bons) para o TK90X não o seja, sabe do que estou falando.

Para os que gostam de programar em linguagem de máquina e possuem um TK90X, como o autor, desejo que este artigo venha ajudar ou, pelo menos, dar o pontapé inicial para um melhor aproveitamento dos recursos desta máquina fantástica.

ROM DO ZX SPECTRUM

O ZX Spectrum possui um monitor ocupando 16 Kb de programas escritos em linguagem de máquina do microprocessador Z80. Este monitor está dividido em três partes principais, que são: rotinas de entrada e saída, interpretador BASIC e operação de expressões.

Rotinas de restart e tabelas – no início do programa monitor estão localizadas as várias rotinas de *restart* que são chamadas por meio da instrução RST. Todos os *restarts* do Z80 são usados; por exemplo, RST 08 é utilizado para mensagens de erro do sistema. Nesta parte do programa monitor estão as tabelas que contêm a forma expandida dos *tokens* e códigos-chave.

Rotina do teclado – é feita uma leitura do teclado a cada 1/60 de segundo (ou 1/50, dependendo da localidade); a rotina de teclado retorna com o código do caráter requerido.

A todo instante o registrador E retorna com um valor na faixa de 03-39, que representa cada uma das 40 teclas do teclado ou 255, caso não haja nenhuma tecla pressionada.

Rotina de Som – o Spectrum possui um alto-falante simples e uma nota é produzida, sendo usada repetidamente a instrução OUT, de modo apropriado, na porta 254 (FEh).
Nota: o TK90X utiliza o alto-falante da própria televisão.

Rotinas de operação com o cassete – ao contrário do ZX81, no Spectrum este é um bloco bastante extenso (1.329 bytes).

Programas escritos em BASIC ou blocos de dados, são precedidos por um cabeçalho de 17 bytes – – – header (Veja a matéria *Analisador de header* em MS nº 53, pág. 34), que é salvo primeiramente. Este cabeçalho descreve as características do bloco de dados que é gravado após ele.

Em todos os casos, (SAVE, LOAD, MERGE e VERIFY) os pares de registros DE contém o comprimento do bloco; o par XI contém o endereço básico, e o registro A é carregado com 00, para o caso de um header, ou FFh (255), para um bloco de dados/programa.

Rotinas de manipulação do vídeo e impressão – todas as rotinas de entrada/saída do Spectrum são vetorizadas através da área de informações dos canais e fluxos.

No Spectrum, normalmente as entradas são possíveis somente pelo teclado, porém saídas podem ser direcionadas para a impressora, parte superior ou inferior do vídeo.

Rotinas de execução – nesta parte do programa monitor, está localizado o procedimento de inicialização e o loop principal de execução do interpretador BASIC.

Toda linha BASIC quando retorna do editor é checada quanto a correta sintaxe e então é salva na área de programa, isto para o caso de iniciar com um número de linha; caso contrário, é executada imediatamente.

Interpretação do comando e a linha BASIC – esta parte do monitor considera uma linha BASIC como sendo um conjunto de declarações e, no seu procedimento, cada declaração é iniciada por um comando particular.

Rotinas aritméticas – o Spectrum opera com dois tipos de números: valores inteiros que variam de –65535 a +65535 ou números de ponto flutuante na forma de cinco bytes (foi mantido o padrão do ZX81).

O calculador de ponto flutuante – o calculador do Spectrum opera números e strings, e estas operações são especificadas por literais. Essa pode, portanto, ser considerada como sendo uma linguagem interna de operação do *stack* no calculador.

Esta parte do programa monitor contém rotinas para todas as funções aritméticas. As aproximações para SIN x, EXP x, LN x e ATN x são obtidas pelo desenvolvimento dos polinômios do *Chebyshev*.

As rotinas de restart e tabelas

0000 Rotina de inicialização (START). As interrupções mascaráveis são desabilitadas, e o par de registradores DE é carregado com o valor máximo para o topo da RAM.
0008 Rotina das mensagens de erro do sistema. O ponteiro de erro é ajustado para a posição de erro.
0010 Imprime o caráter cujo códi-

go está no registrador A.

0018 O conteúdo da posição corrente endereçada por CH-ADD é buscada: um retorno é feito se o valor presente for de um caráter printável, caso contrário, CH-ADD é incrementado e o teste repetido.

0020 Quando uma linha BASIC é interpretada, esta rotina é chamada repetidamente ao longo da linha.

0028 Desvia para o calculador de ponto flutuante.

0030 Esta rotina cria espaço na área de trabalho. O número de espaços é determinado pelo valor corrente no par BC.

0038 Interrupção mascarável, sempre que ocorre o clock é incrementado e é feita uma varredura no teclado.

0066 Interrupção não mascarável (NMI). Se a variável do sistema NMIVCT (23728) for 0, ocorrerá um reset (no TK90X, executará um salto para o endereço que estiver armazenado nesta variável, porém, 0 não reseta).

007D O valor presente no registro A é testado para ver se este é printável.

0095-0204 Tabela dos *tokens*; o último byte de cada *token* está com o bit 7 setado.

0205 – 022B Tabela das chaves principais (modo L e CAPS SHIFT).

022C – 025F Tabela do modo estendido.

0260 – 0269 Tabela dos códigos de controle (tecla digitada + CAPS SHIFT).

026A – 0283 Tabela dos símbolos (tecla + SYMBOL SHIFT).

0284 – 028D Tabela do modo estendido (CAPS SHIFT + SYMBOL SHIFT + tecla).

Rotinas do teclado

028E Rotina de leitura do teclado; se nenhuma tecla estiver pressionada, o par DE retorna com o valor 65535.

02BF Esta rotina é chamada sempre que ocorre uma interrupção mascarável. Em uso normal, ocorre uma vez a cada 20 ms. A finalidade desta rotina é ler o teclado e determinar o valor da tecla pressionada.

0310 Sub-rotina de repetição de tecla. Haverá uma repetição de tecla, se na primeira ocasião ela ficar pressionada por um período de 0,7 segundos, e, posteriormente, se ficar pressionada 0,1 segundo.

031E O valor da tecla é testado e um retorno é feito, se não houver tecla pressionada ou somente SHIFT; caso contrário, o código desta tecla é achado.

0333 Rotina de decodificação do teclado. Esta rotina é entrada com o código principal do registro E, o valor das FLAGS em D, o valor no modo do registro C e o byte invertido no registro B.

Rotinas de som

03B5 Sub-rotina que gera som no alto-falante. A entrada é feita com o par de registradores DE contendo o valor "F-T", onde uma nota de freqüência F terá uma duração de T segundos, e o par HL conterá um valor igual ao número de T *states do loop* de temporização dividido por 4.

03F8 Comando SOUND; esta sub-rotina é entrada com dois números na pilha do calculador. O número no topo representa a tonalidade da nota e o mímero abaixo, a duração.

046E – 04A9 Tabela dos SEMI-TONS.

Rotinas de operação com cassete

04C2 Salva o cabeçalho (header) e posteriormente o programa corrente na memória ou bloco de dados.

053F Esta sub-rotina é comum tanto para o SAVE como para o LOAD. A borda é setada com a cor original, e a tecla BREAK é testada.

0556 Carrega o header e depois carrega (ou verifica) um bloco de dados.

0605 Este ponto de entrada é usado por todas as quatro instruções (SAVE, LOAD, VERIFY e MERGE). O valor contido na variável T-ADDR (23668) é que permitirá fazer a distinção entre as quatro instruções.

0642 Indica erro do tipo F – – – nomc inválido.

0670 Idem erro tipo 2 – – – variável inexistente.

07CB Rotina de controle do VERIFY.

0802 Sub-rotina para carregar um bloco de dados. É utilizada por todas as rotinas de carregamento (LOAD, VERIFY e MERGE).

0808 Rotina de controle do carregamento de um programa BASIC, variáveis ou um array.

08B6 Rotina de controle do MERGE.

092C Sub-rotina para executar MERGE numa linha ou variável.

09A1 – 09F3 Tabela de mensagens do cassete. Cada mensagem é mostrada com o último caráter invertido.

Rotinas de manipulação do vídeo e impressora

09F4 Esta rotina é entrada com o registrador A contendo o código de um caráter de controle, um caráter printável ou um *token*.

0A11 – 0A22 Tabela dos caracteres de controle.

0A23 Sub-rotina para mover o cursor para a esquerda.

0A3D Sub-rotina para mover o cursor para a direita.

0A4F Sub-rotina de *carriage return*.

0A5F Sub-rotina para imprimir usando a tabulação com vírgula (coluna 0 ou 16). Ex.: PRINT,, "ok!".

0A69 Imprimir uma interrogação sempre que for feita uma tentativa de imprimir um código não printável.

0A6D Rotina de caracteres de controle com operandos.

0ADC Esta sub-rotina guarda a posição da linha e coluna, bem como o endereço do pixel na variável do sistema apropriada.

0B03 Sub-rotina para buscar os parâmetros da posição corrente na variável do sistema apropriada.

0B24 Sub-rotina para imprimir um caráter qualquer.

0B7F Sub-rotina usada para imprimir todos os bits dos caracteres (matriz 8x8). Na entrada, o par de registradores DE contém o endereço base da tabela de caracteres; o par HL, o endereço de destino; e o par BC, os valores da linha e coluna correntes.

0BDB Seta byte na área de atributos.

0C0A Sub-rotina usada para imprimir mensagens e *tokens*. O registro A é carregado com o número da mensagem ou do *token* da tabela. O par DE é carregado com o endereço base da tabela.

0C41 Retorna com o par DE apontando para o caráter inicial da entrada requerida.

0C55 Esta rotina é chamada a qualquer momento para forçar a necessidade de um scroll na tela.

0D4D Faz uma cópia temporária dos itens de cor.

0D6B Comando CLS.

0DAF Limpa toda a área de vídeo. É usada pelo CLS, listagem automática e rotina principal de execução.

0DD9 Seta parâmetros para CLS.

0DFE Sub-rotina de scroll. O registro B é carregado com o número de linhas a serem roladas (B = 23).

0E44 Esta sub-rotina limpa as "n" linhas da parte inferior do vídeo, sendo que o número de linhas deverá ser carregado no registro B.

0E88 Limpa a área de atributos.

0EAC Comando COPY.

0ECD Envia para a impressora o conteúdo do buffer de impressão.

0EDF Limpa o buffer da impressora.

0EF4 Copia uma linha na impressora.

0F2C Rotina do editor. O editor é chamado em duas ocasiões: quando da entrada de uma linha BASIC e pela rotina do comando INPUT.

0F81 Esta sub-rotina adiciona um código para o corrente EDIT ou linha INPUT.

0FA0 – 0FA8 Tabela das teclas de edição.

0FA9 Sub-rotina de edição.

0FF3 Cursor para baixo, no modo de edição.

1007 Cursor para esquerda, no modo de edição.

100C Cursor para direita, no modo de edição.

1015 Sub-rotina para apagar um caráter, no modo de edição.

101E Os dois próximos códigos da rotina de entrada são ignorados.

1024 Sub-rotina de entrada para a edição.

1031 Sub-rotina de corte. O endereço do cursor é carregado no par HL e este é decrementado, a não ser que o cursor já esteja no início da linha. Cuidado é tomado para não colocar o cursor entre os caracteres de controle e seus parâmetros.

103E Loop para garantir que os caracteres de controle não sejam separados de seus parâmetros.

1059 Cursor para cima, no modo de edição.

1097 Sub-rotina para limpar a área de edição e espaço de trabalho.

10A8 Esta é uma das importantes sub-rotinas; ela retorna o código da última tecla pressionada no modo CAPS LOCK. A troca no modo e parâmetros de controle da cor são manejados dentro da sub-rotina.

111D Esta sub-rotina é chamada sempre que a linha na área de edição ou INPUT precisem ser printados na parte inferior do vídeo.

1190 Esta sub-rotina retorna com o par HL apontando para a primeira posição e o par DE para a última posição da área de edição ou espaço de trabalho.

11A7 Sub-rotina para remover a *máscara* da marca de início da representação em ponto flutuante em uma linha BASIC (código 14).

Álvaro Ferreira de Freitas Borja está cursando engenharia civil, na UFPA. Atualmente, trabalha como Desenhista-Projetista na Telepará, sendo também usuário dos micros TK83 e TK90X.

Ansioso para continuar descobrindo seu TK90X? Então, prepare-se. Neste número MS traz endereços de mais rotinas internas do seu micro.

O mapa da ROM do TK90X

Álvaro Ferreira de Freitas Borja

Em MS nº 62 iniciamos o estudo dos endereços da ROM do TK90X, mostrando as rotinas de *restart* e tabelas; teclado; som; operação com o cassete; e manipulação de vídeo e impressora. Agora, apresentaremos, nesta segunda e última parte do artigo, os endereços das rotinas de execução; avaliação de expressões; rotinas aritméticas; e calculador de ponto flutuante.

ROTINAS DE EXECUÇÃO

11B7 Comando NEW.
12A2 Loop de execução principal. O loop principal estende-se do endereço 12A2h para o endereço 15AEh e ele controla o modo de edição, a execução de comandos diretos e a produção de mensagens.
1391-1554 Tabela das mensagens de erro.
155D Sub-rotina para adicionar/mudar uma linha em um programa BASIC que esteja na área de trabalho.
15AF Informações iniciais dos quatro canais de comunicação (K – teclado, S – vídeo, R – área de trabalho e P – impressora).
15C6-15D3 Tabela de dados correntes para o canal.
15D4 Sub-rotina para esperar por uma tecla.
15F7 Sub-rotina para abrir um canal. O registrador A deverá conter o número do canal (normalmente FDh a 03). Então, dependendo do dado corrente, um canal particular será aberto.
1615 Seta as flags apropriadas para os diferentes canais.
162D-1633 Tabela de códigos para os canais.
1634 Seta as flags para o canal K.
1642 Seta as flags para o canal S.
164D Seta a flag para o canal P.
1652 Sub-rotina para criar um espaço em uma área. O par de registros HL deverá conter o endereço da posição onde se deseja criar o espaço.
1655 Esta sub-rotina é usada pela anterior, sendo que permite criar N espaços. O par BC deverá conter o número de espaços que se deseja criar e o par HL o endereço da primeira posição a partir da qual deverá ser criado.
169E Esta sub-rotina é chamada quando se usa em RST 30h. Ela cria espaço entre a área de trabalho existente e a pilha do calculador.
16B0 Sub-rotina para limpar a área de edição e todas as posteriores.
16E5 Comando CLOSE #
1716-161B Tabela de dados para fechar o canal corrente.
1736 Comando OPEN #
1795 Comandos LIST e LLIST.
17F5 Ponto de entrada do LLIST. Abre o canal para a impressora.
17F9 Ponto de entrada do LIST. Abre o canal para o vídeo.
1855 Sub-rotina para imprimir uma linha BASIC. O par de registradores HL deverá conter o endereço de início da linha, isto é, o endereço do byte de maior valor do número da linha.
18C1 Imprimir um caráter em FLASH. O cursor de erro e os outros cursores são usados por esta sub-rotina.
18E1 Imprime o cursor (modo C, E, G, K ou L).
1925 Todos os caracteres/*tokens* em uma linha de um programa BASIC são impressos fazendo-se repetidas chamadas a esta sub-rotina.
196F Acha o endereço de uma linha. O número da linha deverá ser carregado no par de registradores HL. Esta sub-rotina retorna com o endereço inicial da linha ou da primeira linha seguinte (caso esta não exista) no par HL, e da linha anterior, no par DE.
19B8 Usada para achar a "próxima linha" na área de programa ou a próxima variável na área de variáveis.
19FB Esta rotina é usada para ler o número da linha na área de edição. Se esta não tiver número, isto é, uma instrução direta, então o número da linha é considerado 0 (zero).
1A1B Imprime o número das linhas de um programa em BASIC. Valores acima de 9.999 serão impressos incorretamente.
1A48-1B16 Tabela dos comandos BASIC.
1B17 Cheque da sintaxe.
1BB2 Comando REM.
1C01-1C0C Tabela da classe dos comandos.
1C0D Comandos classe 03. São comandos que podem ou não serem seguidos por um número. Exemplo: RUN e RUN 100.
1C10 Comandos classe 00. Não precisam de operando. Exemplo: COPY.
1C11 Comando classe 05. Exemplo: PRINT.
1C1F Comandos classe 01, 02 e 04. São usados por comando contendo variáveis. Exemplo: LET, FOR e NEXT.
1C22 Fornece os valores apropriados para as variáveis do sistema DEST (23621) e STRLEN (23666).
1C56 É usada pelas instruções LET, READ e INPUT para primeiramente avaliar e, então, atribuir valores para as variáveis previamente designadas.
1C96 Seta os parâmetros de cores.
1CBE Rotina usada pelas instruções PLOT, DRAW e CIRCLE para especificar a condição *default* (FLASH 8, BRIGHT 8 e PAPER 8).

1CDB Rotina usada pelas instruções SAVE, LOAD, VERIFY e MERGE.
1CEE Comando STOP.
1CF0 Comando IF. Na entrada o valor da expressão entre o IF e o THEN é o último valor na pilha do calculador. Se este é de lógica verdadeira, então, a próxima declaraçao é executada; caso contrário, a linha é considerada encerrada.
1D03 Comando FOR. A entrada é feita com o valor inicial e o limite de FOR no topo da pilha do calculador.
1D86 Esta sub-rotina é usada para detectar a ocorrência de outros DATA, DEF FN ou NEXT em um programa BASIC.
1DAB Comando NEXT.
1DDA Sub-rotina de controle do loop do NEXT.
1DEC Instrução READ.
1E27 Instrução DATA. Durante a checagem da sintaxe, uma instrução DATA é checada para ver se contém uma série de expressões válidas, separadas por vírgulas. Porém, durante a execução do programa, ela é pulada.
1E42 Comando RESTORE.
1E4F Comando RANDOMIZE.
1E5F Comando CONTINUE.
1E67 Comando GOTO.
1E7A Função OUT. Os dois parâmetros da declaração OUT são colocados na pilha do calculador e usados diretamente.
1E80 Instrução POKE. Funciona de modo similar ao OUT.
1E85 Coloca os valores do topo da pilha do calculador em registradores simples (cada um em um registrador).
1EA1 Comando RUN.
1EAC Comando CLEAR.
1EED Comando GOSUB.
1F05 Testa a quantidade de memória disponível.
1F23 Comando RETURN.
1F3A Comando PAUSE. O período da pausa é determinado pela contagem do número de interrupções mascaráveis que ocorrem a cada 1/60 segundos.
1F54 Esta rotina é executada constantemente para fazer a leitura da tecla BREAK. Retorna com a flag de Carry setada, caso as teclas SHIFT e BREAK estejam pressionadas ao mesmo tempo.
1F60 Função DEF FN.
1FC9 Comandos LPRINT e PRINT. O canal apropriado é necessariamente aberto.
1FF5 Sub-rotina usada para imprimir um retorno de carro.
1FFC Esta sub-rotina é chamada pelas rotinas dos comandos PRINT, LPRINT e INPUT.
2045 Teste do fim de impressão.
2089 Comando INPUT.
21B9 Sub-rotina chamada repetidamente, para cada valor *inputado*.
21D6 Retorna com a flag de zero resetada somente quando o canal K está em uso.
21E1 Rotinas de manipulação dos itens de cores.
2211 Rotinas de manipulação das variáveis do sistema que controlam as cores.
2294 Comando BORDER.
22AA Sub-rotina usada pelo comando PLOT para determinar o endereço de um pixel. A entrada é feita com o par de registradores BC, contendo as coordenadas do pixel, e retorna com o par HL, contendo o endereço do byte da área de vídeo que contém o pixel e o registro A apontando para a posição do pixel dentro do byte.
22CB Sub-rotina usada pela varredura da função POINT. Ela é entrada com as coordenadas de um pixel na pilha do calculador e retorna com o valor 1, se o pixel é cor do INK; ou 0, se da cor do PAPER.
22DC Instrução PLOT.
2307 Esta sub-rotina transfere dois números em ponto flutuante para o par BC.
2314 Carrega o registro A com um número em ponto flutuante que esteja no topo da pilha do calculador. O número deve ser entre 0 e 255.
2320 Instrução CIRCLE.
2382 INSTRUÇÃO DRAW.
247D Sub-rotina usada pelas instruções CIRCLE e DRAW para setar os parâmetros iniciais.
24B7 Sub-rotina usada pela instrução DRAW para traçar uma linha reta.

AVALIAÇÃO DE EXPRESSÕES

24FB Esta sub-rotina é usada para executar uma avaliação do resultado da próxima expressão. O resultado retornado é o último valor na pilha do calculador. Para um resultado numérico, o último valor será um valor em ponto flutuante. Porém, para um resultado string o último valor consiste de um conjunto de parâmetros. O primeiro valor dos cinco bytes é desprezado, o segundo e o terceiro bytes contêm o endereço de início da string, e o quarto e o quinto bytes contêm o »

GRAVE SEU PROGRAMA DE GRAVADOR PARA GRAVADOR SEM PROBLEMAS

COM FILTRO FCA

Elimine erros de passagem de programas p/ seu micro

TECNISERVICE
EQUIPAMENTOS ELETRÔNICOS LTDA.

DISTRIBUIÇÃO EXCLUSIVA

MAGNODATA informática ltda.
Av. Paulista, 2644 — 8º Cj. 86
F: (011) 255.7653

Peek & Poke

SOFTWARE PARA CP 400 E COMPATÍVEIS

APLICATIVOS: Processador de Textos PPTEXTO c/ acentuação em português, Planilhas Eletrônicas, Bancos de Dados, Pacotes Integrados.
UTILITÁRIOS: Editor/Assembler, Expansor de Tela, Supertec, Compiladores, Copiadores, Etc.
GRÁFICOS: DUMP P-500, Esquemas Eletrônicos, Bjork Blocks, Comerciais, Graphicom e outros.
LINGUAGENS: LOGO em português, PILOT para professores e uso didático, PASCAL, CBasic.
JOGOS: Muitos títulos para mencionar aqui! Nossos catálogos mostram as telas dos jogos. Temos jogos exclusivos. Escolha melhor na Peek & Poke!

SOFTWARE PARA CP 500 / CP 300

— Processador de textos PPTEXT 500 DISCO c/ acentuação.
— Planilha Eletrônica CALC-300. Única em Z-80 para versão K-7.
— Utilitários copiadores para fita e disco.
— Editor/Assembler, monitor MACMON e tutor de linguagem de máquina. Em fita ou disco. Aprenda Z-80!
— Compilador ZBasic. Produz código objeto com rapidez.

SOFTWARE PARA LINHA MSX

Revendemos cartuchos e fitas c/ os melhores preços. Desenvolvemos títulos próprios. Entre em contato.

A PEEK & POKE demonstra, garante e dá suporte permanente a seus programas. Venha conhecê-los ou solicite catálogo completo para o seu equipamento.

PEEK & POKE Microcomputação e Comércio Ltda.
Av. Brig. Faria Lima, 1664 - Conj. 1102
01452 - São Paulo - SP - Fone: (011) 813-3277

TUDO SOBRE INFORMÁTICA AO SEU ALCANCE!

Seja qual for o assunto de seu interesse na área de Informática, vale a pena conhecer nossa seção especializada.
Nela você encontrará livros e revistas de todos os níveis, para principiantes, estudantes e profissionais, selecionados das mais diversas editoras.

Seção de Informática da
LOJA DO LIVRO ELETRÔNICO
SP — Rua Vitória, 383 — Tel.: (011) 221-0683 — CEP 01210.
RJ — Av. Mal. Floriano, 143 — Sobreloja Tel.: (021) 221-2442 — CEP 20060.

ATENDEMOS PEDIDOS DO INTERIOR - CONSULTE-NOS

tamanho da string.
2596-25AE Tabela das funções de avaliação.
25AF Rotinas da avaliação de funções.
26C9 Rotina de avaliação de variável.
2795-27AF Tabela dos operadores OR, AND, <, >, =, + etc.
27B0-27B6 Tabela das prioridades dos operadores.
27BD Esta sub-rotina é chamada para avaliar a função definida por DEF FN.
28B2 Esta sub-rotina é chamada a qualquer momento para fazer uma procura na área de variáveis ou de argumentos sempre que uma declaração DEF FN for requerida.
2996 Sub-rotina normalmente usada para encontrar os parâmetros que definem uma string existente na área de variáveis ou para retornar com o par HL apontando para o endereço base de um elemento particular ou um número array. Quando chamada pela declaração DIM, esta sub-rotina somente faz um cheque da sintaxe.
2AB1 Passa os valores que estão nos registradores A, B, C, D e E para a pilha do calculador.
2AFF Comando LET.
2BA6 Passa qualquer valor numérico (da pilha do calculador) ou string (da área de trabalho), para a devida posição dentro da área de variáveis.
2BAF Continuação do comando LET.
2BC6 Pega os parâmetros de uma nova string para testar se há espaço na memória para então a string ser finalmente transferida.
2BF1 Transfere o valor no topo da pilha do calculador (ponto flutuante) para os registros B, C, D, E e A, nesta ordem, sendo que B assume o valor do byte mais no topo e assim sucessivamente.
2C88 Retorna com a flag de Carry setada, se o valor presente no registro A representa um dígito ou letra válida.
2C8D Retorna com a flag de Carry setada, se o valor presente no registro A representa uma letra válida do alfabeto.
2C9B Converte um número decimal em ponto flutuante. Esta sub-rotina lê o número decimal, dígito por dígito, e coloca o resultado no topo da pilha do calculador.
2D1B Retorna com a flag de Carry resetada, se o valor presente no registro A representa um dígito válido.
2D28 Converte um valor binário absoluto que esteja no registrador A em um número com ponto flutuante e o coloca no topo da pilha do calculador.
2D2B Converte um valor binário absoluto que esteja no par de registros BC em um número com ponto flutuante e o coloca no topo da pilha do calculador.
2D3B Converte um número inteiro para ponto flutuante.

ROTINAS ARITMÉTICAS

2D4F Esta sub-rotina mostra o valor que está no topo da pilha do calculador, que é o resutado da conversão de um número na forma xEn, onde n é um inteiro negativo ou positivo. A entrada na sub-rotina é feita com x no topo da pilha do calculador e n no registro A.
2D7F Esta sub-rotina coloca em DE um valor inteiro n (onde n varia de -65535 a +65535) da posição endereçada por HL. Sendo n normalmente o primeiro valor no topo da pilha do calculador.
2D8C Empilha um valor inteiro n (onde n varia de -65535 a +65535) da posição endereçada por HL e as quatro posições seguintes, ou seja, recoloca o primeiro valor do topo da pilha do calculador.
2DA2 Transfere um número em ponto flutuante que esteja no topo da pilha para o par de registradores BC. Se o resultado for maior que 65535, retorna com a flag de Carry setada. Se o resultado for negativo, então, a flag de zero é resetada. O byte de menor valor é copiado também no registro A.
2DD5 Transfere um número em ponto flutuante que esteja no topo da pilha para o registrador A.
2DE3 Imprime um número em um ponto flutuante.
2F8B Multiplica cada byte DE por 10 e retorna com a parte inteira do resultado no registro C.
2F9B Prepara um número em ponto flutuante para a adição.
300F Operação de subtração.
3014 Operação de adição.
30C0 Prepara um número em ponto flutuante para divisão ou multiplicação, retornando com a flag de Carry setada, se o número for zero, colocando o sinal do resultado no registro A.
30CA Operação de multiplicação.
31AD Operação de divisão.
3293 Recoloca na pilha do calculador dois números inteiros transformando para cinco bytes em ponto flutuante, provenientes da operação binária da adição, multiplicação e divisão.
3297 Recoloca um número no topo da pilha sob a forma de cinco bytes em ponto flutuante.

CALCULADOR DE PONTO FLUTUANTE

32C5-32D6 Tabela das constantes (0, 1, 1/2, Π/2, 10).
32D7-335A Tabela dos endereços.
335B Sub-rotina do calculador. Usada para executar os cálculos em ponto flutuante. Estes podem ser considerados de três tipos:

a) Operações binárias. Ex.: divisão, onde dois números em ponto flutuante são subtraídos para gerar um valor no topo da pilha do calculador;

b) Operações únicas. Ex.: co-seno, onde o valor no topo da pilha é trocado pelo resultado da função, gerando um novo valor no topo da pilha;

c) Operações de manipulação, onde o valor no topo da pilha é copiado para os principais cinco bytes da área de memória do calculador.
33A1 Sub-rotinas DELETE.
33A2 Executa uma operação aritmética simples.
33A9 Teste de memória suficiente para colocar outro número em ponto flutuante na pilha do calculador.
33C0 Move um número em ponto flutuante para o topo da pilha do calculador ou do topo da pilha para a área de memória do calculador.
3406 Localiza o endereço base para cada porção de cinco bytes da área de memória do calculador para (ou de) onde um número em ponto flutuante será movido da (ou para a) pilha do calculador.
341B Usada para empilhar uma constante.
342D Armazenar dados na área de memória do calculador.
343C Troca o primeiro número do topo da pilha do calculador com o segundo, tipo um SWAP do FORTH.
3449 Rotina para gerar as Séries de *Chebyshev*.
346A Função ABS.
346E NEGATE. Troca o sinal do valor no topo da pilha do calculador.
3492 Função SGN.
34A5 Função IN.
34AC Função PEEK.
34B3 Função USR.
34BC Função USR string.
34E9 Testa se um número em ponto flutuante é zero. Este teste exige que os quatro primeiros bytes do número sejam zero. Retorna com a flag de Carry setada se o número é zero.
34F9 Testa se o valor no topo da pilha é maior que zero. Coloca 1, se o valor for igual a zero, e 0, caso contrário.
3506 Testa se o valor no topo da pilha é menor que zero. Coloca 1, se o valor for menor que zero, e 0, caso contrário.
350B Seta o valor no topo da pilha para 0, se a flag de Carry está resetada; caso contrário, coloca 1.
351B Operação lógica OR.
3524 Operação lógica AND.
352D Operação AND entre um número e uma string.
353B Operações de comparação(<=, <>, >= etc.).
359C Executa concatenação de strings.
35C9 Função CHR$.
35DE Função VAL e VAL$.
361F Fupção STR$.
3669 Função CODE.
3674 Função LEN.
367A Esta sub-rotina é usada somente pelo gerador de séries, o seu efeito é semelhante a um DJNZ, porém o contador é a variável do sistema BREG (23655).
368F Executa um salto condicional, se o valor no topo da pilha do calculador é verdadeiro.
369B Finaliza as operações no calculador (byte 38h).
36A0 Retorna o valor inteiro do quociente INT (X/Y) no topo da pilha do calculador, e o resto X-INT (X/Y) é colocado abaixo na pilha do calculador. Onde Y é um número inteiro positivo colocado no topo da pilha e X um valor inteiro que esteja na pilha, porém, logo abaixo de Y.
36AF Função INT.
36C4 Função EXP.
3713 Função LN.
37AA Função COS.
37B5 Função SIN.
37DA Função TAN.
37E2 Função ATN.
3833 Função ASN.
3843 Função ACS.
384A Função SQR.
3851 Executa uma operação binária tipo X elevado a Y (X ↑ Y).
386E-3CFF Área livre.
3D00-3FFF Tabela do conjunto de caracteres, 8 bytes para cada caráter.

Álvaro Ferreira de Freitas Borja está cursando engenharia civil, na UFPA. Atualmente, trabalha como Desenhista-Projetista na Telepará, sendo também usuário dos micros TK83 e TK90X.

Conhecendo alguns macetes não mencionados no manual do seu TK90X, você pode criar interessantes efeitos visuais na tela do seu micro.

Atributos no TK90X

Cláudio Bittencourt

Aí, tomado por súbita fúria, você digita no seu TK90X:
PRINT # 0; PAPER 8; INK 9; BRIGHT 8;
FLASH 8; "enlouqueci !" : PAUSE 0
E o tiquinho de juízo que ainda lhe resta desaparece de vez, porque o micro aceita numa boa, sem erros de sintaxe nem nada. E ainda goza:
0 Executado 0:2

Mas como executado!? A você (e a mim) é dado esse direito, o de pirar, ao micro jamais. Dele se espera um comportamento exemplar, coerente e sóbrio, eu diria até um pouco bitolado. A obrigação dele seria recusar semelhante disparate, pois o manual não deixa margem a dúvidas: PAPER e INK variam de 0 a 7, enquanto FLASH e BRIGHT podem assumir os valores 0 ou 1. Fora disso, o micro emite a mensagem de erro "K Cor inválida X:Y". E o que dizer então do estapafúrdio PRINT # 0?

Calma, leitor amigo, trata-se de uma inacreditável omissão do manual brasileiro. Estamos pois livres do Pinel: eu, você e o micro; por enquanto.

Aproveitemos então a recuperada lucidez para estudar o assunto. Mas desde já alerto: para um bom aproveitamento, você deve conhecer com razoável precisão as informações do citado opúsculo, em particular as do capítulo 17, referente as cores.

Como todos sabem, a imagem enviada pelo micro ao televisor compõe-se de duas partes. Duas telas, podemos dizer. A principal, de uso irrestrito para o usuário, aparece na parte superior do vídeo, enquanto a outra, reservada para mensagens de erro, entrada de dados e edição, ocupa a parte inferior. Geralmente, a primeira é constituída de 22 linhas e a segunda de apenas duas.

Os atributos de ambas ficam armazenados em uma área contínua de memória, com 768 bytes, o chamado arquivo de atributos, que, junto com o arquivo de imagens (6 Kb), é periodicamente varrido pelo Z80 na tarefa de enviar imagens e cores para o vídeo.

Para atualizar valores do arquivo de atributos, a UCP recolhe informações previamente armazenadas em outras posições da memória RAM, na área das variáveis do sistema.

Observe, no exemplo a seguir, que as instruções relativas a cores podem ser de duas naturezas, que chamaremos de permanente e temporária.

```
10 FLASH 1: BRIGHT 1
20 PAUSE 0
30 PRINT ; FLASH 0;"Estou"
40 PRINT "entendendo"
```

As duas instruções da linha 10 fazem com que a UCP armazene dados permanentes em alguma variável do sistema. Os efeitos dessas instruções não são imediatos, só vindo à tela por ocasião de um PRINT, CLS ou se for teclado ENTER no modo direto.

O FLASH da linha 30 é de natureza temporária, porquanto, embora prevalecendo sobre o da linha 10, só produz efeito particular naquele PRINT ao qual está subordinado, perdendo-se em seguida.

Estes e outros exemplos aqui apresentados devem ser rodados com os atributos de inicialização do TK90X, ou seja, com aqueles que são estabelecidos por ocasião do acionamento da máquina ou da execução de um comando NEW ou de um RESET. As instruções de pausa têm por objetivo permitir que sejam observados os tempos de execução das demais instruções.

Os atributos permanentes da tela superior são armazenados na variável do sistema chamada ATCLR P (endereço 23693) com a mesma estrutura de bits descrita no manual, ou seja:

ATRIBUTO	FLASH	BRIGHT	PAPER	INK
NÚMERO DO BIT	7	6	5 4 3	2 1 0

Quando a UCP inicia a execução de uma instrução PRINT, o conteúdo de ATCLR é integralmente copiado na variável ATCLR T (endereço 23695). Se houver, dentro do PRINT, alguma instrução de natureza temporária, os seus dados substituirão, bit a bit, os dados correspondentes no conteúdo de ATCLR T. Por fim, esse conteúdo é copiado na posição adequada, no arquivo de atributos, que é, em última análise, o que a gente vê na tela. Repare que, com esse mecanismo, o conteúdo de ATCLR P não é alterado, embora tenha contribuído apenas parcialmente para o resultado final.

Um CLS e um ENTER no modo direto são mais simples, pois a UCP apenas copia o conteúdo de ATCLR P em todas as posições do arquivo de atributos relativos à tela superior.

Mecanismo idêntico rege as instruções OVER e INVERSE, que também podem ser temporárias ou permanentes. Um par de bits para cada uma delas na variável SFLAG4 (endereço 23697) é suficiente, pois a informação a armazenar resume-se a ligado ou desligado.

Isso posto, vamos começar a esclarecer aquelas instruções estapafúrdias aceitas pelo micro no início do artigo.

O argumento 8 lá presente nas instruções de cores faz com que o UCP ignore o conteúdo de ATCLR P, preservando, com isso, os dados existentes no arquivo de atributos. Rode o exemplo e veja com seus próprios olhos:

```
  10 FOR f=0 TO 6: PAPER f: PRIN
T STR$ f;: NEXT f
  20 PAUSE 0
  30 PRINT AT 0,0; PAPER 8;"ABCD
EFG"
```

As variáveis do sistema responsáveis pela façanha são MASKCLRP (endereço 23694) e MASKCLRT (enredeço 23696).

Arguto que é, o amigo leitor já percebeu que a primeira armazena dados permanentes e a segunda abriga dados temporários. Esses dados funcionam como máscara (assim a chamamos), da seguinte forma: se, por exemplo, o bit 7 de MASKCLR T for igual a 1 (e ele será em decorrência de FLASH 8), significa que o bit correspondente no arquivo de atributos deve permanecer como está (0 ou 1), a despeito do que possa haver em ATCLR P.

A máscara permanente pode ser estabelecida também através de POKEs em MASKCLR P, o que permite efeitos diversos, porquanto podemos assim discriminar bits dentro das trincas de INK e PAPER, impossíveis de serem distingüidos por INK 8 e PAPER 8.

Note-se, aliás, que todas as variáveis que abrigam dados de natureza dita permanente podem ser poqueadas, economizando-se memória, tempo e comandos. Já para as de natureza temporária, o efeito de POKE é inócuo.

Quanto ao argumento 9, sua função é fazer com que, igualmente, o conteúdo de ATCLR P seja ignorado. Restrito às instruções INK e PAPER, ele não atua em proveito do arquivo de atributos, e sim do que podemos chamar de "cor de contraste". Essa cor, que só pode ser 0 (preto) ou 1 (branco), é automaticamente estabelecida de tal forma a permitir o melhor contraste entre o INK e o PAPER, segundo a tabelinha a seguir:

BASE	CONTRASTE
0 preto	7 branco
1 azul	7 branco
2 vermelho	7 branco
3 magenta	7 branco
4 verde	0 preto
5 ciano	0 preto
6 amarelo	0 preto
7 branco	0 preto

Quem tem tradição em software, tem tudo.

FOLHA DE PAGAMENTO
Cz$ 3.720

Faz a Folha de Pagamento de sua empresa, emitindo relatórios como Guia de IAPAS, Guia de FGTS, Relação de Empregados, Relação para I.R., Relação para Banco, Informe de Rendimentos, Acumulados Anuais, RAIS e Recibo de Pagamento. A folha pode ser semanal ou mensal. As tabelas são modificadas pelo próprio usuário.
Permite também, adiantamentos de salário, reajuste salarial, alterações de acumulados e outras funções que agilizem o processamento da Folha de Pagamento da empresa.

CONTABILIDADE
Cz$ 3.720

A Contabilidade de um mês em apenas 2 horas! Este Sistema permite o cadastramento de históricos padronizados e de plano de contas com até 5 níveis.
Emite Diário,Razão, Balancete, Balanço, Demonstração de Resultados, Demonstração de Lucros e Prejuízos acumulados, Listagem por centro de custo e extrato de contas, entre outras funções.

CONTROLE DE ESTOQUE
Cz$ 2.790

Controla o estoque de itens com Especificação, Estoque Mínimo, Unidade, Fornecedor, Localização e outras informações relacionadas no item como Custo Médio, Entradas e Saídas no período, etc.
Fornece Listagens Geral e Parcial dos produtos, Listagem Físico-Financeira, Listagem dos produtos abaixo do estoque mínimo, Lista de Preços e Etiquetas, entre outras.
Admite também, Reajuste de Preços, Alteração de Dados e Exclusão de Produtos.

A NASAJON oferece assistência técnica total, garantia permanente e mantém à sua disposição programadores e analistas para desenvolver sistemas específicos sob encomenda. Conte com a NASAJON SISTEMAS

Compatíveis com as linhas TRS-80 e Apple. Também disponíveis para IBM-PC. Procure-nos para maiores informações.

nasajon sistemas

Av. Rio Branco, 45 - Grupo 1.311
Rio de Janeiro - CEP 20.090
Tels.: (021) 263-1241 e 233-0615

Empresa filiada à ASSESPRO.

speroni

Assim, PAPER 9 resultará branco se INK tiver valor de 0 a 3 e preto nos demais casos. Veja o exemplo a seguir:

```
10 PAPER 9
20 FOR f=0 TO 7
30 PRINT INK f;f
40 NEXT f
```

Se você ainda não tinha reparado, isso é exatamente o que ocorre, de forma automática, com o INK da tela inferior quando mudamos a cor do BORDER. Experimente variá-lo de 0 a 7 e observe a mensagem "Executado".

O argumento 9 também pode ser temporário ou permanente, e a informação é armazenada em um par de bits na mesma variável SFLAG4, já citada para os casos de OVER e INVERSE. Os bits pares do conteúdo dessa variável referem-se a condições temporárias; e os ímpares, a condições permanentes. Um bit no estado set (igual a 1) significa que a condição está ligada. Veja a estrutura a seguir:

BIT	CONDIÇÃO
T P	
0 e 1	OVER
2 e 3	INVERSE
4 e 5	INK 9
6 e 7	PAPER 9

E, por fim, o PRINT#0, que serve para imprimir na tela inferior, com todos os recursos normais de uma instrução PRINT, ou seja, com AT, TAB, OVER, cores, etc.

Os atributos permanentes na tela inferior são armazenados na variável BORCLR (endereço 23624), cujos bits de PAPER definem também a cor do BORDER. Essa variável só pode ser manipulada pelo usuário através da própria instrução BORDER, que é bastante restritiva, ou mais livremente com POKE. Os atributos temporários são armazenados na mesma variável ATCLR T usada pela tela superior.

CUIDADOS ESPECIAIS

Alguns cuidados especiais são necessários no uso da tela inferior, em virtude das seguintes particularidades:

1) A tela inferior cresce automaticamente (sem a mensagem "scroll?"), diminuindo o espaço útil da tela superior);

2) Mensagens de erro e instruções INPUT apagam a tela inferior e reduzem-na à duas linhas originais.

3) A instrução BORDER atualiza a cor do PAPER, muda o INK para a cor de contraste e apaga o BRIGHT e FLASH.

Deixo para o leitor a tarefa de criar os seus próprios exemplos com PRINT#0.

Mas, a mente imprevisível de *Sir Clive Sinclair* houve por bem inventar nomes diferentes no manual do ZX Spectrum, para designar as mesmas variáveis aqui citadas. Para evitar possíveis confusões, segue-se a correlação entre elas:

TK90X	ZX SPECTRUM	ENDEREÇO
BORCLR	BORDCR	23624
ATCLR P	ATTR P	23693
MASKCLR P	MASK P	23694
ATCLR T	ATTR T	23695
MASKCLR T	MASK T	23696
SFLAG4	P FLAG	23697

A propósito, abro um parêntese para registrar que, no TK90X, a variável do sistema chamada STKEND não aponta o endereço do fim da pilha de cálculo, como sugere o nome, e sim o início. Faço essa ressalva, que nada tem a ver com cores, para poupar surpresas aos desavisados, pois existe uma STKEND no ZX Spectrum, apontando para o fim da pilha; e uma STKBOT, apontando para o início.

Fechado o parêntese, vai aqui uma dica interessante. O manual do TK90X nos ensina que, para conhecer os atributos de uma dada posição da tela, usamos a instrução ATTR (Y,X), onde Y e X têm o mesmo significado que em PRINT AT Y,X. O retorno da instrução é um número entre 0 e 255, resultando da seguinte conta:

Atributo = 128*FLASH+64*BRIGHT+8*PAPER+INK

Mas suponha que você queira saber os valores de FLASH, BRIGHT etc. e não a combinação deles. Aqui estão as instruções BASIC que permitem calculá-los:

```
LET FLASH=(atributo >127)
LET BRIGHT=INT(atributo/64)-2*(atributo>127)
LET PAPER =INT(atributo/8)-8* INT(atributo/64)
LET INK =atributo-8* INT(atributo/8)
```

ATRIB. GGMI

Se você ainda nos acompanha, dedicado leitor, eu o parabenizo pela persistência e ofereço, sob a chancela de GGMI, o programinha ATRIB. como aplicação prática do que foi abordado.

Atrib. serve para escrever-se na tela com o arquivo de atributos, e não com o de imagens.

E para que vou desejar uma coisa dessas? — Estará você perguntando.

Pois lhe asseguro que Atrib. não é de todo inútil, como parece à primeira vista. Por acaso você conhece o famoso joguinho *Manic Miner*? Se conhece, deve ter notado (e se não conhece fique sabendo) que a imagem apresentada durante o carregamento do programa, ou seja, as duas palavras Manic e Miner, de efeito inusitado, são escritas com atributos em uma tela limpa. Jogando engenhosamente com PAPER, INK e FLASH, a palavra Manic fica acesa enquanto Miner permanece apagada, e vice-versa. Há, além do belo efeito, a vantagem de que o tempo gasto para carregar a imagem é dez vezes menor do que o LOAD ""SCREEN normal.

ATRIB. é bastante versátil, pois, além dos caracteres normais da ROM, você pode usar caracteres previamente definidos com UDG 2 e criar belas figuras piscantes. O programa denomina de palavra ao conjunto de um a quatro caracteres, que são impressos em bloco, a partir de uma posição X/Y definida por você, indicando o canto superior esquerdo. Há a alternativa de centralização automática da coordenada X.

As palavras são organizadas em dois grupos, que irão piscar alternadamente. Esses grupos são identificados pelos algarismos 0 e 1, que aparecem à esquerda, no INPUT da palavra. Na entrada do programa apresenta-se o grupo 0, que assim ficará até que você, em vez de uma nova palavra, pressione a tecla ENTER, abrindo o grupo 1. Neste grupo, se você teclar ENTER numa palavra vazia, avançará para as opções <R> de Retoca, <P> de Pára, <S> de Save , ou <L> de Load. Estas opções realizam respectivamente o retorno ao grupo 0, interrompem a execução do programa, efetuam a gravação automática em fita do arquivo ou ainda carregam um arquivo previamente gravado com o ATRIB.

Você também pode apagar uma região da tela onde tenha cometido algum equívoco, digitando uma palavra só de espaços (código 32 do manual). Mas se houver pelo menos um caráter diferente deste, a palavra será normalmente impressa.

A cor de cada caráter é escolhida por você, havendo as alternativas adicionais 8 e 9 com os seguintes efeitos:

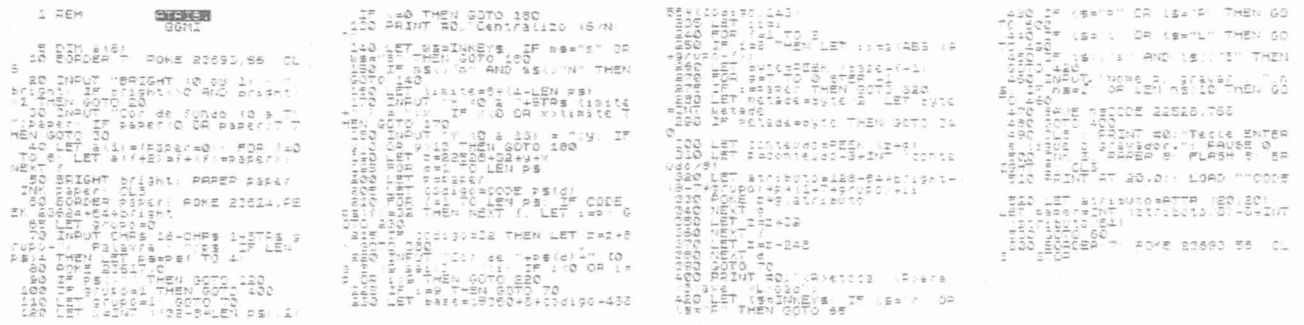
Cor 8 = imprime o caráter utilizando todas as cores, em matizes crescentes ou decrescentes, conforme se esteja no grupo 0 ou 1.

Cor 9 = interrompe a execução da palavra e solicita uma nova.

Observe que a cor de fundo, definida por você no início do programa, é proibida para os caracteres.

Para obter um belo efeito, as palavras de um grupo podem e devem ser sobrepostas às do outro, mas nunca às do seu próprio grupo.

Atrib. GGMI



Se, por acaso, você interromper a execução do ATRIB., digite logo RUN 600, para restabelecer as cores de inicialização do micro, sob pena de não poder listá-lo.

Para transformar o trabalho produzido em "tela de carregamento" de um programa de sua própria lavra, você deve utilizar-se de um programinha carregador, como o abaixo listado, que se encarregará de automatizar as operações:

```
  10 SAVE "carregador" LINE 20:
STOP
  20 INK 8: PAPER 8: FLASH 8: BR
IGHT 8: CLS
  30 PRINT AT 20,0;: LOAD ""CODE

  40 PRINT AT 20,0;: LOAD ""
```

Finalmente você deve colocar a seguinte seqüência na fita: o programa carregador (gravado com RUN e não com SAVE), a tela de atributos e, por último, o seu programa. Convém que este entre rodando (grave-o com SAVE "nome" LINE X), para evitar que a UCP tente continuar a execução do carregador, que já não mais existe.

Repare que o programinha carregador transfere a impressão dos nomes dos programas seguintes para a última linha da tela superior. Como ATRIB. impede a utilização dessa linha, PAPER e INK são, aí, iguais, fazendo com que os nomes não apareçam. Contibuem também para isso os argumentos 8 usados no carregador. Convém ainda que, no seu programa, você desabilite esse argumento, se necessário. Se desejar que o nome do seu programa apareça durante o carregamento, simplesmente substitua o INK 8 da linha 20 por INK 9, no programa carregador.

Cláudio de Freitas B. Bittencourt é formado em engenharia metalúrgica e Professor de Pós-Graduação em engenharia nuclear do IME, Instituto Militar de Engenharia, no Rio de Janeiro.

Editora Campus
A PRIMEIRA EM INFORMÁTICA

VISITE-NOS NA INFORMÁTICA SUCESU 86 — Stand 66 Pavilhão Central

Riocentro, de 18 a 24/08/86

LANÇAMENTOS 86

1. DICIONÁRIO ENCICLOPÉDICO DE INFORMÁTICA. *Fragomeni, A.H.* — **Cz$ 498,00**
2. C — A LINGUAGEM DE PROGRAMAÇÃO. *Kernighan, B.W. e Ritchie, D.M.* — **Cz$ 165,00**
3. PLANILHAS ELETRÔNICAS: COMO USÁ-LAS. *Berry, T.* — **Cz$ 145,00**
4. PC ASSEMBLER. *Quadros, D.G.* — **Cz$ 79,00**
5. SUB-ROTINAS BASIC PARA CP500 E COMPATÍVEIS (TRS-80). *Sinclair, I.* — **Cz$ 69,00**
6. CIRCUITOS SINCLAIR. *Santos Lima, D.* — **Cz$ 75,00**
7. BANCO DE DADOS PARA TK90X. *Rodrigues, M.S.* — **Cz$ 65,00**
8. BASIC PARA CRIANÇAS. *Watt, S. e Mangada, M.* — **Cz$ 59,90**
9. INFORMÁTICA: UMA INTRODUÇÃO. *Velloso, F.C.* — **Cz$ 95,00**
10. VISICALC: GUIA DO USUÁRIO BRASILEIRO. *Alcantara, R.B. e Alcantara, P.M.* — **Cz$ 86,00**
11. VISITREND/VISIPLOT: GUIA DO USUÁRIO BRASILEIRO. *Christmann, R.U.* — **Cz$ 67,00**
12. COMO PROGRAMAR SEU PC. *Hartnell, T.* — **Cz$ 61,00**
13. MUMPS. *Lobo, M.P.C.* — **Cz$ 73,00**
14. PASCAL PARA MICROS. *James, M.* — **Cz$ 92,00**
15. LINGUAGENS DE PROGRAMAÇÃO PARA MICROS. *Marshall, G.* — **Cz$ 90,00**
16. TÉCNICA DE GERENCIAMENTO DE ARQUIVOS. *Claybrook, B.Y.* — **Cz$ 155,00**
17. LOGO: INTRODUÇÃO AO PODER DO ENSINO ATRAVÉS DA PROGRAMAÇÃO. *Goodyear, P.* — **Cz$ 98,00**
18. CONTABILIDADE SISTÊMICA EM MICROCOMPUTADORES. *Secomandi, E.R.* — **Cz$ 67,00**
19. PROJETO ESTRUTURADO DE SISTEMAS. 2ª Edição Totalmente Revisada. *Stevens, W.P.* — **Cz$ 110,00**
20. INTELIGÊNCIA ARTIFICIAL EM BASIC. *James, M.* — **Cz$ 112,00**

Editora Campus

Rua Barão de Itapagipe, 55 — 20261 RJ — RJ Tel.: (021) 284.8443

( ) Anexo cheque nº.....do Banco....., nominal à Editora Campus Ltda., livre de despesas postais.

( ) Através do reembolso postal, acrescido de despesas postais.

( ) Desejo receber catálogo completo de Informática: o CAMPUSWARE.

Desejo receber os livros assinalados abaixo:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

Nome.....

Endereço .....

CEP .....Cidade .....

Estado .....Data...../...../.....

MS/AGOSTO/86

Aprenda a manipular a memória de tela do TK90X e aproveite ao máximo todos os recursos deste pequeno, porém versátil, micro.

A tela do TK90X

Álvaro Ferreira de Freitas Borja

Os usuários de microcomputadores que resolveram migrar para o TK90X (ZX Spectrum), constataram que a estrutura da tela desse equipamento não é linear, isto é, não é seqüencial.

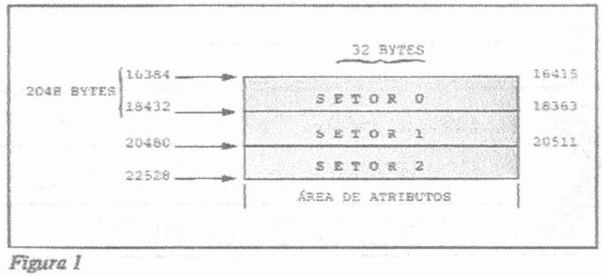


Figura 1

Tanto é verdade que, para pokear diretamente em uma determinada posição da tela, não basta adicionar o número de posições que se deseja deslocar ao valor inicial da área de vídeo, como nos outros micros, pois após as 32 primeiras posições a coisa começa a complicar. O importante é não entrar em desespero e procurar entender o que se passa.

O problema se deve ao fato de a área de vídeo ser de 6Kb (6144 bytes) e a área de atributos ser de apenas 768 bytes. Para contornar este problema, Sir Clive Sinclair usou de uma estrutura um tanto complexa para mapear o vídeo.

Primeiro, rode o programa a seguir para entender melhor o problema.

```
10 FOR i=16384 TO 22527
20 POKE i,255
30 NEXT i
```

Após isto, um navegante desavisado estará pensando como fazer para pokear uma determinada posição da tela. Primeiramente, devemos saber como é dividida a tela.

A área de vídeo foi separada em três setores (podemos chamar assim) de 2048 bytes, conforme a figura 1.

Cada setor é dividido em oito linhas (figura 2), dando um total de 24 linhas, sendo que as duas últimas são as linhas de edição, mensagens de erro etc.

As linhas, por sua vez, são divididas em oito sublinhas; sendo que o "passo" é de 256 bytes. Para uma melhor compreensão, na figura 3 é mostrada a divisão da linha 0 (zero), as outras são de modo análogo.

Pois bem, com o que foi visto até aqui já é possível determinar uma posição na tela, bastando apenas utilizar a seguinte equação:

16384 + 2048 * NST + 32 * NLS + 256 * NSL + C = POSIÇÃO NA TELA

Onde:

NST – Número do setor (0 a 2)
NLS – Número da linha no setor (0 a 7)

Mensagem de erro

Em **MS nº 52**, na Seção Dicas, pág. 54, as dicas **Amplie caracteres** e **Memória disponível** apresentaram erros. Em **Amplie caracteres** o problema está na primeira linha; logo após o comando DATA, o número é 160, e não 165.

Já na dica **Memória disponível**, o erro está na linha 30, que fica assim:

```
30 DATA 205,26,31,42,178,92,237
66,68,77,201
```

Em **MS nº 51**, no programa **Simulador de provas**, pág. 40, a listagem saiu com algumas imperfeições. Na parte "Preparação" as seguintes linhas apresentaram falhas:

```
50 LET [illegible]
[illegible] LET [illegible]
[illegible] LET
[illegible]
[illegible] FOR [illegible] TO 20: PRINT AT [illegible]
[illegible]
[illegible] CLS : PRINT AT [illegible]"MUDANCA
 DE QUESTAO"[illegible]"[illegible]
[illegible]
```

Já na parte "Execução", redigite as linhas abaixo:

```
[illegible] LET [illegible]
[illegible] IF [illegible] THEN GO
TO 160
```

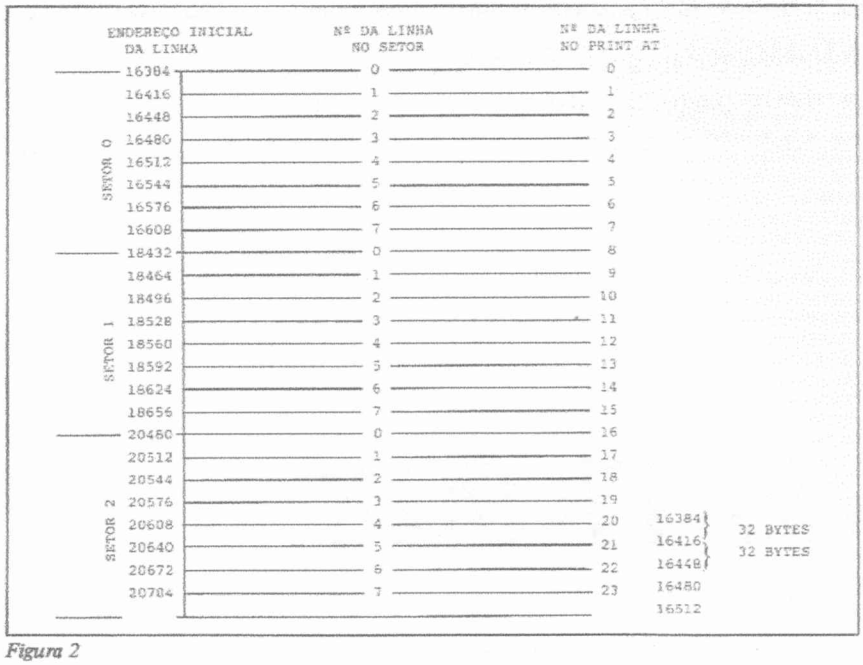


Figura 2

NSL – Número da sub-linha (0 a 7)
C – Número da coluna (0 a 31)

Para finalizar, apresentamos um programa que permite dar um pokeamento linear na tela do TK90X. Lembre-se da primeira tentativa.

```
 5 LET a=0
10 LET HL1=16384: GOSUB 50
20 LET HL1=18432: GOSUB 50
30 LET HL1=20480: GOSUB 50
40 STOP
50 LET HL=HL1
60 FOR b=0 TO 7
```

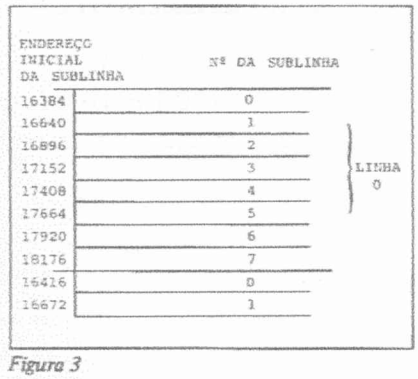


Figura 3

```
 70 FOR d=0 TO 31
 80 POKE HL+d,255
 90 NEXT d
100 LET HL=HL+256
110 NEXT b
120 LET HL1=HL1+32
130 LET a=a+1: IF a > 7 THEN LET a=0:
RETURN
140 GOTO 50
```

Espero que estas explicações tenham ajudado a tirar algumas de suas dúvidas sobre o mecanismo de mapeamento da tela do TK90X.

Álvaro Ferreira de Freitas Borja está cursando o quarto ano de Engenharia Civil, na UFPA. Atualmente, ele trabalha como Desenhista Projetista da Telepará, sendo também usuário dos micros TK83 e TK90X.

MSX
APLICATIVOS FALANDO PORTUGUÊS
EDUCATIVOS FALANDO PORTUGUÊS
JOGOS EMOCIONANTES

PROGRAMAS QUE VOCÊ ENTENDE!

• APLICATIVOS
101 — CONTROLE DE ESTOQUE MSX
102 — AGENDA DE ENDEREÇOS/MALA DIRETA MSX
103 — PLANILHA ELETRÔNICA MSX
• UTILITÁRIO
201 - EDITOR ASSEMBLER/ DISSASSEMBLER MSX
• COLEÇÃO PRINCIPIANTES
301 — PRINCIPIANTE - E
302 — PRINCIPIANTE - U
303 — PRINCIPIANTE - D

• JOGOS EMOCIONANTES
501 — KALEIDOSCÓPIO DIGITAL
502 — MS XADREZ
503 — MÁQUINA QUENTE
504 — MISSÃO: RESGATE DO SATÉLITE
505 — LABIRINTO DOS DIAMANTES
506 — BANCO FANTASMA
507 — VISITANTE DO FUTURO
508 — O AVENTUREIRO DO CAMPO MAGNÉTICO

carlos amaral

mistersoft
mistersoft
um senhor programa

À VENDA NOS REVENDEDORES MSX DE TODO O BRASIL.

RUA URUGUAIANA, 39/903 - CEP 20050
Rio de Janeiro - RJ - Tel.: (021) 221-1674 - C.P. 832/20001

Conheça, neste artigo, os processos de gravação, carregamento e verificação no TK90X. Veja também como funciona a instrução MERGE.

Gravação no TK90X

Aldo Barduco Jr. e Pierluigi Piazzi

O TK90X possui um bom sistema de gravação e leitura com recursos de verificação e MERGE. Existem cinco formas de gravação, quatro formas de carregamento, três de verificação e um MERGE (junção de programas).

O TK90X, como a maioria dos microcomputadores, armazena em fita o programa byte a byte. Nele, a velocidade de gravação e leitura é de 1200 bauds (quase 8 Kb por minuto).

Para se familiarizar com todos esses processos, digite o programa da figura 1 (extraído do livro *Explorando o TK90X*, da Editora Aleph). Observe que nas linhas 50 e 60, a parte que executa o movimento do lápis usa IN ao invés de INKEY$. Desta forma, podemos fazer a leitura de várias teclas simultaneamente, o que não ocorre quando se utiliza INKEY$. Assim, duas teclas juntas formam uma diagonal. Para deslocar o lápis, use as teclas Q, A, Z e S nas direções indicadas. Observe a figura 2.

Digamos que você, depois de brincar um pouco com o programa lápis, queira desligar o computador. Para não ter que redigitar o programa toda vez que for utilizá-lo, vamos gravá-lo.

```
  10 GOSUB 1000
  20 BEEP .1,10
  30 IF a=1 THEN OVER 1
  40 PLOT INK ink;x,y: LET a=x
LET b=y
  50 LET x=x+(IN 65022=62 AND x<
254)-(IN 65022=62 AND x>1)
  60 LET y=y+(IN 64510=62 AND y<
175)-(IN 65278=61 AND y>1)
  70 PLOT INVERSE (a=x AND b=y);
a,b
  80 LET a$=INKEY$
  90 IF a$<>"" AND a$<"7" THEN
LET n(1)=VAL a$
 100 IF a$="8" THEN PRINT AT 0,0
;6: FLASH 0: OVER 1: INVERSE 0;
INK 3;"": LET a=a+1-13 AND
a=2): PRINT AT 0,a+6; FLASH 1;"O
VER 1; INVERSE 0; INK 3;""
 110 OVER 0: GOTO 30
1000 LET x=127: LET y=87: LET in
k=0: LET a=0
1010 PRINT BRIGHT 1; INK 4;AT 7,
12;"CURSORES": BRIGHT 0; INK 1;
AT 10,4;"NORTE ... Q";TAB 17;"SU
L ... Z": PRINT AT 12,4; INK 1
;"LESTE ... S";TAB 17;"OESTE ...
A"
1020 PRINT  PRINT  PRINT BRIGH
T 1; INK 4;"DOIS CURSORES AO M
ESMO TEMPO FORMAM UMA DIA
GONAL"
1030 PRINT AT 21,0; FLASH 1; INK
 1;"APERTE QUALQUER TECLA
"
1040 IF INKEY$="" THEN GOTO 104
0
1050 IF INKEY$<>"" THEN GOTO 1050
1060 RUN
1070 PRINT FLASH 1; INK 3; PAPER
 9;"PINTA"; FLASH 0;" MOVE APAG
A"
1080 PLOT 0,167: DRAW 255,0
1090 RETURN
```

Figura 1

SAVE (LOAD)

Conecte o gravador com o cabo na posição MIC do computador e MIC do gravador. Digite **SAVE "Lápis"** e ligue o gravador no modo de gravação (REC). Assim, gravamos a parte do programa BASIC e os valores de suas variáveis. Mas será que gravamos certo? Para resolver este outro problema, vamos verificar a gravação usando a instrução VERIFY.

Rebobine a fita até o início do programa e conecte o gravador com o cabo nas posições EAR-EAR. Digite, então, **VERIFY "Lápis"** (lembre-se que o título do programa deve ter, no máximo dez caracteres). Ligue o gravador (no PLAY). Após as bordas ficarem com barras vermelhas e azuis vibrando, deverá aparecer o nome do programa no quadro central. A seguir, elas voltarão a vibrar e no fim aparecerá a mensagem **0 EXECUTADO 0:1**.

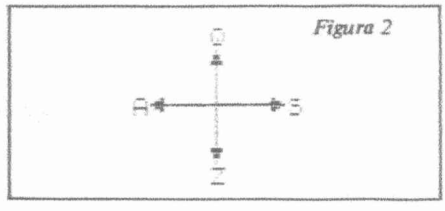


Figura 2

Caso não haja essa mensagem, houve gravação ou leitura errada. Neste caso, execute novamente a operação, alterando o volume e verificando se os agudos estão no máximo. Notando que não é um erro de carregamento, grave o programa novamente e repita a operação. Se o erro ainda persistir, limpe o cabeçote, troque de fita e ajuste o azimute (veja MS nº 47, pág. 70).

A gravação agora está boa, mas como carregar? Digite NEW para limpar a memória (lembre-se: NEW apaga o programa e tudo o que estiver na memória até a Ramtop). Mantenha o cabo na posição EAR-EAR e faça **LOAD "Lápis"**. O micro irá proceder como no VERIFY, porém, ao invés de comparar os dados, ele os carregará na RAM. Ao final da operação, deverá aparecer **0 Executado 0:1**.

Para gravarmos com *Auto Start*, ou seja, gravarmos um programa que rodará após o carregamento a partir de uma linha pré-determinada, fazemos o **SAVE "Nome" LINE X**, onde **X** é a linha do programa que queremos que seja executada após a carga. Então, faça **SAVE "Lápis" LINE 10**.

Não se esqueça: nas instruções SAVE, devemos conectar o cabo MIC-MIC, e nas instruções LOAD, EAR-EAR. Para verificarmos e carregarmos programas deste tipo, digitamos **VERIFY "Lápis"** e **"LOAD "Lápis"**. Não é possível digitar **VERIFY "Lápis" LINE 10** e/ou **LOAD "Lápis" LINE 10**.

Digite NEW, rebobine a fita e faça **LOAD""**. Quando não indicarmos o nome do programa, o micro carregará o primeiro que encontrar.

SAVE (LOAD) CODE

Agora que você já gravou o programa, crie um desenho que faça o melhor uso dele. Digamos que você queira guardá-lo para mostrar aos amigos. Há duas formas de fazer isso: a primeira é através do comando SAVE... CODE. Para gravar, digite BREAK e, a seguir, o comando **SAVE "Lápis" CODE 16384, 6912** (16384 é o endereço do início da D-FILE, ou seja, do local onde o equipamento guarda a tela, e 6912 é o número de bytes ocupados pela tela).

Quando gravamos a tela em fita não podemos verificar diretamente se ela foi gravada corretamente ou não, pois ao jogar o nome do programa no vídeo, ele altera os bytes da tela, que passam a não conferir com os do cassete. Digite **BREAK,CLS** e faça **LOAD "Lápis" CODE**.

Observe que não indicamos o endereço inicial e nem o número de bytes, pois o programa já traz consigo estas informações. (Note como a montagem da tela é esquisita, tipo *persiana*).

O **SAVE... CODE** serve para gravarmos qualquer seqüência de dados (bytes) guardados em endereços específicos, sendo possível com o **LOAD... CODE** carregá-los em outro endereço (ou no mesmo). Digite **UDG 2** e redefina alguns caracteres. Faça **SAVE "Teste" CODE USR "a", 168** (USR "a" é o endereço onde começa a tabela de caracteres). Depois, verifique se foi gravado corretamente, desligue o computador, digite novamente **UDG 2**, rebobine a fita e digite **LOAD" " CODE**. Agora, acabamos de carregar os caracteres especiais do TK90X (UDGs), já redefinidos por você.

```
10 DIM a$(10,5)
20 FOR f=1 TO 10
30 FOR g=1 TO 5
40 LET a$(f,g)=CHR$ (30+f+g)
50 NEXT g
60 NEXT f
```

Figura 3

```
10 LOAD "" DATA a$()
20 FOR f=1 TO 10
30 PRINT a$(f)
40 NEXT f
```

Figura 4

```
10 FOR f=0 TO 60
20 BEEP .25,f: NEXT f
30 PRINT "Aleph"

30 PRINT "Beit"
40 FOR f=-60 TO 0
50 BEEP .25,f: NEXT f
```

Figura 5

```
SAVE "N"                - Grava em fita cassete, com nome N, o programa
                          em BASIC.
SAVE "N" LINE X         - Grava como o anterior, mas com Auto-Start a
                          partir da linha X.
SAVE "N" CODE X,B       - Grava em fita cassete a partir do endereço X
                          tantos bytes quanto indicar B.
SAVE "N" SCREEN$        - Grava a tela em fita cassete.
SAVE "N" DATA X ()      - Grava em fita cassete os valores da variável X,
                          desde que esta já esteja dimensionada (X pode
                          ser uma variável alfanumérica ou numérica).
LOAD "N"                - Carrega na memória o programa de nome N, que
                          deve estar no cassete e ser em BASIC.
LOAD ""                 - Carrega na memória o primeiro programa em BASIC
                          que encontrar.
LOAD "N" CODE X,B       - Carrega na memória o programa N, que deve ter
                          sido gravado na forma de bytes (Save...Code),
                          a partir do endereço X tantos bytes quanto indi
                          car B.
LOAD "" CODE            - Carrega como o anterior, mas neste caso, não é
                          especificado N,X e B.
LOAD "N" SCREEN$        - Carrega na memória a tela de nome N.
LOAD "" SCREEN$         - Carrega na memória a primeira tela que encontrar.
LOAD "N" DATA X ()      - Carrega na memória os valores da variável X.
LOAD "" DATA X ()       - Carrega na memória o primeiro programa que forne
                          ça os valores de X.
VERIFY "N"              - Verifica se o programa de nome N, confere com o
                          que está na memória.
VERIFY ""               - Verifica se o programa contido na memória confere
                          com o do cassete.
VERIFY "N" CODE         - Verifica se os bytes gravados com nome N, conferem
                          com os contidos na memória.
VERIFY "" CODE          - Verifica se os bytes contidos na memória conferem
                          com os do cassete.
VERIFY "N" DATA X ()    - Verifica se os valores da variável gravada com
                          nome N conferem com os da memória.
VERIFY "" DATA X ()     - Verifica se os valores da variável X conferem com
                          os do cassete.
MERGE ""                - Executa o MERGE de um programa em BASIC.
MERGE "N"               - Executa o MERGE do programa de nome N, "emendando-
                          o" com o que está na memória.
```

Figura 6

Existe também um meio de só gravarmos os valores de uma variável desde que esta seja definida (ex.: DIM A (10, 10)). Para isso, usamos o **SAVE "nome do arquivo" DATA X ( )**, onde X pode ser uma variável numérica ou alfanumérica. Digite o programinha da figura 3. Terminado o programa, digite **SAVE "A$" DATA A$( )** e depois faça **VERIFY " " DATA A$ ( )**. Se a gravação estiver correta, digite NEW e depois o programinha da figura 4. Ele deverá carregar a variável A$ na sua memória. Se não houver um dimensionamento da variável (para gravação), ele acusará o erro 2.

MERGE

O TK90X possui ainda o MERGE, uma instrução que nos permite sobrepor programas e funciona da seguinte forma: ao carregar o programa, ele vai inserindo nos endereços respectivos as novas linhas, sem alterar as demais (o MERGE só atua na área do programa BASIC). Digite as rotinas da figura 5 e, no fim, grave cada uma delas com **SAVE "UM"** e **SAVE "DOIS"**.

Carregue o primeiro programa usando **LOAD "UM"** e depois, para carregar o segundo, digite **MERGE "DOIS"**. Terminando a execução desta instrução, observe a linha 30 (veja o que prevaleceu), rebobine a fita para carregarmos o primeiro programa e digite **MERGE "UM"**.

A linha 30 foi alterada em ambos os casos, porque os dois programas possuiam linhas 30 diferentes e, no MERGE, a que prevaleceu é a do último programa a ser carregado. As linhas ocupam sempre uma mesma posição seqüencial na memória, onde MERGE as carrega.

Resumindo: o SAVE salva o programa e as variáveis em fita; LOAD carrega o programa da fita no computador; VERIFY compara se os bytes que foram gravados no cassete conferem com os que o computador possui; e MERGE carrega um programa no computador, sobrepondo-o ao que já está na sua memória.

Para digerir todas as informações dadas até aqui, seria conveniente que você estudasse a tabela da figura 6.

Aldo Barbosa Jr. está cursando a primeira série do 2º grau e é autodidata em BASIC e Assembler. Possui um TK85 e um TK90X.

Pierluigi Piazzi é formado em Química e Física pela USP. Foi professor do Curso Anglo Vestibulares e já editou a revista "Microhobby". É autor de vários livros e atualmente dirige a Aleph Publicações, responsável por muitos sucessos na área editorial, dada a sua orientação didática.

Os usuários da linha Sinclair no Brasil sempre tiveram problemas com a limitação gráfica de seus micros. Mas agora, com o TK 90X, surgem novas e poderosas instruções. Veja uma delas neste artigo.

A função UDG do TK90X

José Rafael Sommerfeld

A função permite a definição do formato de 21 caracteres diferentes. Esta definição poderá se fazer de três maneiras, que são: definir caracteres acentuados da língua portuguesa; definir caracteres acentuados da língua espanhola e definir os próprios caracteres.

Para definir os caracteres com acentuação do português e do espanhol, o equipamento traz rotinas embutidas, bastando digitar UDG 0 (aciona a rotina de definição do português) ou UDG 1 (aciona a rotina do espanhol). Já para definir os próprios caracteres, o equipamento possui uma rotina que permite a manipulação, ponto a ponto, de cada caráter. Esta rotina é ativada quando se digita UDG 2, que será explicado adiante.

Os caracteres definidos ocuparão os lugares de caracteres gráficos das letras de A a U. Assim, para utilizá-los, deve-se colocar o equipamento no modo gráfico (pressionam-se simultaneamente as teclas CAPS SHIFT e 9).

Observem que as letras maiúsculas de A a U não deixam de existir no conjunto de caracteres, pois, para acessá-las, basta digitar sua tecla correspondente no modo maiúscula (pressiona-se CAPS SHIFT).

DEFININDO CARACTERES

Como já foi dito, existe uma rotina utilitária no equipamento que permite a manipulação dos pontos que compõem cada caráter. Ao ligar o equipamento e ativá-la (UDG 2), aparecerá na tela uma matriz de 8 x 8 com a ampliação da letra A e, logo abaixo, as letras de A a U com seus correspondentes caracteres gráficos. No início dessa utilização, os caracteres gráficos correspondentes serão os próprios caracteres, conforme mostra a figura 1.

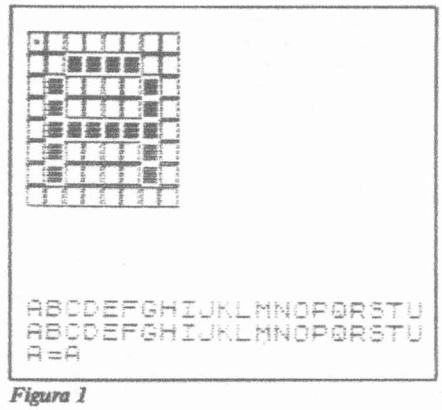


Figura 1

A letra A foi colocada em evidência, na matriz e antes do sinal de igualdade (abaixo na figura 1), somente por ser a primeira. Mas, caso se deseje utilizar qualquer outra das 20 restantes, basta pressionar a tecla correspondente a ela; então, será a nova letra em evidência, tendo como caráter gráfico o formato definido pelo usuário.

Observe que no canto esquerdo superior da matriz existe um ponto dentro de um quadrado. Este ponto poderá se deslocar dentro da matriz e pintar ou apagar cada um desses quadrados. Se desejar limpar toda a matriz, pressione as teclas CAPS SHIFT e 1 simultaneamente, o que fará com que todos os quadrados fiquem apagados. Para deslocar o ponto, basta pressionar as teclas de movimento, que são: 5 — esquerda; 6 — baixo; 7 — cima e 8 — direita. Desejando pintar os quadrados, pressione CAPS SHIFT e a tecla de movimento correspondente, simultaneamente, e o quadrado em que estava o ponto será pintado, deslocando o ponto para o sentido indicado na tecla de movimento. Para apagar os quadrados, pressione SYMBOL SHIFT e a tecla de movimento correspondente, também simultaneamente. Assim, o quadrado em que estava o ponto será apagado e deslocará o ponto para o sentido indicado na tecla de movimento.

Chamemos às tarefas de pintar e apagar os quadrados de **EDIÇÃO**. Assim, durante a edição do caráter poderá se observar, depois do sinal de igualdade (abaixo, na figura 2), a forma, em tamanho natural, do caráter que se está definindo.

Nas figuras 3 e 4, se apresentam mais dois caracteres definidos, e, na figura 5, um pequeno exemplo de uso destes ca- »

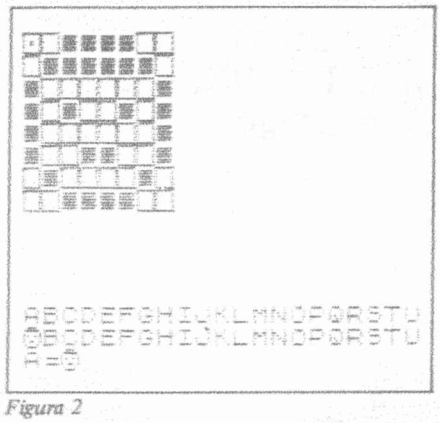


Figura 2

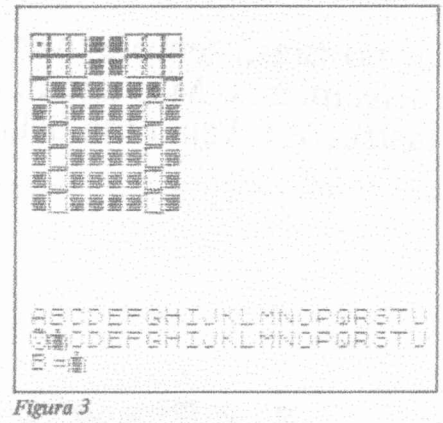


Figura 3

racteres. Para finalizar a edição de caracteres, é só pressionar as teclas CAPS SHIFT e 0 (zero), simultaneamente, e a rotina UDG retorna ao BASIC.

COMO GUARDAR OS CARACTERES

A função UDG permite a definição e uso dos caracteres enquanto o computador estiver ligado. Nem mesmo um comando NEW poderá apagar os caracteres definidos. No entanto, ao se desligar o computador, esses caracteres se perderão, pois ficam na memória RAM do equipamento, que, ao ser ligado de novo, não mais terá aquele conteúdo.

Não se esqueça de que, mesmo que o programa tenha sido gravado em fita, ao desligar e ligar o computador, lendo-se esta fita, ele interpretará que os caracteres gráficos ali contidos são os que ele tem definidos na memória RAM no momento, e estes, como já dissemos, são as próprias letras de A a U. Desta forma, o nosso exemplo da figura 5 seria entendido de acordo com o que mostra a figura 6.

Portanto, é necessário criar uma forma de gravar a definição dos caracteres que o programa for utilizar. Antes, é necessário que saibamos como cada caráter gráfico se organiza na memória.

Cada linha apresentada na matriz corresponde a um byte. Cada byte tem 8 bits, que é cada um dos quadrados da linha. Assim temos que, para guardar o formato de um caráter, o equipamento se utiliza de 8 bits. Cada quadrado pintado corresponderá a um bit com valor um, e cada quadrado em branco corresponderá a um bit com valor 0. A formação de 8 bits dará um valor de 0 a 255 ao byte.

Mostraremos agora como definir os caracteres dentro do programa. Será feita uma pequena rotina para ler os bytes, após o endereço dado na variável RAMTOP do sistema (que é onde ficam as definições dos caracteres gráficos), e imprimi-los na tela, anotando seus valores. A figura 7 mostra a rotina, e a figura 8 indica seu resultado. Na linha 9030, foi colocada entre parênteses a fórmula 8 * 3, onde 8 é o número de bytes por caráter e 3 o número de caracteres que se deseja como resultado. Se o número de caracteres for diferente de 3, este número deverá, então, substituir o número 3 na linha.

Para proceder desta maneira, utilize os caracteres mais à esquerda, ou seja, primeiro o **A**, depois o **B** e assim por diante, pois só pegaremos as definições dos **N** primeiros caracteres; em nosso caso, os três primeiros.

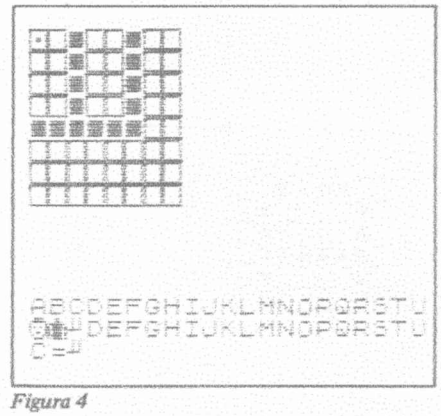


Figura 4

```
10 FOR X=31 TO 0 STEP -1
20 PRINT AT 10,X;"0"
30 PRINT AT 11,X;"■ "
40 PRINT AT 12,X;"▯ "
50 PAUSE 5
60 NEXT X
```

Figura 5

```
10 FOR X=31 TO 0 STEP -1
20 PRINT AT 10,X;"A "
30 PRINT AT 11,X;"B "
40 PRINT AT 12,X;"C "
50 PAUSE 5
60 NEXT X
```

Figura 6

```
9000 REM  Le bytes, apos RAMTOP
9010 REM  Imprimindo-os na tela
9020 LET x=PEEK 23730+256*PEEK 2
3731+1
9030 FOR X=x TO x+(8*3)-1
9040 PRINT X;" = ";PEEK X
9050 NEXT X
```

Figura 7

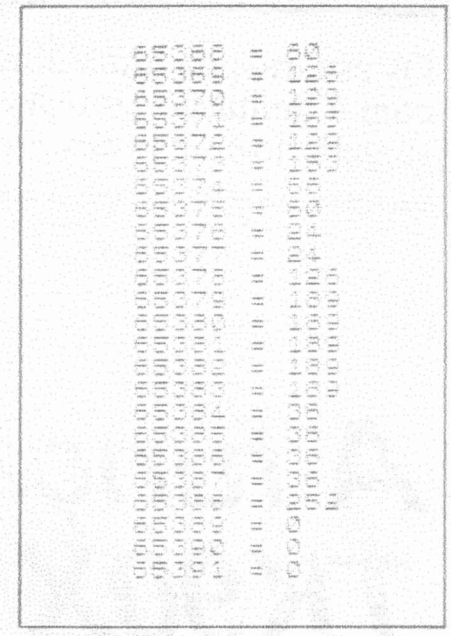


Figura 8

Já estando com os valores dos bytes conhecidos, deve-se, então, colocar uma nova rotina no início do programa, que será a responsável pela montagem dos caracteres gráficos novamente na memória, conforme mostra a figura 9.

Coloque quantos números quiser em cada instrução DATA. O importante é que eles estejam na mesma ordem em que foram mostrados pela rotina 9000. Na linha 6 foi colocada novamente a fórmula 8 * 3 já explicada. Substitua o 3 pelo número de caracteres gráficos que for utilizar. Os números colocados após DATA nas linhas 1, 2 e 3 se referem aos três caracteres definidos. Assim, deve-se colocar nestas linhas os números que forem dados pela rotina 9000 aos seus caracteres. Quanto mais caracteres for utilizar, mais números terão que ser definidos.

A rotina de montagem dos caracteres não precisa estar na posição definida na listagem, mas é necessário que ela seja executada logo no início do programa. Finalmente, elimine a rotina 9000 e grave o programa em fita.

UM MÉTODO MAIS SIMPLES

Até agora vimos como podemos colocar dentro do programa a definição dos caracteres gráficos. Existe, porém, uma maneira de salvar a parte da memória em fita, independente do programa. Assim, poderíamos gravar a área de memória que contém a definição dos caracteres gráficos. Este método, no entanto, nos levará a dois procedimentos na hora de carregar o programa: ler o programa em fita e ler a área de memória que contém as definições dos caracteres da fita.

Para que fique mais fácil, faremos

```
   1 DATA 60,126,129,165,129,153
,66,60
   2 DATA 24,24,126,189,189,189,
189,189
   3 DATA 36,36,36,36,252,0,0,0
   5 LET i=PEEK 23730+256*PEEK 2
3731+1
   6 FOR x=i TO i+(8*3)-1
   7 READ y
   8 POKE i,y
   9 NEXT x
```

Figura 9

```
   5 LOAD "garotog"CODE USR "a"
  10 FOR x=31 TO 0 STEP -1
  20 PRINT AT 10,x;" "
  30 PRINT AT 11,x;" "
  40 PRINT AT 12,x;" "
  50 PAUSE 5
  60 NEXT x
  70 STOP
9000 SAVE "garoto" LINE 5
9010 SAVE "garotog"CODE USR "a"
168
```

Figura 10

com que o próprio programa, após carregado na memória, se encarregue de ler os bytes gravados em fita.

Na figura 10 aparece agora nosso programa exemplo implementado para operar neste método. Vejamos o que fazem suas linhas:

5 – Lê da fita a gravação dos bytes chamada de "garotog", colocando estes bytes na área de definição de caracteres gráficos;

10 a 60 – Movimento do boneco;

70 – Pára a execução do programa;

9000 – Grava o programa com o nome "garoto" e com a instrução de início automático na linha 5;

9010 – Grava os bytes da área de definição de caracteres gráficos, em fita, chamando esta gravação de "garotog".

Assim, para gravar o programa e os bytes, digite 9000.

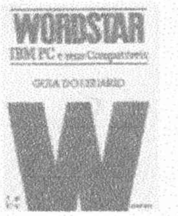
Observe que a linha 9000 comanda um início automático do programa, após leitura, a partir da linha 5. Esta linha, por sua vez, tem um LOAD para os caracteres gráficos. Temos, então, um processo de leitura encadeada com programa e bytes. Simples, não?

Finalizando o artigo, aqui vai uma dica que pode salvá-lo de futuros dissabores: cuidado durante a edição dos caracteres. Ao sair e retornar ao utilitário (o que é comum se fazer para testar os caracteres em movimento, por exemplo), quando for digitar UDG 2, não digite UDG apenas, sem o número, pressionando o ENTER logo em seguida (isto pode acontecer se a tecla 2 não for pressionada com força suficiente), pois o equipamento entende UDG puro como sendo UDG 0, e faz, num piscar de olhos, a definição dos caracteres em português, perdendo todas as definições já feitas.

José Rafael Sommerfeld trabalha há mais de cinco anos com programação de microcomputadores. É colaborador de MS desde o número 22 e autor de um dos livros da série SOFTWARE, destinado às aplicações do microcomputador em casa. Atualmente, presta assessoria em microinformática.

McGRAW-HILL - INFORMÁTICA

Lançamentos



Wordstar – Guia do Usuário – IBM PC e seus Compatíveis – de Richard Curtis. Guia completo sobre o Wordstar, para micros de 16 bits, incluindo acentuação em português. Ferramenta indispensável na vida do profissional moderno.
CÓDIGO 450.085 Cr$ 49.000

WORDSTAR – Guia do Usuário – de W. Ettlin, Versão 8 bits – CP/M. Leitura indispensável às secretárias, executivos em geral, profissionais de jornalismo, direito etc. É um guia completo sobre este software.
CÓDIGO 450.134 Cr$ 49.000

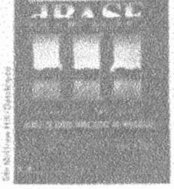
Apple II – Guia do Usuário – Incluindo Apple II plus e IIe, de Lon Poole. O "best seller" de todos os tempos na área de informática apresenta sua segunda edição com o Apple IIe – versão 64 k.
CÓDIGO 450.328 Cr$ 75.000

Manual do Apple Macintosh – de William Sanders. Primeiro no Brasil, este livro apresenta as principais características e conceitos do Mac – incluindo seus ícones, comandos e operações. Inclui também dois capítulos especiais sobre *Gráficos* e *Processamento de Palavras*.
CÓDIGO 450.410 Cr$ 38.000

TK 90X – Programas para Jovens Programadores – de Linda Hurley. É um guia contendo instruções sobre como programar o novo microcomputador TK 90X. Contendo uma coletânea de programas compreensivos, cada capítulo inicia com um programa curto e de fácil assimilação que produz um resultado imediato.
CÓDIGO 450.188 Cr$ 35.000

dBASE II – Para Principiantes – de Alan Freedman. Este é um verdadeiro guia visual de introdução ao dBASE II. Foi projetado para familiarizar os leitores com este software, seus comandos e suas capacidades. É ideal para os leitores iniciantes na computação, por ser rico em ilustrações de comandos, gráficos e exemplos em geral.
CÓDIGO 450.159 Cr$ 75.000

Framework – Aplicações Financeiras, Administração e Negócios – de Clark Fishbach. Diretamente destinado àqueles interessados em usar o Framework de imediato, ou seja, programas práticos para entrar na máquina sem maiores complicações. Suas aplicações também abrangem as áreas de Marketing e Gerência de Pessoal, entre outras, que podem se utilizar deste software para implementar ou aumentar sua produtividade. Um guia ideal para os executivos e profissionais que não têm tempo para aprender.
CÓDIGO 450.156 Cr$ 75.000



dBASE III – Banco de Dados para Todas as Aplicações – de Robert A. Byers. Neste livro, o autor introduz os leitores na nova geração de gerenciamento de banco de dados: o dBASE III. O texto fornece uma visão geral de todas as otimizações possíveis de se obter com a utilização deste software, juntamente com informações valiosas sobre como criar arquivos, manipular dados e aplicar seus comandos. É um texto ideal e interativo para os usuários deste programa.
CÓDIGO 450.044 Cr$ 75.000

Framework para Principiantes – Guia do Usuário – de Bill Harrison. Este livro foi projetado para colocar a grande flexibilidade deste software à disposição de seus usuários. No texto, o leitor encontrará as explicações sobre suas principais características: *Folha Eletrônica, Banco de Dados, Processamento de Tipo "Outlining"* (um verdadeiro "processamento de idéias"), pacote de comunicações e muito mais, além de numerosos exemplos de cada uma de suas funções.
CÓDIGO 450.186 Cr$ 75.000

dBASE II – Aplicações Comerciais – de Robert A. Byers. Este livro traz demonstrações de como o dBASE II pode ser utilizado para se programar uma variedade de aplicações comerciais. Com uma seção completa de referências, este guia oferece uma visão geral de inúmeros tópicos de interesse, desde o software como linguagem de programação, a técnicas de depuração, com ênfase no uso do programa em atividades profissionais/comerciais.
CÓDIGO 450.043 Cr$ 75.000

PEDIDO DE LIVROS

Gostaria de receber os seguintes livros:

Nome:

Endereço:

Cidade: Estado: Cep:

Anexo Cheque ☐ Reembolso Postal ☐

Rua Tabapuã, 1105 - CEP 04533 - Tel. (011) 881-8604 - Itaim Bibi - São Paulo - Brasil

* Em todas as livrarias ou diretamente/solicite catálogos

Operação Resgate

Salvador Rubens Marinho

O programa Operação Resgate roda no micro TK90X, requerendo 16 Kb de memória RAM, e foi construído inteiramente em BASIC. Após ter sido recuperado (LOAD), o programa entrará rodando e construindo automaticamente os caracteres gráficos especiais.

Operação Resgate é um jogo de habilidade, com efeitos visuais e sonoros. Neste jogo você é o piloto de uma nave espacial em missão de resgate, tendo para isto que posicionar sua nave exatamente sobre o seu companheiro que se encontra no solo, por tempo indeterminado e em local inconstante, resgatando-o com seu raio de transporte.

A cada 25 resgates você terá direito a uma nova rodada, e a cada nova rodada aumentará o nível de dificuldade. O jogo terminará se você não conseguir resgatar 25 companheiros, mesmo tendo disparado 100 raios.

Para controlar a nave, pode-se usar o joystick ou o teclado, sendo a tecla **6** para deslocar para a esquerda, **7** para deslocar para a direita e **0** para disparo (com o joystick utilizar o botão).

A primeira tela do programa contém as instruções do jogo, e já na segunda tela temos um cenário espacial, contendo em sua margem inferior contadores de disparo, companheiros recolhidos e o número da atual rodada.

Na digitação, deve-se tomar cuidado com os valores contidos em DATA e com as variáveis em geral. Para acessar os caracteres especiais rode as linhas 6000 e 6010, introduzindo posteriormente as demais linhas. Acione o cursor em modo gráfico e digite **a** para se obter a nave e **b** para o companheiro.

Para gravar o programa, dê em comando direto um GOTO 6050. Convém verificar se o programa foi bem gravado antes de apagá-lo da memória. Para recuperar o programa, volte a fita para o início e dê o comando LOAD "resgate" (a palavra resgate deve ser digitada em letras minúsculas).

Salvador Rubens Marinho é projetista de máquina na indústria Engesa-elétrica, unidade Amplimag, onde programa e opera um computador APPLE II. Possui atualmente um computador TK90X.

```
  1 GOSUB 4000
  2 GOSUB 6000
  3 LET rr=39: PAPER 0: BORDER
1: INK 7: CLS
  4 FOR q=1 TO 50: LET c=INT (R
ND*255)+1: LET l=INT (RND*175)+1
: PLOT INK 5; BRIGHT 1;c,l: NEXT
 q
  5 LET ro=1: FOR r=1 TO 7: CIR
CLE 24,154,r: NEXT r
  6 LET dp=0: LET re=0: PRINT
 INK 6;AT 20,2; BRIGHT 1;"DISPAR
O N.=";dp;"  ";AT 21,2;"RECOLHI
DOS =";re;"  ";AT 20,20;"rodada
 ";ro; INVERSE 1;ro; INVERSE 0
  7 IF rr<4 THEN LET rr=39
  8 LET a$=" ▲": LET b$=" ▞ ":
DIM l(2): DIM c(2): LET l(1)=0:
LET l(2)=18: LET c(1)=16: LET c
(2)=16
  9 REM
    ┌──────────────────────┐
    │ SALVADOR RUBENS MARINHO │
    │ DATA....13/08/85        │
    │ 16 K                    │
    │ PROGRAMA  "resgate"     │
    │ PARA  TK 90X E COMPAT.  │
    └──────────────────────┘
 10 PLOT 0,20: DRAW 255,0: DRAW
 0,(175-20): DRAW -255,0: DRAW 0
,(80-175)
 12 LET r=INT (RND*rr)+1: PRINT
 AT l(2),c(2);" ": LET c(2)=INT
 (RND*30)+1: IF c(2)<2 OR c(2)>28
 THEN GOTO 12
 13 IF r<4 THEN GOTO 12
 14 LET a=0
 20 IF c(1)<2 THEN LET c(1)=2
 25 IF c(1)>28 THEN LET c(1)=28
 30 PRINT BRIGHT 1; INK 6;AT l(
1),c(1);a$
 40 IF INKEY$="6" THEN LET c(1)
=c(1)-1: SOUND ,.013,39
 50 IF INKEY$="7" THEN LET c(1)
=c(1)+1: SOUND ,.013,39
 60 IF INKEY$="0" THEN GOSUB 10
00
 70 LET a=a+1: IF a=7 THEN PRIN
T AT l(2),c(2);" ": GOTO 12
 75 PRINT INK 7; BRIGHT 1;AT l(
2),c(2),b$
 100 GOTO 20
1000 PLOT OVER 1;c(1)+8+12,144
DRAW BRIGHT 1; OVER 1;0,(58-175)
: PAUSE 18: PLOT OVER 1;c(1)+8+1
2,144: DRAW OVER 1; INK 0;0,(58-
175)
1005 LET dp=dp+1: PRINT INK 6; B
RIGHT 1;AT 20,14;dp
1010 SOUND ,.025,0: LET c=INT (RN
D*255)+1: LET l=INT (RND*175)+1:
 PLOT BRIGHT 1;c,l
1020 IF c(1)=c(2) THEN GOTO 2000
1028 IF dp=100 AND re<25 THEN GO
TO 3000
1030 RETURN
2000 SOUND ,.8,-30: FOR n=l(2) TO
 l(1)+1 STEP -1: PRINT BRIGHT 1;
AT n,c(1)+1;"▚": SOUND .025,2: P
RINT AT n,c(1)+1;" ": NEXT n: LE
T re=re+1: PRINT INK 6; BRIGHT 1
;AT 21,15;re
2020 IF dp=100 AND re<25 THEN GO
TO 3000
2030 IF re=25 THEN LET ro=ro+1
2035 IF re=25 THEN LET rr=rr-5
2040 IF re=25 THEN PRINT INK 0;A
T l(1),c(1);" ": GOTO 6
2200 GOTO 12
3000 PRINT AT 5,0; INVERSE 1;"V
OCE CHEGOU NA RODADA N.";ro; PR
INT FLASH 1;ro; PRINT AT 7,0; I
NVERSE 1;" MAS, NESTA RODADA NAO
 RESGATOU  25 COMPANHEIROS";
FLASH 1;"       NOVAMENTE ? (S/N
)"
3010 IF INKEY$="s" THEN SOUND 2,
-28: RUN 3
3020 IF INKEY$="n" THEN STOP
3030 GOTO 3010
4000 BORDER 6: PAPER 0: INK 6: C
LS : PRINT AT 1,0; INVERSE 1;"
     OPERACAO RESGATE
4010 BRIGHT 1: PRINT "    Voce
esta no comando de uma  nave de
 resgate. Sua missao e' recolher
 seus companheiros com  raios de
 tracao."''
4020 PRINT "    Apos 25 resgates
, iniciara  uma nova rodada, ma
s se em 100  disparos voce nao c
onseguir os  25 salvamentos, a m
issao sera  encerrada."
4030 PRINT '"    Use joystyck ou
 teclado  "<6>" para a esquerda"
"<7>" para a direita"
"<0>" dispara."; INVERSE 1; FLA
SH 1;"    pressione qualquer tec
la  "; PAUSE 0: RETURN
6000 FOR x=0 TO 15: READ s: POKE
 USR "a"+x,s: NEXT x
6010 DATA 36,24,60,126,102,255,2
55,66,152,80,60,16,40,40,40,54
RETURN
6050 SAVE "resgate" LINE 1
```

Operação Resgate

O microcomputador TK-90X oferece muitos recursos sonoros e musicais. Leia este artigo e aprenda a explorar melhor este potencial.

Componha no TK-90X

Pierluigi Piazzi

Uma das descobertas científicas mais sensacionais deste século foi a da especialização dos hemisférios cerebrais: o lado esquerdo de seu cérebro gerencia as atividades lingüísticas e matemáticas. O lado direito, por sua vez, é responsável pelo processamento das relações espaciais e musicais.

Numa analogia grosseira, poderíamos dizer que o lado esquerdo é digital e técnico e o direito é analógico e poético.

Fazer música no computador é uma tarefa desafiante e gratificante, pois exige a utilização de ambos os lados, simultaneamente.

Vejamos, portanto, alguns rudimentos de como se iniciar nesta "artécnica" maravilhosa.

AS NOTAS MUSICAIS

Se você olhar para o teclado de um piano, verá a repetição periódica de um mesmo padrão de teclas brancas e pretas (figura 1). Cada módulo de sete teclas brancas (e cinco pretas) é denominado "oitava".

As sete brancas estão associadas às sete notas musicais (figura 2), e cada nota está fisicamente associada a uma freqüência de vibração.

Se você medir a freqüência de vibração sonora do DÓ, por exemplo, e multiplicá-la por, aproximadamente, 1,122, obterá o RÉ. Da mesma forma, do RÉ se obtém o MI, ou do FÁ o SOL.

Entre o MI e o FÁ e entre o SI e o DÓ seguinte, porém, o fator de multiplicação é aproximadamente 1,059.

Dizemos que o intervalo musical entre o DÓ e o RÉ (correspondente ao fator 1,122) é de um TOM e entre o MI e o FÁ é de um SEMITOM.

As teclas pretas do piano servem, justamente, para fornecer uma nota intermediária entre as que estão separadas por um TOM inteiro. Na tecla preta, entre o DÓ e o RÉ, você toca o DÓ # (# = sustenido que é igual a um Semitom acima) ou o REb(b =bemol que é igual a um Semitom abaixo).

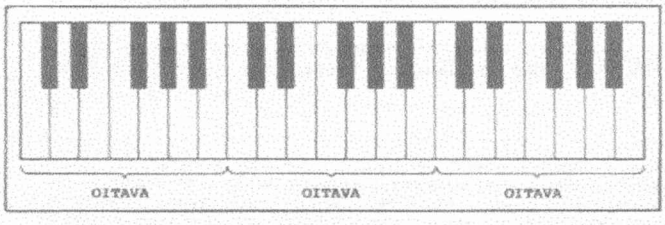


Figura 1

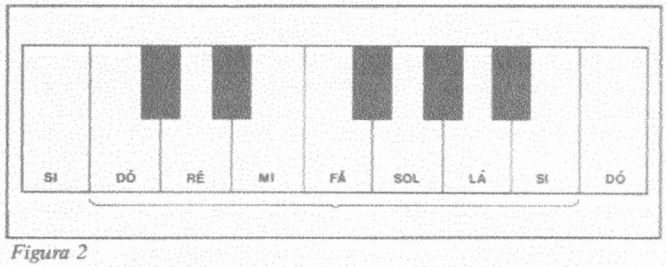


Figura 2

```
10 LET S=2↑(1/12)
20 LET T=S↑2
30 PRINT "S=";S
40 PRINT "T=";T
```

Figura 3

Nesta altura, seu lado direito (artístico) deve estar reclamando: "Afinal de contas, viemos aqui para tocar música ou para fazer cálculos?".

Em compensação, seu lado esquerdo (científico) deve estar indagando, curioso: "De onde vêm esses 1,122 e 1,059?"

Vamos então atender aos dois: quando você toca o DÓ de uma oitava e o DÓ da oitava seguinte, seu cérebro "ouve" DÓ nos dois casos (só que um mais grave e outro mais agudo), porque o segundo tem exatamente o dobro da freqüência do primeiro.

Partindo de um DÓ e tocando todas as teclas até o DÓ seguinte, estamos subindo uma escada (escala) musical em que cada degrau tem a altura de um Semitom.

Como, até dobrar a freqüência, você deve subir 12 degraus, o fator S de multiplicação, para passar de um degrau a outro será tal que:

SxSxSxSxSxSxSxSxSxSxSxS =2, ou seja, $S^{12} = 2$.

Para descobrir o valor de S (Semitom) e de S^2 (Tom) basta, então, rodar o programinha da figura 3.

TOCANDO UM POUCO

No TK-90X (quase totalmente compatível com o ZX Spectrum da Sinclair), a instrução BASIC que envia um sinal sonoro ao alto-falante da TV é SOUND, cuja sintaxe é: SOUND D,A. Onde D é a duração da nota em segundos e A é a sua altura em Semitons, sendo o valor zero atribuído ao DÓ central.

Por exemplo, SOUND 2,0 tocará o DÓ central durante dois segundos, enquanto que SOUND .5,2 tocará o RÉ da mesma oitava durante meio segundo (note que entre o DÓ e o RÉ há um intervalo de dois Semitons).

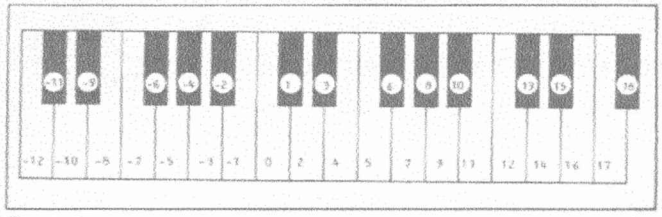


Figura 4

Se você está familiarizado com um piano ou órgão, poderá se orientar pela figura 4. Você já deve ter percebido que para subir de uma oitava, basta acrescentar 12 ao A da instrução.

Para elaborar um programa que toque música, não é necessário escrever um SOUND para cada nota a ser emitida, pois isso seria muito trabalhoso.

Se todas as notas tiverem a mesma duração, podemos colocar a instrução SOUND dentro de um laço e armazenar a seqüência das notas nos DATA.

Para exemplificar, digite o programinha da figura 5, que corresponde à mão esquerda de um BOOGIE-WOOGIE de oito tempos (para quem não sabe, o BW é o ancestral direto do ROCK). Note que as linhas 100, 110 e 130 são idênticas, bastando editar a primeira e alterar apenas o número para obter as outras.

Digamos que após tocá-lo, seu lado direito o tenha achado um pouco lento e sem graça. Tudo bem, agora o lado esquerdo vai alterar o programa para satisfazer o seu lado artístico; mude, por exemplo, a linha 30 para: SOUND .2,A. Melhorou, não?

Seu lado direito ainda acha que está faltando um pouco de "champignon"? Fácil, acrescente a linha: 32 SOUND .05,A e duvido que você não comece a marcar o tempo com os pés.

Se bater um vento cor-de-rosa e você quiser algo mais romântico, elimine as linhas 30 e 32, substituindo-as por:

30 FOR J= 0 TO 4
32 SOUND .1,A+12*J
34 NEXT J

Agora, é só dar asas à sua imaginação e você descobrirá mil variações e truques para melhorar qualquer música.

LENDO O PENTAGRAMA

Se você obtiver a partitura da música que você quer tocar, é extremamente simples traduzi-la para que o computador possa entendê-la. Basta seguir a codificação da figura 6.

Obviamente, toda vez que aparecer o símbolo # (sustenido) ao lado de uma nota, basta acrescentar um à sua altura. Analogamente, aparecendo b(bemol), devemos subtrair um.

Os valores de D, na figura 6, são apenas uma sugestão e deverão ser ajustados em função do andamento da música: num Allegro Vivace, você os reduzirá; num Andante Maestoso, eles deverão ser ampliados.

OS RECURSOS DO TK-90X

Ao contrário de seus irmãos mais velhos, o TK-85 e o CP-200, o TK-90X é capaz de subir escada e mascar chiclete, simultaneamente. Isso significa que você pode processar som e imagem em conjunto.

```
10 FOR I=1 TO 40
20 READ A
30 SOUND .5,A
40 NEXT I
50 RESTORE
60 GOTO 10
100 DATA -12,-8,-5,-3,-2,-3,-5,
-8
110 DATA -12,-8,-5,-3,-2,-3,-5,
-8
120 DATA -7,-3,0,2,3,2,0,-3
130 DATA -12,-8,-5,-3,-2,-3,-5,
-8
140 DATA -10,-8,-7,-6,-5,-7,-8,
-10
```

Figura 5

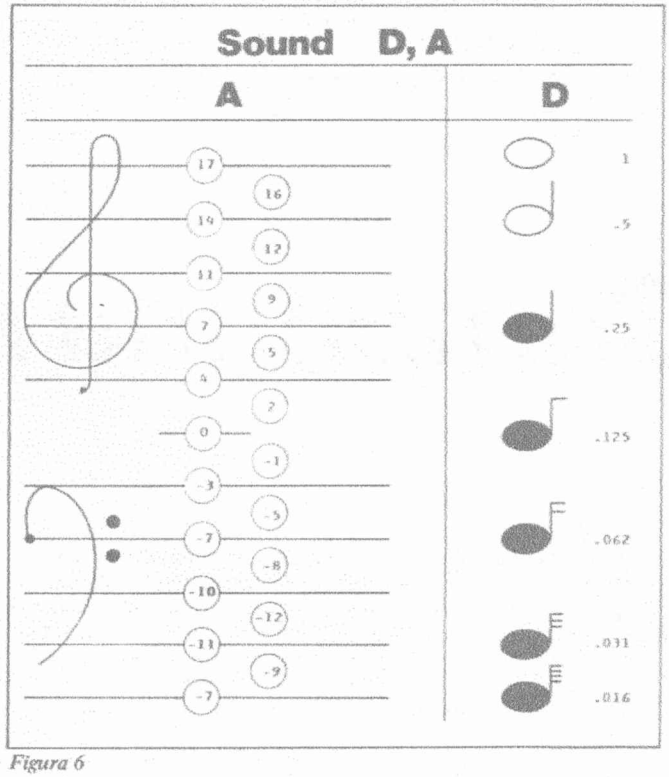


Figura 6

```
10 FOR I=0 TO 20*RND
20 SOUND .01,-20
30 PLOT 100+I*RND*5,100+I*RND*

40 NEXT I
50 GOTO 10
```

Figura 7

Se você retomar o programa da figura 5 e acrescentar as linhas:

28 PLOT 2*I,2*A+50
55 CLS

você poderá "visualizar" a subida e a descida das notas, enquanto a música está sendo tocada.

Ao adaptar o ZX Spectrum, a Microdigital perdeu a oportunidade de implementar a pausa musical. Você pode eliminar este inconveniente, inserindo na seqüência um PAUSE (lembrando que PAUSE 60 equivale a um segundo), ou um laço inútil (FOR X= 1 TO 200 : NEXT I toma aproximadamente um segundo de processamento). Uma sucessão de: SOUND .05,-60 também corresponde a um silêncio, cuja duração será determinada pelo número de instruções assim enfileiradas.

O fato de existir apenas um canal de som, constitui-se também numa limitação bastante séria, mas que não é muito importante quando a finalidade do som não é música, mas sim ruídos.

Digite o programa da figura 7 e entre no também fascinante mundo do barulho!

O texto deste artigo foi adaptado do livro *Curso de Música no TK-90X*, que será lançado, brevemente, pela Aleph Publicações.

Pierluigi Piazzi é formado em Química e Física pela USP. Foi professor do curso Anglo Vestibulares e já editou inclusive a Revista Microhobby. Atualmente, dirige a Aleph Publicações, responsável por muitos sucessos na área editorial, dada a sua orientação didática, voltada para os usuários de micros pessoais.

S-7000/TRS-80 Disco

Formatando em MBASIC

Faça a formatação de campos monetários do tipo XXX.XXX.XXX,00 com esta sub-rotina em MBASIC, na qual o valor a ser formatado será transferido para **VI$** e o valor formatado estará em **VF$**. Teste esta dica e observe sua rapidez e performance sob o compilador BASIC:

```
490 PRINT CHR$(12):DEFDBL V
500 INPUT "VALOR ";V:REF%=SGN(V):IF REF%=0 THEN VF$="
0,00":GOTO 530
510 VI$=STR$(ABS(INT(V))):VI$=RIGHT$(VI$,9):GOSUB 10000
520 REM PODE IMPRIMIR VF$, GRAVAR OU ARMAZENAR EM VARIAVEL
530 PRINT VF$:GOTO 500
10000 VF$="                ,00":VI$=STRING$(9-LEN(VI$),32)+VI$:C1$=LE
FT$(VI$,3):C2$=MID$(VI$,4,3)
10010 IF C1$="   " THEN MID$(VF$,1,4)="    " ELSE MID$(VF$,1,4)=
C1$+"."
10020 IF C2$="   " THEN MID$(VF$,5,4)="    " ELSE MID$(VF$,5,4)=
C2$+"."
10030 MID$(VF$,9,3)=RIGHT$(VI$,3):IF REF%=-1 THEN VF$="-"+VF$ EL
SE VF$=" "+VF$
10040 RETURN
```

Madias -MG

Linha SINCLAIR

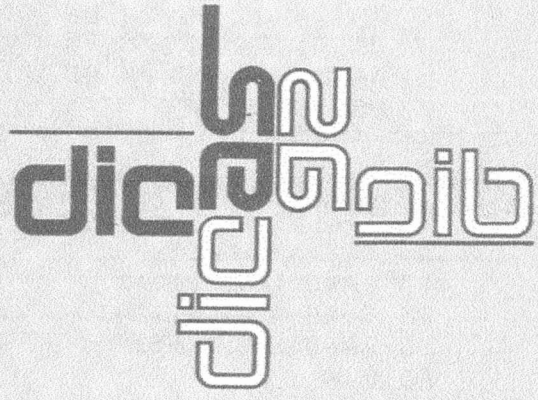
Teste as linhas

Use o **REM** e o **EDIT** para checar se uma determinada linha do programa está funcionando. Primeiro, edite a linha que será testada; depois, coloque um **REM** na frente desta linha, como neste exemplo:

```
10 IF X=0 THEN GOTO 290....(Edit)
20 LET A=1
.
.
.
10 IF X=0 THEN GOTO 290...(Edicao)
10 REM IF X=0 THEN GOTO 290...(New Line)
```

Isto fará com que a linha (10, neste exemplo) seja ignorada pelo computador, permitindo que se avalie em que a ausência desta linha implicará para o programa. Para repor novamente a linha no programa, basta editá-la e deletar a instrução **REM**.

Fábio Antonio R. Corrêa-SP



Se você tem pequenas rotinas e programas utilitários realmente úteis tomando poeira em seus disquetes ou fitas cassetes, antecipe-se aos piratas e trate de divulgá-los. Envie-os para a REDAÇÃO DE MICRO SISTEMAS – SEÇÃO DICAS: Av. Presidente Wilson, 165/grupo 1210, Centro, Rio de Janeiro, RJ, CEP 20030. Não se esqueça de dizer para qual equipamento foram desenvolvidos. Desta forma, sua descoberta poderá ser útil para muitos e muitos, em vez de desmagnetizar-se com o tempo em suas fitas e disquetes...

Linha TRS-80

Otimizando ainda mais

Na Seção Dicas de MS nº 32 (maio de 84), a dica "Otimização de caracteres duplos" transformava o vídeo de 32 CPL para 64 CPL, sem que fosse necessário limpar a tela. Agora eu apresento a minha sugestão, que é um pouco mais simples mas produz o mesmo efeito. Basta digitar em BASIC:

PRINT CHR$ (28);

que a surpresa será a mesma. Para confirmar, troque a linha 40 do exemplo publicado na dica de MS nº 32 para:

40 PRINT CHR$ (28);

Rui Bastos-RS

Linha SINCLAIR

Efeito dinâmico em string

Faça, com esta sub-rotina em BASIC, um efeito rolante na impressão de *strings:* ela deve ser chamada com a variável X, contendo a linha em que será impressa a *string*, e com a própria *string* **A$**.

```
5000 LET TAM=LEN A$+(LEN A$)/2
5010 FOR V=1 TO LEN A$
5020 FOR W=TAM TO V STEP -1
5030 PRINT AT X,W;A$(V);" "
5040 NEXT W
5050 NEXT V
5060 RETURN
```

Ricardo Souza Viana-BA

Linha SINCLAIR

Ilusão de ótica

Seja enganado pela rapidez do Assembler com esta dica: primeiro, digite a linha:

1 REM. . . . (30 caracteres). . .

Em seguida, entre com os códigos em hexadecimal, usando, por exemplo, o Modo A do Monitor Assembler, publicado em MS nº 23, pág. 10:

```
16514 - 00 2A 0C 40 23 06 16
        0E 21 7E FE 76 28 04
        3A 82 40 77 23 0D 20
        F3 10 EF C9
```

Depois de entrar com os códigos, digite este programa em BASIC:

```
5 POKE 16514,6
10 RAND USR 16515
15 POKE 16514,134
20 RAND USR 16515
25 RUN
```

Rode esta dica e aceite o desafio de tentar descobrir qual o sentido do movimento apresentado na tela. Para obter o mesmo efeito, faça a tentativa de trocar as linhas 5 e 15 do programa em BASIC por:

```
5 POKE 16514,137
15 POKE 16514,138
```

Jean Marcelo de Oliveira-SP

Linha APPLE

Representação do seno

Construa a representação gráfica de um seno no seu vídeo, que lentamente vai mudando de cor, com este programa em BASIC:

```
10 REM SENAMO
20 HGR:HCOLOR=3:FOR K=0 TO 279:HPLOT K,90:NEXT
30 FOR J=1 TO 6:HCOLOR=J
40 FOR I=2.1 TO 25 STEP .1:T=I*10+10:U=200*SIN(I)/I+90
50 HPLOT T,U-5 TO T,U+5
60 NEXT:NEXT:GOTO 30
```

Observe ainda o que pode ocorrer se alterarmos a linha 50 para

```
50 HPLOT T-5,U-5 TO T+5,U+5
```

ou então para

```
50 HPLOT T+5,U-5 TO T-5,U+5
```

Para rodar esta dica no TK-2000, é só incluir a linha

```
55 SOUND I*J,15
```

Armando O. Cavanha Fº-RJ

Linha SINCLAIR

Escrita vertical

Escrever na horizontal no micro todo mundo sabe. Aprenda agora com esta dica simples a escrever na vertical:

```
10 DIM A$(1,21)
20 LET A$(1)="ESCRITA VERTICAL"
30 FOR L=1 TO 21
40 PRINT AT L,0;A$(1,L)
50 NEXT L
```

Rui Maurício R. Ribeiro-RJ

Linha SINCLAIR

Mini ordenador

A maioria dos programas ordenadores fazem a mesma coisa: dimensionam uma matriz de variáveis onde são introduzidos, de qualquer maneira, os elementos (números ou letras) a serem ordenados, e só depois da entrada de todos os dados é que começa a ordenação (em geral, muito tediosa) do arquivo inteiro.

Esta dica é diferente por duas razões: não existe pré-dimensionamento, pois o arquivo é uma *string* que *cresce* a cada novo dado introduzido; e a listagem dos elementos ordenados é quase instantânea, já que logo após a introduçao do dado ele é ordenado no arquivo, e ao fim da entrada dos dados a listagem está, como sempre esteve, pronta.

Em termos de limitação, a pergunta "tamanho máximo?" (feita na linha 20 do programa) refere-se ao número de caracteres que cada dado a ser trabalhado possui, pois o arquivo propriamente dito só é limitado pela quantidade de memória disponível. É interessante notar que esta dica também permite ordenar números, sendo que para se voltar o valor numérico da *string* basta usar o comando **VAL**. Dois avisos antes de digitar o programa: para não atrapalhar o **SCROLL**, o vídeo só mostra até a 32ª posição da linha digitada; e para listar o programa é só digitar. (ponto).

```
10 FAST
20 PRINT "TAM.MAXIMO?"
30 INPUT N
40 LET R$=""
50 INPUT I$
60 IF I$="." THEN GOTO 210
70 FOR X=1 TO N-LEN I$
80 LET I$=I$+" "
90 NEXT X
100 LET I$=I$( TO N)
110 SCROLL
120 IF N(=32 THEN PRINT I$
125 IF N)32 THEN PRINT I$( TO 32)
130 FOR X=1 TO LEN R$ STEP N
140 IF I$)=R$(X TO X+N-1) THEN GOTO 180
150 LET P$=I$
160 LET I$=R$(X TO X+N-1)
170 LET R$(X TO X+N-1)=P$
180 NEXT X
190 LET R$=R$+I$
200 GOTO 50
210 CLS
220 FOR X=1 TO LEN R$ STEP N
230 PRINT R$(X TO X+N-1)
240 NEXT X
```

Edgar SantosRocha-RJ

Linha TRS-80 (III)

Descubra a senha

A melhor forma de *achar* uma senha mestra desconhecida é mudar a senha para outra conhecida. Como? Simples, siga estas etapas e comprove:

1 – Coloque no drive **0** um disquete com senha conhecida (SENHA).
2 – Digite **PROT : 0 (PW) < ENTER >**.
3 – O micro perguntará **"SENHA MESTRA?"**.
4 – Digite SENHA < ENTER > .
5 – O computador fará outra pergunta: **"NOVA SENHA MESTRA?"**.
6 – Não responda ainda.
7 – Retire o disquete do drive **0** e coloque o disquete em que a senha é desconhecida.
8 – Responda agora ao computador, digitando: **SENHA < ENTER >** .

Ari Morato – MG

Linha SINCLAIR

Contando bytes

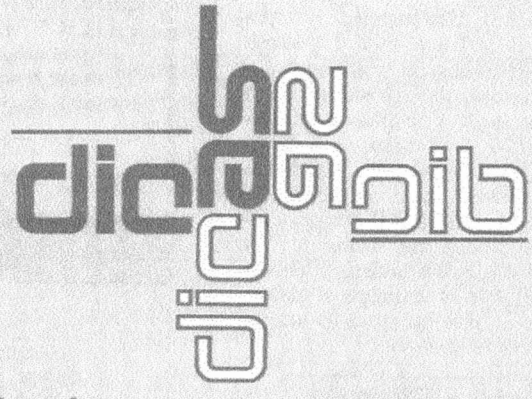
Coloque esta dica no final de seus programas e saiba, a qualquer momento, o total de memória ocupada:

```
9997  LET BYTES=PEEK 163
      96+256*PEEK 16397-
      16509
9998  PRINT "ESTE PROGRA
      MA OCUPA";
9999  PRINT BYTES-115;"
      B"
```

E se no programa houver variáveis dimensionadas, para incluí-las na contagem é só trocar a linha 9997 para:

```
9997  LET BYTES=PEEK 16
      404+256*PEEK 1640
      5-16384
```

Luís Carlos de Almeida Serpa – RJ



Se você tem pequenas rotinas e programas utilitários realmente úteis tomando poeira em seus disquetes ou fitas cassetes, antecipe-se aos piratas e trate de divulgá-los. Envie-os para a REDAÇÃO DE MICRO SISTEMAS – SEÇÃO DICAS: Av. Presidente Wilson, 165/grupo 1210, Centro, Rio de Janeiro, RJ, CEP 20030. Não se esqueça de dizer para qual equipamento foram desenvolvidos. Desta forma, sua descoberta poderá ser útil para muitos e muitos, em vez de desmagnetizar-se com o tempo em suas fitas e disquetes...

Linha GERAL

Imprimindo cruzeiros

Esta sub-rotina em BASIC roda em computadores compatíveis com as linhas TRS-80, PC-IBM e I-7000 (da Itautec), sendo que foi originalmente feita em um equipamento Nexus, da Scopus. Simples, mas útil, esta dica deve ser colocada, em caso de programas longos, logo no início, pois o BASIC procura sub-rotinas a partir da linha inicial, ou seja, de cima para baixo no programa.

Para usá-la, defina primeiro a variável de entrada igual ao valor a imprimir, depois desvie o programa para esta sub-rotina e comande a impressão da variável de saída. Esta variável terá sempre um comprimento fixo e, dentro dela, as casas vazias serão preenchidas com brancos. Caso se queira reduzir o seu tamanho, pode-se fazê-lo, antes da impressão, usando para isso os comandos **CR$=RIGHT$** (sendo que **CR$** refere-se ao tamanho desejado), mas é preciso ter cuidado para não truncar o número.

É interessante observar ainda algumas características desta sub-rotina: o conteúdo da variável original não é destruído; o arredondamento e a notação normal dos números são mantidos, sendo que os negativos precedidos do sinal "–"; e se o programa gerar somente números positivos, as quatro últimas linhas desta dica não serão executadas, podendo-se, portanto, eliminá-las.

Evaldo Mendes – DF

```
10000 REM ROTINA PARA IMPRESSAO EM FORMATO CRUZEIRO SEM DECIMAIS
10010 REM DIMENSAO MAXIMA DE ENTRADA = 999.999.999.999.999 OU 14
 INTEIROS + DECIMAIS
10020 REM VARIAVEL DE ENTRADA: CR#    VARIAVEL DE SAIDA: CR$    *
LEN(CR$=20)*
10030 REM ------------------------------------------------------
10040 IF CR#=0 THEN CR$=STRING$(20," "):RETURN
10050 IF CR#>0 THEN X=.5 ELSE X=-.5
10070 CR$=STR$(FIX(CR#+X)):CR$=STRING$(16-LEN(CR$)," ")+MID$(CR$
,2,LEN(CR$))
10080 FOR X=3 TO 15 STEP 4
10090 IF MID$(CR$,X,1)=" " THEN XX=32 ELSE XX=46
10100 CR$=MID$(CR$,1,X)+CHR$(XX)+MID$(CR$,X+1,LEN(CR$)-X):NEXT
10110 CR$=" "+CR$:IF CR#>0 THEN RETURN
10120 FOR X=19 TO 1 STEP-1
10130 IF MID$(CR$,X,1)=" " THEN CR$=LEFT$(CR$,X-1)+"-"+RIGHT$(CR
$,20-X):X=1
10140 NEXT
10150 RETURN
```

Linha SINCLAIR

Letreiro no micro

Uma boa dica para ilustrar seus jogos e programas: faça um letreiro luminoso, do tipo usado em *outdoor*, na tela do seu micro. Para facilitar, use o comando **E** do MICRO BUG para criar uma linha **REM** (nesta dica a linha **REM** tem 34 caracteres), e o comando **M**, também do MICRO BUG, para inserir os blocos em Assembler. A variável **A$**, que será definida quando o programa começar a rodar, contém a frase ou o texto que se quer colocar no letreiro.

ROTINA EM ASSEMBLER

```
16514   2A 0C 40 23 01 21 16 CD
16522   94 40 C5 06 00 09 C1 10
16530   F6 C9 E5 C5 06 1F 7E 23
16538   4E 2B 71 23 10 F9 77 C1
16546   E1 C9
```

ROTINA EM BASIC

```
 20 FAST
 30 INPUT A$
 40 FOR H=1 TO 31
 50 PRINT AT 8,H;"■";AT 12,H;"■
"
 60 NEXT H
 65 SLOW
 70 FOR H=1 TO LEN A$
 80 PRINT AT 8,31;"■";AT 12,31;
"■"
 90 PRINT AT 10,31;A$(H)
100 RAND USR 16514
110 NEXT H
120 GOTO 70
```

Ricardo Nunes Bandarra — RS

Linha TRS-80 (III)

Ressuscite arquivos

O pessoal que leu o artigo "O NEWDOS que não está nos manuais" (MS nº 31, pág. 22) e ficou triste por não poder aplicar no TRSDOS/DOS 500 a técnica de recuperação de arquivos apagados pelo comando **KILL**, não precisa mais ficar desanimado: é possível modificar o TRSDOS/DOS 500, de forma que o comando **KILL** passe a zerar apenas o primeiro byte-tal como no NEWDOS-, em vez de zerar todos os bytes da entrada do diretório de um arquivo. Para isso, basta utilizar o seguinte **PATCH**:

PATCH * 3 (ADD = 4FAE,FIND = 3600D554, CHG = CBA61808)

Após esta alteração, é só seguir as instruções do artigo de MS nº 31, pág. 22, e (se o espaço do disco anteriormente colocado ainda não tiver sido realocado), ressuscitar os arquivos.

Roberto Quito de Sant'Anna — RJ

Linha SINCLAIR

Duplique programas

Coloque esta rotina em linguagem de máquina no topo da memória (RAM-TOP) e duplique qualquer programa por mais *fechado* que esteja: primeiro, digite **POKE 16388, 241** e depois dê **POKE 16389, 127** e **NEW**. Em seguida, entre com os seguintes códigos:

32753 CD 44 03 CD
32757 BB 02 2C 28
32761 FACD 23 0F
32765 C3 F5 02

e dê **NEW** novamente. Para duplicar o programa, digite **FAST, RAND USR 32753** e carregue o programa. Assim que o programa estiver carregado, a tela ficará escura. Coloque então a fita para gravar o programa, posicione o gravador para gravar, e aperte uma tecla qualquer do micro. Aguarde alguns segundos, pois o micro estará gravando o programa *fechado*.

E para duplicar para *high-speed*, e só trocar a linha **32765** por: **32765 C3 D5 20**.

Uma última observação: se o programa *fechado* foi gravado pela instrução **SAVE**, logo após a gravação em duplicata ocorrerá erro C/0. Mas não é preciso se preocupar, pois esse problema não afetará o carregamento do programa na fita.

Renato Moreira Prado — SP

Linha SINCLAIR

Complementando o SCROLL

Na Seção Dicas de MS nº 25, pág. 126, foi publicada uma dica muito boa: 'SCROLL em Assembler". Só que não funciona exatamente como o **SCROLL** da máquina, pois não posiciona o próximo **PRINT** na última linha. Para tornar aquele **SCROLL** uma simulação idêntica ao do equipamento, coloque este adendo:

SCROLL ANTERIOR	COMPLEMENTO
16514 - 2A 0C 40 E5 11 21 00 19 D1 01 D6 02 ED B0	01 00 15 CD F5 08 C9

Desta forma pode-se mudar a linha e a coluna da próxima impressão, alterando-se os bytes:

```
16520  coluna. Varia de 0 a 32
16521  linha.  Varia de 0 a 23
```

Um lembrete final: para inicializar a rotina, deve-se chamar **RAND USR 16514**.

Alexandre R. Goidanich — RS

Linha SINCLAIR

Surpresa em BASIC

Além de criar um efeito especial na tela do micro, esta dica relata uma interessante experiência: tudo começou quando aprendi a *printar* a tela através de **POKE** no arquivo de imagem. A partir daí, comecei a fazer isto em BASIC e depois em Assembler. E quando tentei dar um **SCROLL** em Assembler, em vez da imagem ir para o lado, ela desceu! Depois de analisar cuidadosamente o erro, descobri o que estava ocorrendo: o programa arrastava um **NEW LINE** da linha anterior para a tela, e o micro, por isso, *pensava* que havia uma linha a mais!

Mas vamos à prática desta descoberta com a rotina (que pode ser colocada em qualquer programa em BASIC, desde que se faça, obviamente, modificações nos números das linhas deste exemplo):

```
1 PRINT,,
10 FOR A=1 TO 255
20 PRINT CHR$ A;
30 NEXT A
40 FOR A=1 TO 183
50 PRINT CHR$ A;
60 NEXT A
70 LET A=PEEK 16396+256*PEEK 16397
80 FOR B=1 TO 24
90 POKE A+B,118
100 NEXT B
110 FOR B=24 TO 1 STEP -1
120 POKE A+B,0
130 NEXT B
```

Antes de verificar o impressionante efeito e, principalmente, a rapidez desta dica em BASIC, é preciso saber de alguns detalhes:

- As linhas 10 até 60 desta rotina podem ser substituídas para *printar* qualquer coisa.
- A linha 70 contém o valor de **D-FILE**, que é o endereço do início do arquivo de imagem.
- As linhas 80, 90 e 100 formam um *loop* para que se coloque, nas primeiras posições da tela, 24 **NEW LINEs**, fazendo com que o micro *pense* que existem 48 linhas!
- As linhas 110, 120 e 130 fazem o micro retornar ao normal.
- NÃO DÊ **BREAK** durante a execução da rotina, pois será muito difícil fazer com que o micro volte ao normal, mesmo dando **CONT**.
- Para fazer com que a imagem não desça totalmente, é só diminuir, igualmente, os *loop* das linhas 80 e 100.

Por fim, tente acrescentar estas linhas (mas não use o comando **CLS** entre as linhas 100 e 110):

101 PRINT AT 1,0; "9 SPACES ESTOU DE VOLTA 9 SPACES"
102 PRINT "32 SHIFT GRAPHICS SPACE"

Eduardo Kyoshi Nishie — SP

dicas

Se você tem pequenas rotinas e programas utilitários realmente úteis tomando poeira em seus disquetes ou fitas cassetes, antecipe-se aos piratas e trate de divulgá-los. Envie-os para a REDAÇÃO DE MICRO SISTEMAS — SEÇÃO DICAS: Av. Presidente Wilson, 165/grupo 1210, Centro, Rio de Janeiro, RJ, CEP 20030. Não se esqueça de dizer para qual equipamento foram desenvolvidos. Desta forma, sua descoberta poderá ser útil para muitos e muitos, em vez de desmagnetizar-se com o tempo em suas fitas e disquetes...

Linha SINCLAIR

Destacando o texto

Aí vai um recurso interessante para chamar a atenção sobre um texto em seus programas: rode esta dica e veja a mensagem centralizada e com uma faixa em vídeo inverso passando, letra por letra:

```
10 LET X$="MICRO SISTEMAS"
20 FOR N=1 TO LEN X$
30 PRINT AT 10,7+N;X$(N)
40 NEXT N
50 FOR N=1 TO LEN X$
60 LET Y$=CHR$ (CODE X$(N)+128)
70 PRINT AT 10,7+N;Y$
80 NEXT N
90 GOTO 20
```

Como é fácil perceber, a *string* **X$** pode conter outra mensagem qualquer, mas deve-se respeitar os valores máximos dos códigos (ou então fazer as modificações necessárias no programa).

Cláudio S. Matsuoka — PR

Linha SINCLAIR

Destravando fácil

Muita gente pensa que usando o comando **USR 757** está realmente bloqueando um programa, mas a utilização desta pequena rotina permite que se destrave qualquer programa: **FAST**

POKE USR 837, 256

Max Oliver — SP

Linha SINCLAIR

SCROLL *multidirecional*

Esta dica apresenta três tipos de **SCROLL**: para os lados, para cima e para baixo, e os dois simultaneamente. Vamos à primeira sub-rotina, que faz com que metade da tela desloque-se para a esquerda e a outra metade para a direita. Para executá-la, dê **RAND USR 16514**.

```
16514 - 0E 10 2A 0C 40 06 18 C5
16522 - 01 0F 00 23 E5 D1 23 C5
16530 - ED B0 C1 97 12 09 03 E5
16538 - D1 2B ED B8 0E 12 09 C1
16546 - 10 E5 11 00 07 1B 7B B2
16554 - 20 FB 0D 20 D5 C9
```

Agora é a vez da segunda sub-rotina, que permite o deslocamento da imagem para as extremidades superior e inferior da tela. Rode-a com **RAND USR 16560**.

```
16560 - 06 0C C5 ED 5B 0C 40 21
16568 - 21 00 19 01 6B 01 ED B0
16576 - 06 20 2B 36 00 10 FB 2B
16584 - 01 8B 01 09 11 21 00 EB
16592 - 19 EB ED B8 C1 11 00 07
16600 - 1B 7B B2 20 FB 10 D3 C9
```

E para obter um interessante **SCROLL** multidirecional, utilize conjuntamente as duas sub-rotinas anteriores, mas antes faça as seguintes alterações: coloque o código **01** nos endereços **16515, 16549, 16561** e **16598**; e o código **00** nos endereços **16550** e **16599**. Para colocar este **SCROLL** em funcionamento, crie um *loop* em BASIC do tipo:

```
10 FOR N=1 TO 12
20 RAND USR 16560+USR 16514
30 NEXT N
```

Gilson Roberto Viana — PR

Linha SINCLAIR

Números invertidos

Imprima qualquer número ao contrário (inclusive número negativo ou fracionário) na tela do micro com esta dica, onde **N** é o número a ser invertido e deve ser definido antes de se entrar na rotina.

```
9800 LET A=LEN STR$ N
9801 LET S$=STR$ N
9802 FOR X=1 TO A
9803 PRINT CHR$ (CODE S$(X)+128);
9804 NEXT X
9805 RETURN
```

Marco André A. Vieira — MG

Linha SINCLAIR

Super SCROLL

Para quem está sempre à procura de novas incrementações para colocar em seus jogos e programas, aí vai uma boa dica: um **SCROLL** por linha, em que uma linha vai andando para a direita e a outra para a esquerda, sucessivamente. Para rodar, primeiro entre com este bloco em Assembler:

```
16514 : 2A 0C 40 23 01
        42 0C E5 C5 06
        1F 7E 23 4E 2B
        71
16530 : 23 10 F9 77 C1
        E1 C5 06 00 09
        C1 10 EA 2A 0C
        40
16546 : 23 01 21 00 09
        01 42 0B C5 01
        1F 00 09 06 1F
        4E
16562 : 2B 7E 23 77 2B
        10 F9 71 C1 C5
        06 00 09 C1 10
        E8 C9
```

E para testar, passe a rotina usando o seguinte programa:

```
  1  REM (ROTINA...)
 10  LET A=16514
 20  LET A$=""
 30  IF A$="" THEN INPUT
     A$
 40  SCROLL
 50  PRINT A,A$(1 TO2)
 60  LET A=A+1
 70  LET A$=A$(3 TO)
 80  GOTO 30
 85  LIST
 90  RAND USR 16514
100  GOTO 90
```

Depois de passar o programa, digite **GOTO 85** e comprove o resultado. Agora, uma surpresa para os curiosos: mudem o efeito com **POKE 16533, 112** e **POKE 16569, 112** e vejam o resultado.

Rogério C. Bastos — SP

Linha SINCLAIR

CLS *incrementado*

Em vez de usar o tradicional **CLS** em seus programas, coloque esta rotina:

```
16514 : AF 32 90 40 2A 0C
        40 23 06 16 0E 20
        7E FE C0 20

16530 : 02 36 00 23 0D 20
        F5 23 10 F0 3A 90
        40 3C FE 40

16546 : 20 02 C6 40 32 90
        40 FE C0 C2 86 40
        C9
```

Agora troque a linha do **CLS** por **RAND USR 16514** e entre, na linha seguinte, com **PRINT AT 0,0**. Rode esta dica e veja a diferença...

Rogério C. Bastos — SP

Se você tem pequenas rotinas e programas utilitários realmente úteis tomando poeira em seus disquetes ou fitas cassetes, antecipe-se aos piratas e trate de divulgá-los. Envie-os para a REDAÇÃO DE MICRO SISTEMAS — SEÇÃO DICAS: Av. Presidente Wilson, 165/grupo 1210, Centro, Rio de Janeiro, RJ, CEP 20030. Não se esqueça de dizer para qual equipamento foram desenvolvidos. Desta forma, sua descoberta poderá ser útil para muitos e muitos, em vez de desmagnetizar-se com o tempo em suas fitas e disquetes...

Linha TRS-80

Sensação na tela

Provoque um efeito sensacional no vídeo do seu micro, com esta dica que faz a mensagem surgir, caráter por caráter, do canto direito da tela até a posição desejada. Teste e confira.

```
>CLEAR 200
10 CLS:INPUT"MENSAGEM";M$:CLS:M$
=M$+"      "
20 FOR K=1 TO LEN(M$):FOR I=63 T
O K STEP -1:PRINT@I,MID$(M$,K,1)
;"  ":NEXT I,K:GOTO 20
```

Antonio Carlos L. de Santana — SP

Linha SINCLAIR

Centralize o texto

Aí vai uma dica para os iniciantes aprenderem a centralizar os textos no seu micro:

```
10 LET A$="CENTRALIZACAO"
20 PRINT TAB 15-LEN A$/2;A$
```

ou

```
20 PRINT AT 10,15-LEN A$/2;A$
```

Ruy Maurício R. Ribeiro — RJ

Linha SINCLAIR

Datilografia no micro

Escreva na tela do seu equipamento como se estivesse batendo a máquina com esta dica: primeiro, digite o seguinte programa:

```
1 REM .....(40 pontos).....
10 FOR A=16514 TO 16554
20 INPUT B
25 PRINT A;"=";B
30 POKE A,B
40 NEXT A
```

Depois, dê **RUN** e entre com estes códigos em decimal:

16514	205	16531	8
	187		8
	2		254
	68		12
	77		200
	121		6
	254		1
	255		17
	62		255
	0		255
	40		33
	244		50
	205		55
	189		25
	7		56
	126		253
	205		16
			248
			24
			219
			248
			24
			216

Após a entrada dos códigos, elimine as linhas de 10 a 40, ficando apenas com a linha 1 (linha **REM**), e rode esta dica com **RAND USR 16514**. É bom lembrar que deve-se usar o espaço usado normalmente para separar as palavras (como em datilografia). Quando terminar de *bater o seu texto* no micro, basta apertar **SHIFT** e **BREAK** para encerrar.

Manoel Silva Rodrigues — RJ

Linha TRS-80

Gráficos em string

Enfeite seus programas ou logotipos com este programa, em que o micro pergunta o gráfico desejado (dentre os códigos de 128 até 191), e dentro de uma string da linha 20 surgirá então, repetido 64 vezes, o gráfico pedido. Por falar em linha 20, é necessário deixar 64 espaços em branco nesta linha.

```
4 CLS
5 INPUT"DIGITE O CARACTER DESEJA
DO (128 -> 191) ";T
10 IF T<128 OR T>191 THEN GOTO 5
20 LET G$="
                                "
30 G=VARPTR(G$)
40 PA=PEEK(G+2)*256+PEEK(G+1)
50 FOR I=PA TO PA+63
60 POKE I,T
70 NEXT I
80 CLS
90 PRINT"LISTE O PROGRAMA E OBSE
RVE A LINHA 20"
```

Miguel Angelo Henley G. Fº — RJ

Linha TRS-80

Tela dançante

Coloque esta dica em linguagem de máquina no monitor residente do seu micro e veja a tela tremer como se estivesse a ponto de explodir:

```
7000   3E 04 D3 EC
       3E 00 D3 EC
       C3 00 70
```

E para quem desejar fazer este mesmo efeito, só que bem mais lento, é só entrar com esta linha em BASIC:

```
10 OUT 236,0:OUT 236,4:GOTO 10
```

Alexandre Costa Gaeta — RJ

Linha SINCLAIR

INKEY$ *em* FAST

Quem pensava que só podia ver **INKEY$** em **SLOW**, agora verá que estava enganado: esta dica fácil permite visualizar o **INKEY$** em **FAST** também! E para isso, basta colocar antes de qualquer linha com **INKEY$**, uma linha que tenha: **RAND USR 681**. Muito simples, não?

Adriano Pascoal Pereira — RJ

Linha SINCLAIR

Simulando DEFM

Eis uma dica bastante útil para o pessoal que lida com Assembler: ela simula, com rapidez e segurança, a instruçao **DEFM**. Nesta rotina que daremos a seguir, as linhas 9030 e 9035 mudam a posição do **PRINT** do arquivo de imagem para uma área qualquer da RAM (designada pela variável **E**), fazendo com que o **PRINT** funcione como um **POKE** mais poderoso do que o normal.

```
9000  PRINT "ENDEREÇO ?"
9005  INPUT E
9010  PRINT "MENSAGEM ?"
9015  INPUT M$
9020  LET X=PEEK 16398
9025  LET Y=PEEK 16399
9030  POKE 16398,E-256*
      INT(E/256)
9035  POKE 16399,INT(E/
      256)
9040  PRINT M$
9045  POKE 16398,X
9050  POKE 16399,Y
9055  PRINT "MENSAGEM D
      EFINIDA."
```

Frederico dos Santos Liporace — RJ

Linha SINCLAIR

Implemente READ *e* DATA

Não raro vemos em tabelas de compatibilidade de BASIC, entre equipamentos que usam lógicas diferentes, a afirmação de que não há correspondente das funções **READ** e **DATA** nos micros da lógica Sinclair. A rigor, de um teclado para o outro, não há mesmo. Mas MICRO SISTEMAS já publicou como conseguir simular estas funções (MS nº 25, pág. 8, na matéria "DATA, READ e RESTORE no TK"). Eis agora uma outra rotina que também permite a utilização destas instruções nos micros da linha Sinclair.

Para executá-la, faça primeiro a entrada de dados da listagem 1 com os comandos **E** e **M** do MICRO BUG. Ao terminar de digitar os códigos da listagem 1, digite **P 4082, 40F9** para obter o resultado **2BDD**. Se o resultado obtido não for este, para e reveja os códigos, pois algum deve estar errado.

Na listagem 2, o comando **RAND USR 16514** corresponde a **READ A$**, ou seja, transfere tudo o que está na linha **REM TO** (tudo mesmo, até a vírgula) para **A$**. O Comando **REM TO** corresponde ao **DATA**, e é a linha onde deverão ser escritos os **A$**.

Para facilitar ainda mais a compreensão, a listagem 2, que é um programa-exemplo desta dica, tem nas suas principais linhas as seguintes *tarefas:* linha 1 — linha reservada para o programa em LM que está contido na linha 7; linha 2 — dimensionada **A$** com tantos caracteres quantos os do maior número da linha 7; linha 4 — coloca em **A$** um dos valores da linha 7; linha 7 — contém os valores que **A$** assumirá (um a um); linha 11 — executa a rotina em LM contida na linha 1.

LISTAGEM 1

```
4082  3A 29 75 FE 00 20 15 21
408A  FE 40 7E FE EA 28 03 23
4092  18 F8 23 7E FE 0F 20 F7
409A  23 22 2A 75 2A 10 40 7E
40A2  FE 46 28 03 23 18 F8 23
40AA  5E 23 56 23 EB 19 22 2E
40B2  75 ED 53 2C 75 21 29 75
40BA  34 2A 2A 75 7E FE 76 20
40C2  07 21 29 75 36 00 18 0B
40CA  FE 1A 20 15 23 22 2A 75
40D2  ED 5B 2C 75 2A 2E 75 ED
40DA  52 CB EB 36 00 23 EB 18
40E2  F3 23 22 2A 75 ED 5B 2C
40EA  75 2A 2E 75 ED 52 C8 EB
40F0  77 23 22 2C 75 18 C2
```

LISTAGEM 2

```
1 REM ************************
**
2 LET A$=CHR$ 0+CHR$ 0+CHR$ 0
3 FOR A=16639 TO 16666
4 RAND USR 16514
5 POKE A,VAL A$
6 NEXT A
7 REM TO 42,12,64,35,14,22,6,32,
126,203,127,32,4,203,255,24,2,20
3,191,119,35,16,241,35,13,200,24
,234
8 FOR A=0 TO 139
9 PRINT CHR$ (128+INT (RND*64));
" ";
10 NEXT A
11 RAND USR 16639
12 FOR A=1 TO 10
13 NEXT A
14 RUN 11
```

Linha TRS-80

Quantos bytes?

Saiba exatamente quantos bytes de memória um programa ocupa com esta simples linha:

```
0 CLS:ZZ$="TAMANHO DESTE PROGRAM
A:===) ":ZZ=(PEEK(16634)*256+PEE
K(16633))-(PEEK(16549)*256+PEEK(
16548))-103:PRINT@260,ZZ$;ZZ;" B
YTES":FOR I=1 TO 2000:NEXT I
```

O procedimento também é bem fácil: depois de colocar esta linha 0 no seu programa, chame-a com **RUN 0**. E se por acaso seu programa já possua uma linha 0, não tem importância, é preciso apenas arrumar um lugar para colocar esta dica.

Dirceu Pivatto-RS

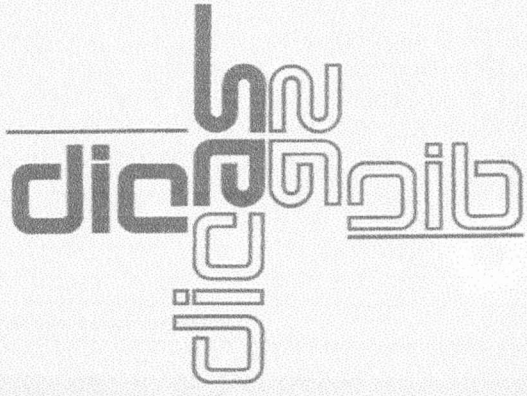
Linha SINCLAIR

Arte oriental

Entre no mundo das **mil e uma noites**: coloque esta dica no seu micro e veja uma série interminável de belos tapetes persas **voando** na tela.

```
4 REM TAPETES PERSAS
6 REM AUTOR: RUY C. GONCALVES
8 REM AGOSTO DE 1984 - PORTO ALE
GRE-RS
10 CLS
20 LET T$=" "
30 FOR I=1 TO 5
40 LET A=INT (RND*11)
50 LET B=INT (RND*11)+128
60 LET T$=T$+CHR$ A+CHR$ B
70 NEXT I
80 FOR I=1 TO 64
90 PRINT T$;
100 NEXT I
110 FOR I=1 TO 100
120 NEXT I
130 GOTO 10
```

Ruy C. Gonçalves-RS



Se você tem pequenas rotinas e programas utilitários realmente úteis tomando poeira em seus disquetes ou fitas, antecipe-se aos piratas e trate de divulgá-los. Envie-os para a REDAÇÃO DE MICRO SISTEMAS — SEÇÃO DICAS: Av. Presidente Wilson, 165 — Grupo 1210 — Centro — Rio de Janeiro — RJ — CEP 20030. Não esqueça de dizer para qual equipamento foram desenvolvidos. Desta forma sua descoberta poderá ser útil para muitos, em vez de desmagnetizar-se com o tempo em suas fitas e disquetes.

Linha SINCLAIR

Dissolvendo o vídeo

Veja a tela do seu micro ir se dissolvendo aos poucos (igualzinho à dica *Derretendo o vídeo*, para a linha TRS-80, publicada em MS nº 34, pág. 72), com esta dica em Assembler. Para usá-la, entre primeiro com o MICRO BUG para criar uma linha **REM** com 48 bytes. Depois, digite estes códigos:

```
16514  FD 36 21 00 2A 0C 40 0E
16522  18 06 20 23 7E FE 80 20
16530  03 3E 3F 77 B7 28 05 35
16538  FD 36 21 01 10 ED 23 0D
16546  20 E7 FD 7E 21 B7 20 D8
16554  C3 F5 08 80 B2 AF A7 80
```

Agora, em vez de dar um **CLS**, digite **LET L= USR 16514** e comprove que efeito! Por fim, um detalhe muito bom para o pessoal que vai incrementar esta rotina nos seus programas: ela pode ser colocada em qualquer posição de memória.

Mário José Bittencourt-SP

Linha TRS-80

Faça uma reta

Os usuários da linha TRS-80 agora já podem traçar uma reta na tela do seu equipamento, é só digitar esta dica:

```
2000 INPUT X1,Y1,X,Y:CLS
2010 IF X-X1=0 M=500 ELSE M=(Y-Y
1)/(X-X1)
2020 IF ABS(M)>1 THEN 2070
2030 FOR X2=X1 TO X STEP SGN(X-X
1)
2040 SET(X2,Y1+M*X2-M*X1)
2050 NEXT X2
2060 END
2070 FOR Y2=Y1 TO Y STEP SGN(Y-Y
1)
2080 IF M<>500 SET((Y2-Y1)/M+X1,
Y2) ELSE SET(X1,Y2)
2090 NEXT Y2
2100 END
```

Henrique Monteiro Cristóvão-ES

TK 2000

Alfabeto grego

Reproduza os caracteres do alfabeto grego no seu TK-2000 com este programa que, inclusive, pode ser usado como sub-rotina em um editor de textos.

```
20 DIM RA(24)
30 FOR U=1 TO 24:READ RA(U):NEXT
 U
40 DATA 8192,8320,8448,8576,8704
,8832,8960,9088,8232,8360,8488,8
616
50 DATA 8744,8872,9000,9128,8272
,8400,8528,8656,8784,8912,9044,9
168
60 HGR:HCOLOR=3:HOME:I=1:L=9
70 VTAB 5:HTAB 1:PRINT"LETRAS GR
EGAS:":PRINT:FOR U=1 TO 39:PRINT
"-";:NEXT U
80 GOSUB 300:N=0
82 FOR C=11 TO 27 STEP 2:N=N+1
84 IF N=27 THEN 98
86 IF C=27 THEN L=L+2
88 IF C=27 THEN 82
90 FOR J=I TO I+7
92 BA=RA(L)+C-1
94 POKE BA+1024*(J-I),M(J)
96 NEXT J:I=I+8:NEXT C
98 VTAB 18:HTAB 18:PRINT"ORLANDO
 MOURA":PRINT TAB(18);"BELEM,AGO
STO/84"
100 END
300 REM ** SIMBOL **
302 DIM M(200)
304 FOR U=1 TO 192:READ M(U):NEX
T U
306 DATA 0,64,44,18,18,44,0,0,48
,72,72,36,68,66,50,2,0,68,42,16,
16,16,16,0,48,8,16,60,34,34,28,0
,0,24,36,4,8,4,36,24,0,26,38,34,
34,34,32,32
308 DATA 4,56,4,8,16,32,36,24,0,
60,66,66,126,66,60,0,0,0,128,4,4
,20,12,0,2,18,10,7,10,82,98,0,2,
5,8,8,20,36,66,0,0,18,30,18,18,8
2,34,2
310 DATA 32,64,66,34,34,18,12,0,
2,28,2,12,2,50,76,32,0,0,28,34,3
4,34,28,0,0,126,36,36,36,36,0,0,
0,24,36,36,20,4,4,0,64,60,2,26,3
4,34,28,0
312 DATA 0,252,18,16,16,20,24,0,
0,0,226,37,36,36,36,24,0,48,74,7
4,60,8,8,8,2,69,40,16,40,68,130,
0,130,85,84,56,16,16,16,0,0,0,2,
65,65,73,54,0
320 RETURN
```

Orlando Moura-PA

Linha SINCLAIR

Curvas incríveis

Faça uma surpresa aos seus amigos com esta dica que traça curvas mirabolantes: é só rodá-la, responder às perguntas que o programa faz e aguardar um pouco porque esta dica é lenta.

```
1 FAST
10 PRINT "STEP?"
20 INPUT S
30 PRINT "A INICIAL?"
40 INPUT Q
50 PRINT "A FINAL?"
60 INPUT W
70 CLS
110 FOR A=Q TO W STEP S
120 PRINT AT 21,25;"A=";A;
130 FOR X=1 TO 43
140 LET Y=(A/2)*(EXP(X/A)+EXP(-X
/A))
150 LET D=24+A-Y
160 LET G=X+21
170 LET H=43-X
180 IF G>63 OR H<0 THEN GOTO 230
190 IF D<2 OR D>42 THEN GOTO 230
200 PLOT G,D
210 PLOT H,D
220 NEXT X
230 NEXT A
```

Adriano Pascoal Pereira-RJ

TK 85

Programas em segurança

De repente, um descuido e... pronto! Lá se foi aquele programa que era tão bom... Quantas vezes isso já aconteceu? Para evitar esta triste situação, tenha sempre cópias de segurança dos seus melhores programas. Esta dica faz para você exatamente isto, e ainda tem uma vantagem: transmite uma mensagem, ao final da cópia ou carregamento, informando se está tudo ok ou não. Ela só tem uma restrição: só pode ser usada no TK-85, pois os comandos das linhas 40 e 70 são especiais deste equipamento. Agora corra para o seu micro e faça logo um *seguro* dos seus arquivos.

```
10 DIM A$(9000)
20 LET Z=0
30 LET Z$="A"
40 LET A=USR 8305
50 PRINT"CARREGAMENTO";"OK" AND
NOT A;"RUIM" AND A
60 PAUSE 4E4
70 LET A=USR 8288
80 PRINT"COPIA";"OK" AND NOT A;"
RUIM" AND A
```

Pedro Paulo P. Santos-RJ

Linha SINCLAIR

Quanto sobra de memória?

Quando você estiver digitando um programa longo e quiser saber exatamente quanto ainda tem de memória disponível, é só entrar com:

```
PRINT(PEEK 16386+256*PEEK 16387)-(
PEEK 16412+256*PEEK 16413)+87
```

Marcio Yamawaki-SP

Linha APPLE

Desafie a velocidade

Aceite este desafio! Veja se consegue ser tão rápido quanto esta dica:

```
10 REM RAPIDO...RAPIDISSIMO
15 REM ARMANDO OSCAR CAVANHA FILHO
20 DIM X(100),Y(100)
30 TEXT:HGR2
35 X(0)=140:Y(0)=80
37 FOR T=1 TO 6:HCOLOR=T
40 FOR I=1 TO 20
50 X(I)=270*RND(1):Y(I)=180*RND(1)
60 HPLOT X(I-1),Y(I-1) TO X(I),Y(I
)
70 NEXT:NEXT:GOTO 30
```

A linha 40 pode ser modificada (sendo que K pode ter qualquer valor até 100) para:

```
40 FOR I=1 TO K
```

E uma boa surpresa para o pessoal que tem TK-2000: para rodar esta dica é só acrescentar esta linha:

```
65 SOUND I*T,3
```

Armando Oscar Cavanha Fº -RJ

dicas

Se você tem pequenas rotinas e programas utilitários realmente úteis tomando poeira em seus disquetes ou fitas cassetes, antecipe-se aos piratas e trate de divulgá-los. Envie-os para a REDAÇÃO DE MICRO SISTEMAS – SEÇÃO DICAS: Av. Presidente Wilson, 165/grupo 1210, Centro, Rio de Janeiro, RJ, CEP 20030. Não se esqueça de dizer para qual equipamento foram desenvolvidos. Desta forma, sua descoberta poderá ser útil para muitos e muitos, em vez de desmagnetizar-se com o tempo em suas fitas e disquetes...

Linha TRS-80 COLOR

Efeitos especiais com PCLS

Se você está cansado das cores monótonas do fundo da tela quando usa o PCLS na tela gráfica, tente agora este programinha para mudar o seu panorama visual:

```
10 PMODE 3,1:SCREEN 1,1
20 FOR A=0 TO 255
30 POKE 179,A
40 PCLS
50 NEXT A
```

Observação: o valor A do **POKE** determina o *padrão* colorido do fundo da tela.

Marcos K. Watanabe-SP

Linha TRS-80

Organize seus programas

Aí vai um programinha simples para fornecer uma listagem impressa e devidamente organizada do conteúdo de todos os seus disquetes:

```
10 CLS:PRINT@18,"***IMPRESSAO DE D
IRETORIOS***"
20 PRINT@82,STRING$(29,"-")
30 PRINT@320,"ENTRE O NOME DO DISC
O :":PRINT:LINE INPUT X$
40 CLS:LPRINT"DIRETORIO DO DISCO==
=>   "X$:LPRINT
50 CMD"Z","ON":CMD"D:0":CMD"Z","OF
F":LPRINT:LPRINT:CLS
60 PRINT@655,"TROQUE O DISQUETE NO
 DRIVE 0 E TECLE"
70 PRINT@715,"<ENTER> PARA CONTINU
AR OU <BREAK> PARA PARAR"
80 LINE INPUT X$:GOTO 10
```

E se você possui dois drives e deseja trabalhar no drive 1, basta modificar a linha 50 (CMD "D:1") e a mensagem da linha 60.

Roberto Quito de Sant'Anna-RJ

Linha TRS-80 III

Contagem regressiva

Eis uma boa dica para ser implementada em seus jogos (ou mesmo em programas sérios): uma rotina que faz a contagem regressiva de 9... até 0.

```
6 CLEAR500
7 CLS
8 INPUT"CONTAGEM A PARTIR DE";F:F=
9-F
10 CLS
11 G=9-F
12 FORI=0TOG
15 U$=CHR$(128)
20 A$=CHR$(168):B$=STRING$(4,131):
C$=CHR$(148)
22 X$=CHR$(168):Y$=CHR$(148):Z$=ST
RING$(4,131):Z2$=STRING$(4,131)
23 H$=CHR$(170)
40 CLS
42 F=F+1:IFF=1THENX$=U$:Z2$=U$:GOT
052
43 IFF=2THENGOTO52
44 IFF=3THENB$=U$:X$=U$:Z2$=U$:GOT
052
45 IFF=4THENC$=U$:GOTO52
46 IFF=5THENX$=U$:C$=U$:GOTO52
47 IFF=6THENZ2$=U$:X$=U$:Z$=U$:GOT
052
48 IFF=7THENA$=U$:X$=U$:GOTO52
49 IFF=8THENA$=U$:Y$=U$:GOTO52
50 IFF=9THENB$=U$:A$=U$:X$=U$:Z$=U
$:C$=U$:Y$=U$:Z2$=U$:PRINT@412,H$:
PRINT@476,H$;:GOTO57
51 IFF=10THENB$=U$
52 CLS:PRINT@410,A$;:PRINT@415,C$
53 PRINT@474,X$;:PRINT@479,Y$
55 PRINT@475,B$;:PRINT@411,Z$;:PRI
NT@539,Z2$;
57 FORJ=1TO25:OUT255,120+N:OUT255,
121+N:NEXTJ
60 NEXTI
65 CLS:F=0:GOTO8
```

Raimundo Antonio Monteiro-GO

Linha SINCLAIR

Centralizando strings

Crie uma moldura na tela com esta rotina simples que centraliza, rapidamente, strings:

```
1 LET H$="**************************
*********"
2 INPUT M$
3 LET T=LEN M$
4 LET N=(28-T)/2
5 CLS
6 PRINT AT 9,N;H$( TO T+4)
7 PRINT AT 10,N;"*";TAB(34+T)/2;"*
"
8 PRINT AT 11,N;"*";TAB(32-T)/2;M$
;" *"
9 PRINT AT 13,N;H$;( TO T+4)
```

Marcel Gameleira-AL

Linha TRS-80 COLOR

Aumente a velocidade

Caso você ache que o seu micro compatível com a linha TRS-80 Color não está trabalhando suficientemente rápido, digite então:

```
POKE 65495,0   e  <ENTER>
```

Observe agora que o cursor está piscando com mais velocidade. Para desativar este *high-speed*, basta dar um **RESET** e a velocidade voltará ao normal. Experimente, para testar, rodar um programa – de preferência com muitos cálculos – e cronometrar o tempo gasto para executar as contas. Coloque novamente o programa e digite esta dica. Viu a diferença? Um lembrete importante: nunca tente salvar em fita um programa se o computador estiver em *high-speed*, pois a gravação e o programa na fita irão para o espaço...

Marcos K. Watanabe-SP

Linha SINCLAIR

Arquivando a tela

Coloque em seu micro esta rotina em Assembler, que é dividida em duas partes: a primeira, que vai do endereço 16514 até 16526, executa o armazenamento de uma tela inteira a partir do endereço 30000; e a segunda, que começa no endereço 16527 indo até 16540, que coloca imediatamente no vídeo a tela que foi armazenada.

```
16514   2A 0C 40 11 30 75 01 D6
16522   02 23 ED B0 C9 21 30 75
16530   ED 5B 0C 40 13 01 D6 02
16538   ED B0 C9
```

Adálbero Fernandes Guimarães-MG

Linha SINCLAIR

DEFM no Sinclair

A dica abaixo é muito útil para programadores que utilizam o Assembler. Ela simula a instrução DEFM (definição de mensagem) dos editores de código de máquina.

```
9000 PRINT "ENDERECO ?"
9005 INPUT E
9010 PRINT "MENSAGEM ?"
9015 INPUT M$
9020 LET X=PEEK 16398
9025 LET Y=PEEK 16399
9030 POKE 16398,E-256*INT(E/256)
9035 POKE 16399,INT(E/256)
9040 PRINT M$
9045 POKE 16398,X
9050 POKE 16399,Y
9055 PRINT "MENSAGEM DEFINIDA."
```

As linhas 9030 e 9035 mudam a posição PRINT do arquivo de imagem para uma área qualquer, designada pela variável de endereçamento E, tornando o PRINT uma instrução POKE muito mais eficiente.

Frederico dos Santos Liporace - RJ

Linha CP/M

Recuperando programas

Às vezes, durante a digitação de um programa (em BASIC ou em outra linguagem, e até mesmo em alguns programas .COM), involuntariamente voltamos ao sistema e perdemos o nosso programa PERDEMOS??!!;;,,,. NEM SEMPRE. Se você não teve que dar um **MASTER RESET** poderemos recuperar o programa perdido. Para isso, depois que surgir o A > , entre com: **SAVE 0 $ $ $. COM**.(sendo que **$ $ $** é o nome do programa). Depois, digite **$ $ $** que o programa estará de volta.

Domingos Aliperti Jr. - SP

Linha SINCLAIR

Explorando gráficos

Picasso? Van Gogh? Dali? Não chega a tanto, mas bem que dá para inventar incríveis desenhos com este programa em BASIC, que explora bastante os caracteres gráficos do micro. Após rodar o programa, teste os seus *dotes artísticos* modificando os valores das linhas de 20 a 30, ou então alterando o comprimento da variável **A$**.

```
5 CLS
10 RAND
15 POKE 16418,0
20 LET A=INT (RND*10)+1
25 LET B=INT (RND*10)+128
30 LET C=INT (RND*10)+128
35 LET A$=CHR$ A+CHR$ B+CHR$ C
40 FOR T=1 TO 256
45 PRINT A$;
50 NEXT T
55 FOR T=0 TO 20
60 NEXT T
65 RUN
```

Claudio Costa - RJ

Envie suas dicas para a Redação de MICRO SISTEMAS, na Av. Presidente Wilson, 165 — grupo 1210, Centro Rio de Janeiro, RJ. CEP 20030.

Linha SINCLAIR

Atire em movimento

Existe uma certa dificuldade no controle do micro pelo teclado durante a execução de um programa, principalmente quando este é em Assembler e há necessidade de se pressionar várias teclas simultaneamente: foi exatamente isto o que observei ao rodar o programa *Laser Base: o tiro do canhão* (MS nº 28, pág. 62), pois o canhão não reagia ao comando do teclado quando duas (ou mais) teclas eram acionadas. Por isso, tomei a liberdade de reescrever parte da sub-rotina que controla o canhão do *Laser Base*.

```
16666 - 2A B4 40 E5 D1 DB FE CB
16674 - 57 28 06 CB 67 20 10 28
16682 - 28 23 7E FE 76 28 08 3E
16690 - 80 36 A6 12 22 B4 40 DB
16698 - FE CB 47 C2 7C 41 18 06
16706 - 00 00 00 00 00 00
```

Esta sub-rotina, que permite o disparo do canhão em movimento, deve ser colocada a partir do endereço **16666**. E, como ela economiza oito bytes, foi inserido um desvio no endereço **16704** para ignorar, desta forma, os bytes não utilizados.

Gilson Roberto Viana - PR

Linha TRS-80

Ligação PTO a PTO

Esse pequeno programa traça uma linha entre dois ponto dados.

```
100 INPUT"ENTRE COORDENADAS X,Y DO PRIMEIRO PONTO";X,Y
110 INPUT"ENTRE COORDENADAS X,Y DO SEGUNDO PONTO";X1,Y1
120 CLS:IF ABS(X1-X)(ABS(Y1-Y)THEN 160
130 FOR X2=X TO X1 STEP 2*(X1<X)-1
140 SET(X2,(X2-X)*(Y1-Y)/(X1-X)+Y+.5)
150 NEXT:END
160 FOR Y2=Y TO Y1 STEP 2*(Y1(Y)+1
170 SET(.5+(Y2-Y)*(X1-X)/(Y1-Y)+X,Y2)
180 NEXT:END
```

Fernando Osvaldo Oliveira Jr. - SC

Linha SINCLAIR

Datilografia no micro

Essa rotina transforma seu micro numa poderosa máquina de escrever. Para digitá-la crie uma linha REM com 40 caracteres e para usá-la basta o RAND USR 16514.

ATENÇÃO: O break pode ser usado como espaço normalmente e para retornar ao BASIC, pressione SHIFT + BREAK.

```
16514 205
16515 187
16516 2
16517 68
16518 77
16519 121
16520 254
16521 255
16522 62
16523 0
16524 40
16525 244
16526 205
16527 189
16528 7
16529 126
16530 205
16531 8
16532 8
16533 254
16534 12
16535 200
16536 6
16537 1
16538 17
16539 255
16540 255
16541 33
16542 50
16543 55
16544 25
16545 56
16546 253
16547 16
16548 248
16549 24
16550 219
16551 248
16552 24
16553 216
```

Manoel Silva Rodrigues - RJ.

Linha APPLE

Brincando com o vídeo I

Eis algumas interessantes sugestões de efeitos para o seu Apple.
Apague o vídeo da esquerda para a direita:

```
10 FOR A=1 TO 40:POKE 33,A:HOME:FO
R B=1 TO 30:NEXT B,A
```

Da direita para a esquerda:

```
10 FOR A=39 TO 0 STEP-1:POKE 32,A:
POKE 33,40-A:HOME:FOR B=1 TO 30:NE
XT B,A
```

Do centro para os lados:

```
10 FOR A=1 TO 20:POKE 32,20-A:POKE
 33,2*A:HOME:FOR B=1 TO 30:NEXT B,
A
```

De baixo para cima:

```
10 FOR A=24 TO 1 STEP-1:VTAB A:HTA
B 1:CALL-868:FOR B=1 TO 30:NEXT B,
A
```

De cima para baixo:

```
10 FOR A=1 TO 24:VTAB A:HTAB 1:CAL
L-868:FOR B=1 TO 30:NEXT B,A
```

De cima para baixo com risco horizontal:

```
10 INVERSE:FOR A=1 TO 24:VTAB A:HT
AB 1:INVERSE:PRINT SPC(40):VTAB A:
HTAB 1:NORMAL:PRINT SPC(40):NEXT A
:HOME
```

José Amim Cury Nasser - DF

Linha TRS-80

POKE no lugar de SET

Para descobrir o endereço POKE de uma instrução SET (X,Y), use esta pequena fórmula:

```
1000 T=FIX(Y/3)*64+FIX(X/2)+15360
```

onde T = endereço, no vídeo, das coordenadas X, Y.

Fernando Osvaldo Oliveira Jr. - SC

Linha APPLE

Faixas aleatórias

Deixe seu micro criar faixas aleatórias com esse pequeno programa Applesoft.

```
10 REM FAIXAS ALEATORIAS
20 REM ARMANDO OSCAR CAVANHA FILHO
30 HGR2:A=80:S=1:X=3:K=2*RND(1):L=
3:J=K/2
80 X=X+S
90 IF A(8 OR A)152 THEN K=-K
100 HCOLOR=L:HPLOT X,A TO X,A+10
110 HCOLOR=L+1:HPLOT X,A+10 TO X,A
+20
120 HCOLOR=L+2:HPLOT X,A+20 TO X,A
+30
150 IF X(3 OR X)275 THEN GOSUB 400
170 A=A+S*K*SIN(J*X/50):GOTO 80
400 I=I+1:S=(-1)*I:K=3*RND(1):J=RN
D(1)*K
410 L=INT(3*RND(1)+1):IF INT(I/7)=
I/7 THEN 500
420 RETURN
500 TEXT:HGR2:RETURN
```

OPCAO PARA O TK:POKE 160 SOUND A,L

Armando Oscar Cavanha Filho - RJ

Linha APPLE

Brincando com o vídeo II

Experimente estes efeitos sofisticados em seu micro da linha Apple.
Apagando a tela em espiral:

```
5 A=1:C=23
10 INVERSE:HTAB 1:VTAB A:PRINT SPC
(40):NORMAL
20 FOR I=A TO 22:VTAB I:HTAB 40-A:
INVERSE:PRINT "  ":NORMAL:NEXT
30 INVERSE:HTAB 1:VTAB C:PRINT SPC
(40):NORMAL
40 FOR I=A TO 22:VTAB I:HTAB A:INV
ERSE:PRINT "  ":NORMAL:NEXT
50 A=A+1:C=C-1:IF A(15 THEN GOTO 1
0
```

Quadriculando a tela (Preto e Branco):

```
10 INVERSE:FOR A=24 TO 1 STEP-1:VT
AB A
20 FOR B=A-INT(A/2)*2+1 TO 40 STEP
 2:HTAB B:PRINT " ";:NEXT B,A
30 NORMAL
```

Quadricule a tela (várias cores):

```
10 HGR:POKE 49234,0:HCOLOR=3:FOR A
=0 TO 191 STEP 4
20 HPLOT 0,A TO 279,A:NEXT A
30 FOR B=1 TO 137 STEP 8:HPLOT B,1
 TO B,191:HPLOT 279-A,1 TO 279-A,1
91:NEXT B
```

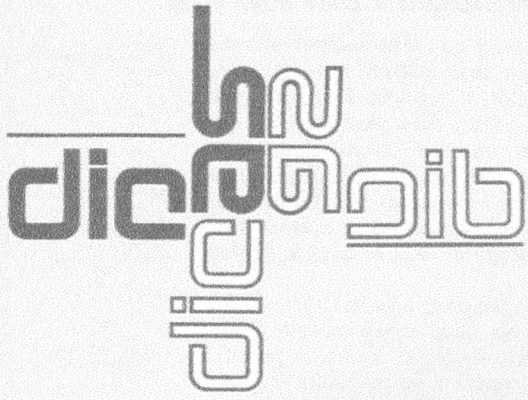
Apague a tela com efeito estrela:

```
5 B=20
10 B=B-1:VTAB B:HTAB 1:CALL-868:IF
 B=5 THEN GOTO 45
20 C=C+1:VTAB C:HTAB 1:CALL-868
30 A=A+1:POKE 32,20-A:POKE 33,2*A:
GOTO 10
45 REM
```

Apague a tela com INVERSE:

```
10 B=1:C=1
20 FOR I=24*A+B TO 24*B+A STEP C:I
F B=1 THEN INVERSE
30 HTAB 1:VTAB 1:PRINT SPC(40):NOR
MAL:NEXT I:IF A=1 THEN GOTO 50
40 A=1:C=-1:B=0:GOTO 20
50 REM
```

José Amim Cury Nasser - DF



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 – grupo 1210. Centro, Rio de Janeiro, RJ. CP. 20030

Linha SINCLAIR

Vire a tela

Inverta a posição dos caracteres do vídeo com a seguinte rotina:

```
16514  2A 0C 40 11 18 79 01 19 03
16523  ED B0 01 30 7C 2A 0C 40 0A
16532  77 08 23 79 FE 17 C2 93 40
16541  78 FE 79 C2 93 40 C9
```

Para testar, use o seguinte programa:

```
10 PRINT "ROGERIO"
20 LIST
30 LIST
40 RAND USR 16514
50 GOTO 40
```

Rogério C. Bastos – SP

Linha TRS-80

Títulos ao centro

Eis uma forma mais simples de centralizar os títulos na tela; dica publicada no nº 23. Use a seguinte modificação:

```
100 A$="TITULO A SER CENTRALIZADO"
110 PRINT TAB (32-LEN(A$)/2)A$
```

Para se formar duas colunas de texto, centralizadas nas duas metades (direita e esquerda) da tela, pode-se mudar o número 32 (linha 110) para 16 (metade esquerda) ou para 48 (metade direita da tela).

José M. Del. Castillo – SP

Linha COLOR

High Speed

Um recurso muito útil em jogos ou programas que têm muitos cálculos numéricos é aumentar a velocidade do equipamento.

Para isto, basta incluir no seu programa uma linha, ou comando direto, do tipo:

```
POKE 65495,0
```

e para voltar ao normal é só usar

```
POKE 65494,0
```

É bom lembrar que em "HIGH SPEED" as escalas PLAY e SOUND são alteradas, e portanto pode ser conveniente incluir uma linha **POKE 65494,0** antes de executar essas instruções. É importante também retornar à velocidade normal antes de gravar (CSAVE) um programa. Agora rode esta seqüência, cronometre o tempo de execução para cada uma delas e veja se "bate" com o timer do micro.

```
POKE 65494,0:TIMER=0:FOR T=1 TO 10000:NEXT:SOUND 128,4:PRINT TIMER/60

POKE 65495,0:TIMER=0:FOR T=1 TO 10000:NEXT:SOUND 128,4:PRINT TIMER/60
```

Cláudio Costa – RJ

Linha SINCLAIR

Números gráficos

Para imprimir na tela qualquer valor numérico em caracteres inversos, use a seguinte subrotina:

```
9800 LET A=LEN STR$ N
9801 LET S$=STR$ N
9802 FOR X=1 TO A
9803 PRINT CHR$ (CODE S$(X)+128);
9804 NEXT X
9805 RETURN
```

onde N é o valor a ser impresso e deve ser definido antes do GOSUB 9800. Essa subrotina funciona inclusive para números negativos ou fracionários.

Marcos André A. Vieira – MG

Linha APPLE

Peripécias no vídeo

Brinque com o vídeo de seu Apple com este pacote de dicas.

Movendo um texto:

```
10 HOME:A=1:B=1:T$="MICRO SISTEMAS
"
20 VTAB 12:HTAB A:PRINT" ";T$;" ";
30 X=PEEK(-16384):IF X)127 THEN GO
TO 70
40 IF B=1 THEN A=A+1:IF A)127 THEN
 B=-1
50 IF B=-1 THEN A=A-1:IF A(2 THEN
B=1
60 GOTO 20
70 REM
```

Fixando a margem inferior do vídeo:

```
10 POKE 35,X  --> ONDE X(0 a 24)
```

Fixando a margem direita do vídeo:

```
ATIVAR => POKE 33,X --> ONDE X=TAMANHO

DESATIVAR => POKE 33,40
```

Fixando o topo do vídeo:

```
ATIVAR => POKE 34,X --> ONDE X(1 a 24)

DESATIVAR => POKE 34,0
```

Apagando a linha:

```
VTAB X:CALL -868 ou

CALL 64541 --> ONDE X=NUM.DA LINHA
```

Movendo o cursor em uma posição para cima:

```
CALL -998 ou CALL 64538
```

José Amim Cury Nasser – DF

Linha APPLE

Marcador

Use essa interessante rotina como um marcador de tempo, ou pontos, etc. Coloque no canto superior da tela a mensagem MARCADOR: 000000 e para maior segurança, coloque um espaço após os 2 pontos (:).

```
16608 - 2A 0C 40 11 0F 00 19 7E
16616 - FE 25 20 05 36 1C 2B 18
16624 - F6 FE 00 C8 34 C9
```

Rode com RAND USR 16608.

Gilson Roberto Viana—PR

Linha TRS-80

Acione os drives

Esta dica aciona os seus drives em Basic com a instrução OUT:

```
OUT 244,1 ----> drive 0
OUT 244,2 ----> drive 1
```

Para acionar os dois drives juntos:

```
OUT 244,3
```

Alexandre Costa Gaeta — RJ

Linha Sharp PC1211

Relógio com alarme

```
10:INPUT "HORA
  INICIAL=";A
20:INPUT "MIN I
  NICIAL=";B
30:INPUT "ALARM
  E? (S OU N)"
  ;C$
40:IF C$="N"
  THEN 70
50:INPUT "HORA
  ALARME=";D
60:INPUT "MIN A
  LARME=";E
70:FOR I=1TO 36
80:PAUSE A;B
90:NEXT I
100:B=B+1
110:IF B=60LET A
  =A+1:B=0
120:IF A=24LET A
  =0
130:IF C$="N"
  THEN 70
140:IF (A=D)*(B=
  E)THEN 160
150:GOTO 70
160:BEEP 5:GOTO
  160
```

Para aqueles que têm compromissos a seguir, e que nem sempre conseguem se lembrar deles, a solução é ter um bom alarme de pulso. Mas para aqueles que não o têm, ou que vivem esquecendo de levá-los, não haverá mais problemas, pois com esse pequeno programa a sua SHARP PC1211 passará a exercer funções de relógio e, caso se queira, de despertador.

O programa pede a hora e minuto iniciais, com os quais o seu relógio começará a marcar o tempo. Também perguntará se se quer o despertador ou não. Com a resposta "S" ele pedirá a hora e minuto desejados para o alarme, e quando o programa chegar a tal instante o beep soará indefinidamente.

Para interromper o programa ou o beep, basta apertar a tecla ON. Convém lembrar que devido aos IFs, desvios, etc. que a rotina do programa possui a precisão do relógio não é a mesma que um cronômetro. Existe um certo erro que se situa mais ou menos em 0,9 segundos por minuto (para cada minuto real a máquina marcará 59 seg.).

Renato Pinto da Cunha—RJ

Linha COLOR

Arquive seus desenhos

Para gravar um desenho ou gráfico diretamente da tela em fita cassete, siga os seguintes passos:

1. Certifique-se de estar em PCLEAR 4.
2. Digite ou Ponha num Programa e rode:

   CSAVEM "nome", 256*PEEK(188), 256*PEEK(188)+6143, 380

3. Para apreciar o desenho entrando na memória, use:

   POKE 359, 57 : SCREEN 1 : CLOADM : POKE 359, 126

Adriano de Arruda Botelho — SP

Linha SINCLAIR

Odnetrevni Sarvalap

Use essa dica para inverter palavras, nomes, etc. Você pode usá-la também para criar um joguinho bastante interessante.

```
10 INPUT A$
20 SCROLL
30 LET K=LEN A$
40 PRINT A$,
50 FOR F=K TO 1 STEP -1
60 PRINT A$(F);
70 NEXT F
80 RUN
```

Marcelo Contart — SP

Linha TRS-80

Vídeo inverso em BASIC

Com esta subrotina em Basic você irá inverter os caracteres gráficos do seu vídeo.

```
    ***** ESTE PROGRAMA GERA 10 LINHAS NO VIDEO *****
10 CLS:FOR I=1 TO 10
20 Y=RND(46):FOR J=0 TO 127:SET(J,Y):NEXT
30 NEXT I

    ***** ESTE PROGRAMA INVERTE O VIDEO *****
100 FOR I=15360 TO 16383
110 IF PEEK(I)=32 THEN C=191:GOTO 145
120 IF PEEK(I)=191 THEN C=32:GOTO 145
130 IF PEEK(I)<=159 THEN C=190-(PEEK(I)-129):GOTO 145
140 IF PEEK(I)>=160 THEN C=159-(PEEK(I)-160)
145 POKE(I),C
150 NEXT:GOTO 100
```

Fernando Osvaldo Oliveira Jr. — SC

Linha TK 2000

Inverta o vídeo em alta resolução

Usando o modo miniassembler (TK-2000), digite a partir do endereço $300 a seguinte rotina:

```
0300-  A0 20     LDY  #$20
0302-  8C 15 03  STY  $0315
0305-  8C 1C 03  STY  $031C
0308-  20 11 03  JSR  $0311
030B-  C8        INY
030C-  C0 40     CPY  #$40
030E-  D0 F2     BNE  $0302
0310-  60        RTS
0311-  A2 00     LDX  #$00
0313-  BD 00 20  LDA  $2000,X
0316-  49 FF     EOR  #$FF
0318-  29 7F     AND  #$7F
031A-  9D 00 20  STA  $2000,X
031D-  E8        INX
031E-  E0 FF     CPX  #$FF
0320-  D0 F1     BNE  $0313
0322-  60        RTS
0323-  00        BRK
0324-  00        BRK
0325-  00        BRK
```

usando o modo monitor, digite os códigos seguintes:

```
0301:A0

030D:C0
```

Para rodar a rotina use os comandos:

```
0301:40

030D:60
```

para inverter a página 2 de alta resolução, alterar no TK-2000:

```
@0300 --) Se no monitor-disassembler
>CALL768 --) Se no BASIC
```

No Apple:

```
0300- A0 20 8C 15 03 8C 1C 03
0308- 20 11 03 C8 C0 40 D0 F2
0310- 60 A2 00 BD 00 20 49 FF
0318- 29 7F 9D 00 20 E8 E0 FF
0320- D0 F1 60
```

Vladimir Oliveira Di Iorio — MG

Linha SINCLAIR

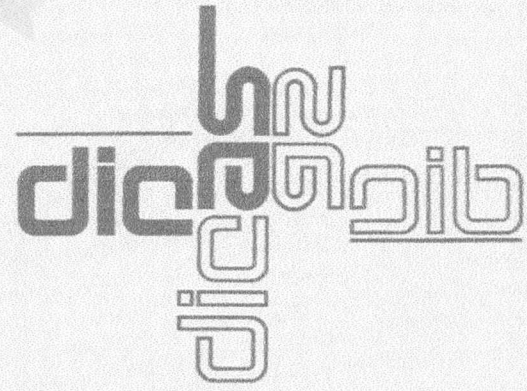
Pausa incrementada

Substitua o comando PAUSE e a pausa entre duas linhas FOR-NEXT por:

```
RAND RND**RND
```

Existe a vantagem de economia de memória e durante a pausa o programa não pode ser parado por BREAK. Para aumentar a duração da pausa, acrescente mais ** RND.

Ricardo Saldanha Freitas — RJ



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 — grupo 1210. Centro, Rio de Janeiro, RJ. CP. 20030

Linha APPLE

Criador END

Esta dica irá auxiliar a operação com arquivos binários em disco. O programa CRIADOR END cria um arquivo seqüencial designado END que, quando chamado por uma instrução EXEC, irá fornecer o endereço inicial e o comprimento do programa em Assembler na memória sob a forma:

```
A end. inicial ,L comprimento (em decimal)
```

Isto torna útil quando se quer gravar o programa binário na memória: o END dará os dados necessários para a instrução BSAVE que exige os parâmetros, endereço inicial e comprimento. Ajudará também quando se quiser editar ou modificar um programa Assembler, pois dará os pontos de referência necessários. Este utilitário foi idealizado para ser utilizado como arquivo seqüencial EXEC porque, ao ser acessado, não comprometerá o atual conteúdo da memória. O programa fornecerá sempre os dados referentes ao ÚLTIMO ARQUIVO BINÁRIO carregado do disco.

```
5 REM
10 REM       CRIADOR END
20 REM    ********************
25 REM    * NEXUS SOFTWARE *
30 REM    * COPYRIGHT-1984 *
35 REM    ********************
40 REM
45 HOME:DB=43634:DL=43616
50 D$=CHR$(4):Q$=CHR$(34)
55 PRINTD$"OPEN END":PRINTD$"DELET
E END":PRINTD$"OPEN END":PRINTD$"W
RITE END"
60 PRINT"PRINT:PRINT"Q$"A"Q$"PEEK(
"DB + 1")*256"Q$",L"Q$"PEEK("DL")+
PEEK("DL + 1")*256"
65 PRINTD$"CLOSE"
```

Para utilizar o programa, digite a listagem e dê RUN. Ele criará o arquivo END diretamente no disco que estiver no drive. Guarde o CRIADOR END para futura instalação em outros discos.

Claudio Milione Dutra e
Leandro Marques Rezende — RJ

Linha COLOR

PEEKs e POKEs no Color

Teste se o computador está preparado para um WARMSTART ou um COLDSTART:

```
PEEK(113)
```

Este PEEK retorna o número 85 se o computador está preparado para um WARMSTART ou um número diferente de 85, se o computador estiver preparado para um COLDSTART.

Retorne o endereço de desvio (JUMP ADDRESS) num WARMSTART:

```
PEEK(114)*256+PEEK(115)
```

Se PEEK (113)=85 e PEEK (PEEK(114)*256+PEEK(115))=18, o INTERPRETADOR COLOR BASIC saltará para este endereço quando o botão ou as teclas de RESET forem pressionadas.

Desabilite o botão ou as teclas de RESET:

```
POKE 114,0:POKE 115,0
```

Quando for pressionado o RESET, o computador executa um COLDSTART.

Retorne o número 127 em um computador com 32K de memória RAM ou o número 63 em um computador com 16K de memória RAM:

```
PEEK(116)
```

Retorne a posição atual de impressão na tela:

```
PEEK(198)
```

Este PEEK indica o número de caracteres que já foram impressos a partir da margem esquerda do vídeo.

É semelhante ao comando POS (0) do INTERPRETADOR COLOR BASIC EXTENDIDO.

Marcel T. de Fontoura — RJ

Linha TRS-80

Incremente seu DOS 500

Incremente o seu DOS 500 ou TRS DOS com essa série de interessantes PATCHES:

```
1) Patch para o DEBUG mostrar caracteres gráficos
        PATCH *Ø (ADD=51D2,FIND=7B,CHG=CØ)

2) Patch para o DOS aceitar (ENTER) em resposta a DATA
        PATCH *Ø (ADD=4EC1,FIND=FØ,CHG=Ø7)
        PATCH *Ø (ADD=4ED4,FIND=3A414F,CHG=C3FE4E)

3) Patch para aceitar a hora no formato HH:MM
        PATCH *Ø (ADD=4F95,FIND=2B,CHG=2A)

4) Patch para o DOS não verificar a senha do arquivo
        PATCH  *2 (ADD=4ED6,FIND=2Ø,CHG=18)

5) Para o comando "CMD "D:n" " mostrar arquivos invisíveis
        PATCH *1Ø (ADD=4E72,FIND=4Ø,CHG=ØØ)
        PATCH *1Ø (ADD=4E47,FIND=Ø2,CHG=Ø3)

6) Patch para o DEBUG poder mudar o BASIC/CMD:
        PATCH *5 (ADD=52EB,FIND=CB,CHG=36)
        PATCH *5 (ADD=52ED,FIND=BE,CHG=ØØ)

7) Para que o DEBUG mostre e altere os endereços abaixo de 56ØØH:
        PATCH *5 (ADD=4ED9,FIND=ØØ56,CHG=ØØØØ)
        PATCH *5 (ADD=4EFF,FIND=ØØ56,CHG=ØØØØ)
        PATCH *5 (ADD=5068,FIND=ØØ56,CHG=ØØØØ)

8) Para fazer a função ROUTE, na vers.1.3 funcionar
        PATCH *9 (ADD=599A,FIND= 3E1FC3094A,CHG=3A2F44FE8Ø)

9) Para retirar logotipo:
        PATCH *Ø (ADD=4E85,FIND=216C51,CHG=C39D4B)
```

Alexandre Costa Gaeta — RJ

Linha TRS COLOR

Desconectando o teclado

Desative todo o teclado do micro durante a execução de um programa com esta dica. Inclua uma linha no início com o comando:

```
POKE 65281,50
```

E para voltar ao normal:

```
POKE 65281,180
```

Estes dois POKEs podem ser aplicados em programas que rodam com a UCP em velocidade alta (POKE 65497,0), pois nesta situação a tela sai de sincronismo e se a tecla **BREAK** for acidentalmente pressionada, o programa poderá ser perdido. Para fazer um teste, rode o programa abaixo e tente interromper sua execução usando **BREAK**. Depois elimine as linhas 10 e 50 e repita a operação:

```
10 POKE 65281,50
20 POKE 65497,0
25 CLS
30 FOR X=1 TO 250:PRINT X;:SOUND 50,1:NEXT X
40 POKE 65496,0
50 POKE 65281,180
```

Geraldo Simonetti Bello — RJ

Linha TRS COLOR

Teclado sonoro

Incremente o teclado do seu compatível com o TRS COLOR, usando essa pequena rotina:

```
TECLADO SONORO
10 DATA 52,86,85,169,169,86,53,214
20 X=PEEK(40)+256*PEEK(39)-8 : CLEAR 200, X
30 Y=PEEK(40)+256*PEEK(39)+1
40 FOR N=Y TO Y+7
50 READ X : POKE N, X
60 NEXT
70 X=INT(Y/256) : POKE 363, X : POKE 364, Y-256*X
80 END
```

Adriano de Arruda Botelho — SP

Linha SINCLAIR

Mude o topo da RAM

Para mudar o topo da RAM sem precisar remover o programa que está na memória, digite essa pequena rotina em BASIC com uma pequena sub-rotina em Assembler:

```
9992 LET A$="21ØØØ22Ø44Ø2R363E28F9
2B2B22Ø2R4ØC3134"
9993 INPUT K
9994 FOR X=1 TO 19
9995 POKE 16443+X,CODE A$(X*2-1)*16
+CODE A$(X*2)-476
9996 NEXT X
9997 POKE 16445,K-INT(K/256)*256
9998 POKE 16446,INT(K/256)
9999 RAND USR 16444
```

Rode o programa e forneça o local do novo topo da memória. Após alguns segundos, seu programa será listado com a memória reorganizada. Durante a execução da rotina, não digite BREAK pois, desse modo, a parte Assembler será apagada. Caso isso aconteça, reinicie a execução da rotina. Ela pode estar em qualquer ponto do programa e ser deletada após uso.

Mário José Bittencourt — SP

Linha Sharp PC

PLOTTER/ Impressora ausente

Para prevenir erros de comando para impressora, quando o PC-1500 não estiver conectado a uma, usar a seguinte rotina:

```
100: IF PEEK &A000<
   >192THEN PRINT
   "ESTOU SEM IMP
   RESSORA"
```

Aldo Rodrigues da Costa — MG

Linha SINCLAIR

Linhas mais rápidas

Esta dica atenua a lentidão do processo de entrada de linhas de um programa na tela. Para que o programa funcione sem prejuízo, porém, é necessário que o número da primeira linha seja maior que 1. Caso o programa tenha subrotinas em linguagem de máquina na linha 1 REM, transfira-o para a linha 2 ou digite **POKE 16510,0**. Quando as linhas começarem a entrar naquele processo lento, digite:

1 ⟨ENTER⟩

Agora você verá que a última linha digitada sumiu. Digite a próxima linha normalmente. A tela será limpa e só aparecerão a linha que havia sumido e a última linha digitada.

Daniel Turini — SP

Linha SINCLAIR

Descubra o nome do programa

Se você não sabe o nome daquele programa armazenado em fita cassete, descubra-o com a ajuda desta rotina em Assembler:

```
40B2 CD230FCD8A401BFB0E01
40BC 0600E7FD8FED3FF1FD2
40C6 A203171738111ØF1F1CD
40A0 0A40CB7A7920Ø1D71730
40AA F41B1DD51E94061A1DD8
40B4 FE17CB7B7B3BF510F501
40BE 2004FE5630CB3FCB1130
40C8 C3C9C9
```

A rotina "NOME" lê o nome do programa em alguns segundos e o imprime na tela, retornando ao BASIC sem carregar o programa e sem destruir arquivos existentes na memória. Dependendo do tipo de uso, poderá ser criada uma linha BASIC para chamar a rotina ou então digitar em modo direto: **RAND USR 16514**.

Carlos Ribeiro Dietrich — RJ

Linha TRS-COLOR

Grave a tela

Ao fazer desenhos em alta resolução, você deseja guardá-los. Através do SAVE normal você gravaria os desenhos juntamente com o programa. Para terminar com esse problema, utilize esta dica que grava uma tela gráfica ou de texto em disco ou fita cassete.

```
CSAVEM "TELA",1024,1535,413     p/ tela de texto

CSAVEM "TELA",256*PEEK(186),256*PEEK(183)-1,413
p/ tela grafica
```

Entre com CLOADM (ou LOADM para disco) para chamar a tela de volta. Veja como esta dica funciona neste programa:

```
10 PMODE 4:COLOR 0,5:PCLS:SCREEN 1
,1
20 LINE (46,16)-(206,176),PSET,BF:
CIRCLE (126,96),80,5:PAINT (126,96
),5
30 FOR J=0 TO 360 STEP 6:A=J/57.29
:LINE (126+80*SIN(A),96)-(126,96+8
0*COS(A)),PSET:NEXT:SOUND 200,5
40 IF INKEY$="" THEN 40
50 CSAVEM "TELA",256*PEEK(186),256
*PEEK(183)-1,413
```

Inserida essa linha de programa, a dica economiza memória. Lembre-se de trocar o CSAVEM por SAVEM (no caso de disco) e note que no lugar de "Tela" pode ser utilizado qualquer outro nome de até oito dígitos. Digite:

```
PMODE 4:PCLS
```

Retroceda a fita, inclua estas duas linhas e dê RUN:

```
1 CLOADM:PMODE 4:SCREEN 1,1
2 GOTO 2
```

Cláudio Costa — RJ

Linha TRS-80

Formata valores

Esta dica formata valores de até 14 dígitos que são listados na impressora:

```
* CLEAR 2000
10 INPUT"VALOR ";VN:IF VN=0 THEN VF$=
"          0.00 " ELSE GOSUB 100
20 LPRINT VF$:GOTO 10
100 V$=MID$(STR$(VN+.0001),2,17):PX=I
NSTR(V$,".")-1:VI$=LEFT$(V$,PX-1)
110 VI$=SPACE$(12-LEN(VI$))+VI$:VF$=M
ID$(VI$,10,3)+","+MID$(V$,PX+1,2)+LEF
T$(STR$(VN),1)
120 FOR IX=7 TO 1 STEP -3:IF MID$(VI$
,IX+2,1)=" " THEN VF$=SPACE$(18-LEN(V
F$))+VF$:RETURN
130 VF$=MID$(VI$,IX,3)+","+VF$:NEXT I
X:RETURN
```

Márcio Antônio Dias — MG

Linha SINCLAIR

Hi FORTH

Se você tem um TK-85 ou consegue simular o HIGH SPEED e possui o compilador FORTH, não deve ficar sem esta palavra: HISAVE. Ela faz uma cópia do compilador em HIGH SPEED (4200 bauds).

```
CREATE HISAVE  2AC,DP,22C,4014,
               CDC,52A4,CDC,20D5,
               CDC,52A4,NEXT
```

Esta palavra tem que ser criada em hexadecimal. Para utilizá-la, basta digitar HISAVE e <NEWLINE>, com o gravador anteriormente preparado para gravar. Para carregar tecle:

```
RAND USR 8630
```

Se você teclar <BREAK> durante a gravação, o sistema se reinicializa não voltando ao BASIC.

Ítalo Jurisato Junior — SP

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 — grupo 1210, Centro, Rio de Janeiro, RJ. CP. 20030

Linha TRS-COLOR

PEEKs e POKEs I

Posicione o endereço de início dos programas BASIC, na página gráfica número 1 (P CLEAR 1):

```
POKE 25,12:POKE 26,1:POKE 3072,0:NEW
```

Posicione o endereço de início dos programas BASIC, na página gráfica número 2 (P CLEAR 2):

```
POKE 25,18:POKE 26,1:POKE 4608,0:NEW
```

Na página gráfica número 3 (P CLEAR 3):

```
POKE 25,24:POKE 26,1:POKE 6144,0:NEW
```

Na página gráfica número 4 (P CLEAR 4):

```
POKE 25,30:POKE 26,1:POKE 7680,0:NEW
```

Aumente a memória disponível para programas BASIC (sem a utilização das páginas gráficas):

```
POKE 25,6:POKE 26,1:POKE 1536,0:NEW
```

Obs: Estes POKEs não funcionam com o INTERPRETADOR COLOR DOS.

Marcel T. da Fontoura — RJ

Linha SINCLAIR

REM Condensada

Quando se está desenvolvendo um programa extenso em Assembler, é comum ter que se digitar várias linhas REMs para o armazenamento de dados ou do próprio programa. Isso ajuda o "pirata", pois ele tem acesso direto no vídeo a essas linhas o que facilita o disassemblamento do programa.

```
9000 PRINT "NUMERO DE LINHAS?"
9005 INPUT N
9010 FOR I=1 TO N-1
9015 LET GUNT=PEEK 16511+256*PEEK
16512
9020 LET TOI=16515+GUNT
9025 LET TOT2=PEEK TOT+256*PEEK(TO
T+1)+TOT-16511
9030 POKE 16511,TOT2-256*INT(TOT2/
256)
9035 POKE 16512,INT(TOT2/256)
9040 NEXT I
```

Rode esta rotina e entre com o número de linhas a serem unidas (sempre com a primeira linha do programa) e faça o pirata "suar" um pouco mais.

Frederico dos Santos Liporace — RJ

Linha TRS-COLOR

Figuras no COLOR

Observe o micro criar figuras simétricas coloridas com esta dica de apenas uma linha:

```
10 CLS RND(8):FOR T=0 TO 90:C=RND(
8):X=RND(63):Y=RND(31):SET(X,Y,C):
SET(X,31-Y,C):SET(63-X,Y,C):SET(63
-X,31-Y,C):SOUND RND(128),RND(3):N
EXT:RUN
```

Cláudio Costa — RJ

Linha SINCLAIR

Efeito em BASIC

Esta dica, embora simples, produz um belo efeito na tela do seu Sinclair. As linhas 60 e 70 contêm o centralizador de textos publicado em MS nº 40.

```
10 POKE 16418,0
20 CLS
30 LET A$="MICRO SISTEMAS"
40 LET A=11
50 LET B=A
60 PRINT AT A,15-LEN A$/2;A$
70 PRINT AT B,15-LEN A$/2;A$
80 LET A=A+1
90 LET B=B-1
100 IF B=0 THEN RUN
110 GOTO 60
```

Alexandre Gouvêa Longo — MG

Linha APPLE

Formate com 36 trilhas

Rode esta dica e formate seu disco com 36 trilhas (16 setores a mais).

```
5 HOME
10 POKE 48894,36:POKE 46063,36:POKE
44725,144
15 PRINT "COLOQUE O DISCO A SER INIC
IALIZADO NO DRIVE"
20 PRINT "DIGITE : INIT HELLO    (RE
TURN)"
```

José de Lima H. Neto — SP

Linha TRS-80

MERGE em cassete

Se você tem dois programas e quer concatená-los em versão BASIC-cassete, carregue o primeiro programa e digite:

```
E=PEEK(16633)+PEEK(16634)*256-2  (ENTER)
'O programa em BASIC termina dois bytes
antes do início da memória de dados.
B2=INT(E/256):B1=E-B2*256  (ENTER)
POKE16548,B1:POKE16549,B2  (ENTER)
'Desproteger o primeiro programa, con-
catenando-o com o segundo.
```

O primeiro programa deve ter os números das linhas menores que as do segundo.

Luciano Chaves Arantes — GO

Linha SINCLAIR

Simule o comando LIST

Esta rotina, baseada no programa SCAN (MS nº 38, pg. 64), simula o comando LIST da linha Apple ou TRS-80, listando apenas a linha especificada:

```
16514  21 00 00
16517  CD D8 09
16520  1E 00
16522  CD 45 07
16525  C9
```

Digite este programa-exemplo:

```
10 PRINT AT 21,0;"NUMERO DA LINHA
?"
20 INPUT X
30 POKE 16515,X-256*INT(X/256)
40 POKE 16516,INT(X/256)
50 CLS
60 RAND USR 16514
70 GOTO 10
```

Os endereços são válidos para o TK, podendo haver diferenças em outros equipamentos.

Fábio Antônio R. Corrêa — SP

Linha TRS-80

ERROR desaparecido

Comparando o NEWDOS/80 com o DOS 500, pode-se notar que o comando ERRORxx do LIB do DOS-500 está "desaparecido" o que não acontece com o NEWDOS/80. Para solucionar este problema, digite:

```
PATCH *1(ADD=5149,FIND=2020202020,
CHG=4552524F52) (ENTER)
```

Um exemplo:

Quando em DOS-500 Ativo, digite: ERROR 12<ENTER>. Será mostrada a seguinte mensagem:

```
Time Out On Disk Drive
```

Luciano Chaves Arantes — GO

Linha APPLE

"CALL + HOME"

Em seus programas, experimente:

```
CALL 63538:HOME
```

Ao invés do uso de um ou de outro separadamente, junte as duas instruções e veja o efeito.

Morivardo Macedo F. Junior — BA

Linha SINCLAIR

Abertura de programas

Esta rotina cria um efeito muito interessante para apresentar seus programas:

```
10 INPUT X$
20 LET C=INT((31/2)-(LEN X$/2))
30 FOR A=1 TO LEN X$
40 FOR B=1 TO 7
50 PRINT AT B,C;X$(A);AT B,C;" "
60 NEXT B
70 PRINT AT B,C;X$(A)
80 LET C=C+1
90 NEXT A
```

Para torná-la mais incrementada, altere e acrescente as seguintes linhas:

```
35 LET Y$=CHR$(CODE X$(A)+128)
50 PRINT AT B,C;Y$;AT B,C;" "
70 PRINT AT B,C;Y$
```

Edison Smejoff — RS

Linha SINCLAIR

Desenhos

Esta dica gera desenhos interessantes no seu micro compatível com o Sinclair. Basta modificar o valor da variável "M".

```
10 FAST
20 PRINT"QUAL O VALOR ?"
30 INPUT M
40 FOR N=-4*PI TO 4*PI STEP .1
50 LET R=4*COS(M*N+PI/4)
60 LET X=INT(7*(4+R*COS N))
70 LET Y=INT(5*(4+R*SIN N))
80 PLOT X,Y
90 NEXT N
```

Fernando Ponzoni Kiehn — SC

Linha TRS-COLOR

PEEKs e POKEs II

Desabilite a função CONT depois do comando BREAK:

```
POKE 45,0:POKE 46,0
```

O comando CONT, depois desses POKEs, causará um erro tipo "?ON ERROR".

Desabilite o botão ou as teclas de RESET:

```
POKE 113,0
```

O computador voltará para a tela inicial se o botão ou as teclas de RESET forem pressionadas.

Desabilite o botão de RESET após o POKE 113,0:

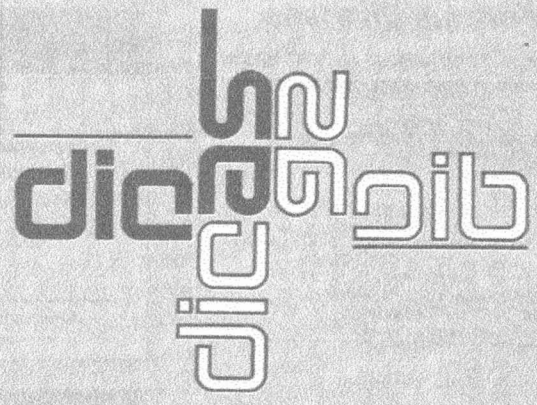
```
POKE 113,85
```

Provoque um COLDSTART:

```
POKE 113,0:EXEC 40999
```

Este POKE apaga 1K (1024 Bytes) de memória RAM começando no endereço 0.

Marcel T. da Fontoura — RJ



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 — grupo 1210. Centro, Rio de Janeiro, RJ. CP. 20030

Linha SINCLAIR

Checando batidas

Às vezes, em jogos, temos que checar se o nosso carro, nave ou avião bateu contra o obstáculo ou não. Para realizar esta tarefa, ponha em seu jogo as seguintes linhas:

```
xxxx PRINT AT X,Y;
```

onde xxxx é o número da linha e X,Y são respectivamente a linha e a coluna da posição onde se quer checar se há obstáculo ou não. A outra linha é esta:

```
xxxx+10 IF PEEK (PEEK 16398+256*PEEK
16399)=yyyy THEN GOTO zzzz
```

onde yyyy é o código do obstáculo e zzzz é a linha onde será imprimida a mensagem: "VOCÊ BATEU", ou onde será realizada a explosão.

Marco Túlio de O. Valente — MG

Linha TRS-COLOR

RESET total

Quando estamos trabalhando com linguagem de máquina, muitas vezes, após o EXEC, o computador entra em looping e o RESET não funciona. Somos, então, obrigados a desligar e religar o equipamento. Para evitar este transtorno, que a longo prazo prejudica a CPU, digite antes de começar as experiências:

```
POKE 113,0   e   <ENTER>
```

Ao ser ressetada, a máquina reinicializa as variáveis do sistema. Para utilizar a dica em BASIC (um NEW completo), digite:

```
POKE 113,0:EXEC 40999
```

Hênio de Araújo Bezerra — RN

Linha TRS-80

Desfile de caracteres

Digite essa pequena dica e assista a um verdadeiro desfile de caracteres na tela de seu TRS-80.

```
10 CLS
20 DEFINT X,Y,K
30 FOR X=192 TO 255:FOR Y=1 TO 64
STEP 2
40 PRINT CHR$(22)
50 POKE(15360+(14*64+Y)),X
60 FOR K=1 TO 70:NEXT K
70 NEXT Y,X
```

Ronaldo Ferreira dos Santos — SP

Linha APPLE

Formas diversas

Execute inúmeras formas na tela de seu Apple com esta dica:

```
1 REM FORMAS DIVERSAS
2 REM ARMANDO OSCAR CAVANHA F*
3 HGR2:HCOLOR=3
4 FOR Q=0 TO 3
6 FOR T=0 TO 3
8 FOR P=0 TO 6.3 STEP .03
10 X=90 * COS (P) + 130
12 Y=50 * SIN (P) + 90
14 Z=X+40 * COS (2 * Q * P)
16 W=Y+40 * SIN (2 * T * P)
18 HPLOT Z,W TO X,Y:NEXT
22 FOR U=1 TO 1000:NEXT
26 HGR2:NEXT:NEXT:GOTO 3

Para o TK2000: 17 SOUND W,8 TO Y,8
```

Armando Oscar Cavanha Filho — RJ

Linha SINCLAIR

Duplicador II

Duplique seus programas com esta dica para os micros da linha Sinclair. Digite uma linha REM com 50 caracteres e entre com o bloco Assembler:

```
16514 - 21 E8 7F 22 04 40 01 31
16522 - 00 11 EE 7F 21 A2 40 ED
16530 - B0 C9 B1 AA B4 B0 AB B0
16538 - A6 B0 B3 BA B3 AA B8 00
16546 - CD E7 02 CD 44 03 CD BB
16554 - 02 2E 28 FA CD 23 0F C3
16562 - F5 02
```

Digite agora a parte em BASIC e rode o programa.

```
1 REM ...(50 caracteres quaisquer)
...
2 PRINT "          PROGRAMA DUPLICADO
R          ",,"PARA GRAVAR EM","DIGI
TE","LOW SPEED","GOTO 7","HI SPEED
","C/ENTER",,,,,"PARA USAR > RAND
USR 32750 >","> LE A FITA > QUALQU
ER TECLA >","> GRAVA A FITA"
3 STOP
4 POKE 16561,213
5 POKE 16562,32
7 RAND USR 16514
8 NEW
10 SAVE "DUPLICADOR"
11 RUN
```

Se a cópia for feita com o LOAD normal, digite GOTO 7. Se for em HIGH SPEED, digite C (CONT) e <ENTER>. A rotina irá para a RAMTOP. Para começar a duplicação de programas, por mais "fechados" que estejam, digite RAND USR 32750 e carregue o programa da fita. Ao término da operação, a tela ficará escura. Coloque a fita para gravar o programa, prepare o gravador e digite qualquer tecla. Se o programa a ser copiado foi gravado pela instrução SAVE, logo após a gravação ocorrerá erro C/0 o que não afetará o carregamento do programa.

Leo F. A. Nunes — RJ

Linha TRS-80

Memorize a tela

Esta dica agiliza o processo de memorização da tela publicado na Seção Dicas, MS nº 33:

```
10 CLS:CLEAR 1030:DIM N$(5),M$(5)
20 FOR X=0 TO 3
30 N$(X)="":N(X)=VARPTR(N$(X))
40 POKE N(X),255:POKE N(X)+1,0:POKE N(X)+2,60+X
50 NEXT X
60 '
70 ' AREA RESERVADA PARA O PROGRAMA GERADOR DE DESENHOS
80 '
90 FOR X=0 TO 3:M$(X)=N$(X):NEXT
100 ' PARA VISUALIZAR O SEU DESENHO:
110 CLS:FOR X=0 TO 3:PRINT M$(X);:NEXT:GOTO 110
```

Alexandre Costa Gaeta — RJ

Linha TRS-COLOR

Ilusão de óptica

```
10 PMODE 4:CLS:COLOR 0,5:PCLS
20 LINE (5,73)-(238,73),PSET:LINE
-(256,117),PSET:LINE -(246,129),PS
ET:LINE -(18,129),PSET
30 LINE (5,85)-(228,85),PSET:LINE
-(236,107),PSET:LINE -(11,107),PSE
T:LINE (11,95)-(218,95),PSET
40 LINE -(228,85),PSET:LINE (218,9
5)-(222,106),PSET:LINE (256,117)-(
18,117),PSET
50 CIRCLE (6,79),5,,1.4:CIRCLE (12
,101),5,,1.4:CIRCLE (19,123),5,,1.
4
60 SCREEN 1,1:GOTO 60
```

Verifique a quantas anda a sua visão após passar horas sentado em frente ao seu COLOR Computer. Afaste-se uns dois metros da televisão, observe o desenho e diga onde está ligado o pino do meio.

Cláudio Costa — RJ

Linha SINCLAIR

Inicialize com CONT

Para não correr o risco de apagar as variáveis de um programa; provocar algum erro por não saber a linha de início de um programa ou não ter que fazer um programa específico para gravar, verificando somente após a gravação se ele foi bem gravado (em HIGH-SPEED), use, antes de gravar:

```
SLOW
GOTO ou RUN linha de inicio
BREAK
```

O BREAK deve ser dado antes que a tela seja apagada totalmente ao se dar o <NEWLINE> nos comandos GOTO ou RUN. Digite:

```
RAND USR 8405
RAND USR 8539
```

Assim que o programa for carregado, digite:

```
CONT
```

Seu programa, qualquer que seja, será inicializado com CONT.

Eugênio Betanho — SP

Linha SINCLAIR

Agilizando a digitação

Na digitação de programas longos, a tela sempre fica cheia e a inserção de uma nova linha torna-se bastante demorada.

Para contornar esse problema, altere a seqüência de digitação de algumas linhas, como por exemplo:

```
*
*
400 PRINT X
410 LET A=A+1
420 LET X=X+A
430 GOTO 500
*
*
*
```

Digite as linhas na seguinte ordem:

```
*
*
*
400 PRINT X
420 LET X=X+A
410 LET A=A+1
430 GOTO 500
*
*
*
```

Dessa forma a tela ficará limpa e a introdução de novas linhas será muito mais rápida.

José Eduardo Mesquita Jr. — DF

Linha SINCLAIR

CLS instantâneo

Execute um CLS instantâneo mesmo após vários SCROLLs. Coloque estas linhas, na mesma ordem em que se encontram, em qualquer parte do programa:

```
POKE 16389,74
CLS
POKE 16389,128
```

Rode o seguinte programa e veja a diferença:

```
20 SCROLL
30 PRINT "TESTE"
40 NEXT I
50 POKE 16389,74
60 CLS
70 POKE 16389,128
80 GOTO 10
```

OBS.: Esta dica só é válida para microcomputadores com 16 Kbytes.

Álvaro Ferreira de Freitas Borja — PA

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha TRS-80

PATCHES no TRS DOS

Facilite a leitura do diretório de disco com muitos arquivos com estes dois PATCHES que fazem a listagem do diretório parar a cada tela preenchida e esperar que você tecle ENTER:

```
PATCH *6(ADD=616D,FIND=CD8B6187C9F
E40,CHG=3A803FD620C9CD)

PATCH *6(ADD=6174,FIND=2806C547AF7
BC1,CHG=7C61CDC901AFC9)
```

Este PATCH elimina a linha de pontos que é exibida após a mensagem TRS DOS Ready:

```
PATCH *1(ADD=4E75,FIND=062E3E,CHG=
F37E4E)
```

Com este PATCH é possível fazer uma formatação mais apurada para novos discos. É melhor perder um disco duvidoso do que perder dados importantes:

```
PATCH *7(ADD=5BEE,FIND=E5,CHG=58)
```

Este PATCH elimina a mensagem "Operation Aborted":

```
PATCH *1(ADD=50A9,FIND=4F,CHG=0D)
```

Estes PATCHES foram publicados originariamente na revista 80 Micro de Janeiro de 85.

Antônio Elias V. Mokarzel — PA

Linha SINCLAIR

Destrave em high speed

Esta dica para programas em high Speed do TK85 gravados por RAND USR 8405: `GOTO USR 8630`

Esta dica não funcionará se o programa for copiado através da dica para duplicar programas protegidos anteriormente publicada.

Marcus Antônio Peixoto — SP

Linha SINCLAIR

Mensagem

Esta dica faz com que uma mensagem de qualquer tamanho passe pela linha 21:

```
10 LET A$="  mensagem  "
20 LET B$="  32 espacos  "
30 LET A$=B$+A$+B$
40 FOR I=1 TO LEN A$-31
50 LET C$=A$(I TO I+31)
60 PRINT AT 21,0;C$
70 NEXT I
```

Experimente transformá-la em sub-rotina e trocar a linha 10 por:

```
10 INPUT A$
```

Marcelo Dionisio Penalva — SP

Linha SINCLAIR

Apresentação original

Rode esta dica e entre com diversas frases ou trechos de frases, separadas por um asterisco. No final, coloque também um asterisco e veja uma apresentação muito interessante. Experimente estas frases: "EIS AQUI *UMA MANEIRA ORIGINAL * DE APRESENTAÇÃO ***VIU?**"

```
5 FAST
10 LET B$=""
20 LET L=0
30 INPUT A$
40 SLOW
50 FOR N=1 TO LEN A$
60 IF A$(N)="*" THEN GOTO 100
70 LET B$=B$+A$(N)
80 NEXT N
90 STOP
110 LET B=INT (LEN B$/23)+1
120 FOR M=1 TO 17+B
130 IF M<=LEN B$ THEN LET A=M
140 PRINT AT L,32-M;B$( TO A)
150 NEXT M
160 LET L=L+1
170 LET B$=""
180 GOTO 80
```

Cada trecho de frase não pode ter mais do que 32 posições.

Henrique Rubens Jerozolimsky — SP

Linha SINCLAIR

SAVE mais rápido

Para aqueles que querem economizar tempo e também espaço em sua fita, ao invés de utilizar o comando SAVE, use **RAND USR 777** em modo FAST. Um programa dessa forma:

```
7000 SAVE "PROGRAMA
7010 RUN
```

ficaria assim:

```
7010 RAND USR 777
7020 SLOW
7030 RUN
```

No comando SAVE, quando você pressiona NEW LINE, o computador deixa a tela escura por alguns segundos e mostra, logo após, a imagem de gravação. Já com o **RAND USR 777**, ao se pressionar NEW LINE o computador mostra instantaneamente a imagem de gravação.

Francisco Carlos S. Moraes — CE

Linha SINCLAIR

CLS especial

Esta rotina em Assembler simula um CLS no vídeo dos micros da linha Sinclair. O Cursor vai "varrendo" a tela e, ao mesmo tempo, a limpa:

```
16514  2A 0C 40 23 06 18 C5 06
16522  20 36 AB 0E FF 0D 79 FE
16530  00 20 FA 36 00 23 10 F1
16538  36 76 23 C1 10 E8 C9
```

No endereço 16624 determina-se o cursor e no endereço 16526 determina-se a velocidade (mais rápido =1 ; mais lento =255). Pokeando no endereço 16524 o valor 0 (zero) e no endereço 16526 o valor 1 (um), pode-se criar um CLS semelhante ao do micro, só que bem mais veloz. Teste-a com esse programa:

```
10 FOR N=255 TO 1 STEP  25
20 POKE 16526,N
30 LIST
40 LIST
50 PRINT AT 0,0;
60 RAND USR 16514
70 NEXT N
```

Esta rotina foi elaborada em um TK-83 com 16K, mas com um POKE 16389,255 na primeira linha do programa é possível utilizá-la em micros com 2Kbytes.

Fábio A. R. Corrêa — SP

Linha TRS-COLOR

PEEKs e POKEs II

Retorne o endereço de memória RAM onde foi armazenado o bloco de dados lido de uma fita cassete:

```
PEEK(126)*256+PEEK(127)
```

Retorne o número dois se um bloco de dados for carregado num endereço de memória que não seja do tipo RAM:

```
PEEK(129)
```

Coloque o valor nulo em INKEY$. Ex: A$ =INKEY:POKE 135,0 faz com que A$ ="" :

```
POKE 135,0
```

Indique o endereço atual do cursor na tela de textos. O endereço pode variar de 0 a 511:

```
PEEK(136)*256+PEEK(137)-1024
```

Posicione o cursor na tela de textos. Este comando coloca o cursor na posição X*256+Y-1024 da tela. Lembre-se que este número tem que estar entre 0 e 511:

```
POKE 136,X:POKE 137,Y
```

Coloque o cursor piscando na coordenada X*256+Y-1024. Este macete não pode ser usado no modo direto e nn indica a linha onde colocamos estes comandos (o endereço deve variar entre 0 e 511):

```
POKE 136,X:POKE 137,Y:EXEC &HA1A1:
GOTO nn
```

Marcel T. da Fontoura — RJ

Linha TRS-COLOR

PEEKs e POKEs III

Faça o cursor parar de piscar durante aproximadamente três segundos:

```
POKE 148,0
POKE 148,255
```

Volte o cursor ao estado normal depois de utilizados os POKEs anteriores:

```
POKE 148,11
```

Retorne a cor atual do cursor. A cor atual do cursor é indicada por um número entre 1 e 11:

```
PEEK(148)
```

Marcel T. da Fontoura — RJ

Linha TK 2000

DEBUG no TK-2000

Existe uma rotina em applesoft ideal para "debug" de programas em BASIC nos Apples compatíveis, conforme bem demonstra Carlos A. Diz em seu artigo "Rotina Aponta Erros" (MS 32).

A rotina é a seguinte:

```
10 ONERR GOTO 60000:EPT=63:T=256
60000 E=PEEK(220)+T*PEEK(221):E=E+
5*(PEEK(E)=0):V=PEEK(E):POKE E,EPT
:L$=RIGHT$("0000"+STR$(PEEK(218)+T
*PEEK(219),5):L=PEEK(121)+T*PEEK(1
22)+49:FOR I=1 TO 5:POKE L+I,ASC(M
ID$(L$,I,1)):NEXT:LIST 00000:POKE
E,V:POKE 216,0:RESUME
```

O programa em BASIC a ser "debugado" deve ser inserido entre estas linhas. Porém, esta rotina não roda no TK-2000. Inverta a ordem dos comandos da linha 10 e rode sem problemas esta rotina no seu equipamento:

```
10 EPT=63:T=256:ONERR GOTO 60000
```

João José de Araújo M. Filho — RJ

Linha SINCLAIR

Destravando programas

Para os seus programas que estão contidos em uma variável ou aqueles que precisam sair rodando para organizar o sistema, use esta rotina que parece não existir.

```
CD 23 0F    ;CALL FAST
CD 45 03    ;CALL LOAD
CD 23 0F    ;CALL FAST
CD F5 02    ;CALL SAVE (TEMPO)
21 3B 40    ;LD HL,4000
CB FE       ;SET 7,(HL)
CD F5 02    ;CALL SAVE
CF          ;RST 8
E1          ;POP HL
C9          ;RET
```

Ela chama a rotina LOAD e depois a rotina SAVE (2 vezes). O primeiro SAVE permite a troca das fitas e dos cabos do gravador; o segundo executa a cópia. Esta rotina pode ser colocada em qualquer posição da memória (RAM) que se possa dar um POKE e posteriormente rodá-la.

Faça a RAMTOP igual a 32512 dando o comando POKE 16389,127 e, logo após, um NEW. Com o auxílio do MICRO BUG, entre com a rotina a partir do endereço 32512. Terminada esta operação, dê outro NEW e chame, sempre que quiser, a rotina com RAND USR 32512.

OBS: Se o programa sai rodando em FAST, retire da rotina os seguintes bytes: 21, 3B, 40, CB e FE.

Alex Argona — SP

Linha SINCLAIR

Efeitos com Scroll

Faça pequenas modificações na dica "Letreiro em BASIC":

```
1 LET A$="...MENSAGEM..."
5 SCROLL
10 PRINT AT 21,0;A$( TO 32)
20 LET A$=A$(2 TO )+A$(1)
30 GOTO 5
```

Digite RUN e observe o belo efeito provocado pelo Scroll. É importante que haja palavra no modo graphics no texto para melhorar o efeito.

Marcelo Contart — SP

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha SINCLAIR

Listagem mais rápida

Na dica "Acelere a listagem", publicada em MS nº 36, o comando POKE 16420,255 faz com que a última linha do programa passe para o topo da tela. No entanto, a linha do programa que passa para o topo da tela é a linha indicada pelo cursor, ou seja: o POKE, juntamente com dois NEWLINE, não irá alterar a linha do topo. Se o cursor, porém, estiver indicando a última linha do programa, esta dica funcionará perfeitamente. Para contornar este problema, digite:

```
LIST    (NEWLINE)
        (NEWLINE)
```

E depois:

```
LIST 9999   (NEWLINE)
            (NEWLINE)
```

A última linha do programa será chamada para o topo da tela, sem depender do cursor.

Esta dica não funcionará quando uma linha do programa, em código de máquina, possuir o código 118 em decimal sendo mostrado na tela. O LIST 9999 poderá ser substituído por LIST X, onde X pode ser qualquer número maior que o número da última linha do programa.

Mário Koukiti Kimura — SP

Linha TRS-COLOR

Uso do joystick

Desenhe na tela do seu micro com o auxílio do joystick, utilizando esta dica:

```
10  EXEC 43304:POKE 65495,0
20  PRINT"ESTE PROGRAMA DESENHA
USANDO O JOYSTICK DIREITO QUANDO
SE APERTA O BOTÃO DE TIRO."
30  PRINT"QUANDO O BOTÃO NÃO ES
TA APERTADO O CURSOR PODE SER MO
VIMENTADO PELA TELA SEM PROBLEMA
EXCETO QUANDO ESTIVER EM CIMA DE
UM PONTO JA DESENHADO O QUAL SE
RA APAGADO"
40  PRINT"ENTRE COM '3' OU '4' P/
COMEÇAR OU PARA ESCOLHER O MODO
GRAFICO"
50  PRINT"TECLANDO QUALQUER OUTRA
TECLA VOCE LIMPARA A TELA EXCETO
'P' QUE FINALIZARA O PROGRAMA."
60  EXEC 44539
70  H=VAL(CHR$(PEEK(135))):IF H<3
OR H>4 THEN 60
80  PMODE H,1:PCLS:SCREEN 1,1
90  EXEC 43486:X=INT(256/63)*PEEK
(346):Y=INT(192/63)*PEEK(347)
100 P=PEEK(65280):IFP=126ORP=254
THEN LINE -(X,Y),PSETELSE LINE-
(X,Y),PRESET
110 PSET(X,Y)
120 A$=INKEY$:IFA$="3"THEN PMODE
3,1:SCREEN 1,1 ELSE IFA$="4" THEN
PMODE 4,1:SCREEN 1,1
130 IF A$="P" THEN POKE 65494,0:
EXEC 43304:END ELSE IF A$<>"" AND
A$<>"3" AND A$<>"4" THEN PCLS
140 PRESET (X,Y)
150 GOTO 90
```

Quando terminar a digitação, rode o programa e comece a desenhar!

Dauro Silveira Moura — RJ

Linha TK 2000

Efeitos especiais

Digite esta dica e veja uma bonita figura sonora na tela:

```
1 REM PROGRAMADOR:JORGE PABLO ZA
PATA RIVERA
4 HGR 2:RUN 90
25 FOR A=0 TO C STEP D
30 HPLOT 140,A TO 0+A,159-A TO 2
79-A,159-A TO 140,A
34 FOR M=1 TO 2
35 VTAB 1:HTAB 1:PRINT"EFEITOS E
SPECIAIS"
36 INVERSE
37 NEXT:NORMAL
39 SOUND RND(1)*255,10 TO RND(1)
*11,10
40 NEXT
45 RETURN
50 B=0:C=149:D=3
55 HCOLOR=I
60 GOSUB 25
70 B=149:C=0:D=-3
75 HCOLOR=I
76 GOSUB 25
77 RETURN
78 B=147:C=0:D=-3:HCOLOR=I:GOSUB
 25
85 RETURN
90 I=RND(1)*7+1
100 GOSUB 50
110 I=0
120 GOSUB 78
130 GOTO 90
```

Jorge Pablo Zapata Rivera — BA

Linha CP/M-80

INKEY$ no DBase

Esta dica substitui o comando WAIT que acessa o disco a cada vez que é chamado, tornando-se um processo demorado. Para utilizá-la, deverão ser inseridas as seguintes linhas no programa:

```
SET CALL TO 42000
POKE 42000,14,06,30,255,205
POKE 42005,05,00,103,202,16
POKE 42010,164,50,89,00,201
```

Cada vez que o programa aguardar o pressionamento de uma tecla, coloque a instrução CALL. Desde que não seja usado o comando SORT, a rotina ficará armazenada na memória. Usando o SORT, as linhas acima deverão ser inseridas novamente. Se você desejar obter a tecla pressionada, armazene em alguma variável a seguinte instrução:

```
STORE CHR(PEEK(89)) TO (variavel)
```

Esta dica é compatível com qualquer micro com CP/M 80.

Rizieri Maglio — SP

Linha SINCLAIR

Imprimindo diferente

Incremente a apresentação de seus programas ou mesmo as mensagens que nele aparecem, com esta pequena dica:

```
10 INPUT A$
20 DIM B$(1,LEN A$)
30 LET B$(1)=A$
40 FOR A=1 TO LEN A$
50 PRINT AT 10,A;CHR$(CODE
 B$(1,A)+128)
60 PRINT AT 10,A;B$(1,A)
70 NEXT A
80 CLS
90 GOTO 40
```

Se você quiser imprimir números, acrescente a seguinte linha:

```
5 INPUT C
```

e mude esta linha:

```
10 LET A$=STR$ C
```

Daniel Pirola Rosseli — SP

Linha TRS-80

Mude a velocidade

Para mudar a velocidade de gravação (CASS) sem perder seu programa, use esta dica:

```
10 POKE 16526,26:POKE 16527,96
20 X=USR(0)
30 END
```

José D. Queiroz — RJ

Linha SINCLAIR

Examine o Assembler

Use esta dica para examinar uma rotina Assembler protegida ou para uma linha REM, onde se encontra essa rotina, que não cabe na tela:

```
9997 FOR T=16514 TO 29000
9998 PRINT CHR$ PEEK T;
9999 NEXT T
```

Dê GOTO 9997 e quando a tela encher, digite CONT.

Ricardo Saldanha Freitas — RJ

Linha ZX SPECTRUM

BORDER Multicor

Rode esta dica e confirme. Você pode alterar à vontade os parâmetros do BORDER:

```
1 POKE 23624,71
2 BORDER 0:BORDER 7:BORDER 1:BOR
DER 6:BORDER 2:BORDER 5:BORDER 3
:BORDER 4:BORDER 0:PAUSE 1:GOTO
2
3 REM CP-ETI Lauro Gomes
```

Wagner Hide Ikeda — SP

Linha TRS-COLOR

PEEKs e POKEs

Retorne o tamanho da memória RAM do microcomputador (16,32 ou 64Kb). Não funciona para o COLOR 64 com 128Kb de memória:

```
PEEK(116)*256+PEEK(117)
```

Retorne o número dois quando abrimos um arquivo em cassete no modo OUTPUT. Ex: OPEN "O", # -1, "NOMEARQ".

Retorne o número um quando abrimos um arquivo em cassete no modo INPUT. Ex: OPEN "I", # -1, "NOMEARQ".

Retorne o número zero quando fechamos um arquivo em cassete. Ex: CLOSE # -1.

```
PEEK(120)
```

Retorne o número zero, se for lido um bloco HEADER da fita cassete.

Retorne o número um, se for lido um bloco de dados da fita cassete.

Retorne o número 255, se for lido um bloco EOF (End Of File) da fita cassete.

```
PEEK(124)
```

Marcel T. da Fontoura — RJ

Linha TRS-80

Armazenamento de STRINGS

Entre com o seguinte programa (no BASIC Residente para o BASIC Disco, é necessário diminuir o índice máximo, 15000, para um valor menor):

```
10 CLEAR 0
20 DIM A$(15000)
30 FOR I=1 TO 15000
40 A$(I)="ABCD...(200 caracteres
 quaisquer)...XYZ"
50 NEXT
60 PRINT"MEMORIA DISPONIVEL =>"
; MEM
70 END
```

Observe que foram reservados 0 (zero) bytes para o armazenamento de strings. Independente disto, estamos criando uma tabela de 1500* 200 = 3000000 (três milhões) de caracteres ou 3 Megabytes!

Tal fato ocorre pois o interpretador BASIC, em vez de alocar uma área de memória diferente para cada elemento da tabela, simplesmente usa a memória para criar pointers, índices e controles da tabela, mas o conteúdo de cada elemento da tabela (os 200 caracteres digitados acima, entre as aspas) só ocupa a memória uma única vez (dentro do próprio programa!). Se tentarmos usar qualquer função string (tipo LEN, MID, STRING, LEFT), um erro ocorrerá ("OS ERRO" no BASIC Residente ou "OUT OF STRING SPACE" no Disco). Isto mostra que só é necessário reservar espaço para strings (com a função CLEAR) quando tivermos que usar alguma função STRING.

José Henrique Silva Sousa — DF

Linha SINCLAIR

Edição de valores numéricos

Coloque pontos em números de até 15 dígitos, divididos de três em três, e padronize os valores com esta pequena dica:

```
9000 INPUT E$
9010 DIM A$(15)
9020 LET A$(16-LEN E$ TO 15)=E$
9030 LET B$=A$(1 TO 3)+"."+A$(4
 TO 6)+"."+A$(7 TO 9)+"."+A$(10
 TO 12)+"."+A$(13 TO 15)
9040 FOR I=1 TO 19
9050 IF CODE B$(I)>27 AND CODE
B$(I)<38 THEN GOTO 9060
9055 NEXT I
9060 PRINT TAB 32-LEN(B$(I TO
19));B$(I TO 19)
9070 GOTO 9000
```

Henrique O. E. Ammirabile — DF

Linha TRS-80

Gerador de sons

Se você possui um CP-500 ou compatível, esta é uma interessante dica que pode ser sofisticada em linguagem de máquina. Esta versão, em BASIC, gera sons preestabelecidos. A opção "T" é para você testar nossos sons.

```
10 CLS
20 INPUT"IMITA O QUE";J$
30 IF J$="NADA" THEN END
40 IF J$="CAVALO" THEN J=50
50 IF J$="LAMBRETA" THEN J=15
60 IF J$="CAVALO CORRENDO" THEN
J=30
70 IF J$="AVIAO" THEN J=3
80 IF J$="BATERIA" THEN J=100
90 IF J$="CARRO" THEN J=8
100 IF J$="T" THEN INPUT J
110 FOR I=1 TO 100
120 FOR K=1 TO J:NEXT K
130 OUT 255,I
140 NEXT I
150 GOTO 20
```

Itamar Dionísio Vieira — RJ

Linha ZX SPECTRUM

O incrível DRAW

Digite esta dica e veja que o comando DRAW faz muito mais do que traçar linhas e arcos:

```
1 REM Publ. em SINC 3:8
10 OVER 1
15 FOR N=1 TO 2
20 PLOT 85,27:DRAW 100,100,PI*25
5
25 NEXT N
```

Wagner Hide Ikeda — SP

Linha TRS-80

Melhore o MERGE

Na seção DICAS de MS 47 saiu uma listagem para simular MERGE no TRS 80, só que não funcionou satisfatoriamente no meu CP-300. Fiz então as seguintes modificações:

```
1. E=PEEK(16633)+PEEK(16634)*256-2
2. PRINT E (Obs.? anote este valor)
3. Carregar programa numero 1
4. E=PEEK(16633)+PEEK(16634)*256-2
5. B2=INT(E/256):B1=E-B2*256
6. POKE 16548,B1:POKE 16549,B2
7. Carregar programa numero 2
8. LET E= (valor anotado no passo 2)
9. B2=INT(E/256):B1=E-B2*256
10. POKE 16548,B1:POKE 16549,B2
11. LIST
```

Nota: É necessário que o número das linhas do programa 1 seja menor que os números das linhas do programa número 2. É, também, necessário anotar o valor do passo 2 porque ele será apagado ao se dar o comando CLOAD para carregar o 1º programa.

Marcelo Baradi Spera — RS

Linha TRS-COLOR

Jogos

Se você quer mexer nos programas abaixo, tente estas dicas:

1) — Para aumentar o número de vidas no programa "FROG", digite POKE 12002, X (onde X é um número de 1 a 255).

— EXEC 46946 — lista o programa da memória na tela.

— EXEC 44759 — Gera a mensagem "?SN ERROR".

2) — Para aumentar o número de naves no programa "ASTRO BLAST", digite POKE 6415, X (X = nº de 1 a 255).

— EXEC 44658 — gera a mensagem "?OM ERROR".

— EXEC 42521 — gera a mensagem "?I/O ERROR".

3) — Para auementar o número de naves no programa "SPACE INVADERS" digite POKE 7674, X (X = nº de 1 a 255).

— EXEC 41175 — coloca no vídeo a mensagem do COLOR BASIC.

— EXEC 32768 — coloca no vídeo a mensagem do EXTENDED COLOR BASIC.

OBS: No caso dos programas, carregue-os (CLOADM), digite o respectivo POKE e depois rode-os (EXEC).

Luiz Garcia — PR

Linha APPLE

ONERR GOTO

Em Applesoft, o comando ONERR GOTO permite contornar erros quando os encontra, antes que seja colocada uma mensagem de erro, ou mesmo que pare a programação. Este comando deve ser dado antes que ocorra o erro, de preferência na primeira linha do programa.

Vejamos um exemplo, com o comando ONERR GOTO 200. Nesse caso, ao encontrar um erro, o programa irá pular para a linha indicada no ONERR GOTO, que é a 200. Nessa linha, o programa irá continuar normalmente, seguindo as instruções da própria linha, que tanto pode reiniciar o programa, como terminá-lo.

A forma usual de tratar erros com o comando ONERR GOTO é escrever uma sub-rotina para onde o programa vá quando ocorra um erro. No final dessa rotina, a declaração RESUME causa o retorno para o início da declaração onde ocorreu o erro.

ONERR GOTO não funciona direito em algumas circunstâncias. Por exemplo, o Apple II trava-se se houver um erro com o comando GET e se a sub-rotina de erro tiver em seu final a declaração RESUME, como mostra o exemplo abaixo:

```
10 ONERR GOTO 100
20 PRINT"ESCREVA UM NUMERO ";:GE
T A
30 PRINT A
40 END
100 PRINT"NAO ESCREVA LETRA ! "
110 RESUME
```

Neste programa, se for teclada uma letra em vez de número, o programa trava.

Ricardo F. de Miranda — SP

Linha TRS-80

Oscilosgráfico

Com esta DICA, você terá no seu micro um gráfico diferente, ou, com algumas modificações, uma nova abertura para seus programas.

Aloysio Soares Paredes — RJ

```
10 CLS:PRINT@450,CHR$(23)"* * *
OSCILOSGRAFICO * * *"
20 FOR X=0 TO 1500:NEXT X
30 PRINT:PRINT:INPUT"GRAU OSCILA
CAO ";A
40 PRINT:INPUT"TEMPO (1 a ...) "
;B
50 CLS
60 FOR Y=0 TO B
70 FOR X=0 TO A
80 PRINT TAB(10);","TAB(30);","T
AB(50);","PRINT TAB(X)"(28 espac
os) **"
90 NEXT X:NEXT Y
100 FOR A=0 TO 0 STEP-1
110 PRINT TAB(10);","TAB(30);","
TAB(50);","PRINT TAB(X)"(28 espa
cos) **"
120 NEXT A
130 GOTO 10
```

Linha SINCLAIR

Inverte a Tela

Esta dica inverte a tela de uma forma diferente da usual, substituindo-a instantaneamente por sua imagem refletida, como em um espelho. Os pixels individuais dos caracteres permanecem inalterados, devido à forma como é mapeado o vídeo e à baixa resolução de um micro da linha Sinclair.

```
16514 - 2A 0C 40 11 10 00 19 54
        50 13 06 16 7E FE 76 20
        09 E8 4E 77 E8 7A 20 13
        10 F2 11 31 00 19 54 50
        13 10 E9 C9
```

Digite agora esse programa, em BASIC, que demonstra o funcionamento da sub-rotina e dê RUN:

```
5 FOR X=1 TO 22
10 PRINT "* ESTE EH UM TESTE DO INVERTE *"
20 NEXT X
30 RAND USR 16514
```

Se ao invés de letras colocarmos determinados desenhos em modo gráfico, em LOOP, teremos a ilusão de movimento tridimensional.

Eduardo Mirabelli A. de Medeiros — SP

Linha SINCLAIR

Armazenamento de imagens

Eis uma forma de armazenar qualquer imagem em uma variável, dentro da questão lançada no artigo "Riscos, Rabiscos e Criatividade" (MS nº 27).

```
10 FAST
20 REM IMAGEM EXEMPLO
30 FOR A=1 TO 704
40 PRINT"X";
50 NEXT A
60 PAUSE 4E4
70 POKE 16437,255
100 LET X$=""
110 LET Z=PEEK 16396+256*PEEK 1639
7+1
120 FOR J=Z TO Z+724
130 IF PEEK J=118 THEN LET J=J+1
140 LET X$=X$+CHR$ PEEK J
150 NEXT J
```

Após a execução, este programa pode ser deletado (sem usar NEW). Para testá-lo, digite PRINT X$ e veja na tela o desenho que foi armazenado na variável X$. Não use CLEAR, RUN ou NEW para que a variável permaneça na memória.

Gino Raniero M. Lozada — RJ

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha SHARP PC-1500

Utilizando rótulos

Facilite a estruturação, a compreensibilidade e a manutenção de seus programas no PC-1500, fazendo uso de rótulos como no exemplo abaixo:

Ao invés de:

50 GOSUB 1000
1000 C = P * N: RETURN

Faça:

50 GOSUB "ROTINA DE CUSTO"
1000 "ROTINA DE CUSTO" C = P * N: RETURN

Também aceitam rótulos os seguintes comandos:

GOTO, IF.... THEN, LIST, LLIST, RESTORE, RUN.

Aldo Rodrigues da Costa — MG

Linha APPLE

Mensagem em parábola

Esta dica serve para mostrar no vídeo a mensagem contida na variável A$. Caso a mensagem exceda os limites da tela, basta modificar a fórmula da linha 30.

```
5 HOME:INPUT"NOME ?";A$:HOME
10 FOR A=0 TO 6.2 STEP .2
20 B=SIN(A)
30 PRINT TAB(8*10+15);A$
40 NEXT A
50 GOTO 10
```

Jorge Pablo Zapata — BA

Linha TK 2000

Amplie caracteres

Com esta dica, você poderá ampliar os caracteres do seu TK 2000. Digite as linhas abaixo e rode com RUN, depois apague-os:

```
1 DATA 165,9,162,7,173,0,64,133,
0,6,0,6,0,144,5,165,2,32,0,248,1
36,202,208,243,230,2,173,6,3,24,
105,4,141,6,3,201,64,208,217,96
2 FOR F=768 TO 807:READ P:POKE F
,P:NEXT F
```

Para usar o programa, digite:

```
10 HOME:INPUT"QUAL A COR ?
";A:PRINT:INPUT"QUAL A LETRA ?
   ";A$
20 GR:POKE 48,A:HTAB 1:VTAB 1:PRI
NT A$
30 POKE 774,32:POKE 2,10
40 CALL 768
50 GET A$:TEXT:RUN
```

William Joseph Watterson — SP

Linha ZX81

Caleidoscópio

Aí vai uma Dica que explora a capacidade gráfica do seu compatível com ZX81. Após rodar o programa, mude o seu efeito alterando a linha 30, para:

```
30 LET A$=CHR$ INT (RND*10+128)
```

```
30 LET A$=CHR$ INT (RND*10+128)+
CHR$ INT (RND*10+1)
```

Podendo então alterar o comprimento da variável A$.

```
1 REM -- CALEIDOSCOPIO --
10 CLS
20 LET A=1
30 LET A$=CHR$ INT (RND*10+1)
40 LET B=20
50 LET C=B*RND
60 LET D=B*RND
70 PRINT AT C,D;A$
80 PRINT AT D,C;A$
90 PRINT AT C,B-D;A$
100 PRINT AT B-D,C;A$
110 PRINT AT D,B-C;A$
120 PRINT AT B-C,D;A$
130 PRINT AT B-C,B-D;A$
140 PRINT AT B-D,B-C;A$
150 LET A=A+1
160 IF A=20 THEN RUN
170 GOTO 50
```

José Rogério Prates — SP

Linha ZX Spectrum

UDG 2 do TK90X

A função UDG 2 do TK90X é uma *mão na roda* para redefinir os caracteres do micro, mas nem tudo foi dito no manual. Experimente apertar uma tecla de A a U, acompanhada de CAPS SHIFT e você verá que pode passar a redefinição da tecla pressionada para a letra que está em edição.

Ricardo Costa — RJ

Linha ZX Spectrum

Memória disponível

Para saber quanto resta de memória em seu micro, digite as linhas abaixo e rode o programa. Depois, sempre que quiser saber quanto sobra de memória, use PRINT USR 23300.

A rotina fica guardada na área do buffer da impressora, e se você quiser colocá-la em outro local da memória, basta mudar os endereços da linha 10 antes de rodar o programa.

```
10 FOR F=23300 TO 23310
20 READ A:POKE F,A:NEXT F
30 DATA 205,26,31,42,178,92,23,7
,66,48,77,201
```

Carlos Alberto Guerreiro — MG

Linha APPLE

Impressão profissional

Esta dica serve para escrever em seus joguinhos ou aplicativos pequenas mensagens com mais elegância. A mensagem deve ser introduzida na variável A$, que se encontra na linha 5. Para mudar a linha, onde vai ser impressa a mensagem, basta mudar o VTAB da linha 30.

```
5 HOME:INPUT"FRASE ?";A$:HOME
10 FOR A=1 TO LEN(A$)
30 VTAB 1:HTAB A
40 B$=MID$(A$,A,1)
45 SOUND ASC(B$)+20,4
50 PRINT B$;CHR$(242);CHR$(199)
60 NEXT A
```

Jorge Pablo Zapata Rivera — BA

Linha ZX81

Efeito na tela

Esta rotina provoca nos computadores da linha ZX81 um efeito muito interessante no vídeo, podendo ser usada também em qualquer programa:

```
10 REM .....(5 caracteres
 quaisquer)
20 POKE 16514,62
30 POKE 16515,48
40 POKE 16516,237
50 POKE 16517,71
60 POKE 16518,201
70 RAND USR 16514
80 FOR A=0 TO 100
90 POKE 16515,A
100 RAND USR 16514
110 NEXT A
```

Você notará que após rodar a rotina, todos os caracteres estarão indefinidos; não se alarme, para voltar ao normal é só dar estes comandos diretos:

POKE 16515,30 (new line) e
RAND USR 16514

Quem Woo Moo — SP

Linha APPLE

Conversão DEC-HEX

Muitas vezes, se faz necessária uma rápida conversão Decimal-Hexadecimal. Normalmente, um programa para esse fim, além de ser lento, necessita de uma série de manipulações de "Strings" e é um pouco dispendioso.

Mas, neste programa, a conversão é feita usando uma rotina intrínseca do Monitor, o que nos garante uma alta velocidade e uma simplicidade maior.

Esta rotina "PRINTA" o conteúdo dos registros XY em HEXA, e seu endereço é $FD99.

Observação: O valor máximo decimal é 65535.

```
10 HOME
20 REM DEC-HEX
30 PRINT"ENTRE COM O NUMERO DEC
";:INPUT Z:PRINT:PRINT
40 K=INT(Z/256):F=Z-K*256
50 X=F:Y=K
60 POKE 768,162:POKE 769,X:POKE
770,160:POKE 771,Y
70 POKE 772,32:POKE 773,153:POKE
 774,253
75 POKE 775,96
80 CALL 768
90 PRINT:PRINT
100 GOTO 30
```

Alexandre Magno Rodrigues Alves — SP

Linha ZX81

Funções especiais

Use esta dica para descobrir o nome de um programa em seu TK85.

```
10 LET Z=0
20 LET Z$="A"
30 DIM A$(100)
40 RAND USR 8305
50 FOR I=1 TO LEN A$
60 IF CODE A$(I)<128 THEN NEXT I
70 PRINT "PROGRAMA: ";A$(TO I)
```

Enrique Raul Broens — PR

Linha TRS-80

Destrave o LIST no TRS-80

Com o seguinte POKE, qualquer programa com LIST todo bagunçado poderá ser listado SOMENTE NA TELA. POKE 17129,239. Se não funcionar, desligue e ligue o computador, recarregue o programa e dê POKE 17130,66. Se ainda não funcionar, desligue e ligue o computador e vá direto para o BASIC residente, dê um PRINT PEEK (17129), PEEK (17130), anotando os valores impressos e, depois, entre no DOS. Carregue o mesmo programa e dê POKE 17129, n1: POKE 17130, n2, sendo n1 e n2 os valores que você tinha adotado.

Bruno Lopes Fernando Cabral — PB

Linha APPLE e TK 2000

Gravação de dados

Um dos grandes problemas do Applesoft/Basic é a gravação de dados alfanuméricos em fita, o que dificulta muito a criação de programas do tipo Banco de Dados, por exemplo.

A solução é simular uma função que execute com precisão e segurança. É o que as rotinas abaixo propõem.

ROTINA 1:

```
10 HOME
20 CLEAR
30 DIM A(255),A$(255)
40 INPUT "TEXTO: ";B$
50 A(0)=LEN(B$)
60 FOR I=1 TO A(0)
70 A$(I)=MID$(B$,I,1)
80 NEXT I
90 FOR I=1 TO A(0)
100 A(I)=ASC(A$(I))
110 NEXT I
120 PRINT "PRONTO?":GET X$
130 STORE A
140 PRINT:PRINT "OK"
```

ROTINA 2:

```
10 HOME
20 CLEAR
30 DIM A(255),A$(255)
40 PRINT "PRONTO?":GET X$
50 RECALL A
60 FOR I=1 TO A(0)
70 A$(I)=CHR$(A(I))
80 NEXT I
90 PRINT "TEXTO: ";
100 FOR I=1 TO A(0):PRINT A$(I);
:NEXT I
```

Estes dois pequenos programas rodam sem problema no TK 2000. É ACONSELHÁVEL, porém, que estes sejam depurados para funcionarem perfeitamente, de acordo com as necessidades de cada um.

Carlos Henrique C. Duarte — MG

Linha ZX81

INKEY$ para várias teclas

Diversas vezes, num programa em BASIC, temos necessidade de interpretar mais de uma tecla, pressionada simultaneamente.

Para resolver tal problema, pode ser utilizada uma variável do sistema contida nos endereços 16421 e 16422, conseqüentemente, o valor das teclas pressionadas, será:

```
PEEK 16421+256*PEEK 16422
```

Basta guardar o resultado em uma variável e, a partir daí, fazer todo o resto. Veja um exemplo com este pequeno programa que interpreta apenas o pressionar simultâneo de 5 e 0 ou 8 e 0, muito utilizado em alguns jogos.

```
10 CLS
20 LET X=PEEK 16421+256*PEEK 16422
30 IF X=56807 THEN PRINT AT 10,16;"5
 E 0"
40 IF X=62959 THEN PRINT AT 10,16;"8
 E 0"
50 GOTO 10
```

Os valores das teclas pressionadas podem ser obtidos a partir da seguinte rotina:

```
9000 PRINT AT 21,0;PEEK 16421+256*PE
EK 16422
9010 GOTO 9000
```

Linha TK 2000

Chamadas em Assembler

Duas dicas para os novos usuários do TK 2000 que acham que call's no TK 2000 não podem ser usados com os do Apple. É um engano. Por exemplo, digite a seguinte rotina:

```
50000 DATA 160,1,162,0,138,24,23
3,1,208,252,141,48,192,232,224,2
55,208,242,236,208,237,96
50010 FOR A=768 TO 769:READ B:PO
KE A,B:NEXT
50020 FOR A=1 TO 255 STEP 3:CALL
 768:NEXT
```

E veja que, tanto no Apple quanto no TK 2000, ela produz um interessante efeito sonoro. Portanto, POKEs a partir do endereço 768 rodam perfeitamente no TK 2000.

Para simular o CALL -958 do Apple, basta substitui-lo por PRINT SPC (39): HTAB 1. Os CALL -936 do Integer BASIC de alguns Apple II, também rodam perfeitamente no TK 2000, e sem modificações.

Bruno Lopes Fernando Cabral — PB

Melhorar a impressão no vídeo, criar gráficos em alta resolução ou proteger programas são algumas dicas que este artigo lhe oferece.

Festival de dicas no TK90X

André Koch Zielasko
Alexandre Mariano de Brito

Nada melhor do que uma boa dica quando se está diante de um problema ou mesmo quando é necessário aprimorar uma idéia.

Para auxiliá-lo nesta tarefa, MS reuniu algumas sugestões enviadas por nossos colaboradores. Através delas, você conhecerá um pouco mais dos recursos que o TK90X lhe oferece e aprenderá ainda a explorá-los melhor.

Portanto, se você é um usuário do TK90X, seja *calouro* ou *cobra*, estas rotinas poderão ajudá-lo a colocar seus conhecimentos em prática, ou, no mínimo, distraí-lo.

Por isso, mãos à obra, ou melhor, ao micro!

1 – IMPRESSÃO

Uma característica particular do TK90X é o uso do apóstofo('): quando ele é utilizado, a próxima impressão tem início na linha seguinte (no começo). Podem ser usados tantos apóstrofos quantos desejados, sendo que cada um corresponderá ao pulo de uma linha. Para verificar isto, rode o programa abaixo e repare o resultado.

```
10 PRINT "uma linha"
20 PRINT "outra linha"
30 PRINT "uma linha"''"outra li
nha"
40 PRINT ''''"pula 2 linhas"
```

É possível também imprimir nas linhas de edição, como mostra o programa a seguir:

```
10 FOR i=1 TO 5
20 PRINT #0;TAB i;"linha de ed
icao=";i
30 PRINT "linha de programa=";
i
40 NEXT i
50 PAUSE 0
```

Você deve ter notado que, ao final da execução, aparecem escritas apenas as mensagens da linha 30, enquanto que a da linha 20 é apagada. Acrescente a linha **50 PAUSE 0** ao programa e rode-o novamente. Observe que a instrução **PAUSE 0** interrompe o programa até que qualquer tecla seja pressionada, permitindo que a mensagem da linha de edição seja visualizada.

Outro recurso que pode ser utilizado para se escrever na linha de edição é dar POKEs na área de memória reservada para o vídeo. Rode e analise o programa a seguir (os endereços podem ser encontrados no manual).

```
10 LET a$="uma linha"
20 FOR n=1 TO LEN a$
30 LET b=CODE a$(n)
40 LET c=15360+b*8
50 LET d=20671+n
60 FOR p=0 TO 7
70 POKE d+(256*p),PEEK (c+p)
80 NEXT p
90 NEXT n
100 PAUSE 0
```

Também é possível escrever uma mensagem ao contrário, utilizando-se alguns caracteres que têm funções especiais de controle, como o volta cursor, representado por CHR$ 8. O programa a seguir realiza esta tarefa.

```
10 LET a$="MICRO SISTEMAS"
20 PRINT AT 10,9;a$;
30 LET v$=CHR$ 8
40 FOR i=1 TO LEN a$
45 PAUSE 60
50 PRINT v$+a$(i)+v$;
60 NEXT i
```

Outros caracteres de controle também podem ser usados livremente, obtendo-se resultados diferentes. Experimente o programa a seguir e tente descobrir como os resultados foram obtidos.

```
10 LET a$=CHR$ 22+CHR$ 10+CHR$
10+"==="
20 LET b$=CHR$ 23+CHR$ 20+"XXX
"
30 LET c$=CHR$ 21+CHR$ 1+"///"
40 PRINT a$+b$
50 PRINT AT 10,10;c$
```

2 – ENTRADA DE DADOS

No TK90X, quando uma instrução INPUT que se refere a uma variável do tipo string é executada, um par de aspas aparece na área de entrada de dados, indicando ao programador que uma variável do tipo string está sendo lida. Porém, é possível controlar este aparecimento de aspas através da instrução **INPUT LINE**. Rode o programa abaixo e perceba a diferença de execução das linhas 10 e 20.

```
  10 INPUT "Digite uma palavra:"
;a$
  20 INPUT "Outra:"; LINE b$
  30 PRINT a$,b$
```

A instrução TAB também poderá ser usada no INPUT. Experimente no programa anterior substituir a linha 20 por **20 INPUT TAB 30; "Outra: "; LINE b$**. Os atributos de COR utilizados com o **PRINT(PAPER,INK,FLASH,BRIGHT)**, também poderão ser usados. Experimente este programa.

```
  10 INPUT INK 3; FLASH 1;"Seu n
ome?";n$
  20 PRINT INK 4;"Bom dia, ";n$
```

3 – UTILIZANDO O JOYSTICK

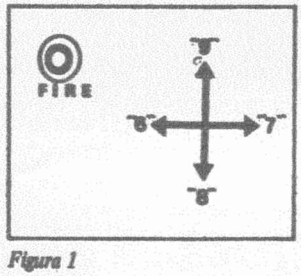


Figura 1

Podemos usar o joystick comum do videogame Atari conectado ao TK90X, obtendo, assim, um ótimo meio de interagir com jogos de ação. A figura 1 mostra a tecla correspondente para cada posição do joystick, e o programa a seguir mostra uma utilização do mesmo.

```
  10 LET a$=INKEY$
  20 IF a$="" THEN GOTO 0
  30 PRINT ("esquerda" AND a$="6
");("direita" AND a$="7");("desc
e" AND a$="8");("sobe" AND a$="9
");("fogo" AND a$="0")
  40 GOTO 10
```

Outro modo de utilizar o joystick (e, por analogia, qualquer tecla!) é através da instrução **IN**. Neste caso, podemos, inclusive, identificar por programa uma direção intermediária. Isto corresponderia, em termos de teclado, a identificar duas teclas pressionadas simultaneamente, o que é altamente desejável ao se programar bons jogos de ação com o uso do joystick, mas não se consegue com a instrução **INKEY$**. Rode o próximo programa e veja os novos resultados. A instrução **IN** é usada neste caso para *ler* o valor de um conjunto de cinco teclas. O endereço 61438, que aparece na linha 10 do programa abaixo, está associado às teclas de 0 a 6.

```
  10 LET s=IN 61438-43
  20 IF s=20 THEN GOTO 10
  30 PRINT ("fogo" AND s=19);("s
obe" AND s=18);("sobe direita" A
ND s=10);("direita" AND s=12);("
desce direita" AND s=8);("desce"
 AND s=16);("desce esquerda" AND
 s=0);("esquerda" AND s=4);("sob
e esquerda" AND s=2)
  40 GOTO 10
```

4 – USANDO VARIÁVEIS DO SISTEMA

Se você quer gerar efeitos não encontrados no BASIC, deixar um programa mais rápido ou torná-lo mais caprichado, a utilização de POKEs na área de variáveis do sistema poderá ajudá-lo e com algumas vantagens. Dentre os endereços e valores existentes, destacamos os seguintes:

a) 23560 - serve para indicar qual tecla foi pressionada (semelhante ao INKEY$). Experimente (e estude) este programinha:

```
10 PAUSE 0: PRINT CHR$ (PEEK 2
3560);: GOTO 10
```

Tecle algo, como se estivesse usando uma máquina de escrever (inclusive com espaço e SHIFT).

b) 23561 - tempo de espera para começar a repetir a tecla que estiver sendo pressionada. O valor mínimo (que dá a maior rapidez) é .5 e o valor normal (quando se liga o micro ou após um NEW) é 35.

c) 23562 - tempo de retardo na repetição de teclas. O valor mínimo é .5 e o normal é 5.

d) 23606 - endereço inicial da tabela de caracteres do micro, que pode
23607 ser modificada à vontade.

e) 23609 - pode-se variar a tonalidade do sinal que é emitido ao se pressionar alguma tecla, colocando neste endereço qualquer valor entre 0 (que é o normal, logo que se liga o micro) até 255.

f) Conjugando duas variáveis, podemos descobrir a quantidade de memória que um programa ocupa. Digite, no modo direto:

```
PRINT (PEEK 23641+256*PEEK 23642
)-(PEEK 23635+256*PEEK 23636)
```

5 – IMPRESSÃO CAPRICHADA

Além dos recursos mencionados no *capítulo 17 do manual do TK90X*, é possível melhorar a impressão no vídeo, utilizando-se das seguintes técnicas:

FLASH - use CAPS SHIFT e SYMBOL SHIFT para obter o cursor E; depois pressione CAPS SHIFT e 9. Tudo o que for escrito agora daqui em diante ficará piscando. Para desativar o FLASH, use (no modo E) as teclas CAPS SHIFT mais 8.

BRIGHT - use CAPS SHIFT e SYMBOL SHIFT para obter o cursor E; depois pressione apenas a tecla 9, sozinha. Agora tudo será escrito com brilho (como BRIGHT 1). Para desativar, pressione apenas a tecla 8, sozinha, após ter entrado no modo E.

PAPER - pressione CAPS SHIFT e SYMBOL SHIFT para obter o modo E; e depois tecle apenas o número correspondente à cor que você deseja para usar no fundo do caráter (PAPER). Para desativar, ponha o microcomputador no modo E e pressione a tecla 7 sozinha.

6 – PROGRAMAS INTERESSANTES

Experimente digitar estes pequenos programinhas, que servirão para demonstrar alguns dos recursos e particularidades do TK90X, além de servir de distração.

```
 10 FOR X=175 TO 0 STEP -1: FOR
Y=0 TO 255: PLOT INK RND*7;Y,X:
 NEXT Y: LET X=X-RND*8: NEXT X:
STOP
```

```
 10 FOR X=175 TO 0 STEP -1: FOR
Y=0 TO 255: PLOT INK RND*7;Y,X:
NEXT Y: NEXT X: STOP
```

```
  10 LET A=1: LET B=1: FOR D=1 T
O 185 STEP 5: CIRCLE D,A,B: LET
A=A+2: LET B=B+2: NEXT D: STOP
```

```
  10 INPUT "Entre com um numero
! ";B: CLS : LET A=0: PLOT 0,175
: DRAW 0,-175: PLOT 0,0: DRAW 17
5,0: FOR D=175 TO 0 STEP -B: PLO
T 0,D: DRAW A,-D: LET A=A+B: NEX
T D: GOTO 10
```

7 – SCROLL COM A ROM

Mesmo possuindo um prático recurso de SCROLL para listagem de programas, o TK90X não tem um comando que realize esta função.

Para se obter um SCROLL em seus jogos, basta utilizar a rotina da ROM (a mesma que é usada na listagem dos programas) chamada por **RAND USR 3280**. Teste o seguinte programa-exemplo:

```
  10 PRINT AT 21,9;"Micro Sistem
as"
  20 FOR F=0 TO 20
  30 RAND USR 3280: NEXT F
```

8 – ZERANDO LINHAS

Às vezes, é necessário zerar a primeira linha de um programa para protegê-la contra um **DELETE** indesejável ou simplesmente zerar uma ROM com o nome do autor para dificultar a ação dos piratas. Porém, é muito mais interessante zerar linhas no meio do programa, a fim de dificultar – e muito – a ação dos piratas, vejamos como:

Digite os comandos abaixo numa linha que antecede a que será zerada:

```
 LET A=PEEK 23637+256*PEEK 23638
: POKE A,0: POKE A+1,0
```

e, a seguir, tecle **GOTO** (o número da linha acima).

Por exemplo, no programa abaixo, para zerar a linha 20, digite os comandos na linha 15, depois tecle **GOTO 15** e pronto! Mas não se esqueça de apagar esta linha depois. Agora, tente apagar a linha zerada.

```
  10 PRINT
  15 LET A=PEEK 23637+256*PEEK 2
3638: POKE A,0: POKE A+1,0
  20 PRINT "Linha a ser zerada"
  30 PRINT
```

9 – GRÁFICOS EM ALTA RESOLUÇÃO

À seguir, quatro programas gráficos decorativos muito simples, que podem ser modificados à vontade, podendo ainda ser usados como efeitos visuais em outros programas.

1º) **Rosca** – Este programa pode gerar aleatoriamente até 17 tipos de gráficos (podem ser triângulos, quadrados, estrelas etc.), que, no final, sempre formam uma figura semelhante a uma *rosca*.

```
  10 BRIGHT 1: CLS : FOR A=95 TO
255 STEP 10

  20 PLOT 75,37: DRAW INK (RND*7
);100,100,A*PI
  30 CLS : NEXT A
```

2º) **Efeitos com DRAW** – Estes dois programas geram lindos gráficos. O primeiro forma uma circunferência e o segundo uma *esteira* que se dobra.

```
  10 BORDER 0: PAPER 0: BRIGHT 1
: OVER 1: CLS
  20 PLOT 65,27: DRAW INK (RND*7
);100,100,PI*255: GOTO 20
```

```
  10 OVER 1: LET X=0: LET X1=255
: LET Y=0: INK (RND*7): CLS
  20 PLOT X,Y: DRAW X1,Y: LET Y=
Y+1: LET X1=X1-6: LET X=X+3: IF
Y<=175 AND X1>-255 THEN GOTO 20
  30 GOTO 10
```

3º) **Retângulos** – Compõe diversos retângulos que formam desenhos harmônicos em toda a tela e depois os apagam.

```
  10 PAPER 6: CLS
  20 LET I=1: LET X=0: LET Y=0:
LET X1=255: LET X2=0: LET X3=-25
5: LET X4=0: LET Y1=0: LET Y2=17
5: LET Y3=0: LET Y4=-175
  30 PLOT X,Y: INK 1: OVER 1: DR
AW X1,Y1: DRAW X2,Y2: DRAW X3,Y3
: DRAW X4,Y4
  40 LET X=X+2: LET Y=Y+2: LET X
1=X1-4: LET X3=X3+4: LET Y2=Y2-4
: LET Y4=Y4+4: IF Y<=175 THEN GO
TO 30: LET I=I+1
  50 GOTO 20
```

4º) **Quadrado** – Desenha uma forma *quadrada* decorando o seu interior.

```
  10 LET X=0: LET X1=175: OVER 1
  20 PLOT X,0: DRAW INK 1;X1,175
: PLOT 0,X: DRAW INK 1;175,X1
  30 LET X1=X1-8: LET X=X+4: IF
X<=175 THEN GOTO 20
  40 GOTO 10
```

André Koch Zielasko estuda as linguagens BASIC, FORTRAN, COBOL e Assembler, além de pesquisar o emprego dos micros de lógica Sinclair nas áreas de Astronomia e Telecomunicações. André desenvolveu as dicas 4, 5 e 6.

Alexandre Mariano de Brito é estudante e programa em BASIC e Assembler nos microcomputadores TK85 e TK90X, sendo ainda o responsável pelas três últimas dicas.

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha SINCLAIR

Título animado

Esta rotina em BASIC cria no seu micro uma interessante apresentação do conteúdo da variável A$ da linha 10, que deverá ter no máximo 32 caracteres. Na linha 9040, você pode controlar a linha final da impressão da sua frase, que, neste exemplo, é 15. Use-a como sub-rotina para incrementar seus programas.

```
  10 LET A$="MICRO SISTEMAS"
  20 GOSUB 9000
  30 STOP
9000 FOR X=1 TO LEN A$
9010 IF A$(X)=" " THEN NEXT X
9020 LET Z=(15-LEN A$/2)+X
9030 GOSUB 9090
9040 FOR A=20 TO 15 STEP -1
9050 PRINT AT A;Z;A$(X);AT A+1,Z
     ;" "
9060 NEXT A
9070 NEXT X
9080 RETURN
9090 FOR A=30 TO Z STEP -1
9100 PRINT AT 21,A;A$(X);" "
9110 NEXT A
9120 RETURN
```

Carlos Alexandre da Silva — RJ

Linha APPLE

PEEKs e POKEs I

- Retorne o endereço de início das variáveis:
  PEEK(105) + 256 * PEEK(106)
- Retorne o endereço de início das matrizes:
  PEEK(107) + 256 * PEEK(108)
- Retorne o endereço do fim das variáveis numéricas:
  PEEK(109) + 256 * PEEK(110)
- Retorne o endereço de início das Strings:
  PEEK(111) + 256 * PEEK(112)
- Retorne o valor do HIMEM:
  PEEK(115) + 256 * PEEK(116)
- Retorne o endereço do fim do espaço das variáveis:
  PEEK(204) + 256 * PEEK(205)
- Faça a primeira linha ser listada continuamente quando dermos LIST:
  POKE 2049,1.

Esta dica também serve para o TK 2000.

PEEKs e POKEs II

- Retorne o valor de Maxfiles:
  PEEK(43607)
- Retorne o último número atribuído ao parâmetro D:
  PEEK(43624)
- Retorne o último número atribuído ao parâmetro S:
  PEEK(43626)
- Retorne o volume do disco:
  PEEK(46017)

Obs.: Estes valores se aplicam a sistemas com 48 Kb. Para os de 32 Kb, subtraia 16384 do valor. Para sistemas com 16 Kb, subtraia 32768 do valor.

Bruno Cabral — PB.

Linha ZX Spectrum

Túnel e pirâmide

Veja como o comando DRAW pode ser útil na criação de figuras geométricas, em seu compatível com o ZX Spectrum:

```
 5 INK 0: PAPER 6: BORDER 1
10 FOR E=0 TO 127
20 PLOT E,E: DRAW 255-E+2,0
30 NEXT E
```

```
 5 INK 0: PAPER 6: BORDER 1
10 FOR F=60 TO 0 STEP -2
20 PLOT F,F: DRAW 255-F+2,0
30 DRAW 0,175-F+2
40 DRAW -255+F+2,0
50 DRAW 0,-175+F+2
60 NEXT F
```

Teotônio Rodrigo de Carvalho — MG

Linha TRS-COLOR

PAUSA

Freqüentemente é necessário interromper a execução de um programa até que o usuário esteja pronto para continuar. Um método empregado para conseguir isto é o que se segue:

```
5000 PRINT "APERTE QUALQUER TECLA PARA CONTINUAR"
5010 A$ = INKEY$ : IF A$ = " " THEN 5010
```

O único problema é ter que ficar repetindo o **IF INKEY$ = " " THEN**... toda vez que uma pausa for necessária. Os usuários do Color podem fazer isso de uma forma mais simples: é só incluir no começo do programa uma linha do tipo:

10 PA = 44539 (ou outra variável qualquer)

A partir daí, basta usar **EXEC PA** toda vez que for necessário interromper o programa até que alguma tecla seja pressionada.

Cláudio Costa — RJ

Linha TRS-80

Trave o BREAK

Dê um POKE 16396,165. A tecla BREAK não BREAKará o programa. Ele fica inerte, sem efeito nenhum. Com isto, no DOS, se você gravar seu programa incluindo o POKE logo no começo, quando o *pirata* der o RUN "nome/sigla" e tentar dar o BREAK, dançará ao som de nossa música.

Bruno Lopes Fernando Cabral — PB

Linha APPLE

Palavras em movimento

É muito simples, usando os pequenos programas abaixo você irá causar um bonito efeito em seu vídeo. Porém há um inconveniente, a frase (que no caso está sendo representada pela string A$) não pode ter mais de 40 (quarenta) caracteres, o que equivale a uma linha.

As duas linhas abaixo serão comuns a todos os programas, ou seja, todos têm as linhas 10 e 20 iguais:

```
10 HOME
20 A$=" * M.S. A PRIMEIRA EM MIC
ROCOMPUTADOR * "
```

A. Da direita para a esquerda:

```
30 FOR X=1 TO 40
40 VTAB 1:HTAB 41-X:PRINT LEFT$(
A$,X)
50 FOR R=1 TO 75
60 NEXT R,X
70 END
```

B. Da esquerda para a direita:

```
30 FOR X=1 TO 40
40 VTAB 1:PRINT RIGHT$(A$,X)
50 FOR R=1 TO 75
60 NEXT R,X
70 END
```

C. De cima para baixo:

```
30 FOR X=2 TO 23
40 VTAB X-1:CALL-868
50 VTAB X:PRINT A$
60 FOR R=1 TO 75
70 NEXT R,X
80 END
```

D. De baixo para cima:

```
30 FOR X=23 TO 1 STEP-1
40 VTAB X+1:CALL-868
50 VTAB X:PRINT A$
60 FOR R=1 TO 75
70 NEXT R,X
80 END
```

Ricardo F. de Miranda – SP

Linha ZX Spectrum

Raios coloridos

O DRAW é um recurso gráfico com efeito, principalmente quando é usado com um ponto de partida (PLOT, CIRCLE). Para provar isso, aí vai uma dica quente:

```
   1 REM  Micro Sistemas
Carlos Henrique Olifiers-12 anos
  10 INPUT "Qual e' o numero de
DRAWS ? ";A
  20 CLS
  30 FOR B=1 TO A
  40 INK RND*5
  50 LET C=INT (RND*70)+2
  60 LET D=INT (RND*80)+1
  70 PLOT C,D
  80 DRAW C,D
  90 NEXT B
 100 GOTO 10
```

Carlos Henrique Olifiers – RJ

Linha TRS-COLOR

INKEY$

A maneira mais comum de simular movimento em jogos animados escritos em BASIC (quando, por qualquer motivo, não se deseja usar o joystick) é utilizar INKEY$ em combinação com as teclas das setas. As seguintes instruções movem um bloco amarelo para a esquerda, quando se pressiona a tecla (CHR$ 8); e para a direita, se for pressionada a tecla (CHR$ 9):

```
5 CLS 0 : X$ = CHR$ (128) + CHR$ (159) + CHR$ (128) : N = 400
7 PRINT @ N,X$;
9 I$ = INKEY$ : N = N + (I$ = CHR$(8) AND N > 383) – (I$ = CHR$
  (9) AND N < 414) : GOTO 7
```

A desvantagem desse método, além de ser muito lento, é que você tem que largar a tecla e pressioná-la novamente toda vez que quiser mover o bloco uma posição. Para resolver os dois problemas de uma só vez, dê uma olhada na tabela abaixo: ela fornece as posições da memória correspondentes às teclas das setas que, quando pressionadas, assumem o valor 247. Agora, é só substituir INKEY$ pelos valores das teclas. Rode o exemplo a seguir e veja a diferença:

```
9 N = N + (PEEK(343) = 247 AND N > 383) – (PEEK(344) = 247 AND
  N < 414) : GOTO 7
```

TECLA	POSIÇÃO
↑	341
↓	342
←	343
→	344

Cláudio Costa – RJ

Linha ZX81

Explosão

A rotina abaixo apresentada se propõe a criar uma explosão na tela. Para digitá-la, crie uma linha (REM) com 22 caracteres e para usá-la, basta dar o (RAND USR) 16514. Uma boa dica para ser utilizada em jogos.

```
16514  06 0A 0E 00 16 1E 7A ED
16522  47 15 20 FA 0D 20 F5 10
16530  F1 3E 1E ED 47 C9
```

Aloisio Carlos Tortelly Costa Jr. – RJ

Linha TRS-80

Desenhos

Desenhe círculos, elipses e outros efeitos com esta dica. Para criar novas figuras, mude os parâmetros da linha 40.

```
5 CLS
20 FORF=0TO2*3.1416STEP3.1416/18
30 FORN=1.5TO20
40 SET(60+2*N*SIN(F),20+N*COS(F))
60 NEXTN
70 NEXTF
80 GOTO5
```

João Luiz Pacheco Vieira Galvão – GO

Linha SINCLAIR

Conversão decimal/binário

O TK90X possui uma função (BIN), que permite se obter um número inteiro a partir do seu correspondente binário.

Use esta dica para conseguir o inverso, ou seja, tendo o número inteiro, conseguir o seu binário correspondente:

Para saber se o bit B do número N é 0 ou 1, faça:

```
LET L=INT (N/2**B)-2*INT ((INT (N/2**B))/2)
depois use PRINT L, para saber o valor do bit.
```

Para ver todos os bits de um número N de 2 bytes, rode o seguinte programa:

```
  10 INPUT N
  20 PRINT N;TAB 10;
  30 FOR B=15 TO 0 STEP -1
  40 PRINT INT (N/2**B)-2*INT ((
INT (N/2**B))/2);
  50 IF B=8 THEN PRINT " ";
  60 NEXT B
```

Para rodar esta dica nos compatíveis com ZX Spectrum, substitua os sinais ** pelo sinal de potenciação(↑).

Cláudio Bittencourt – SP.

Linha APPLE

Gráfico tridimensional

Este programa faz um gráfico que parece saltar da tela. É ótimo para ser inserido no final de seus programas, dando um toque pessoal.

```
5  HGR2 : HCOLOR= 3
10  ROT= 0
20  FOR G = 1 TO 64: SCALE= G
30  DRAW 1 AT 140,96: NEXT
40  POKE 232,0: POKE 233,3
50  FOR X = 768 TO 790
60  READ Y: POKE X,Y: NEXT
70  DATA 1,0,4,0,36,172,46,46,62,
     246,228,22,39,56,246,32,39,3
     7,37,12,54,6,0
80  GET FIM$: TEXT : HOME
```

Mário César Franchi — SP

Linha TRS-80

Pontuação monetária

Use esta dica para pontuar valores monetários em números de até 16 algarismos.

```
10 CLEAR1000
20 CLS:INPUT"DIGITE UM NUMERO";N#
30 C$=STR$(N#):GOSUB92
40 PRINTC$
50 PRINT@860,:INPUT"OUTRA CONSULTA
 S/N";M$
60 IFM$="S"THEN20ELSEEND
92 IFLEN(C$)<=4THENRETURNELSEFORX1
=LEN(C$)-1TO1STEP-3:IFX1<=3THEN94E
LSEJ$="."+MID$(C$,X1-1,3)+J$:NEXT
94 IFX1=0THENX1=4ELSEX1=X1+1
96 C$=MID$(C$,1,X1)+J$
98 J$="":N#=0:RETURN
```

Mervyn de Oliveira — MG

Linha SINCLAIR

Abertura de programas

Rode esta dica no seu Sinclair e verá que é uma abertura interessante para qualquer programa.

```
10 LET A$= "MICRO"
20 LET B$= "SISTEMAS"
30 FOR A=0 TO 7
40 PRINT AT 9,A;A$
50 NEXT A
60 FOR B=23 TO 14 STEP -1
70 PRINT AT B,B;B$
80 NEXT B
90 LET C$= APRESENTA
100 LET C=INT ((31,2)-(LEN C$/2))
110 FOR A=1 TO LEN C$
120 FOR B=21 TO 12 STEP -1
130 PRINT AT B,C;C$(A);AT B,C
140 NEXT B
150 PRINT AT B,C;C$(A)
160 LET C=C+1
170 NEXT A
180 PAUSE 200
190 RUN
```

Paulo Henrique Jurisato — SP

LINHA ZX81/FORTH

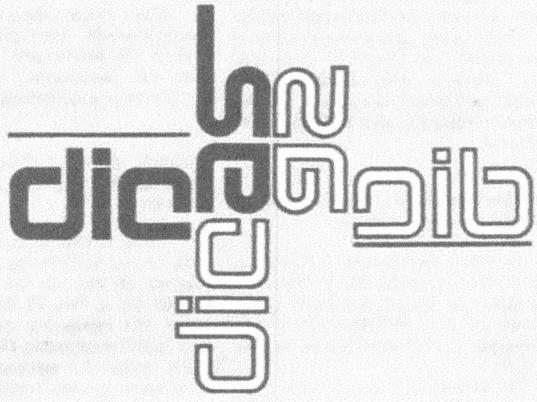
GRAFCOD

Este comando em FORTH lista no vídeo os caracteres gráficos do seu ZX81, com seus respectivos códigos em decimal, hexadecimal ou qualquer outra base de numeração, bastando apenas especificá-la antes de dar o comando GRAFCOD:

2 BASE $ GRAFCOD.

```
DECIMAL
 GRAFCOD
PAGE . "CODIGO DOS CARACTERES GR
AFICOS"
[illegible] + 2 PTC
[illegible] - I EMIT
[illegible] + 16 PTC
[illegible] + DUP . " - " EMIT
LOOP
CR KEY DROP
```

Fabio A. Rodrigues Correa — SP



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha COLOR

Warm Start

Acrescentando à dica publicada em **MS nº 45** (PEEKS e POKES no COLOR), informo que para se conseguir um Warm Start preparado no computador, deve-se digitar: POKE 113, 85 e em seguida RESET.

Fernando Monteiro Guerrero — MG

Linha MC 1000

Mensagem de erro

Quando há problemas em um programa e você está num dos modos gráficos do MC 1000 fica difícil ler a mensagem de erro pois ao voltar para o modo TEXT, normalmente é executado um HOME antes.

Mas, com esta dica, você poderá ver facilmente qual foi a mensagem:

Primeiramente, vamos provocar um erro no modo GR:

```
10 GR:COLOR = 2
20 PLOT 0,0 TO 128,0
```

Como podemos observar, a linha 20 passou os limites do modo GR e para que o erro ocasionado possa ser visto, digite DEBUG. Após ter visto o erro, volte ao BASIC e digite TEXT.

Kleber Desan — PR

Linha TRS-80

POKES no CP 300

Tenho um CP 300 há um ano e estou enviando algumas dicas simples, mas que eu "apanhei" muito para descobrir. Por não possuir impressora, os programas de um amigo que a tinha nunca rodavam no meu micro. Assim, estas dicas servem para ajudar jovens programadores para que não "apanhem" como eu:

Desvio do Vídeo: desvia para a impressora toda informação que normalmente aparece na tela:
POKE 16414,141 : POKE 16415,5

Desvio da Impressora: (1) — desvio automático de todas as instruções que enviam sinais para a impressora.
POKE 16422,67 : POKE 16423,0
(2) — faz com que todas as instruções LPRINT sejam direcionadas para o vídeo.
POKE 16422,88 : POKE 16423,4
OBS.: Não use os dois desvios no mesmo programa. Execute POKE 16422,141 : POKE 16423,5 para voltar tudo ao normal.

Nelson Lima Neto — MG

Linha TRS-COLOR

Cópia de segurança

Veja como tirar cópias de programas em linguagem de máquina que não tenham auto-execução.

Siga as seguintes instruções:

1 – Carregue o programa em LM.

2 – Digite as seguintes instruções:

```
? PEEK(487)*256+PEEK(488)
? PEEK(126)*256+PEEK(127)-1
? PEEK(157)*256+PEEK(158)
```

Deverão aparecer três endereços -1 após cada instrução.

O primeiro endereço indica o INÍCIO do programa; o segundo indica o FIM e o último, o endereço de EXECUÇÃO do programa (EXEC).

Será útil anotar os endereços. Para copiar o programa, após digitar cSAVEM seguido do nome do programa, deverão estar os endereços separados por vírgulas (,), da seguinte forma:

cSAVEM "PROGRAMA", INÍCIO, FIM, EXEC

Observações:

1 – O nome do programa deverá ter no máximo oito caracteres;

2 – Algumas vezes, os endereços de início e fim do programa coincidem, sendo mais provável que ocorram em programas de 16 Kb.

Esta dica foi desenvolvida num CP 400 Color, com 64 Kb de RAM.

Augusto Anderson Brown Santos – BA

Linha TK 2000

Reformulando o INKEY$

Esta dica atua como a função INKEY$. Lê o teclado e associa a tecla pressionada a uma variável, sem parar o programa como acontece com INPUT ou GET.

A primeira parte deve ser digitada e executada no início do programa:

```
10 HIMEN = 38367
20  DATA 32,67,240,141,239,149,96

30  FOR I = 38368 TO 38374: READ
    A: POKE I,A: NEXT I
```

Já a segunda parte faz a leitura do teclado. Pode também ser usada como uma sub-rotina:

```
9000  CALL 38368
9010 K = PEEK (38383) - 128
9020  IF K > 1 THEN K = 32
9030 M$ = CHR$ (K)
```

Roberto Kato Pereira – ES
Cassiano Franco Bernardes – ES

Linha ZX SPECTRUM

Incrível DRAW II

Em MS nº 50, vi a dica "O Incrível DRAW" e mesmo pensando que seria tolice eu a rodei. Fiquei maravilhado com o potencial do comando. Assim, resolvi estudá-lo. Após algum tempo, descobri um efeito bem semelhante, embora mais complexo.

```
  10 OVER 1: PAPER 0: BRIGHT 1
CLS
  20 FOR x=0 TO 455 STEP 35
  30 INK RND+8
  40 PLOT 80,90
  50 DRAW 100,0,PI+x
  60 NEXT x
```

Carlos Alexandre da Silva – RJ

LINHA GERAL

Buble Sort

Na listagem abaixo, você vê um exemplo de SORT, que ordena os caracteres da STRING A$.

```
  10 LET A$="QWERTYUIOPASDFGHJKL
ZXCVBNM"
  80 GOSUB 100
  90 PRINT A$
 100 STOP
 100 LET N=LEN A$
 110 FOR F=1 TO N-1
 120 FOR K=F+1 TO N
 130 IF A$(F)<=A$(K) THEN GOTO 1
70
 140 LET L$=A$(F)
 150 LET A$(F)=A$(K)
 160 LET A$(K)=L$
 170 NEXT K
 180 NEXT F
 190 RETURN
```

Experimente substituir as duas linhas abaixo e compare a velocidade de execução.

```
 120 FOR K=N TO F+1 STEP -1
 130 IF A$(K)<=A$(F) THEN GOTO 1
70
```

Esta rotina foi listada em um SINCLAIR, mas pode ser usada em qualquer micro, com modificações mínimas.

Marco Henrique Alexandre Costa – RJ

Linha ZX SPECTRUM

Variáveis do sistema

Aqui vão algumas dicas que o ajudarão a compreender melhor seu compatível com o ZX Spectrum:

1ª) Para verificar o nº de bytes ocupados por um determinado programa: PRINT PEEK 23627+ 256*PEEK 23628– PEEK 23635- 256*PEEK 23636

2ª) Para verificar o nº de bytes livres na RAM: PRINT PEEK 23730 +256*PEEK 23731 –PEEK 23653 -256*PEEK 23654

3ª) Teste para saber se o micro tem 48 Kb: IF PEEK 23733=255 THEN PRINT "48Kb"

4ª) Para eliminar o espaço dos gráficos definidos pelo utilizador: POKE 23765,255
POKE 23676,255
CLEAR 65534

Hamilton E. L. de Souza – PR

Linha ZX81

Nhoc-Nhoc

Este simples programa em BASIC TK pode, além de causar um bom efeito de explosão em jogos, assustar um "leigo", fazendo-o pensar que a TV está estourando. O que causa tudo isto é a linha 30. O código 126 (número) faz com que os próximos 5 bytes fiquem invisíveis, encolhendo o D-FILE.

Para causar um efeito ainda melhor, basta aumentar o volume da TV ou ligar um rádio FM em torno de 90 MHz, para ouvir um ruído parecido com o de um caminhão ou avião.

Na linha 20 define-se o intervalo de linhas (22 a 7, no caso) onde quer-se causar o efeito. Convém lembrar que os POKES são feitos no início de cada linha, devendo-se evitar a impressão de algo que será reaproveitado lá.

Para anular o efeito basta dar um CLS, POKEAR um código diferente de 126 no local (um looping) ou então imprimir com PRINT, como no exemplo da listagem abaixo:

```
  10 LET x=PEEK 16396+256*PEEK 1
6397+1
  20 FOR N=x+22+33 TO x+7+33 STE
P -33
  30 POKE N,126
  40 NEXT N
  50 PRINT AT 0,0;
  60 FOR N=1 TO 22
  70 PRINT "  32 CARACTERES 128
  80 NEXT N
  90 GOTO 20
```

Fábio Antônio Rodrigues Correa – SP

Linha APPLE

Canário

Rodando este pequeno programa, um efeito sonoro semelhante ao canto de um canário pode ser produzido.

```
10  FOR X = 0 TO 17: READ A
11  POKE 12345 + X,A
12  NEXT
15  DATA  173,48,192,136,208,4
16  DATA  198,0,240,7,202,208
17  DATA  246,166,0,208,239,96
20 N =  RND (1) * 6:R = 1 +  RND
    (1) * 66
21  FOR X = 1 TO N: POKE 0,R
22  CALL 12345: NEXT : GOTO 20
30  END
```

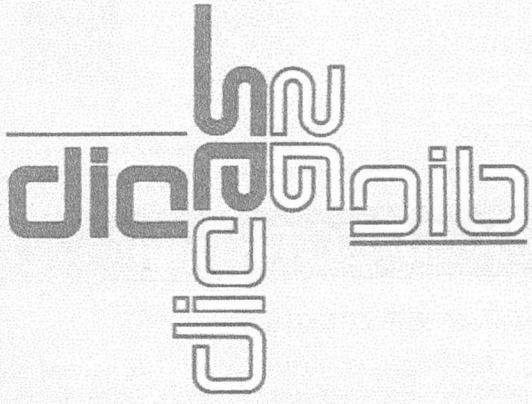
Arthur Luiz M. Bezerra - CE

Linha TRS-COLOR

Mudando o teclado

Devido ao tamanho das teclas, um problema que costuma acontecer aos usuários do CP 400 que digitam muito rápido (ao menos acontece comigo) é apertar acidentalmente o zero ao invés do parêntese; isto inverte os caracteres e obriga a apertar SHIFT e zero novamente para voltar ao normal. Para evitar tais perdas de tempo durante a digitação de programas longos, entre com POKE 282, 1; dessa forma os caracteres permanecerão fixos no modo normal. Ou então você pode fazer um INPUT diferente, onde os caracteres da string (atenção! somente de A a Z) entrem em modo inverso; é só incluir POKE 282, 0 antes do INPUT. Em ambos os casos, POKE 282, 255 retorna o micro ao modo normal.

Cláudio Costa - RJ



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha APPLE

POKEs no Apple

Com os POKEs abaixo, um & seguido de RETURN será interpretado:

```
=>como LIST ...........POKE 1013,76:POKE 1014,165:POKE 1015,214
=>como RUN ............POKE 1013,76:POKE 1014,102:POKE 1015,213
=>como CALL-151 .......POKE 1013,76:POKE1014,165:POKE1015,255
=>como HOME ...........POKE 1013,76:POKE 1014,88:POKE 1015,252
=>como PRINT CHR$(7) ..POKE 1013,76:POKE 1014,228:POKE 1015,251
```

Com estes três POKEs, o acionamento de RESET será interpretado:

```
=>como RUN ............POKE 1010,102:POKE 1011,213:POKE 1012,112
```

Bruno Lopes F. Cabral - PB

Linha TRS-80

PEEKs no TRS-80

Use os PEEKs abaixo em seus programas:

```
PEEK (293)    => Será 73 se o micro for módelo III.
PEEK (16396) => Será 201 se estiver em BASIC residente.
PEEK (16409) => Será 0 se o teclado estiver em maiúsculas.
PEEK (16412) => Será 0 se o cursor for piscante.
PEEK (16419) => Contém o código ASCII do cursor.
PEEK (14312) AND 240 => Será 48 se a impressora estiver pronta.
```

Bruno Lopes F. Cabral - PB

Linha TRS-COLOR

Toque de elegância

Esta dica, para os usuários da linha TRS-Color, consiste em dar um toque de elegância na impressão de uma mensagem. É parecida com a dica de Jorge Pablo Zapata Rivera, publicada em MS nº 52, para a linha Apple, com o nome de impressão profissional.

```
20 FOR F=0 TO 1000:NEXT F:CLS
30 LINE INPUT"MENSAGEM";A$:CLS
40 A=LEN(A$):C=2
50 FOR T=1 TO A
60 B$=MID$(A$,T,1)
70 PRINT@C-1,B$;CHR$(207)
100 SOUND200,1:C=C+1:NEXT T
110 GOTO 20
```

Tárcio Nery - BA

Linha MSX

Efeitos com Circle

Veja os efeitos produzidos com o Circle nesse curioso desenho gráfico.

```
5 REM RICARDO HUNSCHE
  (0512) 40.30.87
10 SCREEN$:F=1:COLOR13,1
20 FOR X=1TO90
30 CIRCLE(120,100),F
40 F=F+1:NEXT3
50 GOTO50
```

Ricardo Hunsche - RS

Linha ZX SPECTRUM

Caleidoscópio

Execute este programa e observe o interessante movimento de formas geométricas coloridas proporcionado por esse caleidoscópio na tela do seu micro.

```
  5 BORDER 0: OVER 1
 10 LET B=5+INT (RND*11)
 20 FOR N=B TO 85 STEP 3
 30 PLOT N,N
 40 LET A=2+N
 50 DRAW 255-A,0: DRAW 0,175-A
DRAW -255+A,0: DRAW 0,-175+A
 60 PLOT 128-N,N
 70 DRAW A,0: DRAW 0,175-A: DRA
W -A,0: DRAW 0,-175+A
 80 INK RND*7
 90 NEXT N
100 CIRCLE 127,88,5+INT (RND*16
)
105 PAUSE 60
110 GOTO 10
```

Márcio José de Carvalho - PA

Linha MSX

POKEANDO

Para quem possui um compatível com o MSX, aqui vão alguns POKEs que irão ajudá-lo no confeccionamento dos seus programas.

1. POKE 62384,N

   Faz o tabulamento horizontal semelhante a WIDTH.
   (mínimo 1 e máximo 40)

2. POKE 62385,N

   Faz o tabulamento vertical.
   (mínimo 0 e máximo 24)

3. POKE 62428,N

   Coloca o cursor na linha N.
   (mínimo 0 e máximo 24)

4. POKE 62429,N

   Coloca o cursor na coluna N.
   (mínimo 1 e máximo 40)

5. POKE 64681,N

   Se N for qualquer valor, exceto 0, o cursor irá aparecer após uma impressão.

6. POKE 64683,N

   Se N for qualquer valor, exceto 0, só poderão ser impressas letras maiúsculas.

7. POKE 64687,255

   Executa um CLS. Se estiver em uma linha de programação, o CLS será executado após SIUF.

8. Para finalizar, use a rotina abaixo para re-inicializar o sistema.

```
10 FOR A=&H4001 TO &HFFFF
20 POKE A,0
30 NEXT A
```

Jorge Pablo Zapata Rivera - BA

Linha APPLE

Quadrados

Experimente esta dica e tente acompanhar o desenho formado por uma série de quadrados. O efeito visual é bonito.

```
10  FOR N = 768 TO 777: READ I: POKE
    N,I: NEXT : POKE 232,0: POKE
    233,3
20  DATA  1,0,4,0,58,36,45,54,7,0

30  HGR2
40  HCOLOR=  RND (1) * 6 + 1
50  FOR S = 20 TO 50
60  FOR N = 0 TO 16 STEP 2
70  ROT= N: SCALE= S: DRAW 1 AT 1
    40,90
80  SCALE= 51 - S: XDRAW 1 AT 140
    ,90
90  SCALE= S - 10: XDRAW 1 AT 140
    ,90
100 NEXT : NEXT
110 GOTO 40
```

Eduardo Eiji Araki - MG

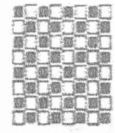
Linha ZX SPECTRUM

Novas cores no TK90X

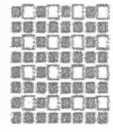
Consiga com esta simples dica, cerca de 200 padrões coloridos em seu TK90X.

```
 10 FOR F=1 TO 2: FOR G=0 TO 7:
FOR H=0 TO 7
 20 PRINT INK G; PAPER H;"BB";
 30 NEXT H: FOR H=0 TO 8
 40 PRINT INK G; PAPER H;"BB";
 50 NEXT H: NEXT G: BRIGHT 1: N
EXT F
```

Nas linhas 20 e 40, os caracteres grifados devem ser redefinidos conforme o desenho abaixo:



A do UDG



B do UDG

Gilberto Figueira da Fonseca - RJ

Linha ZX SPECTRUM

Atributos via teclado

Não consta do manual do TK90X que, se, com o cursor E digitarmos um número sem, CAPS SHIFT, podemos definir o paper (de 0 a 7), Bright 0 (8) e BRIGHT 1 (9).

Experimente digitar, por exemplo:

```
   10 REM  (C/SHIFT + S/SHIFT) 4 (C/SHIFT + S/SHIFT) 9
"EXEMPLO" (C/SHIFT + S/SHIFT) 0 (C/SHIFT + S/SHIFT) 7
"EXEMPLO II"
```

OBS.: Execute o que estiver entre parênteses e digite o restante. Veja o resultado no vídeo.

Flávio Massao Matsumoto - SP

LINHA ZX81

Tecla secreta

Se desejar proteger um programa de "olhares indiscretos" tente a rotina abaixo.

```
   1 FAST
9000 SAVE "PROG"
9001 POKE 16513,230-(INKEY$="C")
9002 GOTO 1
```

Assim, o comando FAST da linha 1 será substituído por um NEW, a não ser que a tecla "C" (pode ser qualquer outra) esteja pressionada no exato momento da entrada do programa. Isto não é uma proteção contra cópia e sim para que só você possa rodar determinados programas. Lembre-se que a tecla escolhida deve ser apertada um pouco antes do programa entrar na memória.

Edgard Santos Rocha - RJ

Linha MSX

Desabilitando o BREAK

Para desabilitar o uso do BREAK (CTRL + STOP) em um programa use este exemplo.

```
10 STOP ON
20 ON STOP GOSUB 100:GOTO20
30 PRINT" MICRO SISTEMAS "
40 END
100 CLS:PRINT"Uma tentativa de B
REAK foi feita."
110 FOR I=1 TO 2000:NEXT I
120 RETURN 30
```

A linha 10 habilita o uso do comando ON STOP GOSUB. Na linha 20 há um LOOP que terminará ao ser acionado BREAK, operando a mensagem da linha 100, e na linha 110 há uma pausa antes do retorno à linha 30. O END da linha 40 é só para a rotina não entrar em LOOP novamente, ao usar esta rotina em seu programa utilize apenas os comandos da linha 10 e 20, sendo que na linha 20 o GOTO 20 pode ser dispensado.

Sérgio Augusto Freire de Souza - AM

Linha APPLE

Minissintetizador no Apple

Esta dica traz um programa que simula um miniórgão, que pode ser rodado em computadores da linha Apple. Após rodá-lo, as teclas numéricas (1 a 0) produzem sons semelhantes aos de um sintetizador.

```
2  POKE 770,173: POKE 771,48: POKE
     772,192: POKE 773,136: POKE
     774,208: POKE 775,5: POKE 77
     6,206: POKE 777,1: POKE 778,
     3: POKE 779,240: POKE 780,9:
      POKE 781,202: POKE 782,208:

3  POKE 783,245: POKE 784,174: POKE
     785,0: POKE 786,3: POKE 787,
     76: POKE 788,2: POKE 789,3: POKE
     790,96: POKE 791,0: POKE 792
     ,0
4  GOTO 100
19  REM
20  GET A$
32  IF A$ = "1" THEN A = 10
33  IF A$ = "2" THEN A = 20
34  IF A$ = "3" THEN A = 30
35  IF A$ = "4" THEN A = 40
36  IF A$ = "5" THEN A = 50
37  IF A$ = "6" THEN A = 60
38  IF A$ = "7" THEN A = 70
39  IF A$ = "8" THEN A = 80
40  IF A$ = "9" THEN A = 90
41  IF A$ = "0" THEN A = 100
60  HOME : POKE 768,A: POKE 769,A
     : CALL 770
70  GOTO 20
100  HOME
110  PRINT "#########SINTETIZADOR
      II##############": PRINT
111  PRINT "TECLAS: 1:2:3:4:5:6:7
     :8:9:0": PRINT : PRINT  TAB(
     9);"(CTRL-RESET PARA SAIR)"
112  FOR I = 1 TO 5000: NEXT I: HOME

113  GOTO 20
120  REM  IVAN S.G.L-24/02/86
```

Ivan S. G. Lima - BA.

Linha dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha MSX

Tamanho do programa

Se você quiser saber quantos Bytes ocupa o seu programa, faça o seguinte:

a) Dê um LIST.

b) Dê o seguinte comando direto:

PRINT 28815 - FRE (1)

O resultado será a memória ocupada pelo seu programa.

Sérgio Augusto Freire de Souza - AM

Linha TRS-COLOR

Construindo com rapidez

Com esta pequena dica, pode-se construir um prédio com quantos andares quiser! Basta inserir esta rotina nos programas que utilizem alta resolução gráfica:

```
10 PMODE 4:PCLS 5:SCREEN 1,1
20 FOR N=0 TO 19 STEP .5
30 CIRCLE(129,191),9,0,N
40 NEXT
50 GOTO 50
```

Se você preferir andares mais "espremidos", modifique a linha 20 para:

```
20 FOR N=0 TO 19 STEP .25
```

Você pode, ainda, encher a tela com prédios de diferentes tamanhos, digitando:

```
15 FOR M=0 TO 240 STEP 18
17 C=RND(10)+8
20 FOR N=0 TO C STEP .5
30 CIRCLE(10+M,191),9,0,N
40 NEXT N,M
```

Winfried H. Schumann - PR

Linha MSX

Ampliando caracteres

Uma das características mais interessantes dos micros da linha MSX é a possibilidade de misturar textos e gráficos.

Sendo assim, elaborei um pequeno programa que amplia o tamanho dos caracteres.

```
10 CLS
20 OPEN "GRP:"AS#1
30 SCREEN 3
40 PRESET (80,72)
50 PRINT #1,"MSX"
60 COLOR RND(1)*15
70 GOTO 40
```

Roberto A. M. do Valle e Silvio Rafael M. dos Santos - RJ.

Linha ZX81

Monitor de programas

Existem alguns programas em Assembler que são executados na RAMTOP do micro, e para serem gravados têm que ser transferidos para a área do BASIC.

A finalidade desta Dica é justamente facilitar este trabalho. Para usá-la você deve colocar seu programa na RAMTOP e seguir os passos abaixo:

1 – Coloque no lugar da variável Y o tamanho do seu programa em código de máquina.

2 – Entre com os POKES abaixo:
POKE 16523, INT (X/256)
POKE 16522, X-PEEK 16523 *256
POKE 16526, INT (Y/256)
POKE 16525, Y-PEEK 16526*256
onde X é o endereço inicial do programa e Y a quantidade de bytes.

3 – Execute o programa com RUN e NEW LINE, mas antes acione o gravador com RECORD/PLAY.

Para fazer esta DICA, crie a linha 1 REM com 33 caracteres, e, após digitar a listagem em HEXA, entre com o programa em BASIC e salve-o em fita.

```
16514  2A 10 40 11 06 00 19 11
16530  40 E8 E0 00 C9 CD 03 40
16538  ED 53 04 40 ED B0 C9

2 CLEAR
3 DIM A$(Y)
4 RAND USR 16528
5 SAVE "PROGRAMA"
6 RAND USR 16535
7 NEW
```

Gilson Roberto Viana - PR.

Linha APPLE TK 2000

Efeito colorido

Veja o bonito efeito desta dica:

```
10  HOME
20  GR
30  FOR A = 0 TO 39
35  COLOR=  INT ( RND (1) * 7 + 1
    )
40  HLIN 0,39 AT A
41  VLIN 0,39 AT A
42  HLIN 0,39 AT 39 - A
43  VLIN 0,39 AT 39 - A
44 SOUNDA,A + 1 TO A,A + 2
50  NEXT A
60  GOTO 30
```

Obs.: Se você possuir um Apple, retire a linha 44.

Jorge Pablo Zapata Rivera - BA.

Linha TRS-COLOR

Seno e co-seno no Color

Este programa faz desenhos abstratos aleatoriamente, em alta resolução, usando comandos como SIN(SENO) e COS(CO-SENO), e ocupando apenas 400 bytes (mais ou menos).

Na linha 20, o **POKE** foi usado para aumentar a velocidade de processamento do programa (isto pode provocar uma aceleração no modo de piscar do cursor, que pode voltar ao normal pressionando-se **RESET**).

```
10 '   SIN & COS NO COLOR
20 L=RND(15):L1=RND(17)
30 M=RND(7):POKE 65495,0
40 PMODE 4,1:PCLS:TIMER=15
50 A=RND(2):SCREEN 1,1
60 FOR F=1 TO 7.3 STEP (TIMER/200)
70 Y=SIN(F+14.16)*55+126
80 X=COS(F+L)*85+94
90 Y1=COS(F+L)*(75+M)+126:X1=SIN(F+L1)*(45+M)+96
100 LINE (Y,X)-(Y1,X1),PSET
110 LINE -(Y1,X),PSET
120 IF INT(A)=2 THEN LINE -(Y,X1),PSET
130 K=INT(Y):IF K>11 THEN K=K-10:SOUND K,1
140 NEXT F
150 FOR F=1 TO 4000:NEXT F:RUN
```

Almir Petrecca Martins - SP.

Linha APPLE

POKES & CATALOG

Com os POKEs abaixo, você pode controlar a amostragem de arquivos do seu disquete no comando **CATALOG**.

a) Coloca uma parada depois de cada arquivo, prosseguindo com o acionamento de qualquer tecla:

```
]POKE 44599,234:POKE 44600,234
```

b) Com estes POKEs, um & seguido de<RETURN> será interpretado como **PRINT CHR$(4) "CATALOG"**:

```
]POKE 1013,76:POKE 1014,110:POKE 1015,165
```

Bruno Lopes F. Cabral - PB.

Linha TRS-80

RUN em BASIC Disco

Em seu DOS 500, sempre que desejar que um programa seja executado ao colocar o disco use o comando:

AUTO BASIC Nome/BAS

Você também pode usar este comando para programas em Assembler,bastando retirar o BASIC e, se desejar ver o diretório logo na entrada, coloque apenas:

AUTO DIR

Bruno Lopes F. Cabral - PB.

Linha ZX Spectrum

O estranho DRAW

Este programa traz uma boa mistura de cores, principalmente quando as duas figuras se encontram. Vale tentar acrescentar mais **DRAW** num loop e ver o que acontece. Após digitar o programa, o usuário pode ativar o flash, independente de qualquer linha, para torná-lo mais interessante.

```
  10 REM O estranho DRAW
  20 FOR J=0 TO 125
  30 PLOT J,-10: DRAW 1,32: DRAW
 OVER 1;20,J: DRAW INVERSE 1;1,-
1
  40 FOR F=1 TO 10
  50 PLOT 80,F: DRAW OVER 1;60,6
0
  60 NEXT F
  70 NEXT J
  80 GOTO 20
```

Antonio Frederico Helvy Dantas - RJ.

Linha TRS-COLOR

Proteção contra o RESET

Com esta pequena rotina você vai proteger seus programas contra um casual ou proposital **RESET**. Colocando esta linha no seu programa, toda vez que for pressionada a tecla **RESET** ele se autoexecutará sem precisar de teclar **RUN** novamente.

```
10 A=PEEK(116)*256+PEEK(117)-20:
X=INT(A/256):Y=A-(X*256):POKE 11
3,85:POKE 114,X:POKE 115,Y:FOR I
=A TO A+17:READ B:POKE I,B:NEXT
I:DATA 18,182,255,3,138,1,183,25
5,3,189,173,33,189,172,239,126,1
73,158
```

Nota: Só pode ser usado com micros que possuem o Extended **Color BASIC**. O CP 400 tem.

Marcos Ortiz - SP.

Linha TRS-80

Folhas para menus

A apresentação dos menus de seus programas ficará mais bonita com a utilização desta dica. Acrescente-a e verá as opções de acionamento de rotinas no meio da tela dentro de "folhas", como fichas de um arquivo.

```
10  'TRS 80 MODELOS I/III
20  'FOLHAS ESFERTAS PARA SEUS
 MENUS
30  'COLOQUE NO MEIO DA TELA
AS OPCOES DE ACIONAMENTO DE RO
TINAS
40  CLEAR 300:A$=CHR$(176) +
STRING$(53,179)+CHR$(131)+CHR$
(131)+CHR$(131):D$=STRING$(60,
140):C$=STRING$(29,140):D$=STR
ING$(60,176)
50  CLS:PRINT@2,A$@64,D$@223,C
$@960,D$;:FORN=0 TO 2:SET(7,N)
:SET(3,N+2):NEXT:FOR N=4 TO 9:
SET(0,N):SET(61,N+1):SET(119,N
):SET(123,N-2):SET(127,N-4):NE
XT:FOR N=10 TO 47:SET(0,N):SET
(119,N):SET(123,N-2):SET(127,N
-4):NEXT:FOR N=120 TO 122:SET(
N,45):SET(N+4,43):NEXT
60  PRINT@166,"TECLE<RETURN>";
70  Y$=INKEY$:IF Y$=CHR$(13)TH
EN 50
80  GOSUB 100
90  GOTO 70
100 FOR N=0 TO 800:NEXT:FOR N=
63 TO 117:SET(N,5):SET(N,9):RE
SET(N-1,5):RESET(N-1,9):NEXT N
:RESET(N-1,5):RESET(N-1,9):RET
URN
110 'JORGE LUIZ MANGINI SILVA-
    RJ
```

Jorge Luiz Mangini Silva - RJ.

Linha ZX Spectrum

Proteção de programas

Nesta dica, segue uma sugestão para a proteção de seus programas:

```
10 POKE 23613,0
20 FOR n=1 TO 704
30 PRINT "a";
40 NEXT n
```

Digite **RUN** e a tela será preenchida com a letra A. Rode de novo o programa e tente interrompê-lo com **BREAK**: Receberá um **RESET** resposta.

João Eriberto Mota Filho - RJ.

Linha ZX Spectrum

Contra a Pirataria

Utilize os POKEs abaixo para proteger seus programas:
POKE 23613,0 – Para fazer o BREAK = NEW
POKE 23757,0 : POKE 23758,0 – Para anular o Merge
Obs.: Não esqueça de colocar os POKEs no início dos programas!

André S. Chaves - RS

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha ZX81

Falso-Louco

Em programas com auto-start (que entram rodando), poderá ser inserida esta pequena rotina:

```
10 SAVE "PROGRAMA"
20 FAST
30 SCROLL
40 CLS
50 SLOW
60 GOTO 20
```

Para um usuário desprevenido poderá parecer que o programa foi mal carregado e não funciona. O problema pode ser contornado com o acionamento da tecla BREAK. Mais uma opção para proteção de seus programas!

Edgar Santos Rocha - RJ

Linha MSX

Recursos Gráficos

Explorando um dos melhores recursos da linha MSX... os recursos gráficos. Rode e veja.

```
10 SCREEN 2
20 C=2+14*RND(-TIME)
30 R=70*RND(-TIME)
40 CIRCLE(255*RND(-TIME),191*RND(-TIME))
,R,C
50 GOTO 20
```

Carlos José Grillo - RS

Linha TRS-80

Cálculo de Determinante

Este programa calcula o determinante de uma matriz de terceira ordem, usando a regra de Sarrus.

```
1 REM ********************************************************
2 REM ***   Calculo de determinantes pela regra de Sarrus   ***
3 REM ***   Por: Silvio Cavalcanti de Araujo, em 16/12/85   ***
4 REM ********************************************************
5 CLEAR1000:DIMA(3,3):CLS:PRINT@12,"* * *    D E T E R M I N A N
 T E S   * * *"@64,STRING$(63,140)
6 PRINT"Entre os elementos da matriz.":PRINT
7 FORX=1TO3:FORY=1TO3:PRINT"Qual o valor de:   A (";X;",";Y;") ";:INPU
TA(X,Y):NEXT:NEXT
8 D=A(1,1)*A(2,2)*A(3,3)+A(2,1)*A(3,2)*A(1,3)+A(3,1)*A(1,2)*A(2,
3)-A(3,1)*A(2,2)*A(1,3)-A(1,1)*A(3,2)*A(2,3)-A(2,1)*A(1,2)*A(3,3
)
9 PRINT:PRINT"O Determinante desta matriz e' = "D@960,"Deseja no
vo calculo ? (S/N)";
10 I$=INKEY$:IFI$="S"ORI$="s",PRINT@256,CHR$(31);:GOTO7ELSEIFI$=
"N"ORI$="n",ENDELSEGOTO10
```

Sílvio Cavalcanti de Araújo - PE

Linha ZX81

Evitando o erro 5

Sempre deparamos com a falta de espaço na impressão de grandes volumes de dados nos micros da linha ZX81. Segue abaixo uma das alternativas de contornarmos o problema:

```
10 FOR A=0 TO 66
20 IF PEEK 16442=2 THEN CLS
30 PRINT "MICRO SISTEMAS"
40 NEXT A
```

Para melhor entendimento, altere as linhas abaixo:

```
30 PRINT "MICRO SISTEMAS",
PEEK 16442
```

Aparecida Pereira - SP

Linha ZX Spectrum

Plot & Draw

Estes dois programinhas mostram algumas façanhas da poderosa dupla Plot & Draw. Deixe os programas rodarem até o final e veja os efeitos obtidos.

```
5 REM TRIANGULO FORMA TRIANGULO
6 OVER 1
9 FOR F=1 TO 125
10 FOR J=125 TO 1 STEP-1
11 FOR F=1 TO 125
15 PLOT F,F:DRAW 34,43:DRAW 56,-F
20 PLOT J,J:DRAW 34,43:DRAW 56,-J
30 NEXT F:NEXT J
40 PAUSE 10:CLS:GOTO 10
```

```
5 REM CAPSULA ESPACIAL
6 OVER
10 FOR J=125 TO 1 STEP-1
20 PLOT J,J:DRAW 34,43:DRAW 56,-J,10
30 NEXT J
40 PAUSE 10:CLS:GOTO 10
```

Antônio Frederico H. Dantas - RJ

Linha APPLE

Desenho Comandado

Esta dica permite que você desenhe no seu micro em alta resolução gráfica. Após rodado o programa, use as seguintes teclas para desenhar:

I – Para subir
M – Para descer
J – Para a esquerda
K – Para a direita
F – Para finalizar

Obs.: Se você ultrapassar as linhas demarcatórias que ficam nas bordas da tela, voltará ao ponto inicial.

```
10  ONERR  GOTO 30
20  HGR2
30 X = 139:Y = 95
40  HCOLOR= 7
50  HPLOT 0,0 TO 0,191 TO 279,191
      TO 279,0 TO 0,0
60  GET A$
70  IF A$ = "I" THEN Y = Y - 1
80  IF A$ = "M" THEN Y = Y + 1
90  IF A$ = "J" THEN X = X - 1
100  IF A$ = "K" THEN X = X + 1
110  IF A$ = "F" THEN  TEXT : END

120  HPLOT X,Y: GOTO 60
```

Ricardo Fernandes de Miranda - SP

Linha TRS-COLOR

Multicurvas II

Este programinha, baseado numa adaptação da dica "Multicurvas", publicada na edição de MS nº 31 (Abril de 84), cria um efeito lindo na tela do seu TRS Color. Os melhores STEPS são os menores de 10.

```
10 CLS
20 INPUT "INTRODUZA O STEP ";S
30 PMODE 4,1:PCLS:SCREEN 1,1
40 FOR P=0 TO 800 STEP S
50 X=.1*P*COS(P):Y=.1*P*SIN(P)
60 LINE-(X+128,Y+96),PSET
70 NEXT P
80 EXEC 44539:RUN
```

Fábio Luís de Paoli - SP

Linha ZX81

Armazenando a Tela

Use esta dica em Assembler para armazenar uma tela numa variável string. Ela pode ser colocada em qualquer lugar da memória. A sugestão é criar uma linha 1 REM com 31 caracteres e armazenar aí a rotina. Usando um monitor Assembler ou o Microbug, introduza o seguinte bloco:

```
16514   2A 10 40 11 08 00 19 ED
16522   5B 0C 40 01 C8 02 1A FE
16530   7F 28 02 77 23 13 0B AF
16538   B8 20 F3 B9 20 F0 C9
```

Em BASIC, faça:

```
1 REM "31 CARACTERES"
2 CLEAR
3 DIM A$(1,704)
```

Paulo Duarte Garcez - SP

Linha TRS-80

Some Horas

Este programa soma qualquer quantidade de horas e as transforma em horas, minutos e segundos exatos. Com um pouco de imaginação, este poderá ser adaptado a outro programa de acordo com a necessidade.

```
10 CLEAR300
20 CLS:INPUT"Quantas horas deseja calcular.: ";W
25 DIMH$(W)
30 FORA=1TOW
40 INPUT"Hora...(hhmmss).: ";H$(A)
50 S$=RIGHT$(H$(A),2):M$=MID$(H$(A),3,2):HT$=LEFT$(H$(A),2)
60 SS=SS+VAL(S$):MM=MM+VAL(M$):HG=HG+VAL(HT$):S$="":M$="":HT$=""
70 NEXTA
80 CLS:PRINT"TRANSFORMANDO HORAS MINUTOS E SEGUNDOS EM HORAS MIN
 E SEG EXATOS":PRINTSTRING$(64,95)
90 FORA=1TOW:PRINTLEFT$(H$(A),2)+":"+MID$(H$(A),3,2)+":"+RIGHT$(
H$(A),2);"   ";:NEXTA
100 PRINT@8*64,STRING$(64,95)
110 PRINT"Total das ";W;"HS SEM TRANSFORMACAO:";STR$(HG)+" HORAS
"+STR$(MM)+" MIN.S"+STR$(SS);" SEG.S":PRINTSTRING$(64,95)
120 IFSS>=60THEN A=SS/60:AA=A-INT(A):SS=SS-(INT(A)*60):M=INT(A)E
LSESS=SS:M=0
130 MM=MM+M:IFMM>=60THEN B=MM/60:BB=B-INT(B):MM=MM-(INT(B)*60):H
D=INT(B)ELSEHD=0
140 HG=HG+HD
150 PRINT@13*64,"Total das";W;"HS JA' transformadas: ";STR$(HG)+
" horas "+STR$(MM)+" min. "+STR$(SS);" segs":PRINT@14*64,"**DESE
JA CONTINUAR <S/N>"
155 PRINT@14*64,"**Deseja continuar <S/N>"
160 O$=INKEY$:IFO$=""THENPRINT@14*64,STRING$(25,128):GOTO155
170 IFO$="S"THENGOTO10ELSECLS:PRINT@64*10+30,"*FIM*":END
180 LPRINTSTRING$(79,"=")
```

José de Lima Filho - SP

Linha TRS-COLOR

Agilizando o Teclado

Digite o programa abaixo e tecle RUN. Após ter feito isso, pressionando a tecla CLEAR você terá um RESET, a tecla seta para baixo você terá um Motor On e a tecla seta para a direita um Motor Off.

```
10 A=PEEK(116)*256+PEEK(117)-600
:X=INT(A/256):Y=A-(X*256):FOR I=
A TO A+37:READ B:POKE I,B:NEXT I
:DATA 56,100,189,161,121,132,127
,129,10,39,11,129,9,39,18,129,12
,38,13,126,160,39,182,255,33,138
,8,183,255,33,134,10,57,189,167,
253,32,240
20 POKE 362,126:POKE 363,X:POKE
364,Y:CLEAR 200,A
```

Marcos Ortiz - SP

Linha ZX Spectrum

Cronômetro no TK90X

Com este programa você tem um cronômetro no centro da tela de seu monitor. Nos endereços 23674, 23673 e 23672 é contado o número de imagens enviadas para o monitor desde que o computador foi ligado.

```
5 POKE 23674,255
10 POKE 23673,255
20 POKE 23672,255
30 LET T=(65536*PEEK 23674+PEEK 23672+25
6*PEEK 23673)/60
40 PRINT AT 11,12;INT(T/60);":";INT(T-IN
T(T/60)*60);"    "
50 GOTO 30
```

Martiniano Coelho - DF

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha ZX Spectrum

CLS em bloco

Esta dica permite um CLS diferente, que divide a tela em quatro partes, apagando-as alternadamente e criando ainda um efeito bastante interessante.

Para usá-la, basta digitar uma linha REM com 76 caracteres e digitar os códigos abaixo. Depois, crie um programinha em BASIC para encher a tela (inclusive as linhas de edição) e chame a rotina com **RAND USR 16536**.

```
16514  F5 C5 CD F5 08 2A 0E 40
16522  06 10 36 00 23 10 FB C1
16530  04 F1 B8 20 EB C9 21 22
16538  40 7E 32 7B 40 36 00 01
16546  00 00 3E 0C CD 82 40 01
16554  10 0C 3E 18 CD 82 40 01
16562  10 00 3E 0C CD 82 40 01
16570  00 0C 3E 18 CD 82 40 01
16578  00 00 CD F5 08 3A 7B 40
16586  32 22 40 C9 00 00 00 00
```

Adail Muniz Rematal — MG

Linha TRS-80

Casas decimais

Com este programa, você poderá executar uma divisão com quantas casas decimais desejar. O resultado exato aparecerá na tela.

```
10 CLS:PRINT "DIVISAO":INPUT "DIVIDENTO=
";X:INPUT "DIVISOR= ";Y:Z=FIX(X/Y):PRINT
Z;",";
20 X=10*(X-Z*Y):IF X=0 THEN END ELSE Z=I
NT(X/Y):PRINT RIGHT$(STR$(Z),1);:GOTO 20
```

Pedro Américo A. M. Júnior — MG

Linha TRS-COLOR

EXEC no Color

Eis algumas formas de utilização do comando EXEC que podem ajudá-lo no manuseio do seu micro.

```
EXEC 42136...equivale ao comando CLOAD

EXEC 43319...imprime na parte superior da
             tela o nome do fabricante de
             seu micro

EXEC 44313...equivale ao comando NEW

EXEC 44321...equivale ao comando RUN

EXEC 46946...equivale ao comando LIST
```

BASIC Computer Clube — SP

Linha MSX

Trava e destrava

Use POKE 64683,X onde:
$1 \leqslant X \leqslant 254$ → trava o teclado em maiúsculas.
$X = 0$ → destrava o teclado em minúsculas.
$X = 255$ → destrava o teclado em maiúsculas.

Eduardo Toledo Santos — SP

Linha TRS-80

Economizando combustível

Esta dica, muito simples, pode ajudá-lo a controlar o consumo de combustível do seu automóvel. Rode-a e veja quanto ele está fazendo por litro.

```
10 CLS
20 PRINT "DIG.KM INICIAL: ";
30 INPUT KI
40 PRINT "DIG.KM FINAL: ";
50 INPUT KF
60 PRINT "DIG.LITROS: ";
70 INPUT LI
80 LET D=KF-KI
100 LET K=D/LI
120 PRINT "MEDIA= ";K;" KM/L"
```

Alexandre Cabral de Azevedo — RN

Linha ZX Spectrum

Em Destaque

Com esta dica, você dará a qualquer mensagem (A$) do seu programa um ótimo efeito de destaque.

```
5 REM Camilo de V.R.
10 BORDER 0:PAPER 0:INK 5:OVER 1:BRIGHT 1
:CLS
20 LET A$="MICRO SISTEMAS"
30 LET A=LEN A$
40 LET B=(INT(16-(A/2)))*8
50 PRINT OVER 1;AT 1,B/8;A$
60 FOR X=B TO B+A*8
70 PLOT X,160:DRAW 0,7
80 NEXT X:GOTO 60
```

Camilo de V. Rebouças — SP

Linha TRS-COLOR

POKEs no Color

Aqui vão alguns POKEs para o Color.

```
POKE 282,0     :"LIGA" AS MINUSCULAS

POKE 282,255   :"LIGA" AS MAIUSCULAS
```

Fábio Luis de Paoli — SP

Linha ZX81

Contador mais veloz

Os microcomputadores da linha Sinclair são lentos na impressão de números ou variáveis numéricas em BASIC, principalmente no modo SLOW. Pode-se, porém, contornar o problema pokeando o arquivo de vídeo com os números relativos aos caracteres numéricos do micro. Por exemplo:

```
10 LET Z=(PEEK 16396+256*PEEK 16397)+1
20 FOR Y=28 TO 37
30 POKE Z,Y
100 NEXT Y
```

Para contadores com mais dígitos, é só dar os POKEs nas posições seguintes da memória. Acrescente estas linhas:

```
40 FOR X=28 TO 37
50 POKE Z+1,X
60 FOR W=28 TO 37
70 POKE Z+2,W
80 NEXT W
90 NEXT X
```

Sugestões:

- Contador decrescente: altere as linhas 20, 40 e 60;
- Variando o contador de x em x elementos: troque os números (28 TO 37) das linhas 20, 40 e 60 por 156 TO 165;
- Contador em qualquer posição do vídeo: 33* (linha)+(coluna) - na linha 10;
- Contador no centro do vídeo: 10 LET Z=(PEEK 16396+256*PEEK 16397)+33* 12+15.

Luciano Nogueira Marmontel — PR

Linha ZX Spectrum

CLS diferente

A rotina abaixo cria, no seu micro, um CLS diferente que apaga a tela aos poucos.

```
 1 REM xxx ( 65 CARACTERES )xxx
 10 FOR I=23760 TO 23824
 20 READ X: POKE I,X: NEXT I
 30 DATA 33,0,33,0,64,205,229,9
8,33,0,72,205,229,92,33,0,80,205
 229,92,201
 40 DATA 229,245,6,8,229,22,0,5
4,0,35,20,122,254,32,32,247,17,0
 32,175,237,186,32,252,225,17,0,1
 50 DATA 25,16,229,241,225,17,3
2,0,25,60,254,8,32,214,175,201
```

Dê um RUN para montar a rotina na linha REM. Mas, para testar, digite um **LIST 10 :RAND USR 23760 e ENTER**.

Se desejar mudar a velocidade de execução, basta dar um **POKE 23799,X**, onde X será um número entre 0 e 255. Quanto maior for o valor, mais lenta será a limpeza do vídeo.

Álvaro F. F. Borja - PA

Linha TRS-COLOR

CSAVE(M) em High-Speed

Para carregar um programa salvo com o micro em alta velocidade, use a seguinte dica, que funcionou no meu CP 400:

POKE 65497,0:CLOAD(ou CLOADM) e ENTER

Este **POKE** triplica a velocidade do micro, porém desabilita a emissão de dados ao vídeo, portanto, a tela ficará embaralhada. Mas não se preocupe, após acabar o carregamento (quando a fita parar), digite cuidadosamente:

POKE 65496,0 e ENTER (você não poderá ver o que está digitando, mas tudo será executado).

Pronto, aí está o seu programa.

Observação: esta dica não poderá ser usada em programas em linguagem de máquina com auto-execução, pois ficaríamos impossibilitados de trazer o micro novamente à velocidade normal (POKE 65496,0).

José Eduardo Linhares - SC

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha TRS-COLOR

Memória do CP 400

Ao ligarmos o CP 400 com 64 Kb e digitaremos:

```
PRINT MEM <ENTER>
```

A resposta será 24871. Para aumentar a memória, basta digitarmos:

```
POKE 25,0:NEW <ENTER>
PRINT MEM     <ENTER>
```

A resposta, agora, será 32551.

Jorge Eduardo M. Meriguette - PE

Linha MSX

Apresentação de programas

Melhore a apresentação dos seus programas com esta dica:

```
10 SCREEN 3
20 OPEN "GRP:"AS1
30 PRESET (10,20):PRINT#1,"MAURICIO"
40 PRESET (50,70):PRINT#1,"POLATO"
50 PRESET (70,130):PRINT#1,"1986"
60 FOR I=1 TO 3000:NEXT I
70 CLOSE
```

Maurício Fonseca Polato - SP

Linha APPLE

Pirâmide

Digite esta dica e observe a pirâmide formada no vídeo. Experimente com números menores que 10; e para retornar, tecle **RETURN**.

```
10 HOME
20 INPUT "DIGITE UM NUMERO ";N
30 FOR A=1 TO 7
40 HGR 2
50 HCOLOR=A
60 FOR X=0 TO 279 STEP N
70 HPLOT 140,0 TO X,191
80 NEXT X
90 NEXT A
100 GET A$
110 TEXT
120 GOTO 10
```

Carlos Alberto Creato - SP

Linha APPLE

Desenhos na tela

Veja a capacidade gráfica do Apple, em modo gráfico de baixa resolução, através de vários desenhos geométricos coloridos que se formam no vídeo ao rodar o programa abaixo:

```
10  HOME : GR
20  FOR A = 5 TO 55 STEP 10
30  FOR I = 1 TO 30: FOR J = 1 TO
    30
40  COLOR= J * 2 / (I + 2) + A /
    10
50  PLOT I,J: PLOT J,I: PLOT 40 -
    I,40 - J: PLOT 40 - J,40 - I

60  PLOT J,40 - I: PLOT 40 - I,J:
    PLOT I,40 - J: PLOT 40 - J,
    I
70  NEXT J,I,A

]
```

Ricardo Fernandes de Miranda - SP

Linha ZX SPECTRUM

Desafie a velocidade no TK90X

Olhando o exemplar de MS nº 42, vi a dica "Desafie a velocidade", adequada para os micros da linha Apple. Como possuo um TK90X, resolvi modificar a mesma para rodar no equipamento.

Acredito que a conversão valeu, pois ficou muito interessante.

```
10 DIM X(100):DIM Y(100)
20 LET X(1)=127:LET Y(1)=87
30 FOR I=2 TO 100
40 LET X(I)=INT(RND*255):LET Y(I)=INT(RND
*175)
50 PLOT X(I),Y(I):DRAW X(I-1)-X(I),Y(I-1)
-Y(I)
60 NEXT I
70 RUN
```

Manoel E. Barbosa de Brito - PE

Linha ZX81

INKEY$ em FAST

Uma dica para utilizar o INKEY$ no modo FAST:

```
10 RAND USR 681
20 IF INKEY$="R" THEN RUN
30 GOTO 10
```

BASIC Computer Clube - SP

Linha MSX

Efeito no vídeo

Tente este efeito na tela do seu MSX:

```
10 SCREEN 2
20 FOR A=255 TO 1 STEP -5
30 FOR B=191 TO 1 STEP -5
40 LET C=INT(RND(-TIME)*14+2)
50 LINE(A,B)-(255-A,191-B),C
60 NEXT B
70 NEXT A
80 END
```

Iaperi G. D. Árbocz - PE

Linha MSX

Texto multicor

Se você quiser trabalhar em modo texto com várias cores (modo texto 2, 32 colunas), rode esta dica ou incorpore-a nos seus programas.

Além da tela ficar mais bonita e esteticamente agradável, a depuração dos programas fica muito facilitada.

```
1 SCREEN 1:IC=BASE(6):READ JC:K=0
2 FOR I=IC TO IC+31
3 K=K+1:IF K=1 THEN READ A,B
4 IF K<=JC THEN VPOKE I,(A*16+B):NEXT I:
END:ELSE K=0:READ JC:GOTO 3
5 DATA 2,7,1,2,1,3,1,7,1,1,15,1,2,12,1,
4,13,1,9,10,1,1,12,1,1,10,1,1,12,1,4,11,
8,3,7,1,1,10,1
```

João Ricardo de Freitas Oliveira - SP

Linha TRS-80

Salva tela

Com esta rotina, que apelidei de Empacota/BAS, fica mais fácil e rápido o processo de empacotamento de strings. Ela empacota em linhas de programa BASIC tudo que estiver na tela quando for executada.

```
5000 REM !-------------------------------------------------!
5010 REM ! EMPACOTA/BAS - UMA ROTINA QUE MEXE COM SUA STRING !
5020 REM !        By - Carlos Henrique Choia - Santos,SP       !
5030 REM !-------------------------------------------------!
5040 REM
5050 DIMT$(16)
5060 T$(01)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5070 T$(02)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5080 T$(03)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5090 T$(04)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5100 T$(05)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5110 T$(06)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5120 T$(07)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5130 T$(08)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5140 T$(09)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5150 T$(10)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5160 T$(11)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5170 T$(12)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5180 T$(13)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5190 T$(14)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5200 T$(15)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5210 T$(16)="   ( COLOCAR 64 CARACTERES EM BRANCO ) "
5220 DEFUSR=31500
5230 FORI=0TO26:READA:POKE31500+I,A:NEXT
5240 DATA 205,127,10,17,0,60,14,16,6,64,26,119,35,19,5,32,249,197,
1,14,0,9,193,13,32,238,201
5250 M=PEEK(VARPTR(T$(1))+1)+256*PEEK(VARPTR(T$(1))+2):X=USR(M)
5260 CLS:PRINT" PRONTO SUA TELA ESTA' SALVA ":END
```

Carlos Henrique Choia - SP

Linha TRS-COLOR

Enter Points

Aí vão algumas rotinas especiais da ROM:

```
$B99C - imprime um espaco em branco
$B9AD - imprime um ENTER
$1339 - espera alguma tecla ser
        pressionada COM o cursor
$4539 - espera alguma tecla ser
        pressionada SEM o cursor
$AC73 - volta ao BASIC
$B9AF - imprime "?"
$A928 - limpa a tela
$A4FE - executa um CLOADM
$A498 - executa um CLOAD
$A282 - CHROUT :rotina especial de
                impressao de caracteres.
                O caracter que vai sair
                na tela devera ter seu
                codigo ASC no registra-
                dor A.
```

Alexandre Carpinelli - SP

MC 1000

Inserção

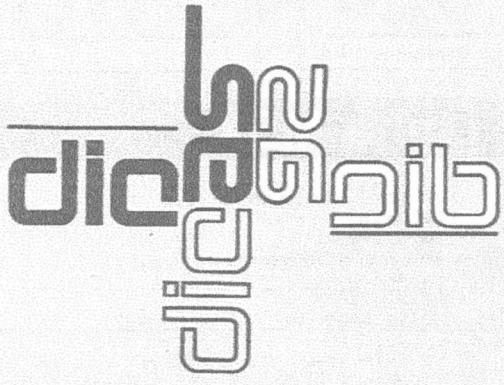
Aqui vão algumas dicas para os usuários do MC 1000:

a) No modo de edição do MC 1000, temos vários comandos, sendo um deles o de inserção de caracteres. No manual não consta se podemos, ou não, sair do modo de inserção. Aí vai minha dica: tecle < CTRL > C.

b) Para listar um programa com uma linha em branco entre as linhas (tanto no BASIC como no Debug), digite no modo direto:

```
HOME:POKE 275,64:WIDTH 32  <RETURN>
```

Carlos A. Batista de Castro — MG



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha ZX Spectrum

Sombreado

Rode este programa e veja o efeito sombreado que ele proporciona em títulos, por exemplo. Experimente aumentar as coordenadas.

```
10 INPUT "NOME ";A$:LET L=LEN A$*8:PRINT
AT 10,5;A$
20 FOR F=1 TO L:FOR N=0 TO 8
30 IF POINT (40+F,86+N)=1 THEN PLOT 40+F
,86+N:DRAW -1,1
40 NEXT N:NEXT F
```

Alexandre Hodapp O. Marques - SP

Linha MSX

Desenho gráfico

Eis aqui um curioso desenho gráfico para os usuários da linha MSX:

```
1 CLS
2 KEY OFF
4 INPUT "DE UM VALOR PARA G (0 ATE 254):
 ";G
9 COLOR 15,1,1
10 SCREEN 2
11 H=0
12 H=H+2
13 IF H=G THEN GOTO 20
14 LINE (127,0)-(H,92),15
15 LINE (127,192)-(H,92),4
16 GOTO 12
20 FOR F=0 TO 255
30 PSET (0,F),15
40 PSET (F,0),15
41 PSET (255,F),15
42 PSET (F,191),15
45 NEXT F
50 FOR R=0 TO 450:NEXT
60 GOTO 1
```

José Guilherme Sáez Júnior - SP

Linha ZX81

Desativando o Break

Use esta dica numa linha REM de 39 bytes (deve ser a primeira do programa) e rode-a com **RAND USR 16514**. Ao apertar a tecla BREAK, o programa em BASIC parará e a única saída será desligar o micro.

```
16514   CD 28 0F DD 21 8A 40 C9
16522   ED 5F 01 01 19 3E F5 CD
16530   B5 02 CD 92 02 CD 20 04
16538   21 00 40 7E FE FF 20 FE
16546   DD 21 8A 40 C3 A4 02
```

Eugênio Betanho - SP

Linha TRS-COLOR

Mini-órgão

O programa abaixo simula um mini-órgão, cujas teclas de controle situam-se na terceira fila do teclado (de A a J). Ao gravar o programa, digite **POKE 65494,0** para deixar o computador em velocidade normal.

```
10 CLS:PRINT"  MINI-ORGAO NO COLOR":POKE
65495,0
15 IF PEEK(339)=254 THEN PLAY"C":GOTO 15
20 IF PEEK(341)=251 THEN PLAY"D":GOTO 20
25 IF PEEK(342)=254 THEN PLAY"E":GOTO 25
30 IF PEEK(344)=254 THEN PLAY"F":GOTO 30
35 IF PEEK(345)=254 THEN PLAY"G":GOTO 35
40 IF PEEK(338)=254 THEN PLAY"A":GOTO 40
45 IF PEEK(340)=254 THEN PLAY"B":GOTO 45
50 GOTO 15
```

Márcio Eduardo Houenstein - PR

Linha ZX Spectrum

Túnel

Esta dica simula um túnel com **DRAW** em terceira dimensão na tela.

```
1 REM tunel
10 BORDER 0:OVER 1
20 FOR T=0 TO 255
30 PLOT T,0:DRAW 255-T*2,175
40 NEXT T
50 FOR T=175 TO 0 STEP -1
60 PLOT 0,T:DRAW 255,175-T*2
70 NEXT T
```

Maximiliano de Medeiros Chaves - CE

Linha TRS-80

Palavras reservadas

Liste as palavras reservadas do BASIC no vídeo com esta dica:

```
10 F=128:PRINT,,,:FOR E=5712 TO 6176
   20 C=PEEK(E):IF C AND 128 THEN PRINT,F;:
   F=F+1
   30 PRINT CHR$(C AND 127);:NEXT
```

João José M. Golçalves - MG

Linha MSX

Círculos

Observe os interessantes efeitos produzidos por este pequeno programa:

```
10 TIME=0:N=1
20 SCREEN 2
30 FOR I=0 TO 60
40 K=INT(RND(-TIME)*15)
50 CIRCLE (120,100),N,K
60 N=N+2
70 NEXT
80 FOR I=0 TO 1000:NEXT:GOTO 20
```

Depois, experimente mudar a linha 50 para:

```
50 CIRCLE (120,100),N,K,,,,4
```

E acrescente:

```
55 CIRCLE (120,100),N,K,,,3
```

Kenedy Pereira de Araújo - RJ

Linha ZX Spectrum

Efeitos visuais I

Veja mais um pequeno efeito de tela que o seu micro pode fazer:

```
1 REM IVAN S.MELO FILHO(13 ANOS)
5 INPUT I:OVER 1
10 FOR A=0 TO 175 STEP I
20 PLOT 0,0:DRAW 128,A
21 PLOT 255,0:DRAW -128,A
22 PLOT 255,0:DRAW -A,128
23 PLOT 0,0:DRAW A,128
30 NEXT A
```

Sugestões:

1) Melhores valores para o **STEP**: 1, 3, 4 e 5;
2) Mudar os números dos **DRAW**s para 175.

Ivan S. Melo Filho - AL

Linha APPLE

Desenhando

Veja o interessante efeito gráfico que esta dica produz:

```
10  REM  NIVARDO A.CAVALCANTE
20  REM  17/01/1986 APPLE II PLUS

30  HOME
40  HGR2
50  FOR K = 1 TO 7
60  FOR G = 1 TO 1000: NEXT G
70  HCOLOR= K
80  FOR F = 1 TO 191 STEP K
90 VA = 88 + F
100  HPLOT F,1 TO 279,F
110  HPLOT 1,F TO VA,191
120  HPLOT  TO F,K
130  NEXT F
140  NEXT K
150  GOTO 50
```

Nivardo Avelino Cavalcante - PA

Linha TRS-COLOR

Editor assembler

Uma dica para quem não tem Editor Assembler (exemplo a partir do endereço &H3F80):

```
10 CLEAR 200,&H3F00:I=&H3F80
20 PRINT "ENDERECO: ";HEX$(I);
30 INPUT "BYTE";B$
40 POKE I,VAL("&H"+B$)
50 I=I+1:GOTO 20
```

Rogério Roth - RS

Linha MSX

Terremoto no vídeo

Se na sua cidade não há terremotos, agora vai ter. Digite este programinha e comprove:

```
10 SCREEN2:COLOR 15,1,1
20 H=255
30 V=191
40 LINE (255-H,191-V)-(H,V),,B
50 H=H-6
60 V=V-6
70 IF H=69 AND V=5 THEN 90
80 GOTO 40
90 C=INT(RND(-TIME)*15)+1
100 SOUND 0,200:SOUND1,0:SOUND 6,20:SOUND
7,238:SOUND 8,16:SOUND 9,16:SOUND 11,90:S
OUND 12,1:SOUND 13,10
110 COLOR,,C
120 GOTO 90
```

Flávio José M. Júnior - SC

Linha TRS-80

OUTs no TRS-80

Mando aqui alguns **OUT**s para serem utilizados nos micros da linha TRS-80.

```
OUT 244,1 ->liga drive 0
OUT 244,2 ->liga drive 1
OUT 244,3 ->liga drives 0 e 1
OUT 244,0 ->desliga drives
```

Edison Assumpção Tacão - PR

Linha APPLE

Modos de impressão

Estas dicas lhe permitem optar por diversos modos de impressão:

1) Escrever da esquerda para o centro da tela:

```
10 HOME:SPEED=210:A$="MICRO SISTEMAS":HT
=20-LEN(A$)/2:FOR A=LEN(A$) TO 1 STEP -1
:X$=MID$(A$,A,1):FOR B=1 TO HT+A:VTAB 10
:HTAB B:PRINT X$;:HTAB B:PRINT" ";:NEXT
B:HTAB HT+A:PRINT X$:NEXT A
```

2) Escrever da direita para o centro da tela:

```
10 HOME:SPEED=210:A$="MICRO SISTEMAS":HT
=20-LEN(A$)/2:FOR A=1 TO LEN(A$):X$=MID$
(A$,A,1):FOR B=39 TO HT+A STEP -1:VTAB 1
0:HTAB B:PRINT X$;:HTAB B:PRINT" ";:NEXT
B:HTAB HT+A:PRINT X$:NEXT A
```

3) Escrever do topo para o centro da tela:

```
10 HOME:SPEED=210:A$="MICRO SISTEMAS":HT
20-LEN(A$)/2:FOR A=1 TO LEN(A$):X$=MID$(
A$,A,1):FOR B=1 TO 10:VTAB B:HTAB A+HT:P
RINT X$;:HTAB A+HT:PRINT " ";:NEXT B:HTA
B HT+A:PRINT X$:NEXT A
```

4) Escrever de baixo para o centro da tela:

```
10 HOME:SPEED=210:A$="MICRO SISTEMAS":HT
=20-LEN(A$)/2:FOR A=1 TO LEN(A$):X$=MID$
(A$,A,1):FOR B=23 TO 10 STEP -1:VTAB B:H
TAB A+HT:PRINT X$;:HTAB A+HT:PRINT " ";:
NEXT B:HTAB A+HT:PRINT X$:NEXT A
```

Eduardo Saito - SP

Linha ZX81

Casas decimais

Com esta pequena dica, os usuários do ZX81 e compatíveis poderão obter o número desejado de casas decimais após a vírgula, numa divisão de dois números reais.

```
10 CLS
20 PRINT "QUAL O DIVIDENDO?"
30 INPUT D
40 PRINT "QUAL O DIVISOR?"
50 INPUT S
60 PRINT "QUANTAS CASAS DECIMAIS?"
70 INPUT I
80 LET X=D/S
90 PRINT "QUOCIENTE= ";INT(D/S);
".";
100 FOR A=1 TO I
110 LET R=D-(INT (D/S)*S)
120 LET D=R*10
130 PRINT INT (D/S);
140 NEXT A
150 PRINT "DESEJA CONTINUAR? S/N"
160 INPUT A$
170 IF A$="S" THEN GOTO 10
180 STOP
```

Narcízio Delamar Roque — SC

Linha ZX Spectrum

Cronômetro II

Com este programa, você pode marcar o tempo em minutos, segundos e décimos de segundos. Ele pode ser usado em jogos e muitas outras aplicações.

```
10 LET D=0: LET S=0: LET M=0
15 IF D=10 THEN  LET S=S+1
20 IF D=10 THEN  LET D=0
25 IF S=60 THEN  LET M=M+1
30 IF S=60 THEN  LET S=0
35 LET D=D+1
40 PRINT AT 10,11;" CRONOMETRO
";AT 12,13;M;" : ";S;" : ";D
45 PAUSE 2.5: GOTO 15
```

Emerson Damasceno Moura — MG

Linha TRS-COLOR

Cores no Color

Rode esta dica e observe os efeitos de cores no vídeo.

```
10 CLS(0)
20 C=RND(8)
30 X1=INT(RND(63)):X2=INT(RND(63))
40 Y1=INT(RND(31)):Y2=INT(RND(31))
50 FOR X=X1 TO X2
60 FOR Y=Y1 TO Y2
70 SET(X,Y,C)
80 NEXT Y,X
90 GOTO 20
```

João Paulo Keller — RJ

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha TRS-80

Busca rápida

Use esta dica para comparar um nome digitado pelo usuário com nomes já arquivados no computador. A rotina é útil para programas que lidam com fichas e cadastros pessoais.

```
10 CLS:CLEAR1500
20 PRINT@473,"QUANTOS ARQUIVOS";:I
NPUTO
30 IFO=0THENGOTO10
40 FORA=1TOO
50 CLS:PRINT@84,"ARQUIVO NUMERO";A
60 PRINT@202,"NOME";:INPUTN$(A)
70 PRINT@266,"RUA";:INPUTR$(A)
80 PRINT@330,"NUMERO DA CASA";:INP
UTN(A)
90 NEXTA
100 CLS
110 PRINT@458,"TECLE <M> PARA VERI
FICAR NOMES"
120 K$=INKEY$:IFK$=""THENGOTO120ELS
EIFK$="M"THENGOTO130ELSEIFK$<>"M"
THENGOTO120
130 CLS:PRINT@448,"DIGITE ALBUM NO
ME ARQUIVADO NO COMPUTADOR";:INPUT
P$
140 CLS
150 FORB=1TOO
160 IFLEFT$(N$(B),LEN(P$))=P$THENP
RINT@448,"ARQUIVO";B;N$(B)
170 FORT=0TO900:NEXTT
180 NEXTB
190 GOTO100
```

Carlos Eduardo de Melo Neiva — RJ

Linha ZX Spectrum

Desenhando no TK90X

Esta dica permite fazer desenhos em seu micro. Use as seguintes teclas para desenhar:

P – Para subir
L – Para descer
Z – Para a esquerda
X – Para a direita

```
  10 INK 2
  20 PLOT 127,87
  30 IF INKEY$="P" THEN  DRAW 0,
2
  40 IF INKEY$="L" THEN  DRAW 0,
-2
  50 IF INKEY$="Z" THEN  DRAW -2
,0
  60 IF INKEY$="X" THEN  DRAW 2,
0
  70 IF INKEY$="L" THEN  INK 2
  80 IF INKEY$=" " THEN  STOP
 100 PAUSE 10
 110 GOTO 30
```

Lincoln Lara Gomes — SP

Linha MSX

Caracteres itálicos no MSX

Com esta dica, você poderá ter todos os caracteres do seu MSX alterados. Mas, atenção: ela só funciona em **SCREEN 1**.

Ao executar as rotinas abaixo, você verá a modificação ser executada e os caracteres permanecerão no novo formato até ser usado um comando **SCREEN** com argumento de tela.

```
5 SCREEN1:CLS:FORX=0TO255:VPOKEX+6144,X:
NEXTX
10 FORX=0TO2048STEP8
20 VPOKEX,VPEEK(X)/8
30 VPOKEX+1,VPEEK(X+1)/8
40 VPOKEX+2,VPEEK(X+2)/4
50 VPOKEX+3,VPEEK(X+3)/4
60 VPOKEX+4,VPEEK(X+4)/2
70 VPOKEX+5,VPEEK(X+5)/2
80 NEXTX:LOCATE6,12:PRINT"TABELA REDEFIN
IDA"
```

David Pfannemuller Guimarães — PR

Linha TRS-80

Redefinindo o READY

Use este programa para redefinir a mensagem READY. O comando **RUN 10-00** carrega a rotina em linguagem de máquina.

```
0 REM JOAO JOSE MARQUES GONCALVES
                    03/06/86
10 CLS:E=28685:A$="  "+CHR$(24)+CH
R$(24)
20 I$=INKEY$:IFI$=""THEN20ELSEPOKE
E,ASC(I$):PRINTI$;A$;:IFI$=CHR$(8)
THENE=E-1:GOTO20ELSEIFI$=CHR$(13)T
HENPOKEE+1,0:ENDELSEE=E+1:GOTO20
1000 FORE=28672TO28680:READA:POKEE
,A:NEXT:POKE16812,195:POKE16813,0:
POKE16814,112
1010 DATA225,205,248,1,205,249,32,
33,15,112
1020 DATA195,40,26,79,107,13,0
```

João M. Gonçalves — MG

Linha ZX81

Procura em cassete

Se você quiser saber os nomes dos programas gravados em *high speed* no seu cassete, acrescente esta rotina no final de cada programa:

```
9000 RAND USR 8405
9010 PRINT "NOME DO SEU PROGRAMA E
 COMO DEVE SER RODADO"
9020 PAUSE 300
9030 RAND USR 8630
```

Sempre que você gravar seus programas, dê o comando **GOTO 9000**. A seguir, carregue a fita e só pressione **BREAK** quando você desejar rodar determinado programa, dando para isso o comando indicado na linha 9010. Para gravações em 300 bps, troque os comandos das linhas 9000 e 9030 por **SAVE** e **LOAD**, respectivamente.

Jorge L. M. Silva — RJ

Linha MSX

REM permanente

Pegue seu programa predileto, crie uma linha REM com qualquer número, desde que seja a última do programa, e coloque nela o que quiser. Depois, digite o programa abaixo e dê o comando **RUN**, seguido de **DELETE-5**.

Observe que a linha REM que você criou mudou de número, e que não é possível apagá-la. Verifique isto dando um comando **LIST** e depois **DELETE 65530**.

```
1 A=PEEK(&HF6C2)+256*PEEK(&HF6C3)-1
2 FOR K=A TO &H8000 STEP -1
3 IF PEEK(K-5)=0 AND PEEK(K)=143 THEN PO
KE(K-1),255:POKE(K-2),250:GOTO 5
4 NEXT
5 END
```

Paulo Duarte Garcez — SP

Linha TRS-COLOR

Efeitos com CIRCLE

Você pode obter ótimos efeitos com CIRCLE, LINE e PSET. Tente esta dica:

```
10 PMODE 4,1:PCLS:SCREEN 1,1
20 FOR X=254 TO 1 STEP -3:CIRCLE(X,INT(X
/5)+100),X/2.5,X/200:NEXT
30 GOTO 30
```

Veja o que acontece fazendo as seguintes substituições na linha 20.

a) `20 FOR X=1 TO 200 STEP 3:CIRCLE(X,INT(X/2)+20),INT(X/4),5,X/2:NEXT X`

b) `20 FOR X=1 TO 255:PSET(X,RND(X)):NEXT X:FOR X=1 TO 254 STEP 2:LINE(X,190)-(INT(X/4)+1,X/3+7),PSET:NEXT X`

Alexandre Cerávolo Bóccoli — SP

Linha APPLE

Gráficos com co-seno

Este programa usa a função co-seno para fazer gráficos em alta resolução.

```
10 HGR
20 INPUT A,B
30 FOR I=0 TO 279:HCOLOR=3
40 HPLOT I,A+B*COS(I/A+B*COS(3))
50 NEXT I
60 GOTO 20
```

Ilo Amy S. Rivero — MT

Linha APPLE

Inverte texto

Esta dica inverte a tela de texto totalmente:

```
10 FOR A=1024 TO 2047
20 NC=PEEK(A)
30 IF NC>159 AND NC<193 THEN NC=NC+64
40 POKE A,ABS(NC-192)
50 NEXT
```

Se você preferir a tela em FLASH, digite:

```
40 POKE A,ABS(NC-128)
```

Experimente mudar a linha 10 para:

```
10 FOR A=1024 TO 2047 STEP 2
```

... e veja o que acontece.

Eduardo Saito — SP

Linha TRS-80

Incremente o INKEY$

Para não ficar sem o cursor quando você usa o INKEY$ em seu programa, execute as linhas abaixo como sub-rotinas:

```
1000 I$=INKEY$:PRINT@P%,CHR$(128);
:FORI=1TO15:NEXT
1010 PRINT@P%," ";:FORI=1TO15:NEXT
:IFI$=""THEN1000
1020 IFI$=CHR$(13)THEN1000ELSEPRIN
T@P%,I$;:RETURN
```

A variável P% representa a linha onde se encontra a mensagem, devendo ser definida antes do GOSUB. Exemplo:

```
100 REM "LINHAS ANTERIORES
110 PRINT@911,"DE SUA OPCAO";:P%=9
11+15:GOSUB1000
120 REM "LINHAS SEGUINTES
```

Moacir José Sophia — SP

Linha APPLE

Contra a pirataria

Esta dica serve como ajuda para os usuários de Apple que querem se proteger contra a pirataria.

```
 5 POKE 1010,102:POKE 1011,213:POKE 1012,1
  12
 7 HOME:VTAB10:HTAB10
 15 PRINT"QUAL E A SENHA? ";:GET A$:GET B$
  :GET C$
 20 D$="T":E$="A":F$="I"
 25 IF A$=D$ AND B$=E$ AND C$=F$ THEN GOTO
  50
 30 PRINT CHR$(7):S=S+1:IF S=3 THEN NEW
 35 GOTO 7
 50 REM SEGUE PROGRAMA
```

OBS.:

- Os POKEs iniciais tornam a função da tecla RESET equivalente à digitação do comando RUN;
- Se o leitor quiser aumentar a senha, é só colocar mais instruções GET$ na linha 15;
- É aconselhável que se grave o programa primeiro, antes de rodá-lo (por causa dos POKEs na primeira linha);
- A senha em questão é TAI podendo ser mudada como o leitor desejar, bastando alterar o valor das variáveis na linha 20.

Tairone Soares de Souza — SC

Linha MSX

Túnel do tempo

Rodando este programa, poderá ser apreciada uma das melhores características da linha MSX: a alta resolução.

```
 5 SCREEN 2
 10 '*******************
 20 '* TUNEL DO TEMPO *
 30 '*******************
 40 LET X=30:LET Y=25
 50 LET C=INT(RND(-TIME)*16)
 60 FOR X=30 TO 220 STEP 5
 70 CIRCLE (X,Y),15,C
 80 LINE (125,100)-(X,Y),C
 90 NEXT X:BEEP
 100 FOR Y=25 TO 165 STEP 5
 110 CIRCLE(X,Y),15,C
 120 LINE (125,100)-(X,Y),C
 130 NEXT Y:BEEP
 140 FOR X=225 TO 30 STEP -5
 150 CIRCLE(X,Y),15,C
 160 LINE (125,100)-(X,Y),C
 170 NEXT X:BEEP
 180 FOR Y=170 TO 25 STEP -5
 190 CIRCLE(X,Y),15,C
 200 LINE (125,100)-(X,Y),C
 210 NEXT Y:BEEP
 220 GOTO 40
```

Luiz Otávio Abdenur — RJ

Linha ZX Spectrum

PAPER rápido

Esta dica simula o comando **PAPER** com a vantagem da tela não ser apagada pelo CLS obrigatório.

```
  10 FOR c=0 TO 21: PRINT AT c,I
NT (RND*32);" "
  20 NEXT c
  30 INPUT "paper? ";p
  40 FOR f=0 TO 21: PRINT AT f,0
; OVER 1; PAPER p;"                    ": NEXT f: GO
TO 30
```

Teotônio Rodrigo de Carvalho — MG

Linha MSX

Minimontador hexa

Este programa é um minimontador hexadecimal. Para usá-lo, digite **RUN**, responda ao **INPUT** com o endereço inicial e digite os códigos hexadecimais.

Em caso de erro, é só recomeçar pelo endereço errado.

```
10 CLS:INPUT"ENDERECO ";E:IF E<0 THEN E=
E+65536!
20 PRINT RIGHT$("000"+HEX$(E),4);:PRINTU
SING" ##### ";E;:PRINT RIGHT$("0"+HEX$(P
EEK(E)),2);" - ";:POKE&HFCAB,255:A$=INPU
T$(2):PRINTA$:C=VAL("&H"+A$):IF A$<>RIGH
T$("0" +HEX$(C),2)THEN END ELSE POKEE,C:
E=E+1:GOTO 20
```

João J. M. Gonçalves — MG

Linha TRS-COLOR

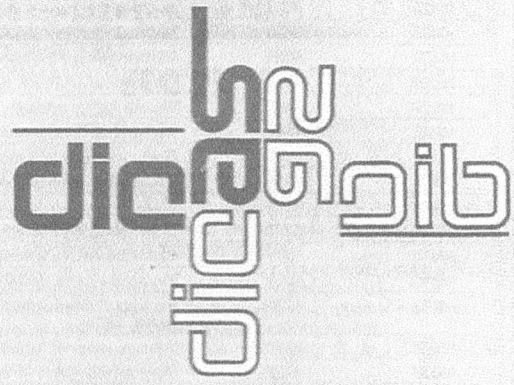
Abertura

Esta dica poderá ser usada para a apresentação de instruções de programas, jogos etc. Para aumentar as mensagens, aumente o número de condições de K entre as linhas 5020 e 5051, sendo esta a última.

Obs.: é mais uma adaptação da dica de **Tárcio Nery**, publicada em **MS** nº 56.

```
9 CLS0:GOSUB 5000
10 REM INICIO DO PROGRAMA PRINCI
PAL
5000 S=225
5005 A$="======*MICRO SISTEMAS*=
====="
5006 A=LEN(A$):FOR T=1 TO A:B$=M
ID$(A$,T,1)
5011 PRINT@ S-1,B$;CHR$(62);
5012 SOUND200,1:S=S+1:NEXT T:FOR
H=1 TO 500:NEXT H
5013 FOR G=224 TO 255:PRINT@ G,"
 ";:PLAY"T245EG":NEXT G
5014 GOSUB 5016
5015 GOTO 5006
5016 K=K+1:S=225
5017 IF K=1 THEN A$=" AV.PRES.WI
LSON-165 GR 1210"
5019 IF K=2 THEN A$="===TELEFONE
-(021)262-6306==="
5020 IF K=3 THEN A$="**RIO DE JA
NEIRO-RJ-BRASIL**"
5051 IF K=<3 THEN 5006 ELSE GOTO
10
```

Ademir Mota de Oliveira — MG



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha APPLE

Alta resolução

Rode o programa abaixo e veja uma pequena amostra do modo gráfico de alta resolução do Apple:

```
5 C = 80
10  HGR2
20  HCOLOR= 2
30  HPLOT 0,0 TO 278,0 TO 278,191
      TO 0,191 TO 0,0
40  FOR I = 1 TO 278
50 Y =  SIN ((6283184 / 278) * I)
      * C + 95
60  HPLOT I,Y
70  NEXT I
80  END
```

Obs.: varie o valor de C na linha 5 (de 2 a 94) e veja o resultado.

Marcelo Lima Reis — RJ

LINHA ZX81

Inversão localizada

Rode esta dica e responda o número de caracteres que você deseja inverter. Após isto, digite **RAND USR 16514** e veja o resultado. Crie uma linha REM com 25 caracteres e entre com o bloco em Assembler.

```
16514 - 2A 0C 40 11 00 00 06 00
16522 - 23 19 7E FE 76 28 07 C6
16530 - 80 77 23 10 F5 C9 23 18
16538 - F1
```

Agora digite a parte em BASIC:

```
10 PRINT "QUANTOS CARACTERES ?"
20 INPUT QC
30 IF QC<0 OR QC>255 THEN GOTO 10
40 POKE 16521,QC
50 CLS
60 PRINT "LINHA => "
61 INPUT A
65 PRINT "COLUNA => "
70 INPUT B
80 CLS
85 LET C=A*33+B
86 LET D=INT (C/256)
90 LET E=C-(D*256)
100 POKE 16518,E
110 POKE 16519,D
```

Obs.: não passe das coordenadas 21,31 para não correr o risco de perder o seu programa.

Max Junginger — RS

Linha MSX

Relógio

Coloque um relógio na tela do seu MSX:

```
10 OPEN"GRP:"FOROUTPUTAS#1
20 CLS:INPUT H,M,S:SCREEN3
30 ONINTERVAL=60GOSUB1000:INTERVALON
40 GOTO 40
1000 BEEP:CLS:S=S+1:IF S=60 THEN S=0:M=M
+1:IF M=60THENM=0:H=H+1:IFH=13THENH=1
1010 PRESET(0,90):PRINT#1,USING"##:";H;M
;:PRINT#1,USING"##";S:RETURN
```

João J. M. Gonçalves — MG

Linha ZX Spectrum

PLOTs

Use esta dica e veja um tipo de utilização do comando **PLOT**:

```
  1 REM MARCOS DIAS DE SOUZA
  8 CLS
  9 INK 0
 10 PRINT AT 0,0;"qual a primei
ra cor desejada?"
 11 INPUT v
 12 FLASH 1: PRINT AT 0,0;"qual
 a segunda cor desejada?": FLASH
 0
 13 INPUT z
 14 CLS
 15 FLASH 1: PRINT AT 0,0;"digi
te o numero de PLOTs deseja-do":
 FLASH 0
 20 INPUT a
 22 INPUT b: CLS
 24 FOR c=0 TO a
 25 FOR d=0 TO b
 30 INK z: PLOT d,c
 31 INK v: PLOT c,d
 32 NEXT d: NEXT c
 33 PAUSE 100
 35 INK 0
 40 GOTO 5
```

Marcos Dias de Souza — RJ

Linha TRS-COLOR

Buraco negro

Rode este programa e tenha um visual típico de filmes de ficção científica, como se estivéssemos na saída de um "Buraco negro", observando um planeta em eclipse ao longe.

```
10 'FICCAO CIENTIFICA
    ARTHUR F. ANKERKRONE
20 PMODE 4,1:PCLS:SCREEN 1,1
30 FOR A=0 TO 255 STEP 15:LINE(A
,191)-(0,A),PSET:LINE(A,0)-(255,
A),PSET:NEXT A
40 FOR X=1 TO 100:PSET(RND(255),
RND(191)):NEXT X
50 CIRCLE(70,96),15,3:PAINT(70,9
6),3,3
60 FOR Y=0 TO 14:CIRCLE(70,96),Y
,2:NEXT Y
70 GOTO 70
```

Arthur F. Ankerkrone — SP

Linha APPLE

Manipulando o vídeo

As dicas abaixo o ajudarão muito na manipulação do seu vídeo:

a) Salvar em disco a tela de texto ou a tela de baixa resolução gráfica:

```
BSAVE TELA,A$400,L$3FF      ou      BSAVE TELA,A1024,L1023
```

b) Salvar em disco a tela de alta resolução gráfica número 1:

```
BSAVE TELA,A$2000,L$1FFF    ou      BSAVE TELA,A8192,L8191
```

c) Salvar em disco a tela de alta resolução gráfica número 2:

```
BSAVE TELA,A$4000,L$1FFF    ou      BSAVE TELA,A16384,L8191
```

Eduardo Saito — SP

Linha TRS-80

Aumente a memória

Esta dica, na verdade um **POKE**, aumenta em pouco mais de 1 Kb a memória disponível para programas em BASIC, mas só funciona com pelo menos um drive.

Basta entrar no BASIC disco e, antes de mais nada, digitar:

```
POKE 16548,1:POKE 16549,102:NEW
```

e dar **ENTER**.

Feito isto, dê um **PRINT MEM** e veja o resultado.

Edison Assumpção Tacão — PR

Linha ZX Spectrum

Efeitos na borda

Provoque um efeito interessante no vídeo do seu micro, fazendo com que a borda fique dividida em várias cores que se movimentam.

```
  10 PAUSE 1: BORDER 0: BORDER 1
: BORDER 2: BORDER 3: BORDER 4:
BORDER 5: BORDER 6: BORDER 0
  20 OUT 0,0: OUT 0,1: OUT 0,2:
OUT 0,6: OUT 0,4: GOTO 20
```

Se você retirar a linha 20 e acrescentar um **GOTO 10**, no final da linha 10, as cores ficarão paradas.

Cleidinei e Cledimar Silva — PA

Linha ZX81

VERIFY no ZX81

Se você possui um TK85, talvez não saiba, mas este micro possui um comando **VERIFY**, a mesma rotina que verifica a gravação de dados e permite checar uma gravação normal de 300 bauds.

Para utilizá-lo, digite o comando direto **PRINT USR 9816**, após ter salvo um programa e rebobine a fita como se fosse ler novamente. Após a verificação, surgirá no vídeo um código de reportagem cujo significado está no capítulo 29 do manual.

Esta dica também é válida para os possuidores do CP-200S e os códigos de reportagem, neste micro, estão no apêndice **E** do manual deste micro.

Paulo Marques Figueira — SP

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

LINHA ZX81

Contagem regressiva

Use esta dica para a abertura de jogos. Crie uma linha REM com 48 caracteres e digite a listagem em Assembler:

```
16514 - 3E 25 2A 0C 40 0E 16 23
16522 - 06 20 77 23 10 FC 0D 20
16530 - F6 3D 06 B0 0E B0 0D 20
16538 - FD 10 F9 FE 1B 20 E3 2A
16546 - 0C 40 0E 16 23 06 20 36
16554 - 00 23 10 FB 0D 20 F5 C9
```

Rode o programa com o comando **RAND USR 16514.**

José Gossain Filho – SP

Linha ZX SPECTRUM

Manipulando o INPUT

Esta rotina permite a introdução de dados em qualquer lugar da tela, especificando a linha (variável L) e a mensagem a ser impressa (variável A$).

```
  10 LET a$="nome ?": LET l=10:
LET po=LEN a$: LET c=1: LET k$="
": LET b$=CHR$ 143
  15 PRINT AT l,0;a$;" ";b$
  20 LET a$=INKEY$: IF a$="" THE
N GOTO 20
  30 IF a$=CHR$ 13 THEN SOUND .0
05,-10: PRINT k$: STOP
  40 IF a$=CHR$ 12 AND c<>1 THEN
 GOSUB 90
  50 PRINT AT l,po+c+1;b$
  60 PRINT AT l,po+c;a$
  70 PAUSE 0: SOUND .005,-10: LE
T k$=k$+a$: LET c=c+1
  80 GOTO 20
  90 LET c=c+1: LET k$=k$( TO c-
1)
 100 PRINT AT l,po+c;b$;" "
 110 SOUND .005,-10
 120 GOTO 20
```

Eduardo Neto Manzi – GO

Linha ZX SPECTRUM

Efeitos com DRAW e PLOT

Digite esta dica e deixe-a rodar por três minutos:

```
   5 BRIGHT 1: CLS : RAND : FOR
j=0 TO 80 STEP 2
   6 PLOT 0,j
   7 DRAW 255,j
   8 NEXT j
   9 FOR j=-175 TO 80 STEP 2
  10 PLOT 0,j
  11 DRAW 255,j
  12 NEXT j
  13 FOR j=175 TO 80 STEP -1
  20 LET s=INT (RND*3)+2
  25 LET t=INT (RND*2)
  30 FOR j=255 TO 0 STEP -5
  35 PLOT j,0: DRAW OVER t;255-2
*j,175
  40 NEXT j
  45 FOR y=0 TO 175 STEP s
  50 PLOT 0,y: DRAW OVER t;255,1
75-2*y
  55 NEXT y
  60 GOTO 9
```

se quiser, acrescente:

```
56 PLOT OVER 1;80,50:DRAW OVER 1
;100,100,PI*761
```

Antonio Frederico H. Dantas – RJ

Linha TRS-80

PATCH no DOS - 500

Os seguintes PATCHs fazem com que o comando **PURGE** não pergunte a senha do disquete. Assim, os "esquecidos" não terão mais problemas na hora de retirar do diretório arquivos indesejáveis.

```
PATCH *11 (ADD=5648,FIND=66,CHG=99)
PATCH *11 (ADD=564D,FIND=21A258,CHG=C36E56)
PATCH *11 (ADD=5693,FIND=28,CHG=18)
```

Não se esqueça de dar **RESET** para que os PATCHs possam surtir efeito. Essa dica funciona tanto para o sistema operacional DOS – 500 como para o TRS–DOS (original).

José H. Fatia da Silva – RJ

Linha ZX SPECTRUM

INKEY$ mais rápido

Em vários programas, observamos a seguinte condição:

```
100 PRINT "outra vez ?(S/N)"
200 IF INKEY$="S" OR INKEY$="s"
THEN RUN
300 STOP
```

Como vimos, é preciso um duplo INKEY$ para registrar uma única opção. Porém se fizermos:

```
100 PRINT "outra vez ?(S/N)"
200 IF CHR$ PEEK 23556="S" THEN
RUN
300 STOP
```

O INKEY$ aceitará tanto S ou s como resposta.

Adriano Pascoal Pereira – SP

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha ZX Spectrum

Códigos UDG

Após redefinir os caracteres gráficos (usando UDG 2), utilize esta dica para se obter os valores decimais correspondentes a esses caracteres. Com isso, você poderá colocá-los nas linhas DATA de seus programas.

```
9988 LET end=65368: LET car=65:
LET udg=144: LET y=0
9989 LET cont=1: LET x=0
9990 LET b=PEEK end
9991 PRINT AT x,y; INVERSE 1;" "
;CHR$ car;" "; INVERSE 0;AT cont
,y;b;AT 10,y+1;CHR$ udg
9992 LET cont=cont+1: IF cont>8
THEN GOTO 9994
9993 LET end=end+1: GOTO 9990
9994 LET y=y+4: IF udg=151 THEN
PAUSE 0: LET car=72: LET udg=151
: LET y=0: CLS
9995 IF udg=159 AND car=80 THEN
PAUSE 0: LET car=80: LET udg=159
: LET y=0: LET end=65503: CLS
9996 IF udg=163 THEN LET end=655
27
9997 IF end=65535 THEN STOP
9998 LET end=end+1: LET car=car+
1: LET udg=udg+1
9999 GOTO 9989
```

Após dar **RUN**, o programa mostrará as letras de A até H. Depois é só apertar a tecla ENTER para que apareçam as outras letras.

Paulo Roberto da Costa – SP

Linha APPLE

SCROLL gráfico

Este programa faz um SCROLL gráfico. Rode e veja o efeito.

```
10  HOME : POKE  - 16304,0: POKE
     - 16298,0
20  VTAB 24
30  FOR A = 1 TO 40: PRINT  CHR$
    (( RND (1) * 94) + 33);: NEXT

40  GOTO 30
```

Cristiano Viana – DF

Linha TRS-COLOR

POKEs na ROM

Após ter digitado as listagens 1 e 2 do artigo "Por dentro do Color (II)" publicado na edição de MS nº 60, veja alguns recursos interessantes:

```
POKE &HABEE,ASC(" "):POKE &HABEF,0
```

Para versão fita. Faz com que a mensagem OK seja substituída por um " ", como num DOS. Para eliminá-lo, no caso de, por exemplo, digitar um programa, pressione ENTER, BREAK ou CLEAR; e para voltar ao sinal de " ", pressione qualquer coisa, como um "?" seguido de BREAK.

Para voltar ao OK, digite:

```
POKE &HABEE,ASC("0"):POKE &HABEF,ASC("K")
```

O comando a seguir faz a tela ficar com caracteres amarelos sobre fundo vermelho escuro, tornando os caracteres mais visíveis, sem prejudicar a vista:

```
POKE &H167,57:SCREEN 0,1
```

Para voltar ao normal, digite:

```
SCREEN 0,0
```

Não utilize este POKE se você usar a tela de gráficos, pois o BREAK não fará voltar ao normal. Caso ocorra isso, tecle SCREEN 0,0. Você não poderá ver o que está digitando, mas terá efeito.

```
POKE &HA1A6,0:POKE &HA1A8,ASC("c")
```

Como já foi dito no referido artigo de Cláudio Costa, estes comandos trocam o cursor. Porém com a opção ASC("c") você pode colocar o caráter que quiser. Teste alguns como: –, =, * e :.

Alex Leandro Morales – SP

Linha MSX

Alienígenas

Esta dica mostra uma maneira fácil de se utilizar sprites.

As linhas 130 a 160 são praticamente iguais, por isso, na linha 130 coloque o cursor em cima de 130; digite 140 e dê um<RETURN>. Faça a mesma coisa nas linhas 150 e 160, dando a seguir um LIST e veja o resultado. Arrume as linhas e divirta-se!

```
10 REM ALIENIGENAS
20 SCREEN 2,3
30 A$=CHR$(&B01000010)
40 B$=CHR$(&B00100100)
50 C$=CHR$(&B00011000)
60 D$=CHR$(&B01111110)
70 E$=CHR$(&B10000001)
80 F$=CHR$(&B11111111)
90 G$=CHR$(&B01000010)
100 H$=CHR$(&B10000001)
110 SPRITE$(0)=A$+B$+C$+D$+E$+F$+G$+H$
120 FOR F=127 TO 1000
130 PUT SPRITE 0,(F-120,F-5),7,0
140 PUT SPRITE 1,(F+20,F-20),4,0
150 PUT SPRITE 2,(F-50,F+120),8,0
160 PUT SPRITE 3,(F-20,F+20),12,0
170 NEXT
180 GOTO 120
```

Sérgio D. Nievola – PR

Linha ZX SPECTRUM

Teia de aranha

Este pequeno programa preenche a tela do micro com uma teia de aranha. Rode e veja.

```
1 REM teia de aranha
10 BORDER 0:OVER 1
20 FOR T=0 TO 255 STEP 16
30 PLOT T,0:DRAW 255-T*2,175
40 NEXT T
50 FOR T=175 TO 0 STEP -16
60 PLOT 0,T:DRAW 255,175-T*2
70 NEXT T
80 FOR T=5 TO 85 STEP 5
90 CIRCLE 127,87,T
100 NEXT T
```

Maximiliano de Medeiros Chaves – CE

Linha ZX81

Trocador

Se você tiver um programa que possua muitas instruções PRINT e deseja usar a impressora, esta pequena rotina trocará todos estes PRINT por LPRINT. É importante notar que uma linha REM F separa o programa da rotina.

Na linha 9030 está definido o processo de substituição, podendo ser qualquer outra instrução. Estas instruções entre parêntesis devem ser digitadas após o comando **THEN**; feito isto, elimine-o. Podem ser digitados, também, diretamente na linha, os códigos das instruções.

```
8999 REM F
9000 REM TROCADOR
9010 LET X=16513
9020 IF PEEK X=234 THEN GOTO 9060
9030 IF PEEK X=CODE " PRINT " THEN
 POKE X,CODE " LPRINT "
9040 LET X=X+4+(PEEK (X-2)+256*PEE
K (X-1))
9050 GOTO 9020
9060 IF PEEK (X+1)=43 THEN STOP
9070 GOTO 9030
```

Só serão substituídas as instruções que estiverem no início de cada linha.

Alex Argona – SP

Linha APPLE

Por do sol

Rode esta pequena dica e tenha um por do sol na tela de seu equipamento.

```
10 HGR
20 HCOLOR=5
30 FOR I=1 TO 80
40 HPLOT 139,79 TO RND(1)*279,RND(1)*79
50 NEXT I
60 HCOLOR=6
70 FOR I=79 TO 159 STEP 10
80 HPLOT 139,79 TO 0,I
90 HPLOT 139,79 TO 279,I
100 NEXT I
110 FOR I=0 TO 279 STEP 10
120 HPLOT 139,79 TO I,159
130 NEXT I
```

Bruno Lopes F. Cabral – PB

Linha TRS-80

Um AUTO diferente

Utilize esta dica para se criar um disco auto-executável (*self-boot*), isto é, um disco que execute qualquer programa /CMD imediatamente após o boot. Para isso, é necessário o sistema operacional NEWDOS 80 e seu utilitário Superzap. Proceda, então, da seguinte forma:

1) Formate um disco de dados e copie para ele o programa /CMD desejado. O programa deverá ser o único desse disco, além dos programas do sistema BOOT/SYS e DIR/SYS;

2) Verifique, com o comando **DIR A**, o número de extensões desse programa. Se ele tiver mais do que uma extensão, deverá ocupar mais que 32 grânulos. Acima deste número, as chances de conseguir sucesso são poucas. Mas não se preocupe, a maioria dos programas não chega a esse número;

3) Usando a opção DFS do Superzap anote o DRS do setor relativo 0 do programa /CMD, no disco de dados. Depois, use a opção DD e responda com o número do drive e o número do DRS anotado acima. Anote agora o número do TRS (em hexa);

4) Usando a opção DFS, agora para o programa BOOT/SYS, modifique o setor relativo 1, da seguinte forma:

- Byte 05: TRS do programa /CMD anotado no item 3,
- Byte 3E: mude de C8 para C9;

5) Coloque o disco no drive 0 e aperte RESET. O programa entrará automaticamente, diferente do que se consegue com o comando **AUTO**. Note que este disco só poderá ser usado para esse programa.

José Henrique Fatia da Silva – RJ

Linha ZX SPECTRUM

Escrevendo a laser

Esta dica permite colocar na tela do seu micro uma mensagem com um interessante efeito visual, que muito se parece com um canhão laser.

```
1 REM Soft-Rody TK90X
      Micro Sistemas
2 CLS
5 PRINT AT 14,0;"(digite 31…)"
10 INPUT LINE A$:IF LEN A$>16 T
HEN GOTO 10
15 INPUT"INVERSO=0/NORMAL=1";IN
18 IF IN>1 OR IN<0 THEN GOTO 15
20 PRINT AT 21,0:PAPER 1;INK 1;
A$
30 FOR F=0 TO LEN A$*8-1
40 FOR G=0 TO 7
50 IF POINT(F,G)=IN THEN PLOT 1
27,28:DRAW OVER 1;(-127,+F)+F,(
48+G)+G:DRAW 0,2:PLOT 127,8:DRA
W OVER 1;(-127+F)+F,(48+G)+G
60 NEXT G:NEXT F
70 INPUT C$:GOTO 1
```

Obs.: na linha 5 você deve digitar, entre as aspas, 31 vezes o caráter gráfico obtido com SIMBOL SHIFT + 0.

Teotônio Rodrigo de Carvalho – MG

LINHA ZX81

Somatório

Use esta dica para corrigir programas em Assembler, quando se tiver o somatório em decimal depois do oitavo byte, e veja que será bastante útil e rápida a procura do erro.

```
9000 CLS
9010 PRINT "ENDERECO INICIAL: ";
9020 INPUT I
9030 PRINT I
9040 PRINT "ENDERECO FINAL: ";
9050 INPUT E
9060 PRINT E;AT 7,4;" SOMATORIA D
E 8 EM 8 BYTES"
9070 LET C=0
9080 LET S=0
9090 FOR F=I TO E
9100 LET C=C+1
9110 IF INKEY$="A" THEN STOP
9120 IF C=9 THEN GOSUB 9170
9130 LET S=S+PEEK F
9140 PRINT AT 10,11;F;" ";S
9150 NEXT F
9160 STOP
9170 LET S=0
9180 IF INKEY$="" THEN GOTO 9180
9190 LET C=1
9200 PRINT AT 10,16;"
9210 RETURN
```

João Batista Negro – PR

Linha MSX

Bateria

Se você gosta de ritmo e deseja conhecer mais um pouco sobre os recursos sonoros do MSX, rode esta dica que simula o som de uma bateria.

```
10 SOUND 7,247:SOUND 8,16:SOUND 13,6
20 SOUND 6,11:SOUND 11,0:SOUND 12,7
30 FOR F=1 TO 1000:NEXT
40 SOUND 12,3:SOUND 6,5
50 FOR F=1 TO 200:NEXT
60 GOTO 20
```

Roberto dos Reis Alvarez – RS

Linha MSX

A misteriosa <SELECT>

Para se ativar em BASIC a misteriosa tecla <SELECT>, do Expert, use esta dica como sub-rotina e terá uma tecla útil:

```
10 GOSUB 20
20 A=INP(170):B=(240 AND A) OR 7:OUT 170
,B:C=INP(169):IF C=191 THEN PLAY"CDEFGAB
":RETURN
30 GOTO 10
```

Obs.: se trocar o 191 por 247, você ativará a tecla <TAB>.

Marco Aurélio M. Barros – MG

Linha ZX Spectrum

POKEs úteis

Aqui vão alguns POKEs úteis para auxiliá-lo com o seu TK90X.

```
POKE 23658,8          -> Caps Lock direto

POKE 23624, de 0 a 7  -> cor da Border


MEXENDO NO CONTROLE DO AT:

   REM xxx (QUALQUER COISA!!!) -PRIMEIRA LINHA DA SUA LISTAGEM-

 Digite em comando direto:

   POKE 23760,22 - Controle AT
   POKE 23761,linha
   POKE 23672,coluna
```

Ewaldo Ramos Araújo — RJ

Linha TRS-COLOR

Efeito tridimensional

Esta dica mostra como se pode obter efeitos tridimensionais, usando as funções seno e co-seno.

```
5 ' EFEITO 3 DIMENSOES
10 PCLEAR 8
20 PMODE 4,1 : PCLS
30 PMODE 4,5 : PCLS
40 PI=4*ATN(1)
50 V=5
60 FOR Z=0 TO 360 STEP 3
70 Y=90-60*SIN(Z*PI/180)
80 X=65+60*COS(Z*PI/180)
90 PMODE 4,1:SCREEN 1,1
100 CIRCLE (128,Y),X,V,.5
110 PMODE 4,5:SCREEN 1,1
120 CIRCLE (128,Y),X,V,.55
130 NEXT Z
140 IF V=0 THEN 50
150 V=0:GOTO 60
```

Pode-se alterar os valores das linhas 70 e 80, para obter diferentes formas. Experimente!

Edson Dalia Vecchia — SP

Linha TRS-80

Retirando o delta

A maioria dos usuários do processador de texto Superscripsit (e seu similar nacional Protexto) não gosta do símbolo delta, que indica dois espaços em branco. Usando em conjunto a tecla SHIFT e a barra de espaço esse problema desaparece.

Esses dois Patchs eliminam a necessidade de se usar a tecla SHIFT, fazendo com que o símbolo delta não mais apareça na tela.

```
PATCH SCR64/CTL (ADD=7F8A,FIND=20,CHG=18)

PATCH SCR35/CTL (ADD=99C9,FIND=20,CHG=18)
```

José Henrique Fatia da Silva — RJ

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

LINHA ZX81

INPUT localizado

Coloque o cursor em qualquer posição da tela entre as 24 linhas disponíveis, nos equipamentos da linha ZX81.

```
10 POKE 16418,N
20 POKE 16389,66
30 INPUT A$
40 PRINT AT 11,0;A$
50 SCROLL
60 GOTO 10
```

O "N" da linha 40 significa o número da linha onde será executado o INPUT, o qual deverá estar entre dois e 25. O POKE da linha 20 serve para acelerar o processamento.

Este outro programa executa um Scroll valendo a mesma observação acima:

```
10 POKE16418,N
20 SCROLL
30 PRINT"SUBINDO"
40 GOTO10
50 SCROLL
60 GOTO 10
```

Fernando César Morellato — RS

Linha MSX

Superefeito

Um interessante efeito é obtido com a seguinte rotina:

```
10 REM * SUPER EFEITO *
20 REM
30 SCREEN 0
40 COLOR 4,15,14
50 KEYOFF
60 FOR A=0 TO 896
70 PRINT CHR$(PEEK(49400!));
80 NEXT A
90 PRINT
```

Após digitá-la, pressione a tecla F5, espere um pouco, e quando aparecer um OK, passe o cursor em cima do nome escrito e veja o que acontece.

Isso ocorre porque, cada quadrado azul é nada menos que o próprio cursor, que está guardado no endereço 49400 do micro.

Obs.: esta rotina foi elaborada num Expert.

Roberto Ramos Stancial — SP

Linha APPLE

Do BASIC para o Assembler

Para executar um comando do monitor diretamente de um programa BASIC, basta usar a pequena linha, a seguir, como uma sub-rotina de seus programas:

```
10 CO$ = "COMANDO DO MONITOR":CO$
     = CO$ + " D823G": FOR A = 1
   TO  LEN (CO$): POKE 511 + A
   , ASC ( MID$ (CO$,A,1)) + 12
   8: NEXT : POKE 72,0: CALL -
   144
```

Na variável CO$ deverá ser dado o comando monitor. Para usar vários comandos, basta deixar um espaço entre eles, o qual funcionará como um RETURN.

Eduardo Saito — SP

Linha ZX81

Criatividade II

Este programa roda uma mensagem no vídeo em letras gigantes. A mensagem é escrita através de um INPUT em BASIC e pode ter no máximo 2000 caracteres (dependendo da quantidade de memória do seu micro).

Digite a parte em Assembler em uma linha REM com 100 caracteres:

```
16514  01 00 03 3E 00 B7 CB 27
16522  10 FC 4F 30 02 06 01 21
16530  00 1E 09 E5 D1 01 20 01
16538  2A 0C 40 23 09 0E 08 06
16546  0B 1A CB 7F 28 04 36 B0
16554  18 02 36 00 23 CB 07 10
16561  F1 C5 01 19 00 09 C1 0D
16569  28 03 13 18 E2 3E 08 F5
16577  2A 0C 40 01 0B 01 09 3E
16585  08 01 1F 00 23 E5 D1 23
16593  C5 ED B0 2B 70 C1 3D 28
16601  03 23 18 F0 F1 3D C8 18
16609  DE
```

Agora entre com a parte em BASIC e rode o programa.

```
1 REM...(100 CARACTERES)...
10 INPUT A$
20 PRINT AT 4,8;"CRIATIVIDADE -
II";AT 20,9;"BY MARCIO - 86"
30 FOR A=1 TO LEN A$
40 POKE 16518,CODE A$(A)
45 RAND USR 16514
50 NEXT A
60 GOTO 30
```

Márcio Elias de Morais — PE

Linha TRS-80

Formatando a melhor impressão

Para poupar tempo na formatação de linhas de impressão em seus programas e, ao mesmo tempo, melhorar a documentação dos mesmos, use a seguinte dica: ao invés de calcular cada coluna na linha de impressão para TAB, utilize uma variável com o formato em que os dados serão impressos e depois imprima-os usando LPRINT USING (formato); dado1, dado2 etc... Por exemplo, para emitir uma lista de cobrança no seguinte formato:

```
NOME              DUPLICATA    VALOR      VENCIMENTO
XXXXXXXXXXXXXXX   XXXXXXXX   999,999.99   99/99/99
```

Use o seguinte processo: primeiro defina o formato.

```
XXXXX F$="%              %  %      %  999,999.99  %      %"
```

Na hora de imprimir:

```
XXXXX LPRINT USING F$;N$,D$,V,DT$
```

Ismenio de Melo Souza — PE

Linha APPLE

Cronômetro analógico

Este programinha simula, em micros da linha Apple, um cronômetro analógico, criado por uma tabela de formas. Os ponteiros indicam minuto e segundo, sendo que o limite do cronômetro é de 12 minutos.

Ao dar o RUN, o programa criará o relógio. Pressione qualquer tecla para iniciar ou parar, sendo que neste último caso você também poderá voltar ao BASIC pressionando qualquer tecla.

Eduardo Saito — SP

```
10 REM CRONOMETRO - EDUARDO SAITO
20 POKE 768,1:POKE 769,0:POKE 770,4:POKE
 771,0:POKE 772,36:POKE 773,0:POKE 232,0
:POKE 233,3:HGR2:S=5.4
30 SCALE=40:FOR A=0 TO 64 STEP S:ROT=A:X
DRAW 1 AT 139,95:NEXT A:SCALE=30:HCOLOR=
0:FOR A=0 TO 64 STEP S:ROT=A:DRAW 1 AT13
9,95:DRAW 1 AT 139,96:DRAW 1 AT 139,96:D
RAW 1 AT 140,95:DRAW 1 AT 139,94:DRAW 1
AT 140,94:NEXT A
40 HCOLOR=3:HPLOT 0,0 TO 279,0 TO 279,19
1 TO 0,191 TO 0,0:WAIT-16384,128:POKE-16
368,0:PRINT CHR$(7)
50 FOR A=1 TO 2:M=X:H=Y:HCOLOR=A+2:H=H+M
/60:SCALE=25:M=M/60:M=M*64:ROT=M:DRAW 1
AT 139,95:SCALE=16:H=H*5:FOR B=1 TO 1:NE
XT B
60 ROT=H:DRAW 1 AT 139,95:DRAW 1 AT 139,
96:DRAW 1 AT 140,95:DRAW 1 AT 139,94:DRA
W 1 AT 138,95
70 IF A=1 THEN FOR B=1 TO 570:NEXT B:IF
PEEK(-16384)>128 THEN PRINT CHR$(7):POKE
 -16368,0:WAIT-16384,128:TEXT:HOME:END
80 NEXT A
90 X=X+1:IF X>60 THEN X=1:Y=Y+1
100 GOTO 50
```

Linha ZX Spectrum

POKEs e PEEKs

Com estes POKEs e PEEKs você tem novos recursos para auxiliá-lo na feitura de seus programas. Experimente.

```
POKE 23617,1:INPUT A$ -> faz o INPUT com cursor E

POKE 23617,2:INPUT A$ -> faz o INPUT com cursor G

POKE 23658,0          -> passa para o modo CAPS LOCK (cursor C)

POKE 23658,8          -> volta ao cursor L

POKE 23613,0          -> (na primeira linha do programa) da NEW

(PEEK 23672+256*PEEK 23673+65536*PEEK 23674)/60 -> indica o tempo em
                                                   segundos, desde que
                                                   o micro foi ligado
                                                   pela ultima vez

POKE 23756,0          -> zera a primeira linha do programa
```

Wilson Roberto Afonso — RS

TRS COLOR

Entrada controlada

Esta dica controla a entrada de dados pelo número de caracteres desejados, que é atribuído na variável N, linha 10.

```
10 CLS:INPUT"QUANTOS CARACTERES
";N
20 CLS:DIM N$(N)
30 FOR F=1 TO N:PRINT@64+F,"-";:
NEXT:NO$=""
40 FOR H=1 TO N
50 PRINT@64+H,CHR$(207);
60 A$=INKEY$:IF A$="" THEN 60
70 SOUND 200,1
80 IF A$=CHR$(8) THEN CLS:GOTO 3
0
90 PRINT@64+H,A$;
100 N$(H)=A$:NO$=NO$+N$(H)
110 NEXT H
120 PRINT:PRINT NO$
```

Não é possível apagar apenas um caráter: caso necessite, serão apagados todos eles através de ← (o CHR$ (207) servirá como cursor).

Tárcio Nery — BA

Linha MSX

Som em jogos

Aproveite esta pequena rotina em seus programas, principalmente jogos, enriquecendo-os com o efeito sonoro produzido.

```
10 REM MSX SUL-CAIXA POSTAL 196- CAMPO M
OURAO -PR- CEP:87300 - TEL 0448-231778
20 DEFINTR:SOUND 10,9:SOUND 5,0:SOUND 9,
9:SOUND 3,0:FOR G=4 TO 127 STEP.25:FOR R
=0 TO 255 STEP G:SOUND 4,R:SOUND 2,63+R*
.5:NEXT
30 FOR R=255 TO 0 STEP-G:SOUND 4,R:SOUND
 2,63+R*.5:NEXT:NEXT:SOUND 10,0:SOUND 9,
0
```

Luciano Nogueira Marmontel — PR

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha MSX

Goonies

Se você está cansado de jogar nas primeiras fases de *Goonies*, então passe adiante com as seguintes palavras-chave:

Fase 1: *Goonies*
Fase 2: *Mr. Sloth*
Fase 3: *Goon Docks*
Fase 4: *Doubloon*
Fase 5: *One Eyed Willy*

Para usá-las, pressione CONTROL K e entre com a palavra-chave, não perdendo desse modo os bônus já adquiridos Bom divertimento!

Luiz Alberto Rodrigues — DF

Linha ZX Spectrum

CLS diferente

Rode o seguinte programinha e veja o resultado. Ele pode ser utilizado como sub-rotina de um outro programa (neste caso, talvez seja preciso fazer algumas alterações).

```
1 OVER 0
5 FOR I=0 TO 703:PRINT "C";:NEXT
I
10 FOR I=0 TO 87
20 PLOT I,I
30 DRAW  255-I*2,0:DRAW 0,175-I*
2
40 DRAW -255+I*2,0:DRAW 0,-175+I
*2
60 NEXT I
70 OVER 1:FOR I=0 TO 87:PLOT OVE
R 0;I,I:DRAW 255-I*2,0:DRAW 0,17
5-I*2
80 DRAW -255+I*2,0:DRAW 0,-175+I
*2:NEXT I:OVER 0:STOP
```

Andreas Christian Hasenack — RS

Linha APPLE

Apresentação

```
10  HOME
20  FOR A = 1 TO 14
30  READ A$
40  FOR I = 39 TO A STEP  - 1
50  VTAB 12: HTAB I
60  PRINT A$
70  VTAB 12: HTAB I + 1
80  PRINT " "
90  NEXT I: NEXT A
100  DATA  M,I,C,R,O, ,S,I,S,T,E,
     M,A,S
```

Esta dica poderá ser usada para a apresentação de programas, jogos etc..

Para modificar a mensagem, basta mudar o valor do laço FOR/NEXT da linha 20 de acordo com o número de letras da nova palavra e colocá-la dentro da instrução DATA da linha 100.

Adriano M. P. Brazão — SP

Linha MSX

Novas imagens

Esta dica usa a instrução DRAW e os seus recursos de escala e rotação. Com ela, você pode alterar a seu gosto a geometria do desenho, criando assim novas imagens:

```
10 REM CARLOS R. MACEDONIO
20 SCREEN 2:PSET(128,96)
30 FOR X=0 TO 255
40 FOR T=0 TO 3
50 V=INT(RND(TIME)*15):COLOR V
60 DRAW "U18F18D8H18"
70 DRAW "S=X;":DRAW "A=T;"
80 SOUND 0,X:SOUND 1,T:SOUND 8,15
90 NEXT T
100 NEXT X
110 GOTO 110
```

Tente estes exemplos alterando a linha 60:

Losango:

```
60 DRAW "E20F20G20H20"
```

Catavento:

```
60 DRAW "E2F2L4"
```

Carlos Roberto Macedonio — SP

Linha TRS-80

Número de execuções

Experimente esta rotina em micros compatíveis com o modelo III e veja que a cada execução do programa, o N muda de valor:

```
10 N$="000":N=VAL(N$)
20 N=N+1
30 M$=MID$(STR$(N),2):L=LEN(N$)-LEN(M$)
40 M$=M$+STRING$(L,32)
50 X!=VARPTR(N$):GOSUB80:N1!=X!
60 X!=VARPTR(M$):GOSUB80:N2!=X!
70 FOR X=0 TO LEN (N$)-1:POKE N1!+X,PEEK(N2!+X):NEXT X
75 END
80 X!=PEEK(X!+1)+256*PEEK(X!+2):X!=X!+(X!>32767)*65536:RETURN
```

Você poderá utilizar esta dica para saber quantas vezes um programa foi executado. Para isso, adicione a seguinte linha:

```
74 SAVE"PROG"
```

ou ainda utilizá-la como uma técnica de proteção contra a pirataria (ver o item *programas com tempo de vida limitado* no artigo "Copia de programas: defesa" de autoria de Pierre Lavelle, publicado em MS nº 54). Uma idéia meio "terrorista" é adicionar a seguinte linha, que fará o programa ser deletado do diretório depois de cinco execuções:

```
74 IF N>5 THEN NEW ELSE SAVE"PROG"
```

José Henrique Fatia da Silva — RJ

Linha ZX81

Efeitos com retângulos

Esta rotina em Assembler coloca na tela um retângulo qualquer com suas características determinadas através de POKEs nos seguintes endereços:

16507 – número de linhas a saltar (0 a 23)

16508 – número de linhas a imprimir (1 a 24)

16514 – número de colunas a saltar (0 a 31)

16515 – número de colunas a imprimir (1 a 32)

16417 – código do caráter que vai para a tela

Crie uma linha REM com 102 caracteres e entre com o bloco em Assembler:

```
16514 - 00 00 F3 16 00 2A 7B 40
16522 - 7D 84 FE 19 30 03 E5 CB
16530 - C2 2A 82 40 7D 84 FE 21
16538 - 30 03 E5 CB CA 7A FE 03
16546 - 30 05 FE 00 CB C1 C9 D1
16554 - C1 D5 2A 0C 40 11 21 00
16562 - 79 FE 00 28 04 19 0D 20
16570 - FC D1 4A 7B 82 57 3E 21
16578 - 00 92 57 D5 C5 C1 D1 D5
16586 - C5 7B FE 00 28 04 23 1D
16594 - 20 FC 3A 21 40 23 77 0D
16602 - 20 F8 23 15 20 FC C1 05
16610 - C5 20 E2 C1 D1 C9
```

Esta rotina pode ser utilizada como um CLS dimensionado (apagando determinadas partes da tela) e intercalando-se chamadas com alterações nos endereços, chega-se a produzir efeitos de movimento. Acrescente este programa à rotina (executada com RAND USR 16516) e veja a sua versatilidade:

```
300 LET SL=11
302 LET SC=15
304 FOR X=4 TO 24 STEP 2
306 POKE 16508,X
308 POKE 16515,X
310 LET SL=SL-1
312 LET SC=SC-1
314 POKE 16507,SL
316 POKE 1614,SC
318 POKE 16417,128
320 RAND USR 16516
322 POKE 16508,X-2
324 POKE 1615,X-2
326 POKE 16507,SL+1
328 POKE 16514,SC+1
330 POKE 16417,0
332 RAND USR 16516
334 NEXT X
336 GOTO 300
```

Francisco de Aguiar – PI

Linha ZX Spectrum

Senha

Esta pequena dica *tranca* um programa exigindo a senha e após três tentativas, o programa se autodestrói. Grave-o com GOTO 9999.

```
  1 REM PASSWORD - TK 90X
      AGOSTINHO F R FILHO
      24/05/86
  2 LET x=1: CLS
  3 INPUT "informe a senha: ";
LINE a$: IF a$="senha" THEN GOTO
 6
  4 LET x=x+1: IF x=4 THEN NEW
  5 GOTO 3
  6 CLS : PRINT PAPER 6;AT 10,1
0;"diga mestre."
  7 STOP : REM coloque a partir
 daqui as informacoes
9999 SAVE "senha" LINE 2
```

Agostinho Fernandes R. Filho – PA

Linha APPLE

Rotina de som

A pequena linha, abaixo listada, traz ao endereço 768 (300 em hexadecimal) uma rotina da ROM, composta de 14 bytes, que produz variados tipos de sons. Esta rotina é a mesma que produz o *beep* (caráter ASCII 7 – o Control–G).

Para executar o som basta dar um POKE 769, duração do som e POKE 771, número da nota. A linha que copia a rotina no endereço 768 é a seguinte:

```
10 CO$ = "300<FBE2.FBEFM D823G": FOR
    A = 1 TO  LEN (CO$): POKE 51
    1 + A, ASC ( MID$ (CO$,A,1))
     + 128: NEXT : POKE 72,0: CALL
     - 144
20  INPUT "NOTA: ";N: INPUT "DURA
    CAO: ";D: IF N < 0 OR D < 0 OR
    N > 255 OR D > 255 THEN 20
30  POKE 769,N: POKE 771,D: CALL
    768
```

Eduardo Saito – SP

Linha TRS-80

RND do TRS-80

Esta dica ajudará bastante os programas que têm problemas com a instrução RND.

O manual do CP 500 diz como obter números inteiros entre 1 e X (RND (X)); e números de precisão simples, entre 0 e 1. Porém, não diz como simular números inteiros entre X e Y, sendo eles negativos ou positivos. Após alguns cálculos, fiz a seguinte fórmula:

RND (Y – X + 1) + X – 1

Na fórmula, X e Y são dois números inteiros, positivos ou negativos, porém X deverá ser menor que Y.

Edson Assumpção Tacão – PR

ZX SPECTRUM

Mundo de cores

Provavelmente você já se surpreendeu alguma vez com o seu computador apresentando na tela do televisor umas cores não muito familiares.

Rode então esta dica bem simples, que, com efeitos de fusão das cores normais do TK90X, pode fornecer cerca de 105 novas cores.

O INPUT da linha 20 determina o tom predominante.

```
  10 CLS:BORDER 0
  20 INPUT "COR DE FUNDO";COR
  30 LET ACR=0
  40 LET BR=0: GOSUB 60: LET BR=
1: GOSUB 60
  50 GOTO 20
  60 FOR I=1 TO 7
  70 INK 1
  80 FOR F=1 TO 16 STEP 2
  90 PLOT F+ACR,0
 100 DRAW PAPER COR; BRIGHT BR;0
,175
 110 NEXT F
 120 LET ACR=ACR+16
 130 NEXT I
 140 RETURN
```

Augusto Erzinger Jr. – SC

Linha TRS-COLOR

Beep

Com esta rotina, toda tecla pressionada emitirá um *beep* que poderá ser alterado com o comando **SOUND X,Y:**

```
1'**   ROTINA DE BEEP    **
10'* BY RODRIGO TOMAZONI *
20 A=PEEK(116)*256+PEEK(117)-632
30 X=INT(A/256):Y=A-(X*256)
40 FOR I=A TO A+9:READ B
50 POKE I,B:NEXT I
60 POKE 362,126:POKE 363,X
70 POKE 364,Y:CLEAR 200,A
80 DATA 52,118,198,1,189,169,81,
53,118,57
```

Rodrigo Tomazoni – PB

Linha TRS-80 (III)

Diretório incrementado

Esta dica simula uma instrução do tipo CMD"D:X", sendo X um número de zero a três.

```
10 CLS:INPUT"DRIVE: ";D:IFD<0 OR D>3 THEN 10 ELSE CLS
20 X$=STRING$(10,32):V=VARPTR(X$):V1=PEEK(V+1):V2=PEEK(V+2)
30 V=V1+256*V2:IF V>32767 THEN V=V-65536
40 FOR X=V TO V+9:READ Y:POKE X,Y:NEXT X:POKE V+5,D:DEFUSR=V:X=U
SR(0):PRINT"DRIVE: "$D,
50 EN=-4096:FOR X=EN TO EN+1761 STEP 22:FOR Y=X TO X+11:P=PEEK(V
):IF P<>58 THEN PRINTCHR$(PEEK(Y));:NEXT Y
60 PRINT,:IF PEEK(X+22)<>43 THEN NEXT X
70 PRINT:PRINT:END
80 DATA 33,0,240,1,0,0,205,144,66,201
```

A principal diferença entre o CMD e a rotina é o fato dela exibir na tela, também, os arquivos invisíveis.

Edison Assumpção Tacão — PR

Linha ZX Spectrum

Letras espertas

A rotina abaixo pode dar maior requinte às apresentações dos seus programas.

```
10 POKE 23693,68: POKE 23624,68:
 POKE 23658,8: CLS : OUT 254,0
20 INPUT "PALAVRA (MAX. 32 LETRA
S)? ";A$: PRINT AT 0,0;A$
30 INPUT "LINHA (MAIOR QUE 10)?
";A: LET B=0
40 INPUT "COM INCLINACAO? (S/N)
";B$
50 INPUT "ALTURA ";I'"LARGURA ";
C'"ESPACAMENTO ";E
60 FOR H=175 TO 167 STEP -1: FOR
 G=0 TO LEN A$*8-1
70 IF POINT (G,H)=1 THEN FOR S=1
 TO I: PLOT B+G*(E-(C<E)),H-A+S:
 DRAW C,0: NEXT S
80 NEXT G
90 IF B$="S" THEN LET B=B+1
100 LET A=A+I-1: NEXT H
110 GOTO 20
```

Eduard Montgomery Meira Costa — PB

Linha MSX

Esculturas

Veja como fazer belas esculturas usando formas geométricas:

```
10 REM ESCULTURAS
20 K=75
30 C=150
40 SCREEN2
50 PSET(128,177)
60 FOR I=0 TO C*2 STEPC/30
70 K=K-.25
80 IF K<0 THEN 140
90 X=128+K*SIN(I)
100 Y=88+K*COS(I)*1.2
110 LINE-(X,Y)
120 NEXT
130 GOTO 60
140 GOTO 140
```

Jorge Marcelo M. da Silva-RJ

Linha APPLE

Desenhos com pontos

Observe os efeitos produzidos por esta dica com pontos no vídeo. Depois de rodá-la, experimente alterar o STEP do laço FOR ... NEXT. O resultado será o aumento ou a diminuição da densidade dos pontos.

Após ter sido rodado o programa, experimente apertar qualquer tecla, que mudará o desenho.

```
5  REM  ** RUY MENEZES **
10  HGR2 : HCOLOR= 3
20 PI = 4 *  ATN (1)
30 A =  INT ( RND (1) * 100)
40 B =  INT ( RND (1) * 40)
50  FOR N = 0 TO 200 * PI STEP B /
    100
60 W =  PEEK ( - 16384)
70  POKE  - 16368,0
80  IF W > 127 THEN 10
90 P = 140 + B *  SIN (N):Q = B0 +
    B *  COS (N)
100  HPLOT P + B *  SIN (X),Q + B
     *  COS (X)
110 X = X - A / 1000
120  NEXT N
130  GOTO 30
```

Ruy Menezes Neto — SP

LINHA ZX81

CLS cônico

Esta dica desenha molduras concêntricas até que a tela seja apagada. Crie uma linha **REM** com 69 caracteres.

```
16514  2A 0C 40 06 0D 11 16 20
16522  C5 D5 42 23 36 00 10 FB
16530  43 11 21 00 19 36 00 10
16538  FB 19 D1 D5 42 36 00 2B
16546  10 FB 23 43 11 21 00 C6
16554  00 ED 52 36 00 10 FB E5
16562  06 02 0E FF 0D 20 FD 10
16570  F9 E1 D1 C1 15 15 1D 1D
16578  05 C2 8A 40 C9
```

Teste com este programa:

```
10 REM (69 CARACTERES QUAISQUER)
15 POKE 16418,0
20 FRO F=0 TO 21
30 PRINT"ABCDEFGHIJKLMNOPQRSTUVW
XYZ123456"
40 NEXT F
50 PAUSE 4E4
60 RAND USR 16514
```

É aconselhável usar o **POKE** da linha 15 sempre que for rodar a dica.

Renato Fernandes Cantão — SP

Linha ZX Spectrum

Draws

Observe o interessante efeito causado por DRAW em OVER 1.

```
  1 REM &&& ALPS &&&
  2 BRIGHT 1: PAPER 6: INK 1: O
VER 1: CLS
  3 FOR A=0 TO 175: LET Y1=175-
A: LET X1=A-255
  4 PLOT 255,A: DRAW -A,Y1: DRA
W X1,-A: DRAW A,-Y1: DRAW -X1,A
  5 NEXT A
```

Andre Luiz Paes da Silva — PE

Na dica

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha TRS-80

INKEY$/AUX

Esta dica melhora a entrada de dados via teclado de uma forma muito simples. A princípio, ela mais parece uma simulação de INPUT usando INKEY$, mas na verdade faz mais que isso, sendo de muita utilidade para os usuários que só dispõem de cassete.

Normalmente, deverá haver um programa que chame a rotina. No início deste programa, deverá existir uma instrução CLEAR X, onde X é um valor de, no mínimo, 300. Não foi colocado este CLEAR na rotina para que ele não atrapalhasse as variáveis do programa que chamasse a rotina.

Depois, uma linha no formato: P=XXX:GOSUB 5000 chamará a rotina, sendo que XXX é a posição de tela de onde se deseja começar a imprimir.

```
5000 X$="":D=P
5010 A$=CHR$(PEEK(16419)):D$=A$:B=(-1+(PEEK(16416)=1))
5020 IF B<0 THEN B$=CHR$(32) ELSE B$=A$
5030 PRINT@P,D$;:FOR X=1 TO 10:C$=INKEY$
5040 IF C$="" THEN NEXT X:GOTO 5100 ELSE H=ASC(C$):IF H>31 THEN
5090
5050 IF H<>13 AND H<>8 AND H<>24 THEN 5100
5060 IF C$=CHR$(13) THEN PRINT@P,CHR$(32);:RETURN
5070 IF C$=CHR$(24) THEN PRINT@D,STRING$(LEN(X$)+1,32);:P=D:X$=""
:GOTO 5100
5080 IF C$=CHR$(8) AND P>D THEN PRINT@P,CHR$(32);:P=P-2:X$=LEFT$
(X$,LEN(X$)-1):C$="":GOTO 5090 ELSE 5100
5090 X$=X$+C$:PRINT@D,X$;:P=P+1
5100 IF D$=A$ THEN D$=B$ ELSE D$=A$
5110 GOTO 5020
```

Edison Assumpção Tacão — PR

Linha TRS-80 (III)

Dia da semana

Esta sub-rotina calcula o dia da semana a partir da data do sistema operacional e pode ser colocada em qualquer parte do programa. Chame-a com GOSUB 10000.

```
1 GOSUB 10000
2 PRINTDI$
3 END
10000 D$="SABADO DOMINGOSEGUNDATERCA  QUARTA QUINTA SEXTA  "
10010 B=PEEK(16923):M=PEEK(16924):Y=PEEK(16922)
10020 Y=Y+1900:IF M<3 THEN M=M+12:Y=Y-1
10030 N=B+2*M+INT(.6*(M+1))+Y+INT(Y/4)-INT(Y/100)+INT(Y/400)+2
10040 N=INT((N/2-INT(N/7))*7)-1:N=N+1
10050 DI$=MID(D$,N*7-6,7):RETURN
```

Ismênio de Melo Souza — PE

Linha ZX Spectrum

Girotexto

Esta dica, bastante interessante, faz com que qualquer mensagem, posicionada em qualquer parte da tela, gire!

```
10 FOR f=30 TO 0 STEP -1
20 POKE 23606,f
30 PRINT AT 10,10;"MICRO SISTE
MAS"
40 PAUSE 3
50 NEXT f
```

Edgard Dias de Medeiros Neto — CE

Linha ZX Spectrum

Efeitos visuais IV

Observe seu computador criar interessantes efeitos visuais, fazendo o valor de "s" variar na linha 5.

```
1 REM IVAN S.MELO FILHO(13 ANOS)
5 OVER 1:INPUT S
10 FOR A=255 TO 0 STEP -S
20 PLOT 255,175:DRAW -A,-175
21 PLOT 0,0:DRAW A,175
30 NEXT A
```

Ivan S. Melo Filho — AL

Linha MSX

Roda de retas

Veja como você pode criar um círculo utilizando a função LINE.

```
10 COLOR 7,4,4:SCREEN 2:PI=4*ATN(1)
20 FOR N=0 TO 2*PI STEP .1
30 X1=128+90*COS(N):X2=128+90*COS(N+PI/1
.5):Y1=96+90*SIN(N):Y2=96+90*SIN(N+PI/1.
5)
40 LINE (X1,Y1)-(X2,Y2)
50 NEXT N
60 FOR R=45 TO 0 STEP -1
70 CIRCLE(128,96),R
80 NEXT R
90 GOTO 90
```

João Heraclito M. de Castro — RJ

Linha TRS-COLOR

Impressão de diretórios

Para aqueles que têm unidade de disco e impressora e gostariam de imprimir seus diretórios, usem esta dica:

```
POKE 111,254:DIR X    (X=0 a 3)
```

Para um efeito melhor, o programa abaixo resolve:

```
10 DIR
20 LINEINPUT"NOME DO DISCO- ";A$
30 PRINT #-2,"DISCO- ";A$
40 POKE 111,254:DIR
50 PRINT #-2,FREE(0)"GRANULES LI
VRES"
60 PRINT"APERTE QUALQUER TECLA"
70 EXEC 44539
80 CLS:RUN
```

Fábio Luis de Paoli — SP

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - grupo 1210. Centro, Rio de Janeiro. RJ. CEP 20030

Linha MSX

Texto nos jogos

Esta dica é util para se utilizar em instruções de jogos, onde é preciso misturar texto com gráfico.

O tamanho do texto é de 14 linhas por 26 letras; a variável B determina o número de linhas de texto; e a variável A engrossa a letra.

```
10 SCREEN2:COLOR15,1,1:OPEN"GRP:"AS#1
20 LINE(20,20)-(240,180),14,BF
30 LINE(25,15)-(245,175),15,BF
40 FOR A=0 TO 1:RESTORE 80
50 FOR B=1 TO 4:READ M$:COLOR1
60 PRESET(A+35,(B*10)+20):PRINT#1,M$
70 NEXT:NEXT
80 DATA Nelson Hisashi Tamura, Mogi das
Cruzes - SP, MICRO SISTEMAS, 12 de marco
 de 1987
90 GOTO 90
```

Nelson Hisashi Tamura — SP

Linha ZX81

Sinais sonoros

Esta rotina em Assembler envia sinais sonoros para o televisor. O programa deve estar em uma linha REM; e para rodá-lo, use LET L=USR 16527 no modo FAST.

```
16514  C5 D5 41 3E 01 3D 20 FD
16522  10 F9 D1 C1 C9 0E 00 11
16530  FF 00 06 01 D3 FF CD 82
16538  40 DB FE CD 82 40 10 F4
16546  0C 1B 7A B3 20 EC C9
```

O endereço 16518 contém o loop em linguagem de máquina que varia de 1 a 255(01 a FF).

Osmar de Paula Filho — SP

Linha ZX Spectrum

Som para jogos

Utilize esta dica para obter som em seus programas.

```
8000 FOR N=10 TO 60 STEP 5
8010 SOUND .01,N
8020 SOUND .01,N-2
8030 NEXT N
```

Marcos Alexandre M. Sandi — SP

Linha MSX

Efeitos

Esta dica, depois de dado o comando **RUN**, forma um risco diagonal de onde surge um conjunto de cores e brilhos interessante.

```
10 SCREEN2
20 FOR X1=0 TO 255
30 X2=255-X1
40 C=C+1:IF C=16 THEN C=0
50 LINE(X1,0)-(X2,191),C
60 NEXT X1
70 FOR Y1=0 TO 191
80 Y2=191-Y1
90 F=F+.1:IF F=16 THEN F=0
100 D=INT(F)
110 LINE(0,Y1)-(255,Y2),D
120 NEXTY1
```

Wagner de Campos Ramos — MG

Linha TRS-COLOR

Auto-repeat

Utilize esta dica para se obter auto-repeat sem usar PEEKs no buffer de teclado.

```
10 ' AUTO-REPEAT
20 A$="A":A=VARPTR(A$):POKE A,8:
POKE A+2,&H01:POKE A+3,&H52
30 PRINT @ 384,A;:GOTO 30
40 B$="A":A=VARPTR(B$):POKE A+2,
0:POKE A+3,135
100 IF A$=STRING$(8,255) THEN 10
0 ELSE PRINT B$;:GOTO 100
```

Maximiliano Hauer Pereira — RJ

Linha APPLE

Relógio

Com esta dica, os usuários de compatíveis com o Apple II poderão marcar as horas na tela e programar um alarme para outra hora:

```
0  HOME
1  POKE 771,173: POKE 772,48: POKE
     773,192: POKE 774,136: POKE
     775,208: POKE 776,4: POKE 77
     7,198: POKE 778,1: POKE 779,
     240: POKE 780,8: POKE 781,20
     2: POKE 782,208: POKE 783,24
     6: POKE 784,166: POKE 785,0:
      POKE 786,76: POKE 787,3: POKE
     788,3: POKE 789,96
10  HOME
15  PRINT  TAB( 10);"RELOGIO DIGI
     TAL"
16  PRINT : PRINT
20  INPUT "HORA INICIAL => ";HI
21  PRINT
30  INPUT "MIN. INICIAL => ";MI
34  PRINT
35  PRINT "ALARME (S/N)? ";: GET
     R$
45  IF R$ = "N" THEN  GOTO 60
46  PRINT
50  INPUT "HORA ALARME ";HA
55  INPUT "MIN. ALARME ";MA
60  HOME
65  VTAB 1: HTAB 1: PRINT ;HI;":"
     ;MI;":";
66  FOR A = 00 TO 59
67  HTAB 7: VTAB 1: PRINT A;" "
68  FOR B = 1 TO 730: NEXT B
69  NEXT A
75 MI = MI + 1
80  IF MI = 60 THEN MI = 0:HI = H
     I + 1
90  IF HI = 24 THEN HI = 0
95  IF (HI = HA AND MI = MA) THEN
     LE = 15:PI = 10: POKE 0,255 -
     PI: POKE 2,LE: CALL 771:PI =
     5: POKE 0,255 - PI: POKE 1,L
     E: CALL 771
100  GOTO 65
```

Henrique Veiga C. Cunha — GO

Linha ZX81

Derretimento do vídeo

Crie uma linha REM com 37 caracteres e rode esta dica com RAND USR 16514.

```
16514   16 00 2A 0C 40 06 16 23
16522   0E 20 7E FE 80 20 02 D6
16530   41 FE 00 28 04 16 01 3D
16538   77 23 0D 20 ED 10 E8 3E
16546   01 BA 28 DC C9
```

Segue um programa exemplo:

```
10 REM (37 CARACTERES QUAISQUER)
20 FOR F=0 TO 21
30 PRINT"ABCDEFGHIJKLMNOPQRSTUVW
XYZ123456"
40 NEXT F
50 PAUSE 4E4
60 RAND USR 16514
```

Renato Fernandes Cantão — SP

Linha TRS-80

Um CLS diferente

Experimente usar esta dica em seus programas e acrescente sofisticação aos mesmos.

```
5 CLEAR 300:CLS
10 A$=STRING$(64,140):B$=STRING$(64,128)
20 FOR N=0 TO 510:PRINT"MS";:NEXT N
30 IF INKEY$="" THEN 30
40 FOR N=0 TO 448 STEP 64:PRINT@N,A$;@N,B$;@959-N,A$;@959-N,B$;:
NEXT N
50 IF INKEY$="" THEN 50
60 RUN
```

Jorge Luiz Mangini Silva — RJ

Linha ZX Spectrum

Relevo em Pascal

Para os adeptos da programação Pascal, esta dica poderá ser útil àqueles que utilizam o compilador Pascal 1.6 do ZX Spectrum em função da aplicação da procedure PLOT em seus programas:

```
PROGRAM RELEVO;
VAR
  A,X,Y:REAL;
PROCEDURE PLOT(X,Y:INTEGER);
  BEGIN
  INLINE(#FD,#21,#3A,#5C,#DD,#46,#02,#DD,
         #4E,#04,#CD,#E5,#22);
  END;
BEGIN
PAGE;
A:=0;REPEAT
  X:=0;REPEAT
    Y:=-10*COS(3*SQRT(SQR(X-5)+SQR(A-5)))/2+30;
    PLOT(ENTIER(X*24+A),ENTIER(Y+A*10));
    X:=X+0.07; UNTIL X>10;
  A:=A+0.2; UNTIL A>11;
END.
```

Márcio Henrique A. Costa — RJ

Linha TRS-COLOR

Quadrados e círculos

Observe os belos efeitos gerados por esta dica:

```
10 PMODE 4,1:PCLS:SCREEN 1,1:PMO
DE 1,1
20 POKE &HB9,64:'ALTURA DO SISTE
MA GRAFICO P/ 64 BYTES P/ LINHA
30 IF RND(2)=1 THEN POKE &HBB,0
ELSE POKE &HBB,32
40 COLOR RND(4):LINE-(RND(255),R
ND(191)),PSET,BF
50 GOTO 30
```

Para se obter círculos concêntricos, altere as linhas 30 e 40 para:

```
30 K1=RND(4):K2=RND(4)
40 FOR R=1 TO 70 STEP 2:POKE &HB
B,0:POKE 178,K1:CIRCLE(128,96),R
:POKE &HBB,32:POKE 178,K2:CIRCLE
(128,96),R:NEXT
```

Maximiliano Hauer Pereira — RJ

Linha TRS-80

Mudando o cursor

Usando esta pequena dica, pode-se mudar o cursor para qualquer caractere: POKE 16419,1.

Obs.: no lugar do 1, pode-se colocar qualquer número entre 0 e 255.

Carlos Alberto Netto da Fonseca - SP

Linha APPLE

Manipulando o vídeo II

Aí vai a segunda parte de dicas que facilitam na manipulação da tela:

1) Quadricular tela de alta resolução gráfica sem apagá-la:

```
POKE768,1:POKE769,0:POKE770,4:POKE771,0:
POKE772,4:POKE773,0:POKE232,0:POKE233,3:
ROT=0:SCALE=192:FOR B=0 TO 279 STEP 80:
FOR A=B TO B+39:XDRAW 1 AT A,0:NEXT A:NE
XT B
POKE772,36:ROT=16:SCALE=140:FOR B=0 TO
191 STEP 80:FOR A=B TO B+31:XDRAW 1 AT
0,A:NEXT A:NEXT B
```

2) Encher tela de alta resolução gráfica com círculos:

```
POKE768,1:POKE769,0:POKE770:4:POKE771,0:
POKE772,4:POKE773,0:POKE232,0:POKE233,3:
ROT=0:SCALE=10
FOR A=20 TO 279 STEP 30:FOR B=20 TO 191
STEP30:FOR C=0 TO 65:ROT=C:DRAW 1 AT A,B
:NEXT C:NEXT B:NEXT A
```

Obs.: selecione a tela de alta resolução gráfica antes de digitar as dicas.

Eduardo Saito — SP

Linha IBM-PC

Controlando o cursor

A dica a seguir oferece um pequeno procedimento em Turbo Pascal para controlar o tamanho do cursor que é comandado por INT 10h.

Para isso o registro AH deve conter o valor 1, os bits 0-4 do CH o número da primeira linha de varredura do cursor e os bits 0-4 do CL o número da última linha. Note que o cursor usa oito linhas de 0 a 7.

```
PROCEDURE TAMANHOCUR (N,M:INTEGER);
TYPE REGS=RECORD AX,BX,CX,DX,BP,SI,DI,DS,ES,FLAGS:INTEGER;
            END;
VAR INTPARS:REGS;
BEGIN
 WITH INTPARS DO
  BEGIN AX:=$100;BX:=0;CX:=256*N+M;DX:=0;
        BP:=0;SI:=0;DI:=0;DS:=0;ES:=0;
        FLAGS:=0;
  END;
 INTR (16,INTPARS);
END;
```

Obs.: TamanhoCur (0,7) dá o tamanho máximo ao cursor e TamanhoCur (8,7) apaga-o.

George Svetlichny — RJ

Linha ZX81

Escrita na vertical

Digite o seu nome e veja-o impresso no vídeo na vertical.

```
10 PRINT "QUAL SEU NOME ?"
20 INPUT A$
30 LET A$=A$
40 FOR F=1 TO LEN A$
50 PRINT AT F+7,2;A$(F)
60 NEXT F
65 PAUSE 300
67 CLS
70 GOTO 10
```

Fernando T. M. Peixoto — MG

Linha APPLE

Bloqueando o RESET

Com o **POKE** abaixo você bloqueará o RESET de seu micro: toda vez que ele for acionado será executado um *Cold Start* (reinicialização do sistema).

POKE 1011,0

Digite isto e pressione RESET logo em seguida: que tal? Descubra como anular este **POKE**, se conseguir.

Atenção: esta dica poderá ser utilizada na listagem do seu programa, dificultando a pirataria.

Alexandre Folle de Menezes — RS

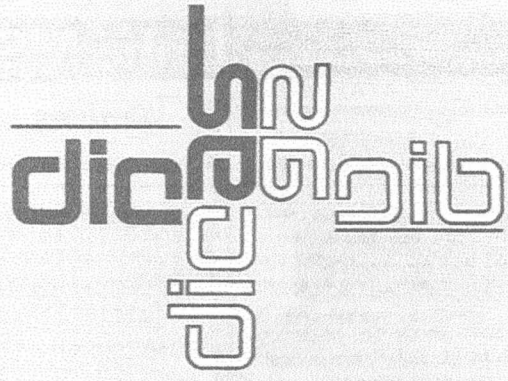
Linha TRS-80

Inversão de tela

Faça uma tela gráfica usando os caracteres gráficos disponíveis em seu computador (códigos de 128 até 191), acrescente as seguintes linhas, e observe o resultado.

```
1000 FOR N=0 TO 1023:Z=PEEK(15360+N)
1010 IF Z<=159 THEN S=ABS(Z-128):A=191-S
1020 IF Z>=160 THEN S=191-Z:A=128+S
1030 IF Z=32 THEN A=191
1040 POKE 15360+N,A
1050 NEXT N
```

Josué de Oliveira Carvalho — MG



Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 — grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha ZX Spectrum

Som em Assembler

Veja o que o seu TK90X/95 é capaz de fazer em termos de efeitos sonoros.

```
 10 CLEAR 28500: FOR N=28500 TO
28527: READ V: POKE N,V: NEXT N
: RAND USR 28500
 20 DATA 33,2,1,25,1,22,0,122,2
11,254,20,124,186,32,248,11,62,0
,184,32,240,36,62,210,188,32,231
,201
 30 BORDER 7
```

Sílvio José Lima Moreira — MA

Linha MSX

Depurador de erros

Esta dica pode auxiliar muito quem depura programas extensos em BASIC, onde é muito provável ocorrer erro de digitação.

A técnica consiste em usar a rotina de manipulação de erros do MSX e o recurso que permite a listagem da última linha digitada ou a linha na qual o interpretador BASIC encontrou um erro, usando-se o comando **LIST**.

```
10 ON ERROR GOTO 10000: 'VAI P/ O FINAL DO PROGRAMA
20 :
30 :
40 :
9999 END
10000 CLS: PRINT "ERRO N.";ERR;"NA LINHA";ERL
10010 LIST.: 'LISTA A LINHA ONDE OCORREU O ERRO
10020 END
```

Roberto Sidnei Chiandotti — SP

Linha TRS-COLOR

Novo RESTORE

Para simular o comando **RESTORE** *número de linha*, inexistente no TRS-Color, utilize a seguinte dica:

```
10 CLS
20 DATA 4F,BD,B3,ED,9E,19,20
30 DATA 02,1F,21,10,AE,84,27
40 DATA 09,10,A3,02,25,04,27
50 DATA 04,20,F0,9E,19,30,1F
60 DATA 9F,33,39
70 OH$=STRING$(31,32)
80 FOR I=1 TO 31:READ X$
90 X=VAL("&H"+X$):MID$(OH$,I,1)=CHR$(X)
100 NEXT I
110 X=VARPTR(OH$)
[illegible]
125 DATA "DICAS"
130 DATA "MICRO SISTEMAS"
140 DATA "1987"
150 X=USR0(130)
160 READ A$,B$:PRINT A$,B$
170 END
```

O argumento da função USR0 na linha 150 indica a linha para qual se deseja *restaurar*. Caso a linha não exista, a rotina irá *restaurar* na primeira linha do programa.

Nivaldo Nunes de Medeiros Junior — GO

Linha TRS-COLOR

Data/BAS

Calcule dias entre datas e localize o dia da semana para qualquer data do século XX.

```
10 DEFFN DA!(A%,M%,D%)=A%*365+INT((A%-1)/4)+(M%-1)*28+VAL(MID$("
000303060811131619212426",(M%-1)*2+1,2))-((M%>2)AND((A% AND NOT
-4)=0))+D%
20 DEFFN DS$(DA!)=MID$("SEXTA  SABADO DOMINGOSEGUNDATERCA  QUART
A QUINTA  ",(DA!-INT(DA!/7)*7)*7+1,7)
30 INPUT"DIGITE O DIA (DD)";D%:INPUT"DIGITE O MES (MM)";M%:INPUT
"DIGITE O ANO (AAAA)";A%:IF A%<1900 OR A%>9999 THEN 30
40 D!=FN DA!(A%,M%,D%):DIA$=FN DS$(D!)
50 PRINT DIA$;D%;"/";M%;"/";A%
55 'abaixo, como calcular dias entre datas.
60 PRINT FN DA!(1987,03,31)-FN DA!(1961,06,23)'substituir as dat
as, por variaveis correspondentes
```

Adalberto Silva — Rj

Linha MSX

Janelas

Aqui vai uma dica para os usuários do MSX que gostam de fazer os seus próprios programas com uma boa apresentação. Esta rotina permite a entrada de dados por uma janela de alta resolução, podendo ser melhorada com a atribuição de variáveis à janela e ao tamanho da string digitada.

Utiliza-se a tecla DEL para corrigir erros de digitação.

```
10 'exemplo de entrada de dados por jane
la em tela de alta resolucao
20 'JOCELYN SPOLAOR JR. F:(0512)33.77.12
30 'AV.GETULIO VARGAS,1501-MENINO DEUS
40 'PORTO ALEGRE-CEP:90.060
50 SCREEN 2:COLOR 1,14,4:CLS:DIMA$(30):L
ET N=1:LET C=80
60 LINE(22,22)-(232,172),1,BF
70 LINE(20,20)-(230,170),11,BF
80 LINE(25,25)-(225,165),1,BF
90 LINE(26,26)-(224,164),11,BF
100 OPEN"GRP:"FOROUTPUTAS#1
110 PSET(30,30),11:PRINT#1,"nome:"
120 A$(N)=INPUT$(1)
130 IFA$(N)=CHR$(127)THENC=C-8:N=N-1:PSE
T(C,30),11:COLOR11:PRINT#1," ":GOTO120
140 IFA$(N)=CHR$(13)THEN 180
150 COLOR 1
160 PSET(C,30),11:PRINT#1,A$(N)
170 N=N+1:C=C+8:GOTO120
180 FORF=1TON:W$=W$+A$(F):NEXTF
190 PSET(30,100):PRINT#1,W$
200 GOTO 200
```

Jocelyn Spolaor Jr. — RS

Linha APPLE

Home colorido

Esta dica limpa a tela com a cor desejada. Veja os comandos da linha 10.

```
5  HGR
7  FOR C = 0 TO 7
10  HCOLOR= C: HPLOT 0,0: CALL 62
    454
15 E =  PEEK ( - 16336)
20  NEXT C: GOTO 7
```

E agora um exemplo:

```
10  HGR2 :O = 280 / 192
15  HCOLOR= 2: HPLOT 0,0: CALL 62
    454
17  HCOLOR= 3
20  FOR Y = 0 TO 191 STEP 10
30  HPLOT 0,Y TO 279 - O * Y,0 TO
    279,191 - Y TO O * Y,191 TO
    0,Y: NEXT Y
```

Repare como a tela é preenchida rapidamente.

Christine Cordula — RJ

Linha ZX Spectrum

Cesto

Este programa cria um interessante efeito gráfico utilizando as funções seno e co-seno do seu TK90X.

```
10 FOR D=0 TO 2*PI STEP .1
30 LET E=D+PI/2
50 LET X=128+60*COS D
40 LET Y=80+20*SIN D
60 LET S=128+60*COS E
50 LET T=150+20*SIN E
70 PLOT X,Y: DRAW S-X,T-Y
80 NEXT D
```

Paulo Maurício Costa — MG

LINHA ZX81

Copy localizado

Com o comando Copy obtemos toda a tela na impressora.

A rotina a seguir é útil na elaboração do formato de fichas, pois permite verificar as formas destas antes da impressão.

```
7000 REM VIDEO -- IMPRESSORA
7010 REM
7020 REM ------------------------
7030 REM  GILBERTO F. DA SILVA
7040 REM      CX POSTAL 292
7050 REM  09700 S. BERNARDO SP
7060 REM ------------------------
7070 REM
7080 LET COUNT=0
7090 LET DFILE=PEEK 16396+256*PE
EK 16397
7100 PRINT "QUANTAS LINHAS"
7110 INPUT X
7120 LET X=X+32
7130 LET DFILE=DFILE+1
7140 LET W$=CHR$ PEEK DFILE
7150 IF CODE W$=118 THEN GOTO 71
30
7160 LET COUNT=COUNT+1
7170 LPRINT W$;
7180 IF COUNT<X THEN GOTO 7130
7190 RETURN
```

Gilberto F. da Silva — SP

Linha TRS-COLOR

Espiral

Com esta dica você poderá ver uma espiral se formando na tela.

```
10 G=7:E=G/57.296
20 PCLS:PMODE 4,1:SCREEN 1,1
30 PSET(128,96)
40 FOR A=0 TO 200 STEP E
50 X=A*COS(A)+128:Y=A*SIN(A)*.85
+96
60 IF X<0 OR X>255 OR Y<0 OR Y>1
91 THEN 80
70 PSET(X,Y)
80 NEXT A
90 GOTO 90
```

Adriano Almeida — RS

dicas

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 — grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

Linha TRS-COLOR

POKEs úteis

Os **POKEs** a seguir auxiliarão os usuários da linha TRS-Color:

1) Carrega um programa gravado em fita cassete com o **POKE** acelerador:

```
POKE 143,13:POKE 145,6:POKE 65495,0:CLOAD (M)
```

2) Carrega um programa gravado em alta velocidade sem alterar a velocidade do microprocessador:

```
POKE 143,8:POKE 145,4:CLOAD (M)
```

3) Valores normais:

```
POKE 143,18:POKE 145,10:POKE 65494,0
```

4) Causa um *coldstart* se algum espertinho tentar listar um programa em BASIC (apaga todo o programa):

```
POKE 383,62:POKE 113,0
```

Fábio Massami Yamamoto — SP.

Linha ZX Spectrum

Multisom

Esta dica produz um belo efeito sonoro no seu ZX Spectrum, podendo ser utilizada em jogos:

```
  10 FOR F=30000 TO 30025: READ
A: POKE F,A: NEXT F
  20 FOR F=35000 TO 35027: READ
A: POKE F,A: NEXT F
  30 FOR F=40000 TO 40030: READ
A: POKE F,A: NEXT F
  40 PRINT AT 0,0:"█ █ █ ""█"""
  50 LET I$=INKEY$: IF I$="" THE
N GOTO 50
  60 IF I$="1" THEN RAND USR 300
00
  70 IF I$="2" THEN RAND USR 350
00
  80 IF I$="3" THEN RAND USR 400
00
  90 IF I$="4" THEN RAND USR 400
00: RAND USR 30000: RAND USR 350
00
 100 GOTO 40
 110 DATA 6,10,197,33,0,3,17,1,0
,229,205,181,3,225,17,16,0,167,2
37,82,32,240,193,16,233,201,17,9
8,100,38,50,58,72,92,31,31,31,14
,254,238,16,237,121,67,16,254,37
,32,244,28,21,32,232,201
 120 DATA 58,72,92,31,31,31,6,24
0,14,254,37,32,6,238,16,237,121,
38,238,45,32,244,238,16,237,121,
46,254,16,236,201
```

Observação: para aumentar a duração da opção número 1, é só dar um **POKE** 30001, duração.

José Serrão de Oliveira Filho — PB.

LINHA ZX81

Formato financeiro

Coloque esta rotina (142 bytes) em qualquer parte da RAM e utilize-a em programas que exigem saída de valores na forma financeira usual (exemplo: 85704063.555 se tornará 85.704.063, 56):

```
2A 10 40 11 10 00 19 36
1C 2B E5 36 1C 2B 36 1A
2B E5 01 64 00 CD 20 15
EF 2D A2 01 A0 34 ED 5B
1C 40 1B 2A 14 40 2B 2B
01 05 00 ED B8 EF 04 0F
24 01 05 2A 1B 34 CD A7
0E 79 D1 2A 1C 40 2B 2B
ED B8 13 EB 4F 5F 3E 1B
ED B1 79 A7 28 15 7B D1
E5 C5 ED B0 C1 0C D1 1B
1B 2B 91 4F EB ED B8 13
EB 18 06 C1 7B 43 2B 10
FD E5 2A 10 40 23 23 4F
3E 0B 91 47 23 36 00 10
FB D1 EB 79 D6 03 D8 A7
CB 4F D5 D5 ED B0 2B 36
1B E1 D1 1B 18 EE
```

Supondo que a rotina tenha sido armazenada em uma linha 0 REM, use a parte em BASIC para teste:

```
0 REM (...rotina...)
10 PRINT "FORMA NORMAL","FORMA F
INANCEIRA",,,,
20 LET W$="12345678901234"
30 FOR X=1 TO 10
40 PRINT"? ";
50 INPUT W
60 PRINT W,
70 RAND USR 16514
80 PRINT W$
90 NEXT X
```

Entre com valores menores que 999999994.4 e maiores que 0.1 e confira os resultados.

Francisco de Aguiar — PI.

Linha MSX

Trava listagem

Este **POKE** trava a listagem de programas BASIC.

```
POKE &HFF89,&HC1
```

Este destrava:

```
POKE $HFF89,&HC0
```

Fernando Moraes — PR.

Linha APPLE

ANTI-BREAK

```
1  REM  <<TESTE DO BREAK>>
2  REM  DEPOIS DE (RUN) TENTE
3  REM  DAR (CTRL+RESET) E VOCE
4  REM  VERA O QUE ACONTECE
10  POKE 1010,224: POKE 1011,03: POKE
     1012,69
15  REM  PROTEJA-SE CONTRA
16  REM  OS P*I*R*A*T*A*S
```

Ivan Souza Guerra Lima — BA.

Linha TRS-80

Relógio

Com esta dica, pode-se ativar e desativar o relógio interno no canto superior direito do seu monitor.

Para ativar use **POKE 16912, 1** e para desativar, **POKE 16912, 0.**

Marcelo Gevaera da Silva — SC.

Linha MSX

SCREEN 2 com 40 colunas

Transforme com esta sub-rotina as 32 colunas da SCREEN 2, em 40:

```
10 SCREEN 2:COLOR 15,1,1:CLS:DEFINT A-Z
20 OPEN"GRP:" FOR OUTPUT AS 1
30 PRESET(60,80):PRINT #1,"MICRO SISTEMA
S":REM FORMATO NORMAL
40 A$="MICRO SISTEMAS":REM STRING A SER
ESCRITA EM 40 COLUNAS
50 C=10:L=12:REM SUBSTITUI LOCATE
60 GOSUB 1000
70 A$="NOTOU A DIFERENCA ?"
80 C=10:L=14
90 GOSUB 1000
100 LINE(50,70)-(182,128),10,B
110 GOTO 110
1000 REM ROTINA 40 COLUNAS
1010 FOR P=1 TO LEN(A$)
1020 B$=MID$(A$,P,1)
1030 PRESET(C*6,L*8):PRINT #1,B$
1040 C=C+1:IF C>40 THEN C=0:L=L+1
1050 NEXT P
1060 RETURN
```

Introduza a rotina da linha 1000 em seu programa, alterando as variáveis A$ (texto), C (coluna) e L (linha). Como exemplo, você pode utilizar o programa a partir da linha 10.

Darci Marcondes Filho — SP.

Linha TRS-80

Novo BREAK

Esta dica permite desativar, parcialmente, a tecla BREAK. Ela estará ativa quando pressionada juntamente com SHIFT/SETA (←) à esquerda.

```
10 REM - RESPONDA A (Mem. usada ?) COM : 65526
20 FOR A=0 TO 8:READ B:POKE -9+A,B:NEXT
30 DATA 58,128,56,230,1,202,141,2,201
40 POKE 16396,195:POKE 16397,247:POKE 16398,255
50 NEW
```

Adauto Delmiro da Silva — DF.

Linha ZX Spectrum

Taça colorida

O programa abaixo desenha uma taça que muda de cor continuamente.

```
  1 REM TACA COLORIDA
 10 CLS : FOR T=0 TO 6
 20 INK T: FOR I=1 TO 14 STEP .
1
 30 LET X=10+I: LET Y=20+5*SIN I+
50
 40 PLOT X,Y: DRAW Y-X,X-Y
 50 NEXT I: NEXT T
```

Leandro Rabello Barbosa — RS.

Linha APPLE

Desenhos com ROT e SCALE

Este pequeno programa produz interessantes efeitos na tela do Apple, usando os poderosos **ROT** e **SCALE**:

```
5 ONERR GOTO 70
10 HGR2:HOME:HCOLOR=3
20 FOR A=1 TO 200
30 ROT=A
40 SCALE=9
50 DRAW 1 AT A,A
60 NEXT A
70 END
```

Substitua as seguintes linhas para obter outro interessante efeito:

```
30 ROT=64
40 SCALE=A
```

ou, a partir do programa original:

```
40 SCALE=20
50 DRAW 1 AT A*2,A
```

Edson Zangiacomi Martinez — MG.

LINHA ZX81

Destrava software

Destrave programas fechados, passando-os para a velocidade de *high-speed* (3200 bauds) com a seguinte instrução: RAND USR 836. Grave-os com RAND USR 8405 e copie-os com RAND USR 8630. Verifique a gravação com RAND USR 8539.

Manuel Fidalgo Neto — MG.

Linha TRS-COLOR

Entrada formatada

Utilize esta dica para controlar a entrada de dados via teclado, limitando a introdução de caracteres conforme a necessidade.

```
50 DIM K$(20)
100 CLS:PRINT@96,"NOME.....:":POKE 1146,122:Z=1
110 K$(Z)=INKEY$:POKE 1130+Z,128:POKE 1130+Z,143:IF K$(Z)="" THEN 110
120 IF K$(Z)=CHR$(8) THEN K$(Z)="":Z=Z-1:GOSUB 500:K$(Z)="":GOTO 110
130 IF K$(Z)=CHR$(13) THEN K$(Z)="":GOTO 170
140 PRINT @106+Z,K$(Z);
150 Z=Z+1:IF Z>16 THEN 170
160 GOTO 110
170 FOR Y=1 TO Z
180 N$=N$+K$(Y):K$(Y)=""
190 NEXT Y
200 '.
210 '.
220 '.
499 END
500 IF Z=0 THEN Z=1
510 RETURN
```

Bob Willem A. Anthonijsz — DF.

Linha APPLE

Protege linhas

O comando a seguir protege o número de linhas indicado pelo programador após a vírgula contra **HOME**:

```
POKE 34,(num. de linhas a serem protegidas)
```

Para desativá-lo, basta digitar:

```
POKE 34,0
```

ou pressionar o **RESET**.

Alexandre Lemes Mestriner — SP.

dicas dicas dicas dicas

ZX SPECTRUM

```
1 REM ------------------------
          EFEITO SOUND
      ------------------------
2  LET Y=3: LET A$=INKEY$
3 IF A$="1" THEN  LET Y=20
4 IF A$="2" THEN  LET Y=30
5 IF A$="3" THEN  LET Y=40
6 IF A$="4" THEN  LET Y=50
7 IF A$="5" THEN  LET Y=60
8 LET K=INT (RND*Y): SOUND .0
  09,K: GOTO 2
9 REM ------------------------
      ANDRE LUIZ P.SILVA - PE
-----------------------------
```

ZX-81

```
1 REM ------------------
      ROTINA DE INVERSAO
      ------------------
2 REM CRIE UMA LINHA REM COM
      25 CARACTERES
3 REM BLOCO ASSEMBLER
4 REM ------------------------------
      16514 - 06 16 2A 0C 40 23 7E FE
              76 28 05 C6 80 77 18 F5
              10 F3 C9
      ------------------------------
5 REM PROGRAMA EXEMPLO
10 FOR A=1 TO 21
20 PRINT AT A,0;"* EXEMPLO DA ROTINA *"
30 NEXTA
40 RAND USR 16514
50 FOR B=1 TO 10
60 NEXT B
70 GOTO 50
80 REM ---------------------------
      CLAUDIO LOMBARDO JORGE - RJ
      ---------------------------
```

MSX

```
1 REM +-------------------------------+
2 REM |        ABERTURA ESPECIAL      |
3 REM +-------------------------------+
10 CLEAR100:DIME$(10):KL=0:CLS
20 E$(1)="PROGRAMADO POR"
30 E$(2)="Micro Sistemas"
40 DF=8:KL=KL+1:IF KL=3 THEN KL=1
50 FOR L=1 TO LEN(E$(KL))
60 A$=MID$(E$(KL),L,1):DF=DF+1:A=1:B=1
70 LOCATE A,B:PRINT A$
80 LOCATE A-1.B-1:PRINT SPC(1)
90 A=A+1:B=B+1:IF A=DF THEN GOTO 110
100 GOTO 70
110 B=B-1
120 LOCATE A-1,B:PRINT A$
130 LOCATE A-1,B+1:PRINT SPC(1)
140 B=B-1:IF B=6 THEN GOTO 160
150 GOTO 120
160 NEXT L
170 FOR H=34 TO 0 STEP -1
180 LOCATE H,7:PRINT CHR$(208);SPC(1)
190 NEXT H:LOCATE 0,7:PRINT"   ":GOTO 40
200 REM +-------------------------------+
210 REM |  SIDNEY AGRA BARROS - AL      |
220 REM +-------------------------------+
```

APPLE

```
5 REM ------------------------------
6 REM      PREENCHENDO A TELA
7 REM ------------------------------
10 HGR
15 VTAB 22
20 INPUT A
25 REM --> A=COR. ( 0 < A < 255 )
30 POKE 28,A : CALL 62454
40 GOTO 20
50 REM ------------------------------
55 REM BRUNO ACCIOLY N. SILVA - RJ
60 REM ------------------------------
```

TRS-80

```
10 REM ------------------------
       ESTE PATCH ELIMINA OS 63
       PONTINHOS QUE SURGEM ABAIXO
       DA MENSAGEM "DOS500 Ativo"
       ------------------------
20 REM ------------------------------
       PATCH #1 (ADD=4E78,FIND=2E,CHG=5F)
       ------------------------------
30 REM ------------------------------
       PODE-SE SUBSTITUIR 5F POR QUALQUER
       VALOR HEXADECIMAL DO CARACTER QUE
       PREFERIR
       ------------------------------
40 REM ------------------------
       EDISON ASSUMPCAO TACAO - PR
       ------------------------
```

TRS COLOR

```
1 '    =-=-=-=-=- -=-=-=-=-=
2 '    :     ABRE-TELA     :
3 '    =-=-=-=-=- -=-=-=-=-=
4 '
10 ZZ$=STRING$(50,32):ZZ=256*PEE
K(VARPTR(ZZ$)+2)+PEEK(VARPTR(ZZ$
)+3):DEFUSR1=ZZ
10 ZZ$=STRING$(50,32):ZZ=256*PEE
K(VARPTR(ZZ$)+2)+PEEK(VARPTR(ZZ$
)+3):DEFUSR1=ZZ
20 FORJ=0TO48:READA$:POKEZZ+J,VA
L("&H"+A$):NEXT
30 CLS:FORJ=2TO16:PRINTSTRING$(J
,32);"MICRO SISTEMAS":NEXT
40 EXEC44539
50 FORJ=1TO16:A=USR1(143):NEXT
60 GOTO60
70 DATA BD,B3,ED,D7,50,8E,03,FF,
30,01,C6,0F,A6,01,A7,80,5A,26,F9
,96,50,A7,84,C6,10,30,88,10,A6,8
2,A7,01,5A,26,F9,30,88,11,8C,05,
FF,23,DF,8E,04,FF,7E,A7,D3
71 '-----------------------------

     Experimente outros valores
       como argumento de USR
     ---------------------------
72 '=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
73 ': LUIZ ALBERTO LOPES - RJ :
74 '=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

Envie suas dicas para a Redação de MICRO SISTEMAS na Av, Presidente Wilson, 165 - Grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

clube do leitor

dicas dicas dic

TRS-COLOR

```
1 '  ////////////\\\\\\\\\\\\
2 '  //                    \\
3 '  //  MOLDURA GIRATORIA  \\
4 '  //                    \\
5 '  \\\\\\\\\\\\////////////
6 '
10 CLEAR200,15000:CLS
20 FORX=15000TO15103
30 READOP$:N=VAL("&H"+OP$)
40 POKEX,N:NEXT
50 PRINT@170,"DICAS";:PRINT@238,
"DE";:PRINT@295,"RODRIGO TOMAZON
I";
60 EXEC15000:FORX=1TO75:NEXT:J=J
+1:IFJ>50THENGOTO120ELSEGOTO60
70 DATA 31,8D,00,5F,1F,21,E6,8D,
00,5D,3A,1F,12,8E,04,00,A6,A0,A7
,80,8D,44,8C,04,20,26,F5,8E,04,3
F,A6,A0
80 DATA A7,00,8D,36,30,88,20,8C,
05,FF,26,F2,8E,06,00,A6,A0,A7,82
,8D,25,8C,05,E0,26,F5,8E,05,C0,A
6,A0,A7
90 DATA 00,30,88,E0,8D,14,8C,04,
00,26,F2,A6,8D,00,18,4C,81,04,26
,01,4F,A7,8D,00,0E,39,81,BF,26,0
4,31,8D,00,01,39
100 DATA 9F,AF,CF,BF,00
110 '
120 '--INICIO DO SEU PROGRAMA--
130 '
140 CLS:END
150 '
160 '\\\\\\\\\\\\\\\\\\\\\\\\\\
170 '   RODRIGO TOMAZONI - PB
180 '  //////////////////////////
```

ZX81

```
10 REM ----------------
       INVERSAO DE TELA
       ----------------
20 REM CRIE UMA LINHA REM
       COM 19 CARACTERES
30 REM BLOCO ASSEMBLER
40 REM ------------------------------
       16514 - 2A 0C 40 06 17 23 7E FE
       16522 - 76 28 05 C6 80 77 18 F5
       16530 - 10 F3 C9
       ------------------------------
50 REM RODE-A COM RAND USR 16514
60 REM ------------------------------
       LUCIANO VANDERLEY T DA SILVA - AL
       ------------------------------
```

APPLE

```
 5 REM ------------------------------
 6 REM        BOLA DE CRISTAL
 7 REM ------------------------------
10 HGR2
20 C = 89 : D = 99
30 FOR I = 1 TO 300
40 X = 90 + 90 * COS (I / 48)
50 Y = 90 + 90 * SIN (I / 48)
60 HCOLOR = 3
70 HPLOT X,Y TO X,D TO C,Y
80 FOR K = 1 TO 2 : P = PEEK ( - 16
   336 ) - PEEK ( - 16336 ): NEXT K
90 NEXT I :GET A$ :TEXT :HOME :END
95 REM ------------------------------
96 REM   JULIANO D. ANTONIO - PR
97 REM ------------------------------
```

ZX SPECTRUM

```
10 REM ------------------------
20 REM        POKES UTEIS
30 REM Para evitar que a tela
       seja afetada devido a
       operacao LOAD, use:
       POKE 23570,16
       Para voltar ao normal:
       POKE 23570,6
40 REM ------------------------
50 REM
60 REM ------------------------
70 REM IVAN F. GARCIA - MT
80 REM ------------------------
```

MSX

```
1 REM ********************************
2 REM *          CIRCULOS            *
3 REM ********************************
5 SCREEN 3
10 FOR A=150 TO 1 STEP -2
15 CIRCLE(125,90),A,RND(1)*16
16 N=N+1:IF N=16 THEN N=2
17 COLOR,,N
20 NEXT
25 FOR A=1 TO 150 STEP 2
30 CIRCLE(125,90),A,RND(1)*16
32 N=N+1:IF N=16 THEN N=2
33 COLOR,,N
35 NEXT
40 GOTO 10
50 REM ********************************
60 REM *   ROBERTO TIETZMANN - RS     *
70 REM ********************************
```

dicasdicasdicas

MSX

```
1 REM ********************************
2 REM *         EFEITO VISUAL        *
3 REM ********************************
10 SCREEN2:COLOR 15,1,1
20 FOR C=2 TO 15
30 FOR Y=-1 TO 8:FOR X=-1 TO 10
40 A=X*30:B=Y*30
50 LINE(11+A,11+B)-(29+A,19+B),C,BF
60 LINE(26+A,26+B)-(44+A,34+B),C,BF
70 LINE(16+B,21+A)-(24+B,39+A),C,BF
80 LINE(31+B,6+A)-(39+B,24+A),C,BF
90 NEXT:NEXT:NEXT:GOTO 20
100 REM ******************************
110 REM * NELSON HISASHI TAMURA-SP  *
120 REM ******************************
```

APPLE

```
  5 REM ------------------------------
  6 REM      ABERTURA COM EFEITO
  7 REM ------------------------------
 10 HGR
 20 C = 7
 30 FOR S = 0 TO 278 STEP C
 40 TAB 22 : PRINT "** BOM DIA, VAM
    OS TRABALHAR! **"
 50 HCOLOR = C
 60 HPLOT S,0 TO 139,159
 70 NEXT S
 80 C = C - 1:IF C > 1 GOTO 30
 90 REM ------------------------------
100 REM   BENEDITO R. FILHO - MG
110 REM ------------------------------
```

TRS-80

```
5 REM-------------------------------
7 REM      ABERTURA DE JOGOS
9 REM-------------------------------
10 CLS
15 FOR X=1 TO 200: NEXT X
20 PRINT CHR$(23)
30 PRINT @400,"GUERRA ESPACIAL"
31 A=A+1: IF A=10 THEN 60
33 FOR X=1 TO 200: NEXT X
40 PRINT CHR$(28)
50 GOTO 15
60 PRINT:PRINT "INICIO"
70 REM VOCE PODE TROCAR O TEMPO DE
72 REM FREQUENCIA ALTERANDO O VALOR
74 REM DA LINHA 31.
76 REM-------------------------------
78 REM NILTON MANOEL DA SILVA - SC
80 REM-------------------------------
```

ZX81

```
10 REM ------------------------
       APRESENTACAO DE PROGRAMAS
       ------------------------

9040 DIM M$(208)
9042 LET M$="-------------------------------
--EU,TK-85 APRESENTO A VOCES UM PROGRAMA M
UITO BOM E' O (NOME DO PROGRAMA) TECLE ENT
ER-------------------------------"
9044 FOR F=1 TO 176
9046 PRINT AT 21,0;M$(F TO F+31)
9048 IF CODE INKEY$=118 THEN GOTO 9060
9050 NEXT F
9052 GOTO 9044
9060 DIM M$(32)
9062 PRINT AT 21,0;M$
9100 REM ------------------------
         FERNANDO T M PEIXOTO - MG
         ------------------------
```

TRS 80

```
4 REM +-----------------------------------------------------------+
5 REM :                    CALEIDOSCOPIO                          :
6 REM +-----------------------------------------------------------+
10 CLS:CLEAR100:RANDOM
12 PRINTCHR$(23);@64*4+10," ** CALEIDOSCOPIO **";@64*8," <1> CAR
AC. GRAFICOS":PRINT" <2> CARAC. ESPECIAIS":PRINT" <3> FIM";@64*1
4+10," PRC-87"
13 A$=INKEY$:IFA$<>"1"ANDA$<>"2"ANDA$<>"3",13ELSEA=VAL(A$)
14 IFA=1,CA=128:FI=191ELSEIFA=2,CA=191:FI=255ELSECLS:END
15 CLS:CA=CA+1:IFCA>FI,RUN
17 IFA=2,GA$=CHR$(21)+CHR$(CA)+CHR$(21)ELSEGA$=CHR$(CA)
18 P=15:CO=0:PRINT@64*14+50,"CARAC. ";CA;
20 P1=RND(P):P2=RND(P)
30 PRINT@64*P1+(P2+20),GA$;@64*P2+(P1+20),GA$;@64*P1+((P-P2)+20)
,GA$;@64*(P-P2)+(P1+20),GA$;@64*P2+((P-P1)+20),GA$;@64*(P-P1)+(P
2+20),GA$;@64*(P-P1)+((P-P2)+20),GA$;@64*(P-P2)+((P-P1)+20),GA$;
40 CO=CO+1:IFCO=10THEN50ELSE20
50 PRINT@64*6,"PARA SAIR <CLEAR>";@64*8,"CONTINUAR <ENTER>";
60 IFPEEK(14400)=1,15ELSEIFPEEK(14400)=2,RUNELSE60
70 REM +-----------------------------------------------------------+
80 REM :              PAULO R. CARVALHO - RJ                       :
90 REM +-----------------------------------------------------------+
```

ZX SPECTRUM

```
  1 REM +----------------------+
  2 REM :   SENOS X COSSENOS   :
  3 REM +----------------------+
 10 BORDER 0: PAPER 0: INK 7: B
RIGHT 1: CLS
 20 FOR N=0 TO 2*PI STEP .1
 30 LET X1=128+60*COS N: LET X2
=128+60*COS (N+PI/1.5)
 40 LET Y1=96+60*SIN N: LET Y2=
96+60*SIN (N+PI/1.5)
 50 PLOT X1,Y1: DRAW X2-PEEK 23
677,Y2-PEEK 23678
 60 NEXT N
 70 REM +----------------------+
 80 REM :  NIVARDO AVELINO-PA  :
 90 REM +----------------------+
```

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - Grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

clube do leitor

dicas

TRS-80

```
10 REM ------------------
       IMPRESSAO DE TELA
       ------------------
20 REM FACA UMA COPIA INSTANTANEA DA
       TELA NA IMPRESSORA COM ESTA
       ROTINA PARA PROGRAMAS EM BASIC
30 REM --> SE FOR BASIC-DISCO:
40 REM ---------------
       10 DEFUSR=473
       ---------------
50 REM --> SE FOR BASIC RESIDENTE:
60 REM -------------------------------
       10 POKE 16526,217:POKE 16527,1
       -------------------------------
70 REM PARA IMPRIMIR ACRESCENTE:
80 REM -- nnn X=USR(0)
90 REM ONDE nnn E' QUALQUER NUMERO
       VALIDO DE LINHA
100 REM ------------------------------
        FRANCISCO AURELIO A FILHO - CE
        ------------------------------
```

MSX

```
5 REM ------------------------------------
            ARQUIVE O VIDEO
      ------------------------------------
10 DATA3E,EC,21,0,0,11,1D,EC,1,0,4,F5,3A
,AF,FC,B9,28,2,26,18,F1,BA,CA,59,0,EB,C3
,5C,0
20 FORY=0TO28:READX$:POKE$HEC00+Y,VAL("&
H"+X$:NEXTY
30 X=&HEC00:DEFUSR=X:DEFUSR1=X+2
40 FORX=0TO479:PRINT".*";:NEXTX:X=USR(0)
50 CLS:FORX=0TO50:NEXTX=X=USR1(0):GOTO50
60 REM DEFUSR=&HEC00:X=USR(0)    ARQUIVA
70 REM DEFUSR1=&HEC02:X=USR1(0)RECUPERA
80 REM &HEC00,&HEFE0,&HEC02   ENDERECOS
90 REM        PARA BSAVE DO ARQUIVO TELA
100 REM ----------------------------------
          IGOR K.DE OLIVEIRA   -   RJ
----------------------------------------
```

ZX SPECTRUM

```
    10 REM ***********************
    20 REM Este programa serve
    30 REM para realizar MERGE de
    40 REM programas protegidos.
    50 REM ***********************
    60 FOR N=23296 TO 23314: READ
  A: POKE N,A: NEXT N: RAND USR 23
  296
    70 DATA 62,1,33,16,91,50,116,9
  2,34,93,92,205,14,6,207,8,34,34,
  13
    80 REM ***********************
    90 REM JAIRSON V.S. FILHO - PE
   100 REM ***********************
```

APPLE

```
  5 REM ------------------------------
  6 REM       ILUSAO DE OTICA
  7 REM ------------------------------
  8 HCOLOR=3
 10 HOME : HGR
 15 C = 0
 20 M = 79:O = M:K = 0:N = 159:C =
    C + 1:VTAB 23: PRINT "AS RETAS
    PARALELAS SE ESTUFAM DE FORA PA
    RA DENTRO"
 30 FOR L = 0 TO 279 STEP 20
 40 HPLOT L,O TO 140,K
 50 HPLOT L,M TO 140,N
 60 NEXT L
 70 FOR X = 60 TO 64
 80 HPLOT 0,X TO 279,X
 90 HPLOT 0,X + 35 TO 279,X + 35
100 NEXT X
110 GET Q$
120 O = K:M = N:K = 79:N = K:C =
    C + 1: IF C = 2 THEN C = 0:VTAB
    23: PRINT "AS RETAS PARALELAS S
    E ESTUFAM DE DENTRO PARA FORA,N
    O CENTRO": GOTO 30
130 GOTO 15
140 REM ------------------------------
150 REM   CHRISTINE C. DANTAS - RJ
160 REM ------------------------------
```

ZX81

```
9000 REM -------------------------
          APRESENTACAO DE PROGRAMAS
          -------------------------
9040 DIM M$(208)
9042 LET M$="----------------------------------
---EU,TK-85 APRESENTO A VOCES UM PROGRAMA
MUITO BOM E' O (AQUI ENTRA O NOME DO PROGR
AMA)TECLE ENTER ------------------------------
-----"
9044 FOR F=1 TO 176
9046 PRINT AT 21,0;M$(F TO F+31)
9048 IF CODE INKEY$=118 THEN GOTO 9060
9050 NEXT F
9052 GOTO 9044
9060 DIM M$(32)
9062 PRINT AT 21,0;M$
9100 REM -------------------------
          FRANCISCO T M PEIXOTO - MG
          -------------------------
```

Envie suas dicas para a Redação de MICRO SISTEMAS na Av, Presidente Wilson, 165-Grupo 1210. Centro, Rio de Janeiro, RJ. CEP 20030

clube do leitor

dicasdicasdicasdic

MSX

```
1Ø ' ----------------------------------
       GRAVACAO EM 18ØØ BAUDS
   ----------------------------------

2Ø '
     Grave seus programas numa
  velocidade intermediaria entre
       24ØØ e 12ØØ bauds:

3Ø '==================================
     POKE 62478,55:POKE 62471,61:
     POKE 62472,25:POKE 62473,3Ø:
          POKE 62474,23
   ==================================

4Ø ' ----------------------------------
     LUIZ ALBERTO RODRIGUES - DF
   ----------------------------------
```

APPLE

```
1Ø ' ----------------------------------
     DE BAIXA RESOLUCAO PARA HGR
   ----------------------------------

2Ø ' Esta dica transfere todos
  os graficos de baixa resolucao
  para a  tela de alta resolucao
        grafica numero 1.

3Ø HGR: HCOLOR= 3: FOR Y= Ø TO 3
9: FOR X= Ø TO 39
4Ø IF SCRN (X,Y) THEN X2 = X * 7
:Y2 = Y * 4:HCOLOR= 2 + (SCRN (X
,Y) > 12 AND SCRN (X,Y) < 16): F
OR N = Ø TO 3: HPLOT X2,Y2 + N T
O X2 + 6,Y2 + N: NEXT N
5Ø NEXT X,Y
6Ø ' ----------------------------------
        EDUARDO SAITO - SP
   ----------------------------------
```

ZX SPECTRUM

```
1Ø REM ----------------------------------
                ESPELHO
   ----------------------------------


2Ø CLEAR 64999
3Ø FOR N=65ØØØ TO 65Ø18: READ A:
 POKE N,A: NEXT N
4Ø DATA 33,Ø,64,78,6,8,2Ø3,17,31
,16,251,119,35,124,254,88,32,241
,2Ø1
5Ø PRINT AT 11,15; "ESPELHO"
6Ø PRINT AT 5,3; "Aguarde o espe
lhamento"
7Ø PAUSE 5Ø
8Ø PRINT AT 5,3;
9Ø RAND USR 65ØØØ
1ØØ PRINT AT 5,3; "Pressione qua
lquer tecla para   voltar ao nor
mal"
11Ø PAUSE Ø
12Ø PRINT AT 5,3;
13Ø RAND USR 65ØØØ
14Ø REM ----------------------------------
          ROBSON P. RANGEL - SP
   ----------------------------------
```

MSX

```
1Ø ' ----------------------------------
          CAVAQUINHO II
   ----------------------------------

2Ø PLAY"O5L14S1ØM2ØØØ","O6L14SM4
5ØØ"
3Ø PLAY"BBBBDGBDGBC#DGA#DGA#","B
BBBDGBDDGA#DGA#DGA#"
4Ø PLAY"DGA#DGBDGBDGBC#C#C#BBBBA
#AG#G..","DGA#DGBDGBDGBC#C#C#BBB
BA#AG#G..."
5Ø ' ----------------------------------
          RENATO LABADIA - SP
   ----------------------------------
```

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - Grupo 1210 - Castelo Rio de Janeiro - RJ - CEP 20030

dicas

MSX

```
5 REM ------------------------------------
          TOCANDO CAVAQUINHO
      ------------------------------------
10 PLAY"O5L14S10M2000","O6L14S2M4500"
20 PLAY"BBBBDGBDGBDGBDGA#DGA#DGA#","BBBB
DGBDGBDGBDGA#DGA#DGA#"
30 PLAY"DGA#DGBDGBDGBC#C#C#BBBBA#AG#G.."
,"DGA#DGBDGBDGBC#C#C#BBBBA#AG#G..."
40 REM    A LINHA 10 DEFINE O TIMBRE
100 REM ----------------------------------
          ALEXANDRE C. GOMES   -   SP
------------------------------------------
```

ZX SPECTRUM

```
5 REM -------------
      GIRA O TEXTO
      -------------
10 FOR F=30 TO 0 STEP-1
20 POKE 23606,F
30 PRINT AT 10,10;"MICRO SISTEMAS"
40 PAUSE 3
50 NEXT F
60 REM ------------------------------------
      CARLOS EDUARDO SCARTEZINI - SP
      ------------------------------------
```

ZX81

```
5 REM ---------
      PROTETOR
      --------
6 REM ESTE PROGRAMA GERA
      UMA SENHA QUE IMPEDE
      O ACESSO AO PROGRAMA
7 REM CRIE UM REM COM 41
      CARACTERES E DIGITE:
8 REM 16514 CD 91 40 FE 39 20 F9 CD
      16522 91 40 FE 30 20 F2 C9 CD
      16530 BB 02 2C 28 FA 2D 44 4D
      16538 CD BD 07 3A 27 40 BE 28
      16546 EE 7E 32 27 40 C9 80 80
      16554 80
9 REM DIGITE A PARTE EM BASIC:
10 PRINT AT 11,3;"DIGITE A SENHA PARA RODA
R"
15 RAND USR 16514
20 CLS
25 REM O PROGRAMA SO RODARA SE
      TECLARMOS "T" E "K".
30 REM MUDE A SENHA DIGITANDO NOS
      ENDERECOS 16518 E 16525 OS
      CODIGOS DAS LETRAS ESCOLHIDAS.
40 REM ----------------------------
      SERGIO RICARDO TEIXEIRA - SP
      ----------------------------
```

TRS-COLOR

```
1 '-----------------------------
          ALTA RESOLUCAO
        DE  PONTA - CABECA
    -----------------------------
5 '
  Vire a tela de alta resolucao
 (PMODE 4) de cabeca para baixo:

10 CLEAR200,32000:FORJ=0TO31:REA
DA$:POKE32000+J,VAL("&H"+A$):NEX
T
20 PMODE4,1:COLOR5,0:PCLS:SCREEN
1,1
30 CIRCLE(60,48),30:LINE(180,120
)-(230,170),PSET,B
40 EXEC44539:EXEC32000:GOTO40
50 DATA 9E,BA,30,89,0B,E0,1F,13,
33,C8,20,86,20,97,50,A6,C4,E6,84
,A7,80,E7,C0,0A,50,26,F4,30,88,C
0,11,93,B7,25,E8,39
60 '-----------------------------
      LUIZ CARLOS MIRANDA - RJ
     -----------------------------
```

MSX

```
0 '       =---------------------=
1 '       ¦                     ¦
2 '       ¦    CLS  PERSIANA    ¦
3 '       ¦                     ¦
4 '       =---------------------=
5 '
6 '   Esta dica rotaciona a tabela
        de caracteres no sentido
        vertical e produz um CLS.
7 '
8 '  Note que caracteres redefinidos
       serao alterados. A rotina e
       relocavel e pode ser usada em
              SCREEN 0 e 1.
9 '
10 CLEAR200,&HF350:FORA=&HF350TO&HF377:
READA$:POKEA,VAL("&H"+A$):NEXT
20 DEFUSR=&HF350
30 A=USR(0):END
40 '
50 DATA AF,32,17,FC,0E,08,06,F7,2A,24
60 DATA F9,C5,CD,A5,0B,1E,17,0E,08,43
70 DATA 10,FE,CD,45,07,0E,08,09,C1,10
80 DATA EC,0D,20,E4,CD,C3,00,C3,1E,07
90 '
100 ' =-----------------------------=
110 ' ¦      CLAUDIO COSTA - RJ      ¦
120 ' =-----------------------------=
```

ZX SPECTRUM

```
 1 REM ------------------------
        EFEITOS VISUAIS - V
      ------------------------
 5 OVER 1: INPUT S
10 FOR A=0 TO 255 STEP S
14 PLOT 255,175: DRAW -A,-175
16 PLOT 0,175: DRAW A,-175
18 PLOT 255,0: DRAW -A,175
20 PLOT 0,0: DRAW A,175
30 NEXT A
40 REM ------------------------
        IVAN MELO FILHO - AL
-------------------------------
```

TRS COLOR

```
5 ' ------------------------------
        RELOGIO DIGITAL
      ------------------------------

10 CLEAR200,32599
20 FORJ=32600 TO 32679:READN:POK
E J,N:NEXT
30 DATA 204,0,0,253,127,252,253,
127,254,48,140,4,191,1,13,57
40 DATA 206,127,164,142,128,0,16
6,130,76,161,192,38,9,111,132
50 DATA 140,127,252,38,242,134,1
,167,132,206,4,32,143,127,255
60 DATA 79,230,130,192,10,45,3,7
6,32,249,195,47,58,237,195,17
70 DATA 131,4,25,47,6,134,58,167
,194,32,229,126,137,76,50,60,60,
13
80 ' Para colocar o relogio na
      tela use EXEC 32600

90 '   Para acertar o relogio
            digite:
      POKE 32764, horas
      POKE 32765, minutos
      POKE 32766, segundos

100 ' ------------------------------
        NILTON GABY - SP
      ------------------------------
```

APPLE

```
10 '------------------------------------
        EXECUCAO AUTOMATICA
      ------------------------------------

20 ' Esta dica simula o comando "-" do
      DOS do TK 3000 IIe no Apple II.
     Com este comando pode-se executar
    qualquer arquivo no disquete, seja
          RUN, BRUN ou EXEC.
30 '  Entre no monitor com CALL -151
              e digite:

=========================================

 B6B3: A0 00 8C 5D AA 20 A4 A1 CD B2 AA
       F0 F8 C9 AD D0 02 A0 1B 8C 62 AA
       88 60 AD 62 AA F0 14 20 A8 A2 AD
       C2 B5 29 7F D0 03 4C C9 A5 C9 02
       D0 03 4C D1 A4 4C 8E A3
 9D52: CA B6
 0300: 20 B3 B6 8C 5F AA C8 D0 45
       9FCD<300.308M

=========================================

40 ' ------------------------------------
        MARCELO MANTA - RJ
      ------------------------------------
```

MSX

```
10 ' ------------------------------------
        A MANSAO MAL-ASSOMBRADA
      ------------------------------------

20 PLAY"T90","T90","T90"
30 PLAY"V15L1604EE-C0AA-A04CE-EE-C03AA-
A04CE-","V1503E2E-4F4","V1502A2A-4F4"
40 PLAY"V15L1604AA-ECBC03B04AA-EC05C04B
DAE","V1503E2E-4F4","V1502A2A-4F4"
50 PLAY"V15L1604EE-C03AA-A04CE-EE-C03AA
-A04CE-","V1503E2E-4F4","V1502A2A-4F4"
60 PLAY"V15L1604AA-ECBC03B04AA-EC05C04B
DAE","V1503E2E-4F4","V1502A2A-4F4"
70 PLAY"V15T3504A4","V15T3504C4","V15T3
502A4"
80 ' Experimente alterar o valor de T
              na linha 20

100 ' ------------------------------------
        RENATO LABADIA - SP
      ------------------------------------
```

ZX SPECTRUM

```
 1 REM ------------------------
        EFEITOS VISUAIS - VI
      ------------------------
 5 INPUT F: OVER 1
10 FOR A=0 TO 255 STEP F
20 PLOT A,175: DRAW 255-A-2,-1
75
30 NEXT A
40 FOR A=0 TO 175 STEP F
50 PLOT 0,A: DRAW 255,175-A-2
60 NEXT A
70 REM ------------------------
        IVAN MELO FILHO - AL
      ------------------------------
```

TRS-COLOR

```
10 ' ------------------------------
        BEEP NO TECLADO
      ------------------------------

20 A=PEEK(116)*256+PEEK(117)-632
30 X=INT(A/256):Y=A-(256*X)
40 FORI=A TO A+9:READB:POKEI,B
50 NEXT:POKE363,X:POKE364,Y:POKE
362,126
60 CLEAR200,A
70 DATA 52,116,198,1,189,169,81,
53,118,57

100 ' ------------------------------
        NILTON GABY - SP
      ------------------------------
```

clube do leitor

dicas

MSX

```
10 ' ================================
        ALFABETOS EM BASIC
   ================================
20 '
30 ' Esta dica permite usar os alfabetos
     do GRAPHOS III num programa BASIC
        nos screens 0, 1, 2 e 3:
40 '
50 CLEAR200,&HD600:CLS:INPUT"NOME DO ALF
ABETO ";A$
60 BLOADA$,&H4400
70 POKE&HF91F,INP(&HA8)/&H40:POKE&HF920,
0:POKE&HF921,&HD6
80 SCREEN1
90 '
100 '-----------------------------------
        ROBERTO LOPES GUIMARAES - RJ
   -----------------------------------
```

APPLE

```
10 ' ------------------------------------
        DIRECIONE A IMPRESSAO
   ------------------------------------

20 '   Para exibir a pagina 1 de alta
       resolucao enquanto se executa
       um HPLOT na pagina 2:
            POKE 230,64

30 '   Para exibir a pagina 2 enquanto
       um HPLOT e executado na pagina 1:
            POKE 230,32

100 '-----------------------------------
        CLAUDIO LUIS M. SAMPAIO - MG
   -----------------------------------
```

TRS-80

```
50 REM ----------------
       RELOGIO DIGITAL
       ----------------
100 CLEAR500:A$=CHR$(26)+STRING$(2,24):FOR
X=0TO10:FORY=1TO2:READZ:N$(X)=N$(X)+CHR$(Z
):NEXTY:N$(X)=N$(X)+A$:FORY=1TO2:READZ:N$(
X)=N$(X)+CHR$(Z):NEXTY,X:CLS:INPUT"HR,MN,S
S ";H,M,S:POKE16919,S:POKE16920,M:POKE1692
1,H:CLS
110 FORX=1TO8:PRINT@471+X*2,N$(ASC(MID$(TI
ME$,9+X,1))-48);:NEXTX:IFPEEK(14400)AND13R
UNELSE110
120 DATA151,149,141,133,170,32,138,32,179,
149,141,132,179,149,140,133,181,149,32,133
,183,145,140,133,183,145,141,133,131,149,3
2,133,183,149,141,133,183,149,140,133,136,
32,130,32
130 REM PARA ACERTAR O RELOGIO, ESTANDO
ELE EM FUNCIONAMENTO, TECLE <ENTER>
140 REM SE QUISER UM CRONOMETRO DIGITAL
PRESSIONE <ENTER> NO LUGAR DA RESPOSTA
PARA HR,MN,SS
150 REM ----------------------
        EDISON A TACAO - PR
        ----------------------
```

ZX81

```
10 REM ------------------------
        CONTAGEM REGRESSIVA
   ------------------------
20 REM CRIE UMA LINHA REM COM 97
     CARACTERES
30 REM ------------------------
16514 - 25 21 82 40 7E FE 1B CA
        E0 40 21 94 40 77 01 00
        03 3E 1C B7 CB 27 10 FC
        4F 30 02 06 01 21 00 1E
        09 E5 D1 01 14 01 2A 0C
        40 23 09 0E 08 06 00 1A
        CD 7F 26 04 36 00 10 02
        36 80 23 CB 07 10 F1 C5
        01 19 00 09 C1 0D 28 03
        15 18 E2 3E 00 21 FF 25
        28 BC C2 D2 40 21 82 40
        7E 3D 77 C3 83 40 36 25
        C9
40 REM ------------------------
50 REM RODE COM RAND USR 16515
60 REM
70 REM        ENDERECOS:
     16593 - TEMPO DA CONTAGEM
     16567 - CARACTER DO NUMERO
     16571 - CARACTER DO FUNDO
80 REM ------------------------
        CLAUDIO L. JORGE - RJ
   ------------------------
```

ZX SPECTRUM

```
5 REM ---------------
      INVERTE A TELA
      ---------------
6 REM ESTA ROTINA INVERTE
      A TELA COMPLEMENTANDO
      SEUS BYTES SEM MEXER
      NOS ATRIBUTOS.
10 DATA 243,33,255,23,1,0,64,1
0,47,2,3,43,62,255,188,32,246,
251,201
20 CLEAR 49999
30 FOR N=50000 TO 50018
40 READ A:POKE N,A:NEXT N
50 REM ---------------------
       DIRCEU M F LIMA - MG
       ---------------------
```

MSX

```
10 '-------------------------------------
      CARACTERES BOLD EM SCREEN 1
   -------------------------------------

20 SCREEN1:E=&HC000
30 FORI=E TO E+24:READA$:POKEI,VAL("&H"
+A$):NEXT
40 DEFUSR=E:A=USR(0)

50 DATA 21,00,00,CD,4A,00,47,CB
60 DATA 0F,B0,CD,4D,00,23,3E,07
70 DATA BC,20,F0,3E,FF,BD,20,EB,C9

100 '-------------------------------------
        FERNAO M. MAUGER -SP
   -------------------------------------
```

dicasdicasdicasdicas

TRS COLOR

```
1 '==============================
      COMPRESSOR DE PROGRAMAS
              BASIC
  ==============================

2 ' Este programa elimina linhas
 REM e espacos que nao estiverem
  contidos em declaracoes PRINT.
   Grave a rotina e  rode-a com
    CLEAR200,30000:CLOADM:EXEC

10 CLEAR200,30000
20 CLS:FORN=30001TO30191:READA$:
POKEN,VAL("&H"+A$):NEXT
30 PRINT:PRINT"PREPARE O GRAVADO
R E TECLE ENTER PARA GRAVAR":EXE
C44539:PRINT"GRAVANDO"
40 CSAVEM"COMPRES.",30001,30192,
30001
50 DATA DE,19,DC,1B,34,6,20,6,A6
,80,26,1C,EE,C4,6F,8D,0,B7,EC,C4
,26,E,9E,1B,9F,1D,9F,1F,35,6,93,
10,8D,B4,F4,39,30,44,20,E0,81,20
,26,13,6D,8D,0,69,26,D6,CC,0,1
60 DATA 34,56,8D,61,35,56,30,1F,
20,C9,81,22,26,0C,E6,8D,0,52,C8,
1,E7,8D,0,4C,20,B9,81,86,26,12,A
6,1E,81,FF,27,AF,E6,8D,0,3C,C8,2
,E7,8D,0,36,20,A3,81,82,27,4
70 DATA 81,83,26,98,A6,1E,81,FF,
27,95,EC,C4,30,1F,34,50,A3,E1,1F
,12,AE,C4,30,1F,34,16,1F,20,A3,6
4,C1,4,26,7,35,6,C3,0,4,34,6,8D,
5,35,56,16,FF,75,0,EC,C4,27
80 DATA A,AE,C4,A3,62,ED,C4,1F,1
3,20,F2,EC,64,A3,62,1F,3,AE,64,A
6,80,A7,C0,9C,1B,25,F8,DC,1B,A3,
62,DD,1B,39
90 '==============================
      FABIO MASSAMI YAMANOTO - SP
  ==============================
```

ZX81

```
10 REM ------------------------------
      DECOMPOSICAO EM FATORES
             PRIMOS
20 REM ------------------------------
30 PRINT "QUAL O NUMERO A SER DE
COMPOSTO?"
40 INPUT A
50 CLS
60 PRINT "NUMERO="; A
70 FOR I=2 TO A
80 LET X=A/I-INT(A/I)
90 IF X=0 THEN LET A=A/I
100 IF X=0 THEN PRINT "       ";I
,A
110 IF X=0 THEN GOTO 60
120 NEXT I
130 REM
------------------------------------
   AGNALDO LOPES MARTINS - MG
------------------------------------
```

APPLE

```
10 '-----------------------------------
          CONVERSAO BINARIO-DECIMAL
   -----------------------------------
20 FORJ=768TO816:READA:POKEJ,A:NEXT
30 POKE1013,76:POKE1014,0:POKE1015,3
40 DATA 133,252,169,128,133,251,169,0,13
3,250,165,252,201,66,208,30,32,177,0,240
,8,201,48,208,7,70,251,144,243,76,149,21
7,201,49,208,10,165,251,24,101,250,133,2
50,76,25,3,76,201,222
50 '-----------------------------------
     Esta rotina simula a funcao &B  do
     MSX e Spectrum nos micros da linha
     Apple. Para usa-la, veja o exemplo:
   -----------------------------------
60    &B0001010101:PRINTPEEK(250)
70 '-----------------------------------
      FREDERICO LAMBERTI PISSARRA - ES
   -----------------------------------
```

MSX

```
1 '          ----------   ----------
2 '          :      ABRE - TELA      :
3 '          ----------   ----------
4 '
5 ' Esta dica funciona em SCREEN 1
6 '
10 DATA 06,10,C5,11,01,1B,06
20 DATA 1B,DD,21,1D,00,C5,D5
30 DATA DD,19,06,0F,DD,E5,D5
40 DATA E1,CD,4A,00,2B,CD,4D
50 DATA 00,13,E1,CD,4A,00,23
60 DATA CD,4D,00,DD,2B,10,E8
70 DATA 2B,3E,20,CD,4D,00,EB
80 DATA 2B,3E,20,CD,4D,00,D1
90 DATA C1,21,20,00,19,EB,10
100 DATA C8,C1,10,BF,C9
120 FORX=&HA000TO&HA043:READM$:POKEX,VA
L("&H"+M$):NEXT:DEFUSR=&HA000:A=USR(0)
140 '
150 '-----------------------------------
160 ':  DENILSON SIQUEIRA GOMES - RJ   :
170 '-----------------------------------
```

TRS COLOR

```
1 '-----------------------------------
2 '        LEITURA SEM ERRO
3 '-----------------------------------
4 '
10 CLEAR 999
20 FORJ=0TO20:READA$:POKE970+J,V
AL("&H"+A$):NEXT
30 CLS:EXEC970:CLS
40 DATA 1A,50,BE,80,0,B7,FF,DE,A
6,80,B7,FF,DF,A7,1F,8C,E0,0,25,F
1,39
50 READB$:GOSUB80:IFB$<>"FFFF"TH
EN50
60 POKE&HF3,4:POKE&HF4,&HC0
70 CLS:PRINT"LEITURA SEM ERRO AT
IVADA":NEW:END
80 B=VAL("&H"+B$):IFB>256THEN90E
LSEPOKEE,B:E=E+1:RETURN
90 E=B:RETURN
100 DATA 9D00,1C,AF,B6,FF,21,84,
F7,B7,FF,21,8E,4,C0,9F,F3,39,9E0
0,9F,76,9E,F3,86,80,A7,80,9F,F3,
9E,76,39,9F00,9F,76,9E,F3,86,9F,
A7,80,9F,F3,9E,76,39,A4E3,12,12,
A531,12,12,A740,4,7E,9E,0,A745,7
E,9F,0,A7E9,7E,9D,0,FFFF
110 '
120 '  Esta rotina permite ler
130 'programas do cassete mesmo
140 'que estes contenham erros.
150 '
160 'Uma barra indicara na tela
170 'a extensao do programa e a
180 'posicao aproximada do erro.
181 '
190 '-----------------------------------
200 ' ELSON DE SOUZA VIANA - SP
210 '-----------------------------------
```

ZX SPECTRUM

```
 1 REM ------------------------------
        EFEITOS VISUAIS - II
   ------------------------------
 5 OVER 1: INPUT I
10 FOR A=0 TO 87 STEP I
20 PLOT 0+A-2.5,A: DRAW 255-A-
   5,A
30 NEXT A
40 REM ------------------------------
        IVAN MELO FILHO - AL
------------------------------------
```

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - Grupo 1210. Castelo, Rio de Janeiro, RJ. CEP 20030.

clube do leitor

dicas

TRS 80

```
5 '-------------------------------
      HEADER DE ARQUIVO EM ASSEMBLER
  -------------------------------
10 ON ERROR GOTO 130
20 FILES:PRINT
30 PRINT"Nome: ";:LINE INPUT F$:PRINT
40 OPEN F$ AS #1
50 FIELD #1, 1 AS X$, 2 AS ST$, 2 AS EN
$, 2 AS EP$
60 GET 1,1
70 CLOSE
80 IF ASC(X$)>254 THEN PRINT"Nao e' pro
grama Assembler":END
90 PRINT"Starting Address: ";HEX$(CVI(S
T$));"H (";CVI(ST$);")"
100 PRINT"Ending Address: ";HEX$(CVI(EN
$));"H (";CVI(EN$);")"
110 PRINT"Entry Point: ";HEX$(CVI(EP$))
;"H (";CVI(EP$);")"
120 END
130 PRINT"Erro":IF ERR=255 THEN PRINT"A
rquivo nao existe":CLOSE:KILL F$
140 RESUME 120
200 '-------------------------------
        DAVID P. GUIMARAES - PR
  -------------------------------
```

APPLE

```
10 '-------------------------------
          DESTRAVADOR
   -------------------------------
20 '  Esta rotina permite interromper
   qualquer programa BASIC que estiver
     travado em fita com a instrucao
       POKE 82,128:SAVE:POKE 82,85
30 HOME
40 PRINT"LOAD"
50 FORI=768TO775:READB:POKEI,B:NEXT
60 DATA 32,201,216,169,0,133,214,96
70 CALL768
80 '
90 '-------------------------------
        MARIO CESAR FRANCHI - SP
  -------------------------------
```

MSX

```
1 REM ----------------
      CIRCLES E LINES
      ----------------
10 SCREEN 2
20 FOR F=0 TO 285
30 Z=RND(1)*255:X=RND(1)*192
40 PSET(Z,X),15
50 NEXT
60 FOR D=0 TO 85 STEP 2
70 Z=RND(1)*255:X=RND(1)*192:H=RND(1)*14
+1
80 LINE(128,80)-(Z,X),4
90 CIRCLE (Z,X),8,H
100 PAINT (Z,X),H
110 NEXT
120 OPEN "GRP:"AS#1
130 PSET(48,8)
140 PRINT#1,"EDNEI DELFIM"
150 GOTO 150
160 REM ----------------
       EDNEI S DELFIM - SP
       ----------------
```

ZX SPECTRUM

```
  1 REM +----------------------+
  2 REM : SENOS X COSSENOS 2  :
  3 REM +----------------------+
  5 PAPER 1: BORDER 1: INK 7: C
LS : LET D=0
 10 FOR A=80 TO 0 STEP -5
 20 LET B=80-A
 30 FOR C=0 TO 6.3 STEP .1
 40 IF NOT D THEN  PLOT A*COS C
+126,B*SIN C: LET D=1
 50 LET X=A*COS C+126: LET Y=B*
SIN C+87
 60 DRAW X-PEEK 23677,Y-PEEK 23
678
 70 NEXT C: LET D=0: NEXT A
 80 REM +----------------------+
 90 REM : NIVARDO AVELINO-PA  :
100 REM +----------------------+
```

TRS COLOR

```
10 '-------------------------
       PCOPY EM SCREEN 0
   -------------------------
20 'Esta rotina simula o comando
     PCOPY no modo SCREEN 0.
   EXEC 32515 salva o conteudo
     da tela e EXEC 32532
   repoe a tela de texto.

30 CLEAR200,32000:FORJ=0TO33:REA
DN$:POKE32515+J,VAL("&H"+N$):NEX
T
40 DATA 10,8E,7D,00,8E,04,00,A6,
80,A7,A0,8C,06,00,25,F7,39
50 DATA 10,BE,7D,00,8E,04,00,A6,
A0,A7,80,8C,06,00,25,F7,39

60 '-------------------------
      FRANCISCO S. SOUZA - SC
   -------------------------
```

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - Grupo 1210. Castelo, Rio de Janeiro, RJ. CEP 20030.

MC 1000

```
1 '-------------------------
          RELOGIO
   -------------------------
2 A$="12/06/88  TOMAS PRATA":GOS
UB 100
10 DATA 21,7,3F,22,39,0,C9,F3,F5
,E5,D5,C5,2A,FC,3F,2B,22,FC,3F,7
C,B5,CC,1B,3F,C3,64,C5,21,1,1,22
,FC,3F,6,0,E,1,3E,3B,CD,DE,3F,CB
,49,28,5,C5,CD,BC,3F,C1,3E,5,D3,
20,21,F8,3F,3A,1B,1,FE,C,20,2,CB
,C1,FE,4,20,2,CB,81,FE,4,38,3,3D
,D3,60,3A,F5
15 DATA 0,FE,3,3B,2,CB,86,CB,87,
CB,46,CB,D3,80,21,18,80,CB,7E,20
,26,3E,20,32,63,1,2E,0,AF,32,55,
0,0,1A,11,40,0,1A,A7,28,5,77,13,
23,10,F7,36,20,23,10,FB,36,3A,2E
,1D,36,3A,E,3,41,11,FF,3F,21,17,
80,1A,3C,36,20,28,2,36,5B,EB,2E,
FB,1E,1F,C5,E5,6E,E,A
20 DATA 26,0,CD,2D,C0,1,B0,B0,9,
EB,72,2B,73,EB,1B,E1,C1,2B,1B,10
,E6,C9,21,FF,3F,11,FA,3F,1A,BE,C
0,2B,1B,1A,BE,C0,6,D,C5,CD,60,C0
,F3,1,FF,0,CD,48,C0,C1,10,F2,2E,
FF,75,C9,21,FB,3F,34,BE,D0,70,C,
2B,34,BE,D0,70,C,CD,60,C0,F3,2B,
34,3E,17,BE,D0
25 DATA 70,C9,1,0,0,0,2,0,0,FF
26 DATA HORA=,16377,MINUTO=,1637
8,HORA/DEC=,14280,MINUTO/DEC=,14
383
30 HOME:PRINT:PRINT:POKE850,1:PR
INTSPC(13)"AGUARDE":POKE850,0
40 FORB=16128 TO 16383:READX$:A=
0:B=0:IFLEN(X$)=1THEN50
45 A=ASC(LEFT$(X$,1)):A=A-48:IFA
>9 THEN A=A-7
50 B=ASC(RIGHT$(X$,1)):B=B-48:IF
B>9 THEN B=B-7
55 N=A*16+B:POKEG,N:T=T+N:NEXT
60 IFT<>25453 THEN PRINT:PRINT"-
----ERRO NA DIGITACAO------":END
70 HOME:PRINT:PRINTSPC(8)"RELOGI
O":PRINTSPC(14)"POR TOMAS PRATA"
:CALL 16128:PRINT
80 FORG=1TO4:READA$,A:POKE345,0:
POKE346,129:POKE347,12:POKE348,1
29
85 PRINTA$;:INPUTH:POKEA,H:OUT12
8,PEEK(32772 )+80:NEXT
90 PRINT:INPUT" ENTRE NOME(ATE 1
7 CARACTERES)";A$:GOSUB100
92 HOME:PRINTTAB(45)"INSTRUCOES"
:PRINT"CTRL/L HABILITA SAIDA P/
TELA"
93 PRINT:PRINT"CTRL/D DESABILITA
":PRINT
94 PRINT"CTRL/(A,B,C,D,E) ALTERA
M O BIP DO TECLADO"
95 PRINT:PRINT"SINAL (I)=DESPERT
ADOR LIGADO"
97 PRINT:PRINT"APOS UM RESET DE
CALL 16128 P/ REINICIAR"
98 END
100 FORG=1TOLEN(A$):X=ASC(MID$(A
$,G,1)):POKEG+63,X+128:NEXT
102 POKEG+63,0:POKE355,0:RETURN
200 '-------------------------
     GRAVE O PROGRAMA E PARA
     APOS LO DIGITE  RUN E NOV
        NO MODO DIRETO.
   -------------------------
300 '=========================
        TOMAS PRATA - MG
   =========================
```

dicasdicasdicas

MSX

```
Ø '-----------------------------------
          HOT-ASM EM DISCO
-----------------------------------
   Para gravar um arquivo texto do
   programa HOT-ASM em disco, siga
   este procedimento:

1 ' Gere o texto usando o HOT-ASM;
2 ' Peca a funcao MAP;
3 ' Chame o BASIC;
4 ' Grave o texto em disco usando
   BSAVE"A:NOME.HAS",INIC,FIM   onde:
         INIC= &H8100
         FIM = end. final
         HAS = ext. do HOT-ASM;
5 ' Salve no disco os dados da funcao
   MAP com o nome do arquivo.

6 ' Para ler, siga este procedimento:

7 ' Leia o texto com BLOAD"A:NOME.HAS"
8 ' Leia os dados da funcao MAP no
   arquivo gerado como acima;
9 ' Chame o editor do HOT-ASM com CALL
   EDT;
10 'Chame a funcao 'M EC00' do editor
   do HOT-ASM e modifique os dados dos
   seguintes enderecos:
11 '        &HEC00 - 3D
            &HEC01 - 00
            &HEC02 - 81
            &HEC03 - LSB-1
            &HEC04 - MSB  ; MSB e LSB do
   endereco final dado pela funcao MAP
12 '-----------------------------------
     LUIZ MARCOS MARTINS MARQUES - MG
-----------------------------------
```

ZX SPECTRUM

```
 10 REM ************************
 20 REM *        CORES         *
 30 REM ************************
 40 BORDER Ø: PAPER 1: INK 7: C
LS : OVER 1: GOSUB Ø070: FOR A=2
 TO 10
 50 FOR B=1 TO 80 STEP A: CIRCL
E 127,87,B: NEXT B: GOSUB Ø070:
GOSUB Ø090
 60 GOSUB Ø070: NEXT A: GOTO 99
99
 70 FOR X=Ø TO 255 STEP 2: PLOT
 X,Ø: DRAW Ø,175: NEXT X
 80 FOR X=Ø TO 175 STEP 2: PLOT
 Ø,X: DRAW 255,Ø: NEXT X: RETURN

 90 FOR X=Ø TO 255: PLOT  OVER
1;X,Ø: DRAW  OVER 1;Ø,175: NEXT
X
 100 FOR X=255 TO Ø STEP -1: PLO
T  OVER 1;X,Ø: DRAW  OVER 1;Ø,17
5: NEXT X
 110 RETURN
 120 REM ************************
 130 REM * CHRISTOF KISTNER-SP *
 140 REM ************************
```

APPLE

```
5  REM  ------------------------
6  REM          MOSAICO
7  REM  ------------------------
10  HGR2
20  FOR M = 1 TO 20
25 P = 45:Q = 25:A = 22:B = 20:C =
    - 1:D =  - 1
30  FOR L = 1 TO 4: FOR K = 1 TO
    5: FOR I = Ø TO 19
35 X = P - C * I:Y = Q - D * I:S =
    P + C * I:T = Q + D * I:E =
    X - A:F = Y - B:G = S + A:H =
    T + B
40  HCOLOR=  RND (1) * 7 + 1
50  HPLOT X,Y TO E,Y TO E,F TO X,
    F TO X,Y
60  HPLOT S,Y TO G,Y TO G,F TO S,
    F TO S,Y
70  HPLOT X,T TO E,T TO E,H TO X,
    H TO X,T
80  HPLOT S,T TO G,T TO G,H TO S,
    H TO S,T
90  NEXT I:P = P + 45: NEXT K:P =
    45:Q = Q + 40: NEXT L: NEXT
    M
100  REM  ----------------------
110  REM  NIVARDO CAVALCANTE- PA
120  REM  ----------------------
```

ZX SPECTRUM

```
 10 REM ------------------------
 20 REM      FUNCAO SENOIDAL
 30 REM ------------------------
 40 CLS : FOR T=Ø TO 7: INK T
 50 FOR I=Ø TO 18 STEP .1
 60 PLOT 15*I,30*SIN I+35: DRAW
15*I^1.2+20-(15*I),30*SIN I+130
-(30*SIN I+35)
 70 SOUND .01,I*5
 80 NEXT I: NEXT T
 90 REM ------------------------
100 REM LEANDRO R. BARBOSA - RS
110 REM ------------------------
```

MSX

```
1 '-----------------------------------
          INVERSE NO MSX
-----------------------------------
2 ' Esta dica simula o comando INVERSE
   no MSX invertendo parte do conjunto
  de caracteres. A rotina e chamada com
   A=USR(0) e a string a ser invertida
   deve ser passada como argumento da
        funcao USR1.

10 DEFINTA:FORA=0TO58:READA$:POKE&HD000
+A,VAL("&H"+A$):NEXT
20 DATA 2A,24,F9,11,00,01,19,E5,11,20,0
4,19,EB,E1,01,D8,02,CD,4A,00,2F,EB,CD,4
D,00,EB,23,13,0B,78,B1,20,F0,C9,3A,63,F
6,FE,03,C0,1A,B7,C8,47,EB,23,5E,23,56,1
A,C6,84,CD,A2,00,13,10,F7,C9
30 DEFUSR=&HD000:DEFUSR1=&HD022:A=USR(0
)
40 ' Veja um exemplo de como obter o
        efeito de FLASH:

50 CLS:KEYOFF:WIDTH39
60 LOCATE11,10:PRINT" Micro Sistemas ":
:LOCATE3,13:PRINT"a primeira revista br
asileira de":PRINTTAB(11)"microcomputad
ores"
70 FORA=1TO500:NEXT:LOCATE11,10:A$=USR1
(" Micro Sistemas "):FORA=1TO500:NEXT:G
OTO60
80 '-----------------------------------
     ROBERTO LOPES GUIMARAES - RJ
-----------------------------------
```

MSX

```
5 REM -------------------------------
          ELVIS ERA ASSIM:
-------------------------------
10 A$="V15M2000S11T25503CCEEGGAAA#A#AAGG
EE"
20 B$="V15FFAACCDDD#D#DDCC03AA"
30 C$="V15GGBBO4DDEEO3FFAAO4CCDD"
40 PLAYA$+A$:PLAYB$+A$:PLAYC$+A$
50 PLAY"O4":BEEP
60 END
70 REM -------------------------------
        LEONARDO BRANDAO      -     RJ
-------------------------------
```

ZX 81

```
1 REM -------------------------
        APRESENTACAO 1
-------------------------
2 REM
3 REM     ENTRE COM O BLOCO EM
      HEXADECIMAL E CHAME COM
        RAND USR 16514
4 REM
5 REM -------------------------

 16514 - 0E 88 3E 80 06 FF C5 2A
 16522 - 0C 40 11 21 00 06 10 23
 16530 - 71 23 77 10 FA 06 0B 77
 16538 - 19 71 19 10 FA ED 52 23
 16546 - 06 10 2B 71 2B 77 10 FA
 16554 - 06 0B 19 ED 52 77 ED 52
 16562 - 71 10 F8 E5 F5 3E 00 21
 16570 - FF 08 2B BC C2 BC 40 F1
 16578 - E1 00 00 FE 80 CA D9 40
 16586 - C1 0E 88 05 3E 00 B8 CA
 16594 - E8 40 3E 80 C3 88 40 C1
 16602 - 0E 80 05 3E 00 B8 CA E8
 16610 - 40 3E 88 C3 88 40 C9 00

-------------------------------
6 REM -------------------------
      CLAUDIO L. JORGE - RJ
-------------------------
```

clube do leitor

dicas

IBM-PC

```
------------------------------------------------------------
          AUTO-BOOT E DISQUETE PERSONALIZADO
------------------------------------------------------------

Esta dica personaliza o disquete e faz com que um
          programa entre rodando sozinho:

        - Carregue o seu sistema;
        - Apos o A) digite:

      A) COPY CON AUTOEXEC.BAT
         PROMPT (seu nome).$N$G
         CLS
         REM ** (seu nome) **
         DIR/W
         NOME DO PROGRAMA (Ex: DBASE)
         ^Z (CTRL Z - para gravar o programa)

------------------------------------------------------------
          SALUSTIANO LUIZ DE SOUZA - SC
------------------------------------------------------------
```

TRS COLOR

```
10 '-------------------------
        PADROES EM S24
   -------------------------

20 '  Veja alguns padroes que
    podem ser produzidos no modo
       semigrafico S24:

30 DIMA(10)
40 PMODE3,1:SCREEN1,1:POKE&HFF22
,B:POKE&HB3,128:PCLS
50 K=PEEK(&HBA)*256+PEEK(&HBB):F
ORT=0TO10:READA(T):NEXT
60 DATA 128,32,96,143,159,175,19
1,207,223,239,255
70 FORCF=0TO10:FORCI=0TO10:FORX=
0TO1:FORY=0TO11STEP2
80 PO=K+X+Y*32+(CI*3)+(CF*16*32)
90 POKE PO,A(CI):POKE PO+32,A(CF
)
100 NEXT Y,X,CI,CF
110 GOTO110
120 '-------------------------
      DANIEL WEILER - RS
   -------------------------
```

APPLE

```
10 '-----------------------------------
          LISTAGEM POR PAGINAS
   -----------------------------------
20 ' Esta dica congela a listagem de um
     programa no video ate que alguma
        tecla seja pressionada.
    CTRL+S <CR> interrompe a listagem.
30 '
35 FORA=768TO836:READB:POKEA,B:NEXT:CALL
768
40 DATA 169,11,162,3,133,54,134,55,76,23
4,3,72,165,37,201,23,240,4
50 DATA 104,76,240,253,152,72,165,36,72,
173,0,192,16,251,172,16,192
60 DATA 201,131,240,16,201,147,240,23,32
,88,252,104,133,36,104,168
70 DATA 104,76,240,253,169,240,133,54,16
9,253,133,55,32,234,3,76,3,224
80 '
90 '-----------------------------------
          MARCELO MAUTA - RJ
   -----------------------------------
```

ZX-81

```
1 REM ------------------------
          APRESENTACAO 2
      ------------------------
2 REM   ENTRE COM O BLOCO 1 A
    PARTIR DO ENDERECO 17250
     E O BLOCO 2 A PARTIR DE
     17350. CRIE UM PROGRAMA
    BASIC CONTENDO OS DESENHOS
      DE ABERTURA EM LINHAS
     'PRINT' E USE ESTA DICA DA
          SEGUINTE FORMA:
10 PRINT "DESENHO"
   :
   :
XX RAND USR 17350
XX CLS
XX RAND USR 17250

5 REM          BLOCO 1
-------------------------------
 17250 - 21 82 40 ED 5B 0C 40 06
 17258 - 20 3E 15 4F C5 13 23 7E
 17266 - 12 CD BD 43 10 F7 C1 05
 17274 - C5 79 F5 01 21 00 F5 09
 17282 - 7E EB 09 77 EB F1 CD BD
 17290 - 43 3D 20 F2 F1 3D C1 4F
 17298 - C5 1B 2B 7E 12 CD BD 43
 17306 - 10 F7 C1 78 FE 0B C8 79
 17314 - 05 C5 F5 01 21 00 F5 ED
 17322 - 42 7E EB ED 42 77 EB F1
 17330 - CD BD 43 3D 20 F0 F1 3D
 17338 - C1 18 B0 C5 06 FF 10 FE
 17346 - C1 C9
-------------------------------

6 REM          BLOCO 2
-------------------------------
 17350 - 2A 0C 40 11 82 40 01 D7
 17358 - 03 ED B0 C9
-------------------------------

10 REM ------------------------
          ELIAS C. NASCIMENTO - RJ
      ------------------------
```

ZX SPECTRUM

```
  10 REM +-------------------+
  20 REM !      QUADROS      !
  30 REM +-------------------+
  40 OVER 1: LET A=INT (RND*255)
: LET B=INT (RND*(255-A)): LET C
=INT (RND*(255-B)): IF C>175 THE
N  LET C=175
  50 GOTO 0070
  60 PLOT A,A: DRAW B,0: DRAW 0,
C: DRAW -B,0: DRAW 0,-C: LET A=A
+2: LET B=B-4: LET C=C-4
  70 IF A>175 OR A+B>255 OR A+C>
175 OR A+B<0 OR A+C<0 THEN  GOTO
 0040
  80 GOTO 0060
  90 REM +-------------------+
 100 REM !   GIL TOMIZAWA-PR  !
 110 REM +-------------------+
```

Envie suas dicas para a Redação de MICRO SISTEMAS na Av. Presidente Wilson, 165 - Grupo 1210. Castelo, Rio de Janeiro, RJ. CEP 20030.

dicas

APPLE

```
10 '-----------------------------------
        GRAFICO DE LINHA
   -----------------------------------
20 HOME:DIM P(10):VTAB 1:HTAB 10:PRINT"G
RAFICO DE LINHA"
30 FOR X=1 TO 10
40 VTAB 5:HTAB 1:PRINT "ENTRE COM O ";:I
NVERSE:PRINT X;:NORMAL:PRINT "o PARAMETR
O : ";:INPUT"";P(X)
50 IF P(X)(0 OR P(X))190 THEN 40
60 NEXT X
70 HGR2
80 HCOLOR=5:HPLOT 0,0 TO 0,190 TO 255,19
0 TO 255,0 TO 0,0
90 FOR A=1 TO 10:AL=190-P(A):HPLOT 1,AL
TO 255,AL:NEXT A
100 FOR B=1 TO 255 STEP 28:HPLOT B,0 TO
B,190:NEXT B
110 HCOLOR=3
120 COL=1
130 HPLOT 0,190-P(1)
140 FOR X=1 TO 10
150 AL=190-P(X):HPLOT TO COL,AL
160 FOR R=1 TO 200:NEXT R
170 COL=COL+28:NEXT X
180 END
200 '-----------------------------------
        RICARDO BASTOS GARCIA - SP
   -----------------------------------
```

TRS-80

```
5 '-----------------------------------
      DUMP COM CARACTERES GRAFICOS
   -----------------------------------
10 FORJ=57344 TO 57400 :READA:POKEJ,A:NE
XT
20 DATA 221,33,0,60,1,0,4,38,64,62,13,20
5,59,0,197,6,7,62,32,205,59,0,16,249,193
,221,126,0,205,59,0
30 DATA 221,35,37,11,120,177,254,0,202,5
1,224,124,254,0,202,7,224,195,25,224,62,
13,205,59,0,201
40 '
-----------------------------------
```

Esta rotina centraliza a tela e faz um DUMP mantendo os caracteres graficos. Entre no DOS ativo e grave-a com DUMP PRSCN/CMD (START=0E000,END=0E030, TRA=0E000) <ENTER>

```
50 '-----------------------------------
```

Para usa-la no DOS, digite PRCSN <ENTER>
Para usa-la no BASIC Disco, digite CMD"L","PRSCN/CMD":DEFUSR=&HE000 <ENTER> e X=USR(0) quando quiser um dump.

```
100 '----------------------------------
      EDISON ASSUMPCAO TACAO - PR
   -----------------------------------
```

IBM-PC

TURBO POR SOFTWARE

Esta dica permite alterar a velocidade TURBO/NORMAL do PC-XT Turbo por software:

Copie o programa DEBUG.COM para um disco formatado, carregue-o com DEBUG <enter> e em seguida digite as linhas abaixo ate' o <enter>:

```
TURBO.COM  <enter> *  nome do programa
A 100      <enter>    end. inicial
IN AL,61   <enter>    le o valor na porta 61
OR AL,4    <enter> *  seta o bit 2
OUT 61,AL  <enter>    troca o valor na porta 61
INT 20     <enter>    retorna para o DOS
<enter>
RCX        <enter>
8          <enter>
W          <enter>
Q          <enter>
```

Para fazer a versao que ativa a velocidade normal, repita o procedimento acima trocando as linhas marcadas com *, respectivamente, por:

```
NORMAL.COM <enter>     nome do programa
AND AL,FB  <enter>     resseta o bit 2
```

EDUARDO NEVES - RJ

MSX

```
10 '-----------------------------------
        PRAXIS COM 75 COLUNAS
   -----------------------------------
20 ' Esta dica permite ajustar a PRAXIS
     20 com interface para um limite de
      75 colunas de impressao. Caso se
       queira alterar este numero de
     colunas, basta modificar o byte 48
     na linha 50 para o valor desejado.
30 FOR I=&HEFE0 TO &HEFF0:READB$:POKE I
,VAL("&H"+B$):NEXT
40 FOR I=&HFD9D TO &HFD9A STEP-1:READB$
:POKE I,VAL("&H"+B$):NEXT:NEW
50 DATA F3,21,15,F4,7E,FE,48,38,07,3E,0
D,CD,A5,00,AF,77,C9,EF,E0,02,F7
60 '-----------------------------------
      GERSON WASHISKI BARBOSA - PR
   -----------------------------------
```

ZX SPECTRUM

```
  10 REM ::::::::::::::::::::::::
  20 REM :   EFEITOS VISUAIS    :
  30 REM ::::::::::::::::::::::::
  40 BORDER 1: CLS
  50 FOR A=0 TO 47
  60 LET L=INT (RND*255): POKE U
SR "A"+A,L
  70 NEXT A
  80 PAUSE 100
  90 FOR A=1 TO 117: PRINT "ABCD
EF";: NEXT A
 100 PRINT "AB";AT 0,0;
 110 GOTO 0050
 120 REM ::::::::::::::::::::::::
 130 REM :OS CARACTERES DAS     :
         :LINHAS 90 E 100 DEVEM:
         :SER DIGITADOS EM UDG.:
 140 REM :FERNANDO S. JUNIOR-SP:
 150 REM ::::::::::::::::::::::::
```

MSX

```
10 '-----------------------------------
        ARQUIVE TODA A TELA
   -----------------------------------
20 '  Esta dica arquiva na memoria e
      repoe o conteudo de toda a  VRAM
       (incluindo os sprites) e pode
       ser usada em qualquer SCREEN.
30 CLEAR200,&HDE00:FORJ=0TO62:READA$:PO
KE&HDE00+J,VAL("&H"+A$):NEXT
40 DATA DB,A8,4F,32,3D,DE,E6,30,1F,1F,B
1,32,3E,DE,21,0,0,11,0,40,1,0,40,D3,A8,
CD,50,0,F3,CD,14
50 DATA 7,3A,3D,DE,D3,A8,C9,3A,3E,DE,21
,0,0,11,0,40,1,0,40,D3,A8,CD,53,0,F3,CD
,48,7,18,E3,0,0
60 '          Exemplo:
70 SCREEN2:FORJ=0TO10:X=RND(1)*255:Y=RN
D(1)*192:C=RND(1)*15:X1=RND(1)*255:X2=R
ND(1)*192:LINE(X,Y)-(X1,X2),C,BF:NEXT
80 DEFUSR1=&HDE00' arquiva
90 DEFUSR2=&HDE26' repoe
100 A=USR1(0)
110 A$=INKEY$:IFA$=""GOTO110
120 CLS:A=USR2(0)
130 GOTO130
140 '-----------------------------------
      FABIO S. MENEZES - SP
   -----------------------------------
```

CLUBE DO LEITOR

DICAS

MSX

PRINT USING COM NOTAÇÃO BRASILEIRA

Esta dica faz a instrução PRINT USING (variável numérica) imprimir valores no formato financeiro utilizado no Brasil. POKE&HFF9D,&HC3 ativa e POKE&HFF9D,&HC9 desativa a dica.

```
10 FORI=&HFFDB TO &HFFFD:READA$:
POKEI,VAL("&H"+A$):NEXT
20 FORI=&HFF9F TO &HFF9DSTEP-1:R
EADA$:POKEI,VAL("&H"+A$):NEXT
30 DATA F5,C5,D5,E5,21,C6,F7,7E
40 DATA FE,2C,28,0A,FE,2E,28,0A
50 DATA B7,28,0B,23,18,F1,36,2E
60 DATA 18,F9,36,2C,18,F5,E1,D1
70 DATA C1,F1,C9,FF,DB,C3
80 CLS:A=999999999999.99#:PRINTA
:PRINT
90 PRINTUSING"NCz$###############
,.##";A
```

MÁRCIO OSÓRIO FARIAS - PE

TRS COLOR

APAGA-TE, TELA!

Esta dica produz um CLS0 decrementando o valor de cada um dos bytes da tela de texto.

```
10 CLEAR 32000:FORJ=0TO24:READA$
20 POKE32000+J,VAL("&H"+A$):NEXT
30 DATA 8E,04,00,5F,A6,84,81
40 DATA 80,23,04,4A,5C,12,12
50 DATA A7,80,8C,06,00,25,EF
60 DATA 5D,26,E8,39
70 CLS0:FORJ=0TO510:A=RND(127)+1
28
80 PRINTCHR$(A);:NEXT
90 EXEC32000:GOTO70
```

LUIZ ALBERTO LOPES - RJ

ZX SPECTRUM

VU-METER NO MICRO

Esta rotina simula um "VU-Meter" do tipo bargraph encontrado nos aparelhos de som. Esta dica pode ser usada para indicar o melhor volume para carregar um programa ou para ajustar o azimute do gravador.

```
 5 LET A=10: LET B=11: LET C=12
10 LET D=13: LET E=14: LET F=15
20 LET C$="cd1f5b21a05978b128063
60e230b18f6062036082310fb3e7fdbf
e1f38e2c9"
30 LET C$=C$+"2100051605f3010000
dbfecb772801032b7cb520f4b7cb18cb
191520f8fbc9"
40 FOR I=1 TO 123 STEP 2
50 POKE 23295.5+I/2,VAL C$(I)*16
+VAL C$(I+1): NEXT I
60 LET C$="00007e7e7e7e0000":FOR
I=1 TO 15 STEP 2
70 POKE USR "A"+I/2-.5,VAL C$(I)
*16+VAL C$(I+1): NEXT I
80 PRINT AT 5,0;" Coloque uma fi
ta (programa ou musica) para rep
roduzir. Se for"
90 PRINT "programa,o melhor nive
l corres-      ponde a faixa ver
de."
100 PRINT AT 13,0;
110 FOR I=0 TO 31: PRINT CHR$ 14
4;
120 NEXT I
130 PRINT '' PAPER 5;TAB 21; PAP
ER 4;TAB 24; PAPER 2;TAB 31;" "
140 RAND USR 23296
```

CARLOS E. C. MATOS - BA

Envie suas dicas para Redação/MICRO SISTEMAS, Av. Presidente Wilson, 165 — grupo 1210, Rio de Janeiro, RJ, CEP 20030

CLUBE DO LEITOR

DICAS

ZX SPECTRUM

IMPRESSÃO DINÂMICA

```
10 LET B$="
          ": REM 32 ESPAÇOS
20 LET A$=B$+"** MICRO SISTEMAS
**"
30 LET A$=A$+B$+B$
40 FOR A=1 TO LEN A$-31
50 PRINT AT 21,0;A$(A TO A+31)
60 IF A>32 THEN PRINT AT 0,0;A$(
A-32 TO A-1)
70 PAUSE 10: NEXT A
80 GOTO 40
```

LUíS CLAUDIO ALBERO - SP

MSX

INPUT PERSONALIZADO

Esta dica altera os dois prompts da instrução INPUT do BASIC. Os caracteres dos prompts estão nos endereços FFF6H e FFF9H.

```
10 FORI=&HFFF5 TO &HFFFD
20 READA$:POKEI,VAL("&H"+A$)
30 NEXT
40 FORI=&HFDE0 TO &HFDE3
50 READA$:POKEI,VAL("&H"+A$)
60 NEXT:NEW
70 DATA 3E,3A,DF,3E,20,DF,C3
80 DATA D5,23,E1,C3,F5,FF
```

DIÓGENES S. LEÃO FILHO - PE

MSX

PLAY IT AGAIN,SAM

```
10 KEYOFF:CLS:DIMA$(2,5):FOR J=1
TO 2
20 READA$(J,1),A$(J,2),A$(J,3),A
$(J,4),A$(J,5)
30 N$="V15O4EFEDCDD.":K=1800:N=1
:GOSUB150
40 N$="EGFEDFF.":N=2:GOSUB150
50 N$="GO5CO4BAGAA.":N=3:GOSUB15
0
60 N$="BO5DCO4BABBO5CCO4GGG":K=2
860:N=4:GOSUB150
70 N$="CR64C.DE..":K=1370:N=5:GO
SUB150
80 PRINT:PRINT
90 NEXTJ:END
100 DATA "YOU MUST REMEMBER THIS
","A KISS IS JUST A KISS"
110 DATA "A SIGH IS JUST A SIGH"
,"THE FUNDAMENTAL THINGS APPLY"
120 DATA "AS TIME GOES BY","AND
WHEN TWO LOVERS WOO"
130 DATA "THEY STILL SAY I LOVE
YOU","ON THAT YOU CAN RELY"
140 DATA "NO MATTER WHAT THE FUT
URE BRINGS","AS TIME GOES BY"
150 PRINT A$(J,N):PLAYN$:FORI=1T
OK:NEXTI:RETURN
```

GEILSON LIMA SANTANA - BA

Envie suas dicas para Redação/MICRO SISTEMAS, Av. Presidente Wilson, 165 — grupo 1210, Rio de Janeiro, RJ, CEP 20030

Cibertécnica
informática entre amigos
PC/XT/AT
• desenvolvimento
• instalação
• manutencão
• software
• treinamento
• venda
impressoras, estabilizadores, formulários, discos, fitas, mesas.
Cibertécnica Informática Ltda. — Rua Senador Dantas, 117 Gr. 1941 — Tel.: (021) 262-8249

CLUBE DO LEITOR

DICAS

MSX

CARACTERES EM SHADOW E OUTLINE

Esta dica permite a impressão de caracteres em formato shade (sombra) ou outline (contornado) no modo gráfico SCREEN 2.
Entra: A$=texto
X,Y=posição na tela
T=tipo(0=shade,1=outline)
IN=incremento para X,Y

```
10 DEFINTA-Z:SCREEN2:COLOR1,15:C
LS
20 OPEN"GRP:"AS1:IN=10:T=0
30 X=10:Y=10:A$="Este texto está
":GOSUB1000
40 X=10:Y=Y+10:A$="no modo SHADE
...":GOSUB1000
50 T=1:X=10:Y=Y+20:A$="E este mo
stra o modo":GOSUB1000
60 X=10:Y=Y+10:A$="OUTLINE...":G
OSUB1000
70 X=10:Y=Y+10:A$="TECLE ":GOSUB
100:A$="[ENTER]":T=0:GOSUB1000
80 A$=INPUT$(1):X=0:Y=Y+20
90 FORA=1TO255:IFA<32THENA$=CHR$
(1)+CHR$(A+64)ELSEA=CHR$(A)
100 FORT=0TO1:GOSUB1000:NEXT
110 IFY=0THENA$=INPUT$(1):CLS:Y=
10
120 NEXT:GOTO90
1000 FORL=1TOLEN(A$):A1$=MID$(A$
,L,1):IFA1$=CHR$(1)THENFL=1:NEXT
1010 IFA1$>" "THENGOSUB1500
1020 X=X+IN:IFX>235THENX=0:Y=Y+I
N:IFY>182THENY=0
1030 NEXT:RETURN
1500 COLOR1:FORX1=XTOX+2:FORY1=Y
TOY+2:PSET(X1,Y1),POINT(X1,Y1)
1510 IF FL THENPRINT#1,CHR$(1);
1520 PRINT#1,A1$:NEXT:NEXT
1530 COLOR15:PSET(X+T,Y+T),POINT
STEP(0,0)
1540 IF FL THENPRINT#1,CHR$(1);
1550 PRINT#1,A1$:COLOR1:FL=0:RET
URN
```

ROBERTO DA COSTA MEDEIROS - PA

ZX SPECTRUM

DESTACANDO MENSAGENS

Para mudar as cores digite POKE 30035,N onde N é o valor do atributo desejado.

```
10 FOR T=30000 TO 30044: READ A:
POKE T,A: NEXT T: INK 7
20 DATA 14,32,33,0,88,6,24,17,32
,0,229,54,18,25,16,251,197,33,20
8,0,17,32,0,205,181,3,193,225
30 DATA 229,6,24,17,32,0,54,15,2
5,16,251,225,35,13,32,217,201
40 PRINT AT 11,8;"MICRO SISTEMAS
":RAND USR 30000: INK 0
```

PAULO EDUARDO GANDIN - RS

APPLE

TROCANDO CORES

Esta dica substitui uma cor na tela de baixa resolução por outra cor especificada.
O byte 30 contém o código da cor a ser alterada e o byte 31 o código da nova cor.

```
10 FOR J=768 TO 813:READ A:POKE
J,A:NEXT
20 DATA 169,0,133,207,165,31,32,
100,248,169,0,133,206,165,207,16
4,206,32,113,248,197,30,208,5
30 DATA 165,207,32,14,248,230,20
6,165,206,201,40,208,232,230,207
,165,207,201,40,208,220,96
40 GR:COLOR=15
50 HLIN 0,39 AT 0:HLIN 0,39 AT 3
9:VLIN 0,39 AT 20
60 FOR T=1 TO 3000:NEXT
70 POKE 30,0:POKE 31,5:CALL 768
```

HENRIQUE S. DE ALMEIDA - SP

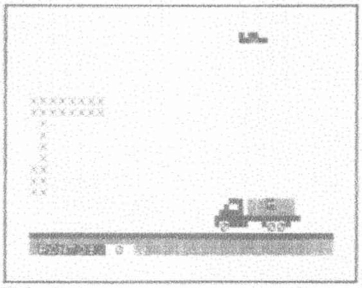
Envie suas dicas para a Redação do MICRO SISTEMAS na Av. Presidente Wilson, 165 Grupo 1210, - Rio de Janeiro - RJ CEP 20.030

Sabotagem

Claudio Costa

O jogo Sabotagem foi criado para micros de lógica Sinclair, em BASIC e com um programa à parte para inserir o desenho mais complexo numa variável string.

O desenho do caminhão, que aparece no programa foi adaptado da ilustração do artigo "Arte e técnica na tela do computador", de Renato Degiovani, publicado em **MICRO SISTEMAS** nº 27. O programa foi escrito em um TK 85 com 16 Kb.



Tela do jogo

```
   1 LET C$="
   2 LET N$="19 20 21 22 23 24 2
5 26 50 51 52 53 54 55 56 57 58
85 82 83 84 85 87 88 89 8C"
   3 LET M$="87 03 05 88 88 AD 8
8 88 87 80 80 82 8A 8A 8A 8A 8A
84 82 10 03 03 02 1C 1C 01"
   4 FOR T=1 TO 77 STEP 3
   5 POKE VAL N$(T TO T+1)+16500
,16*CODE M$(T)+CODE M$(T+1)-476
   6 NEXT T
```

Listagem 1

O JOGO

Uma potência estrangeira está construindo uma estação nuclear num ponto estratégico do planeta. Você é um temível sabotador a serviço de uma potência inimiga, com o objetivo de impedir, a todo custo, que tal estação seja concluída. Em seu avião, você sobrevoa a área e fica observando os caminhões que, a todo momento, chegam transportando material para a estação.

Sua tarefa não será tão difícil, já que você conseguiu sabotar o radar do inimigo, impossibilitando-o de registrar os caminhões até que eles cheguem à torre de desembarque de carga. Assim, tudo o que você tem a fazer é lançar sobre os caminhões sua artilharia de bombas, que neutralizam o material atômico, inutilizando-o completamente.

O único detalhe é que você precisa acertar em cheio as caixas contendo o material, antes que o caminhão chegue à torre de desembarque, já nos limites do radar. Não adianta tentar atirar quando as caixas estiverem sendo desembarcadas, pois você seria descoberto; os mísseis já lançados se autodetonarão neste momento.

Os mísseis que errarem o alvo também se autodestruirão silenciosamente antes de tocar o solo, sem causar dano aos caminhões. Lembre-se de que, para impedir que a estação possa ser concluída, você precisa neutralizar mais da metade da carga.

Esperamos que você, além de temível sabotador, também seja bom de pontaria!

MONTAGEM

Comece pela listagem 1, digitando antes uma linha **1 LET C$ = "(74 espaços)"** (dê **RUN** e **PRINT LEN C$** para conferir). Em seguida, elimine as linhas de 2 a 6 e comande direto ao teclado:

POKE 16510,140

Entre, então, com a listagem 2. Para lançar bombas, aperte qualquer tecla e no final do jogo faça o mesmo para recomeçar.

O número de caixas (linha 160) é escolhido aleatoriamente entre 20 e 50, mas o programa não informa quantas caixas vão ser desembarcadas. Para alterar a dificuldade do jogo, modifique as linhas 110 e 115. As linhas 225 a 300 desenham apenas a apresentação e podem ser omitidas, se você desejar.

Claudio Costa é desenhista free-lancer e tem como hobby a programação de microcomputadores.

```
   1 REM CLAUDIO COSTA
   2 GOTO 125
   3 LET M=0
   4 LET X=3
   5 IF INKEY$<>"" THEN GOTO 27
   7 PRINT AT 0,A;A$;AT 17,C;C$
   9 LET A=A+1
  11 LET C=C-1
  13 IF C=-1 THEN GOTO 53
  15 IF A=28 THEN GOSUB 35
  17 IF NOT M THEN GOTO 5
  19 PRINT AT X,M;" ";AT X-2,M;"
"
  21 LET X=X+2
  23 IF X=17 THEN GOTO 41
  25 GOTO 7
  27 IF M<7 THEN GOTO 7
  29 LET M=A+1
  31 PRINT AT 1,M;"▒";AT 1,M;" "
  33 GOTO 19
  35 PRINT AT 0,28;"    "
  37 LET A=NOT A
  39 RETURN
  41 IF M=C+7 THEN GOTO 47
  43 PRINT AT 15,M;" ";TAB M;"▒"
;AT 15,M;" "
  45 GOTO 3
  47 FOR I=1 TO 5
  49 PRINT AT 17,M;"H";AT 15,M;"
";AT 17,M;"▒"
  51 NEXT I
  53 LET C$(7)="▒"
  55 LET P=P+C+10
  57 PRINT AT 15,M-1;"▒ ▒";AT 15
,M-1;"   ";AT 21,8;P
  59 LET K=K+1
  61 GOTO 3
  63 IF M THEN PRINT AT X-2,M;"▒
";AT X-2,M;" "
  65 FOR I=10 TO 16
  67 PRINT AT I,6;CHR$ 14;AT 0,A
;A$
  69 LET A=A+1
  71 IF A>27 THEN GOSUB 35
  73 NEXT I
  75 FOR I=16 TO 10 STEP -1
  77 PRINT AT I,6;C$(7);TAB 6;"
";AT 0,A;A$
  79 LET A=A+1
  81 IF A>27 THEN GOSUB 35
  83 NEXT I
  85 PRINT AT 10,6;" "
  87 LET N=N-1
  89 IF NOT N THEN GOTO 100
  91 LET C$(7)="▒"
  93 LET C=22
  95 PRINT AT 17,0;L$
  97 GOTO 3
 100 PRINT AT 4,4;"DESEMBARCARAM
";F;" CAIXAS"
 105 PRINT ,,TAB 3;"VOCE NEUTRAL
IZOU ";K;" CAIXAS"
 110 IF K>INT (F/2) THEN PRINT A
T 12,4;"PARABENS - MISSAO CUMPRI
DA"
 115 IF K<=INT (F/2) THEN PRINT
AT 12,6;"VOCE NAO CONSEGUIU"
 120 IF INKEY$="" THEN GOTO 120
 125 CLS
 130 POKE 16418,0
 135 RAND
 140 LET C$="▒▒▒"
 145 LET M$="▒"
 150 DIM L$(96)
 155 LET A=INT (RND*28)
 160 LET N=INT (RND*30)+20
 165 LET C=22
 170 LET P=0
 175 LET K=P
 180 LET F=N
 185 PRINT AT 0,A;A$;AT 8,0;"XXX
XXXXX",,"XXXXXXXX"
 190 FOR Z=1 TO 4
 195 PRINT " X"
 200 NEXT Z
 205 PRINT "XX",,"XX",,"XX",,TAB
C;C$
 210 PRINT AT 20,0;"▒▒▒ PONTOS 0 ▒▒▒"
 215 GOTO 3
 220 SAVE "SABOTAGEM"
 225 PRINT AT 8,5;"▒▒▒▒▒▒▒▒▒▒▒▒"
 230 PRINT TAB 5;"▒▒▒▒▒▒▒▒▒▒▒▒"
 235 PRINT TAB 5;"▒▒▒▒▒▒▒▒▒▒▒▒"
 240 PRINT ,,"XXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
 245 FOR Z=0 TO 24
 250 PRINT AT 0,Z;" ▒"
 255 NEXT Z
 260 FOR Z=1 TO 7
 265 PRINT AT Z,26;" ";TAB 26;"*
"
 270 NEXT Z
 275 PRINT ",*"
 280 FOR Z=1 TO 14
 285 PRINT TAB 31;" "
 290 SCROLL
 295 NEXT Z
 300 RUN
```

Listagem 2

Aprenda neste artigo, compatível com os micros de lógica Sinclair, como utilizar sub-rotinas em linguagem de máquina.

CLS Pincel

Nelson N. S. Santos

Um dos recursos de programação mais comum é o uso de sub-rotinas. Em BASIC, uma sub-rotina é um trecho de qualquer programa, terminado pelo comando **RETURN**. Para execução, usa-se o comando **GOSUB**, seguido do número da linha onde ela inicia. Quando se programa usando sub-rotinas, um cuidado especial se impõe: não *invadi-las* acidentalmente.

Em linguagem de máquina, o uso de sub-rotinas é extremamente semelhante ao BASIC: em vez de **GOSUB**, **CALL**; em vez de número de linha, endereço de início; em vez de **RETURN**, **RET**.

Vamos escrever um programa com efeito de vídeo interessante: o *CLS Pincel* (o batismo fica por conta dos meus filhos), para ilustrar o uso de sub-rotinas em Assembly.

A SUB-ROTINA DE PAUSA

A maneira mais simples de se obter uma pausa é com um loop que não faz nada. Por exemplo, a seção DICAS de **MS nº 28** sugere:

```
. . . . . .
800 FOR A=0 TO 100
801 NEXT A
802 RETURN
```

como forma de produzir uma pausa de aproximadamente três segundos. Já, em linguagem de máquina podemos fazer conforme a figura 1.

A sub-rotina ocupa oito bytes, incluindo RET. A pausa produzida é pequena (menos de um segundo), mas satisfaz nossos propósitos: tornar visíveis simulações de movimento. Ela pode ser diminuída, variando-se o conteúdo inicial do acumulador. Em nosso programa, chamaremos esta sub-rotina de dentro de outra sub-rotina, não havendo nenhum problema.

A SUB-ROTINA PINCEL

Desejamos o seguinte efeito de vídeo: ir *pintando* de preto (espaço inverso) a primeira coluna da tela de cima para baixo, a segunda de baixo para cima, a terceira de cima para baixo, . . ., até a trigésima segunda de baixo para cima, simulando os movimentos de um pincel. Quando a tela estiver completamente preta, chamaremos novamente a sub-rotina para ir apagando (colocando espaços), da mesma maneira. Tal efeito não seria perceptível – devido à velocidade da linguagem de máquina – sem a sub-rotina de pausa que explicamos anteriormente.

Como todos os registradores serão usados, explicaremos suas funções, por ordem de aparecimento.

HL. . . ponteiro para a tela, através de LD HL, (D. FILE).

DE. . . carregando com 33. ADD HL, DE equivalerá a descer uma posição na tela, e SBC HL,DE equivalerá a subir uma posição. Em caso de dúvida, consulte o esquema da tela dos Sinclair, publicado em meu artigo "Como virar sua tela sem fazer força", em **MS nº 45**.

C. . . contador para 16 pares de colunas.

B. . . contador para 21 subidas ou descidas.

A. . . deverá ser pré-definido, no corpo principal do programa, com código do caráter a ser colocado na tela antes da chamada da sub-rotina. Veja a figura 2.

A sub-rotina ocupa 36 bytes, incluindo **RET**.

A SUB-ROTINA PRINT AT

Em linguagem de máquina é possível chamar sub-rotinas que você não escre-

RÓTULO	INSTRUÇÃO	CÓDIGOS	COMENTÁRIOS
	CALL PAUSA	CD XX XX	chamaremos a sub-rotina no endereço em que ela se iniciar; normalmente será colocada no final do programa
PAUSA	PUSH AF	F5	preserva na pilha o conteúdo do acumulador
	LD A,0	3E 00	inicializa o acumulador com 0
DIMINUI	DEC A	3D	decrementa A, que passa a conter 255(FF)
	JR NZ,DIMINUI	20 FD	decrementa o acumulador até 0
	POP AF	F1	recupera o conteúdo original do acumulador
	RET	C9	não volta ao BASIC, mas sim à instrução seguinte a CALL PAUSA

Figura 1

CLS PINCEL

RÓTULO	INSTRUÇÃO	CÓDIGOS	COMENTÁRIOS
PINCEL	LD HL,(D.FILE)	2A 0C 40	
	LD DE,33	11 21 00	
	LD C,16	0E 10	
PROX	INC HL	23	entra na coluna ímpar
	LD B,21	06 15	
POKE 1	LD (HL),A	77	começa a construção da coluna ímpar
	CALL PAUSA	CD XX XX	a pausa faz o efeito visível
	ADD HL,DE	19	descendo...
	DJNZ POKE 1	10 F9	loop
	LD (HL),A	77	termina a coluna ímpar
	INC HL	23	passa para a coluna par
	LD B,21	06 15	
POKE 2	LD (HL),A	77	começa a construção da coluna par
	CALL PAUSA	CD XX XX	
	AND A	A7	zera a CARRY FLAG sem alterar o acumulador
	SBC HL,DE	ED 52	subindo...
	DJNZ POKE 2	10 F7	loop
	LD (HL),A	77	termina a coluna par
	DEC C	0D	decrementa o contador de pares de colunas
	JR NZ,PROX	20 E5	loop para terminar a tela
	RET	C9	não volta ao BASIC, mas sim à instrução seguinte a CALL PINCEL

Figura 2

Podemos variar o tempo de pausa, alterando completamente o efeito visual. O programa abaixo faz isto de maneira espetacular.

```
10 FOR F=255 TO 0 STEP -17
20 POKE 16565,F
30 RAND USR 16514
40 NEXT F
50 PRINT "FIM DA DEMONSTRAÇÃO"
```

Acredito que você agora saiba mais sobre sub-rotinas em Assembly do que sabia antes...

Nelson N. S. Santos é professor de Química e de Matemática e consultor técnico da DIMERJ Sistemas. Usuário de um Sinclair, é também autor do livro Além do BASIC, *sobre linguagem Assembly para a linha Sinclair, recentemente lançado pela Editora Campus.*

veu... Sim, pois estão a nosso dispor todas as sub-rotinas da ROM. Para chamá-las, são necessários dois conhecimentos: o endereço inicial e quais os dados que devem ser fornecidos antes de chamá-las.

A sub-rotina PRINT AT é das mais simples. Seu endereço inicial é 08F5 (2293). Assim, CALL PRINT AT é CD F5 08 (não esqueça da inversão!). Antes de chamá-la, B deve conter a linha e C a coluna para impressão. O programinha a seguir tirará qualquer dúvida. Vamos realizar PRINT AT 11,16; " * ", em linguagem de máquina.

A maioria das sub-rotinas da ROM *destrói* os conteúdos dos registradores, e esta não é uma exceção; logo, vamos chamar PRINT AT antes de carregar o acumulador.

```
LD BC,XXXX          01 10 08    (B=11 e C=16)
CALL PRINT AT       CD F5 08
LD A,23             3E 17
RST 10h             D7
RET                 C9
```

Teste o seguinte programa BASIC:

```
 0 REM aqui está o programinha
10 PRINT AT 11,16;"□"
20 RAND USR 16514
```

Rode-o e só haverá o asterisco no meio da tela e a indicação 0/20. Se você prestar bastante atenção ao vídeo, perceberá o espaço inverso sendo impresso e substituído pelo asterisco.

Esta sub-rotina será necessária em nosso programa porque CLS não deve apenas limpar a tela, mas também posicionar o cursor de impressão em 0.0. O corpo principal do programa está escrito na figura 3.

Vamos agora ao cálculo dos endereços iniciais das sub-rotinas. O corpo principal do programa ocupa 13 bytes, de 16514 e 16526. Logo, o endereço inicial de *Pincel* será 16527. Vamos passar para hexa:

RÓTULO	INSTRUÇÃO	CÓDIGOS	COMENTÁRIOS
	LD A,128	3E 80	o caráter colocado na tela será espaço inverso
	CALL PINCEL	CD XX XX	
	XOR A	AF	zera o acumulador para a colocação de espaços
	CALL PINCEL	CD XX XX	
	CALL PRINT AT	CD F5 08	vindo da sub-rotina PINCEL, B=0 e C=0; tudo certo para chamar PRINT AT na ROM
	RET	C9	retorna ao BASIC
PINCEL			
PAUSA			

Figura 3

```
16527 - 16384 = 143 = 8F
          ↓
        4000h
```

Logo, 16527 = 408F e CALL PINCEL tornam-se CD 8F 40 (não esqueça da inversão!).

Pincel ocupa 36 bytes, ou seja, de 16527 a 16562. Logo, o endereço inicial de PAUSA será 16563. Em hexa: 16563 — 16384 = 179 = B3. Então, 16563 = 40B3 e CALL PAUSA tornam-se CD B3 40.

O programa completo ocupa 57 bytes (13 + 36 + 8). Assim sendo, é necessária uma linha 0 REM com 57 caracteres quaisquer. O programa completo tem a seguinte listagem hexa:

```
16514 - 3E 80 CD 8F 40 AF CD 8F
16522 - 40 CD F5 08 C9 2A 0C 40
16530 - 11 21 00 0E 10 23 06 15
16538 - 77 CD B3 40 19 10 F9 77
16546 - 23 06 15 77 CD B3 40 A7
16554 - ED 52 10 F7 77 0D 20 E5
16562 - C9 F5 3E 00 3D 20 FD F1
16570 - C9
```

O uso de RAND USR 16514 como linha de programa substitui o CLS da máquina, podendo ser usado como você achar melhor. O programinha BASIC abaixo demonstra o efeito.

```
10 FOR F=0 TO 10
20 PRINT TAB F*.6;"DEMONSTRAÇÃO DO CLS PINCEL"
30 PRINT TAB F*.6;"                          "
40 NEXT F
50 RAND USR 16514
60 PRINT "TELA LIMPA"
```

274-8845
Agora em PABX

Fita Impressora Nacional ou Importada
Ligue 274-8845

Formulário Continuo 1, 2 ou 3 vias
Ligue 274-8845

Aquele Arquivo para diskettes 5.1/4" ou 8"
Ligue 274-8845

Pastas para Listagens 80 e 132 colunas
Ligue 274-8845

Etiquetas Adesivas em Formulário Continuo
Ligue 274-8845

Diskettes 5.1/4" ou 8" (5 anos de garantia)
Ligue 274-8845

Reebobinagem em Nylon e Polietileno
Ligue 274-8845

Nós temos tudo isso, e muito mais...

* Pronta Entrega
* Qualquer Quantidade
* Garantia de Qualidade

Suprimento
MATERIAIS PARA COMPUTADORES

R. VISCONDE DE PIRAJA, 550/202 — 274-8845 — IPANEMA — RIO

Analisador de header

Paolo Fabrizio Pugno

Certamente, quem já prestou um pouco de atenção ao sistema de operação com cassete do ZX Spectrum ou TK90X percebeu que sua grande característica é a variedade de comandos disponíveis, aliada a uma grande segurança na transferência de dados do micro para a fita e vice-versa. Porém, é muito normal que queiramos fazer um back-up de segurança de um determinado programa (nunca se sabe...), e é aí que geralmente começam os problemas.

Enquanto o programa a ser copiado for só em BASIC, não teremos maiores dificuldades, pois basta dar um BREAK e, logo em seguida, um banalíssimo SAVE "nome" LINE xx.

Mas é muito freqüente que os programas contenham uma rotina em linguagem de máquina, a qual é carregada logo que o programa em BASIC começa a rodar através de um LOAD " " CODE. Porém, neste caso, fica muito difícil, senão impossível, com procedimentos normais, sabermos qual o comprimento em bytes desta rotina e qual é seu endereço inicial. Isto porque o LOAD "" CODE, sem parâmetro nenhum, assume como *default* o endereço original do bloco de bytes no momento em que ele foi salvo. De alguma forma, porém, o micro deve ser informado destes parâmetros. E é justamente aqui que a *porca torce o rabo...* Aonde vamos buscar estas informações?

O HEADER

Antes de carregar qualquer informação vinda do cassete, o Spectrum tem que saber se trata-se de um programa BASIC, um bloco de bytes, uma matriz numérica ou alfanumérica, para, só então, estabelecer o modo de colocação na memória. O que fornece estas informações é o header ou, simplesmente, o cabeçalho – aquela curta descarga de sons que antecede cada bloco salvo, seja ele um programa, matriz etc. Ele é composto por 17 bytes assim subdivididos:

O primeiro byte informa sobre o tipo de bloco que se segue, podendo assumir quatro valores distintos: 00 (indica programas em BASIC); 01 (indica matriz numérica); 02 (indica matriz alfanumérica) e 03 (indica bloco de bytes).

Os dez bytes seguintes contêm o nome do bloco em questão. Se o nome tiver menos de dez caracteres, o espaço restante será preenchido com códigos ASCII 32 (space). Isto também explica aquela estranha obsessão do micro em não aceitar nomes com mais de dez caracteres. O 12º e 13º bytes contêm o número total de bytes que serão carregados na memória.

O 14º e 15º indicam a linha de programa para a qual o micro deve saltar logo após o carregamento do programa em BASIC, se este foi salvo com SAVE "nome" LINE xx. Caso o bloco seja somente se bytes (primeiro byte do header igual a três), eles indicam o endereço de início do bloco de bytes.

O 16º e 17º devem ser considerados somente se o bloco se referir a um programa em BASIC. Neste caso, eles contêm o número de bytes relativos ao programa sem variáveis. Logicamente, se antes de efetuarmos o SAVE dermos um CLEAR (procedimento recomendado), o valor destes bytes será igual ao indicado pelo 12º e 13º bytes.

Para podermos analisar corretamente o header e, assim, obtermos estas informações, preparei um pequeno programa, o qual contém uma pequena rotina em linguagem de máquina. Esta rotina predispõe o micro a guardar, em um local da memória estabelecido por nós, um certo número de bytes também determinado por nós. Observe como ele funciona e veja sua estrutura principal:

```
LD DE, 17
LD IX, 23372
LD A, 0
SCF
CALL 1366
```

No par de registradores DE colocamos o número de bytes que iremos carregar; IX contém o endereço a partir do qual vamos carregar estes bytes; o acumulador deve conter 0 e devemos setar o *carry flag* (caso contrário, ao invés de um LOAD executaremos um VERIFY). Logo após, chamamos a rotina da ROM no endereço (decimal) 1366, e a brincadeira estará feita. Naturalmente, podemos carregar quantos bytes quisermos, somente alterando o valor de DE, e a partir de qualquer endereço alterando IX. No nosso caso, resolvi colocar a rotina na área reservada no buffer da impressora pelos seguintes motivos:

- O mais certo seria alocá-la no topo da memória, mas isso demandaria duas versões do mesmo programa, uma para quem tem 16 e outra para quem tem 48 Kb de memória. Melhor não complicar.
- Poderia colocá-la numa linha 1 REM, como se fazia nos micros compatíveis com o ZX81, porém no Sepctrum isso não é recomendável, pelo seguinte motivo: microdrives. Eles ainda não foram lançados, mas têm toda a *pinta* de que serão, e muita gente já possui os originais da Sinclair, por meios oficiais ou não. Acontece que a interface que controla os microdrives desloca a área BASIC em, pelo menos, 58 bytes à frente, para criar novas variáveis de sistema, e isso prejudicaria o funcionamento normal do programa.

Optando pela solução mais simples, resolvi colocar, seja o programa, sejam os dados, no buffer da impressora. Este buffer começa no endereço 23296 e é longo (256 bytes), assim, temos espaço de sobra.

Quem já tiver alguma experiência com programação vai perceber que o programa é mais extenso do que o que foi descrito: não há nenhum problema, apenas coloquei uns "embelezamentos" que podem ser facilmente eliminados. Mas, mesmo assim, não creio que o programa seja tão longo a ponto de provo-

car desânimo só em pensar em digitá-lo (muito pelo contrário).

DESCRIÇÃO DO PROGRAMA

Linhas 10 e 20: contêm os códigos que serão pokeados no buffer da impressora, contendo a rotina em linguagem de máquina;
Linha 30: loop que lê um dado e efetua os POKEs;
Linha 40: faz a chamada para a rotina em l/m (linguagem de máquina) e, conforme o tipo de header analisado, manda para a sub-rotina BASIC correspondente;
Linhas de 100 a 130: imprimem as informações relativas a um programa em BASIC;
Linha 200: imprime as informações relativas a uma matriz numérica;
Linha 300: imprime as informações relativas a uma matriz alfanumérica;
Linha 400: imprime as informações relativas a um bloco de bytes;
Linha 500: sub-rotina utilizada por todas as outras, imprime o nome contido no header e o número total de bytes do bloco, bem como calcula os outros dois parâmetros.

VARIÁVEIS UTILIZADAS

n1 – contém o número total de bytes do bloco;

```
  1 REM ************************
  2 REM *      ANALISADOR      *
    *        DE  HEADER        *
  3 REM ************************
  4 REM
  5 REM
 10 DATA 62,2,205,1,22,33,35,91
,6,41,126,215,35,16,251,17,17,0,
221,33,76,91,62,0,55,205,86,5,33
,76,91,6,0,78,201
 20 DATA 22,10,0,17,3,16,6,18,1
,32,79,75,32,76,73,71,65,69,32,7
9,32,71,82,65,86,65,68,79,82,32,
69,32,65,71,65,55,82,68,69,46,32
,25,"REM
 30 RESTORE : FOR f=0 TO 75: RE
AD a: POKE (23296+f),a: NEXT f
 40 BORDER 7: CLS : GOTO 100+10
0*USR 23296
 100 CLS : BORDER 1: PRINT PAPER
 4;"PROGRAMA EM BASIC"'': GOSUB
500: PRINT PAPER 5;"No. de bytes
 de programa: "; PAPER 7;n3''
 110 IF n2<=9999 THEN PRINT PAPE
R 5;"LINE: "; PAPER 7;n2''
 120 IF n1<>n3 THEN PRINT PAPER
3; INK 9;"Bytes de variaveis: ";
 PAPER 7;n1-n3: STOP
 130 PRINT PAPER 3; INK 9;"O PRO
GRAMA NAO CONTEM VARIAVEIS.": ST
OP
 200 CLS : BORDER 2: PRINT PAPER
 4;"MATRIZ NUMERICA"'': GOSUB 50
0
 210 STOP
 300 CLS : BORDER 3: PRINT PAPER
 4;"MATRIZ ALFANUMERICA"'': GOSU
B 500
 310 STOP
 400 CLS : BORDER 4: PRINT PAPER
 4;"BLOCO DE BYTES"'': GOSUB 500
 410 PRINT PAPER 5;"Endereco ini
cial: "; PAPER 7;n2: STOP
 500 PRINT PAPER 5;"Nome: "; FO
R f=1 TO 10: PRINT PAPER 7;CHR$
PEEK (23372+f);: NEXT f: LET n1=
PEEK 23383+256*PEEK 23384: PRINT
 '' PAPER 5;"No. total de bytes:
 "; PAPER 7;n1'': LET n2=PEEK 23
386+256*PEEK 23386: LET n3=PEEK
23387+256*PEEK 23388: RETURN
1000 SAVE CHR$ 18+CHR$ 1+"HEADER
" LINE 1
```

Analisador de header

n2 – contém o LINE (caso o bloco seja um programa BASIC e este tenha um autostart) ou o endereço inicial do bloco (caso o header se refira somente a um bloco de bytes);
n3 – contém o número de bytes de programa BASIC (sempre que o bloco for um programa BASIC).

Para finalizar, aí vão duas observações importantes: 1) quem tem a impressora Alphacom 32 pode ter alguns problemas com este programa, pois ele dificulta o acesso ao buffer. Neste caso, é recomendável desconectá-la. Quem possui uma ZX Printer ou TK Printer não precisa se preocupar; 2) pode acontecer, em alguns casos, do header conter um caráter de controle, e o mesmo, ao ser impresso, causar erro K (cor inválida) ou algo semelhante. Caso isto aconteça, simplesmente digite CONT, e o programa prosseguirá mostrando os dados normalmente.

Paolo Fabrizio Pugno cursa atualmente o primeiro ano da Faculdade de Engenharia Industrial (SP) e possui um ZX Spectrum, com o qual desenvolve programas.

COISA DE MACRO
OS MICROS DA COMPUTERWARE!

Para quem pensa grande, a Computerware tem sempre os melhores preços do mercado, com estoques suficientes para uma entrega imediata e condições de pagamento a sua escolha.
Além de descontos especiais, a Computerware oferece assistência técnica com profissionais altamente especializados, o melhor atendimento, contrato de manutenção e uma infinidade de vantagens, que fazem a nossa macro diferença.

PCPAQ.
Microcomputador Compacto, totalmente compatível com o IBM PC/XT. Duas reduzidas placas de circuito, o PCPAQ reúne no mesmo gabinete: CPU, memória inicial de 256 K, monitor de vídeo 9'', 2 drives de 360 K, além de 2 slots adicionais para expansões e comunicações de dados. É incrível a versatilidade do PCPAQ, com um peso total de 13 kg, pode ser levado para onde quiser, protegido por uma bolsa almofadada . ... Consulte-nos.

UNITRON Ap II
Computador pessoal. Microprocessador 6502, 48 K de memória RAM e 12 K de memória ROM, teclado com maiúsculas e minúsculas. 8 conectores para expansões.

UNITRON T.I.
Além de reunir as características do APII, possui o Teclado Inteligente Unitron, que, diretamente, como em máquinas de escrever, fornece maiúsculas / minúsculas e acentuação da língua portuguesa.
Apresenta a característica especial de permitir a programação de cada tecla com comandos ou funções definidos pelo usuário.
Na Computerware, encontram-se também a disposição, todas as interfases / placas e periféricos do Unitron. ... Consulte-nos.

★ ASSISTÊNCIA TÉCNICA ★
(021) 262-1886
• Técnicos especializados na fábrica
• Atendimento imediato
• Contratos de manutenção

VISITE NOSSO SHOW-ROOM
Rio - INFOSHOPPING
Rua do Catete, 311

COPYRIGHT

COMPUTER WARE Informática Ltda.

Rio - Av. Almirante Barroso, 91 - 11º andar - Tel.: (021) 240-7294
Rio - INFOSHOPPING - R. do Catete, 311 - (Show Room)
São Paulo - Rua Jesuíno Arruda, 797/Grupo 22 - Tel.: (011) 881-7446
Curitiba - Av. João Gualberto, 1375 - 1º andar - Tel.: (041) 253-5433

UDG 3

José Rafael Sommerfeld e Fernando Luiz de Carvalho e Silva

Na edição de novembro, mostramos um interessante artigo sobre o uso da função UDG no TK90X. Essa função permite a redefinição do formato de alguns caracteres, permitindo, assim, acentuação em português (UDG 0), em espanhol (UDG 1) ou até mesmo a criação de pequenos desenhos (UDG 2). Este último recurso facilita bastante o desenvolvimento de jogos, permitindo ainda que se criem seres extraterrenos, naves interplanetárias e muitas outras ilustrações.

Porém, como tudo na vida, a função UDG 2 tem suas limitações. Ela permite que se redefina apenas 21 caracteres, que são os caracteres gráficos do micro de A a U. Se tivermos então a necessidade de redefinir, por exemplo, 50 caracteres, a função não nos permitirá.

No entanto, nem tudo está perdido, pois nessa hora lançaremos mão de uma simples e poderosa ferramenta, que nos permite definir 96 novos caracteres: o programa UDG 3.

COMO FUNCIONA O PROGRAMA

Para ativar sua execução, digite RUN ou, se desejar utilizá-lo com outro programa (durante o desenvolvimento de jogos, isto pode ser interessante), digite RUN 9000 ou GOTO 9000.

Assim, logo aparecerá a pergunta "Transfere Tabela? (S/N)". Essa pergunta tem como objetivo saber se o usuário deseja inicializar o formato dos 96 novos caracteres, ou seja, se o formato a dar aos novos caracteres criados será igual ao dos caracteres normais do micro. Dessa maneira, se esta for a primeira utilização do programa, após colocá-lo na memória, deve-se responder S(sim). Caso o programa já tenha sido utilizado, deve-se então responder N(não) ou simplesmente ENTER, que o programa irá apresentar o menu. Fica, pois, convencionado o seguinte: os caracteres que serão redefinidos pelo programa são os novos criados e não os caracteres normais do micro.

Ao pedir a opção 1(redefinir), o programa exibirá todos os caracteres no vídeo, e no início deles (sobre o caráter espaço), aparecerá um cursor que indicará o caráter que se deseja redefinir. Para movimentar esse cursor, utilize as teclas 5, 6, 7 e 8, que o fazem deslocar-se na direção indicada pelas setas. Quando estiver com o cursor sobre o caráter desejado, tecle ENTER. O programa então transferirá o caráter que estava sob o cursor para a posição da letra A gráfica e entrará em UDG 2, ou seja, a partir daí, os recursos de edição do caráter serão aqueles já conhecidos pelos usuários do equipamento. Ao terminar a edição (CAPS SHIFT e 0), o programa voltará à última tela que apresentou antes de entrar em UDG 2 (a que exibe os caracteres a redefinir). Nesse momento, para visualizar os caracteres redefinidos, digita-se R e para visualizá-los na forma normal, digita-se N.

Observem — na tela de apresentação dos caracteres — que ao teclarmos ENTER, o programa envia para a função UDG 2 a definição do caráter que está aparecendo na tela, no momento. Assim, se quiser alterar um caráter já definido, pressione antes a tecla R para que o formato redefinido apareça.

```
  1 REM
-------------------------------
Redefinidor.          31/09/85
José Rafael Sommerfeld
Fernando Luiz de C. e Silva
-------------------------------
9000 IF PEEK 23731>250 THEN CLEA
R 64599: LET k=32768
9010 IF PEEK 23731<250 THEN CLEA
R 31831: LET k=0
9020 INPUT "Transfere Tabela ? (
S/N) ";a$: IF a$="s" OR a$="S" T
HEN GOSUB 9700
9030 LET x=0: LET y=2: CLS : PRI
NT ">UDG 3";"">1-Redefinir";">2-
Ler CHR$ ";">3-Gravar CHR$ ";">4
-Finalizar"
9040 INPUT op: IF op<1 OR op>4 T
HEN GOTO 9030
9050 IF op=4 THEN CLS : STOP
9060 GOTO 9000+op*100
9100 PRINT AT 0,0;">UDG 3";"";"
"#$%&'()*+,-./0123"
9102 PRINT "456789:;<=>?@ABCDEFG
"
9104 PRINT "HIJKLMNOPQRSTUVWXYZ[
"
9106 PRINT "\]↑_£abcdefghijklmno
"
9108 PRINT "pqrstuvwxyz{|}~©"
9110 PAUSE 10: LET A$=SCREEN$ (Y
,X): PRINT INVERSE 1;AT Y,X;A$ '
9114 PAUSE 10: PRINT AT Y,X;A$
9116 LET X=X+(INKEY$="8" AND X<1
9)-(INKEY$="5" AND X>0)
9118 LET Y=Y+(INKEY$="6" AND Y<6
)-(INKEY$="7" AND Y>2)
9120 IF Y=6 AND X>15 THEN LET X=
15
9122 IF INKEY$=CHR$ 13 THEN LET
Z1=256*PEEK 23608+256*PEEK 23607
: GOTO 9132
9124 IF INKEY$=CHR$ 12 THEN GOTO
 9030
9126 IF INKEY$="N" OR INKEY$="n"
 THEN GOSUB 9400: GOTO 9100
9128 IF INKEY$="R" OR INKEY$="r"
 THEN GOSUB 9500: GOTO 9100
9130 GOTO 9110
9132 LET Z2=((CODE A$-32)*8): LE
T Z=Z2+Z1: FOR F=Z TO Z+7
9134 POKE USR "A"+F-Z,PEEK F
9136 NEXT F: CLS : UDG 2: CLS
9138 FOR F=USR "A" TO USR "B"-1
9140 POKE Z2+31832+k+F-USR "A",P
EEK F
9142 NEXT f: GOTO 9100
9144 GOTO 200
9200 GOSUB 9600
9210 LOAD a$CODE
9220 GOTO 9030
9300 GOSUB 9600
9310 SAVE a$CODE 31832+k,768
9320 GOTO 9030
9400 POKE 23606,0: POKE 23607,60
: RETURN
9500 POKE 23606,88: POKE 23607,1
23: IF k>0 THEN POKE 23607,251
9510 RETURN
9600 CLS : INPUT "Nome do Bloco
 ";LINE a$: RETURN
9700 FOR z=15616 TO 16383: POKE
z+16216+k,PEEK z: NEXT z: RETURN
```

UDG 3

Para voltar ao menu, digite CAPS SHIFT e 0.

Ao pedir a opção 2 (leitura), o micro solicitará o nome do bloco. Isto é, para os casos de antigas redefinições já feitas e gravadas em fita e que se deseje alterá-las ou simplesmente consultá-las. Informe o nome da gravação e posicione a fita no início da gravação desejada. A instrução LOAD já está embutida no programa.

Ao pedir a opção 3 (gravação), o micro solicitará o nome do bloco de caracteres, que se definiu e que se deseja gravar. Este será o nome usado em futuras leituras desse bloco de caracteres. A instrução SAVE também está embutida no programa.

COMO USAR O PROGRAMA

Descreveremos agora duas maneiras básicas de se utilizar o programa:

— Apenas para definir novos caracteres;
— Definir novos caracteres utilizando-os em um programa em confecção.

A primeira maneira será aquela em que o usuário simplesmente quer ver seus caracteres definidos à sua maneira, personalizados, ou, então, aquele usuário que quer fazer um belo desenho para utilizar em aberturas de outros programas ou coisas assim.

A segunda maneira se aplica bem ao desenvolvimento de jogos, onde a Arte se funde com a Lógica para produzir um interessante e bonito passatempo.

Abordaremos então como fazer esse trabalho. Antes de iniciar o desenvolvimento de seu jogo, carregue o programa

UDG 3 na memória. Utilize para numeração das linhas de seu programa os números de 1 a 8999, pois o UDG 3 começa a partir da linha 9000. Comece com o desenvolvimento de seu jogo, e, a medida que necessitar de caracteres especiais, execute UDG 3, criando-os. Lembre-se que durante o uso do UDG 3, indica-se ao micro como desejamos ver os caracteres. Sejam R (Redefinidos) ou N (Normais). Assim, após definir seus formatos, pense no que será feito. Se você quiser testar como ficará o movimento da nova figura, deixe-os então em Redefinidos. Se você quiser continuar a fazer seu programa, deixe-os portanto em Normais.

Como pode ser observado, o programa cria 96 novos caracteres. Porém, se esses caracteres forem utilizados, os 96 caracteres normais do micro não poderão ser vistos. Isto seria muito problemático para um jogo que utilizasse 90 novos caracteres e que além disso, no início, tivesse uma grande tela com instruções. Como exibir as instruções (que são escritas em português, espero) sem estar com a definição correta das letras? Existe, pois, uma maneira de se fazer com que o programa utilize, em determinado momento, os caracteres normais e ainda, em outro, os caracteres novos.

Para utilizar os caracteres novos:

48 Kb – POKE 23606,88 : POKE 23607,251

16 Kb – POKE 23606,88 : POKE 23607,123

Para utilizar os caracteres normais:

16 e 48 Kb – POKE 23606,0 : POKE 23607,60

COMO É FEITO O PROGRAMA

Os caracteres normais do micro ficam definidos na memória ROM, ao contrário dos caracteres gráficos de A a U, que ficam na memória RAM do equipamento, podendo ainda ser alterados. No entanto, o micro possui uma variável chamada PTBLCHR que fica localizada a partir do endereço 23606. Essa variável contém o endereço de definição dos caracteres. Se mudarmos o conteúdo dessa variável para uma área da memória RAM, o micro passará a pegar nessa área da memória a definição dos caracteres, e por esta área pertencer à memória RAM poderemos alterá-la à vontade. Observem que a definição original dos caracteres (aquela que está na ROM) não é (e não pode ser) alterada. Assim, se quisermos usar os novos caracteres definidos, colocaremos em PTBLCHR o endereço que contém os caracteres da ROM, e se quisermos usar os caracteres novos, colocaremos em PTBLCHR o endereço que contém os caracteres da RAM. Esse princípio faz com que, ao teclarmos R, apareçam os caracteres redefinidos e ao teclarmos N, apareçam os caracteres normais. A única coisa alterada foi o conteúdo de PTBLCHR.

A área da memória RAM escolhida para guardar o formato dos caracteres foi a área situada logo abaixo das definições dos caracteres gráficos, ou seja, nos micros de 16 Kb, a partir do endereço 31832; e nos micros de 48 Kb, a partir do endereço 64600. Para proteger essa área, foi então baixado o conteúdo da variável RAMTOP em 768 bytes. Assim, nem mesmo um comando NEW destruirá as definições feitas.

Deve-se lembrar que a área para programas em BASIC fica menor em 768 bytes e, assim, programas que utilizem a memória totalmente terão problemas de espaço.

Esperamos que este programa ajude-o a confeccionar belos jogos!

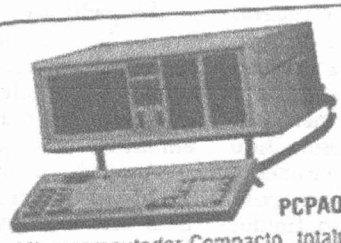
Até breve...

José Rafael Sommerfeld trabalha há mais de cinco anos com programação de microcomputadores. É colaborador de MICRO SISTEMAS desde o número 22 e autor do livro "Aplicações Pessoais para o TK85", da editora Campus.

Fernando Luiz de Carvalho e Silva possui o curso de BASIC Avançado, da UFF. É usuário dos micros CP-200 e TK90X e, atualmente, estagia no Departamento de Telecomunicações da Embratel.

COISA DE MACRO
OS MICROS DA COMPUTERWARE!

Para quem pensa grande, a Computerware tem sempre os melhores preços do mercado, com estoques suficientes para uma entrega imediata e condições de pagamento a sua escolha.
Além de descontos especiais, a Computerware oferece assistência técnica com profissionais altamente especializados, o melhor atendimento, contrato de manutenção e uma infinidade de vantagens, que fazem a nossa macro diferença.



PCPAQ.
Microcomputador Compacto, totalmente compatível com o IBM PC/XT. Duas reduzidas placas de circuito, o PCPAQ reúne no mesmo gabinete: CPU, memória inicial de 256 K, monitor de vídeo 9", 2 drives de 360 K, além de 2 slots adicionais para expansões e comunicações de dados. É incrível a versatilidade do PCPAQ, com um peso total de 13 kg, pode ser levado para onde quiser, protegido por uma bolsa almofadada. ... Consulte-nos.

UNITRON Ap II
Computador pessoal. Microprocessador 6502, 48 K de memória RAM e 12 K de memória ROM, teclado com maiúsculas e minúsculas, 8 conectores para expansões.

UNITRON T.I.
Além de reunir as características do APII, possui o Teclado Inteligente Unitron, que, diretamente, como em máquinas de escrever, fornece maiúsculas / minúsculas e acentuação da língua portuguesa.
Apresenta a característica especial de permitir a programação de cada tecla com comandos ou funções definidos pelo usuário.
Na Computerware, encontram-se também a disposição, todas as interfases / placas e periféricos do Unitron. ... Consulte-nos.

★ ASSISTÊNCIA TÉCNICA ★
(021) 262-1886
• Técnicos especializados na fábrica
• Atendimento imediato
• Contratos de manutenção

VISITE NOSSO SHOW-ROOM
Rio - INFOSHOPPING
Rua do Catete, 311

COMPUTER WARE Informática Ltda.

Rio - Av. Almirante Barroso, 91 - 11º andar - Tel.: (021) 240-7294
Rio - INFOSHOPPING - R. do Catete, 311 - (Show Room)
São Paulo - Rua Jesuíno Arruda, 797/Grupo 22 - Tel.: (011) 881-7446
Curitiba - Av. João Gualberto, 1375 - 1º andar - Tel.: (041) 253-5433

Sintevox

Paolo F. Pugno

"Só falta falar". Isto é o que muitos dizem quando podem ver com os próprios olhos ou experimentar com suas mãos o que um micro da linha ZX Spectrum é capaz de fazer. Pois bem, vamos suprir esta falta e lançar um novo desafio: agora só falta mesmo é ele pensar!

Aproveitando a saída de som de nosso micro, podemos realizar coisas muito interessantes, e uma delas é a emissão de sons que correspondam à voz humana. Isso, podem dizer alguns, demandaria tabelas quilométricas determinando a pronúncia dos vários fonemas, um trabalho decerto não fácil. Porém, há outro caminho: fazer a síntese dos sons vindos de uma fita cassete. Deste modo, nosso micro poderá falar com nossa voz ou ainda cantar uma canção (um trecho dela) com a voz do cantor preferido. Lógico, a saída de som não será de alta-fidelidade, mas é perfeitamente possível distinguir as palavras e sons, ainda por cima com a opção de variarmos a velocidade de execução. Vejamos assim como isso pode ser feito.

A PORTA 254

A porta de entrada e saída 254 (FE em hexadecimal) é uma das mais importantes do micro: quando utilizada como input, lê o teclado e contém o estado da entrada EAR do cassete, no bit 7 (D6). Portanto, o que temos a fazer é monitorar esta porta e ir anotando os valores que formos encontrando na memória. Depois faremos o caminho inverso: examinaremos a memória e, de acordo com as informações estocadas, iremos pilotar a saída de som do micro através da mesma porta 254. Quando for utilizada em output esta porta, além de controlar o som, também pilota a cor do BORDER. Portanto, devemos tomar cuidado com isto, se quisermos evitar um estranho piscar do BORDER.

O PROGRAMA

O programa em si é muito simples: uma pequena rotina em linguagem de máquina fará todo o serviço. Vou explicá-la passo-a-passo: a primeira coisa a fazer é desabilitar as interrupções; sintetizar sons com um computador já não é normalmente tarefa fácil, imagine-se, então, sendo interrompido 50 vezes por segundo! Em seguida, carregamos os registradores HL e DE com o endereço inicial do bloco de bytes reservado à memorização dos dados e com o comprimento deste bloco.

Iniciamos, portanto, um ciclo de oito repetições: lemos a porta 254, verificamos seu bit D6 e, de acordo com seu estado, setamos ou não o bit D0 do registrador C. Este registrador, então, sofrerá uma rotação à esquerda, liberando espaço para mais um bit. Um pequeno retardo também é executado.

Depois que o ciclo for executado oito vezes, o valor de C será estocado no endereço indicado por HL, que será incrementado logo após. DE será decrementado, e, se seu conteúdo for zero, o controle retornará ao BASIC. Caso contrário, começará tudo de novo, desde o ciclo de leitura da porta. Isto é feito com o digitalizador. Já, com o sintetizador, executamos exatamente a operação inversa: carregamos em C o valor do endereço indicado por HL e, examinando-o bit por bit, mandamos ou não um *click* à saída de som (bit D4) da porta 254. Essa operação é feita oito vezes, após o que incrementa-se HL, decrementa-se DE, seu conteúdo é examinado e, se for diferente de zero, volta-se ao início. Caso contrário, retorna-se ao BASIC.

Outra coisa: no começo de cada rotina, as interrupções foram desabilitadas através da instrução DI. Antes do retorno ao BASIC, é necessário que elas sejam habilitadas novamente, portanto, utilizando uma instrução EI antes do RET.

COMO UTILIZAR AS ROTINAS

As duas rotinas estão situadas acima do endereço 32768, mas não há restrições quanto às suas posições na memória, pois não contém nenhum salto absoluto. No entanto, se forem posicionadas nos primeiros 16 Kb de RAM poderão não funcionar perfeitamente, já que a ULA tem prioridades nesta área de RAM. Esta área contém o mapa de vídeo, o qual deve ser explorado periodicamente, a fim de enviar uma imagem à tevê. Se a UCP tentar acessar esta parte da memória enquanto a ULA estiver realizando este trabalho, ela terá seu clock suprimido até que a ULA o tenha terminado. Normalmente, isto não causa grandes problemas, mas, no nosso caso, significaria uma temporização irregular do programa, prejudicando o resultado final. Portanto, atenção também com isto.

Para utilizar as rotinas, você deve colocar uma fita com a mensagem gravada no seu gravador, uma música mesmo serve. Digite **RANDOMIZE USR 32768**, dê partida no gravador e aperte ENTER. O programa memoriza cerca de 10 segundos de sons, contudo isto pode ser alterado mexendo-se no valor dos registradores DE, no início de cada rotina.

A temporização também pode ser alterada, mas há algumas limitações: quanto maior for, menos memória utilizaremos, porém a qualidade também será baixa. Uma temporização pequena gasta a memória rapidamente, mas a qualidade do som de saída é bem melhor. A sugestão é que se mexa à vontade com o programa, tentando encontrar a configuração que melhor se adapte às suas necessidades. Para ouvir o que foi gravado, digite **RANDOMIZE USR 32805** e... curta um bom som!

```
   5 POKE 23609,55: RESTORE : CL
EAR 32767: FOR f=0 TO 77: READ z
: LET a=(32768+f): POKE a,z: NEX
T f
  10 DATA 243,17,24,121,33,232,1
28,14,0,6,8,219,254,203,119,40,2
,203,193,203,9,62,6,61,32,253,16
,239,113,27,35,123,178,32,227,25
1,201
  20 DATA 243,17,24,121,33,232,1
28,78,6,8,58,72,92,15,15,15,203,
121,40,2,238,56,211,254,203,9,62
,4,61,32,253,16,233,27,35,123,17
8,32,224,251,201
 100 BRIGHT 1: CLS : BORDER 5
 110 PRINT AT 0,0;"   SINTETIZAD
OR DE SONS E VOZ   ";AT 5,7;"(c)
 - Carregar som              (o)
 - Ouvir"
 120 LET h=150: LET p=INT (RND*4
)+1
 130 IF h<106 THEN GOTO 120
 140 PLOT 0,h: DRAW OVER 1; INK
p;255,0: LET h=h-1
 150 IF INKEY$="" THEN GOTO 130
 160 LET a$=INKEY$: IF a$="o" OR
 a$="O" THEN GOTO 300
 170 IF a$()"c" AND a$()"C" THEN
 GOTO 130
 180 CLS : PRINT AT 0,0; PAPER 1
; INK 5; FLASH 1;"          CAR
REGAMENTO          "
 190 PRINT AT 10,0;"A partir de
que endereco voce   quer carrega
r os sons? -)";: INPUT e: PRINT
FLASH 1;e
 200 IF e<33000 THEN CLS : GOTO
180
 210 PRINT AT 13,0;"Quantos byte
s serao reservados  para o som?
-)      ";AT 14,14;: INPUT n: PR
INT FLASH 1;n: IF n+e>65000 THEN
 GOTO 210
 220 LET b=INT (e/256): LET c=e-
b*256: POKE 32773,c: POKE 32774,
b: LET b=INT (n/256): LET c=n-b*
256: POKE 32770,c: POKE 32771,b
 230 PRINT #0;AT 1,0;"Ligue o gr
avador, digite enter": PAUSE 0
 240 BORDER 2: RAND USR 32768: B
ORDER 5
 250 CLS : GOTO 100
 300 CLS : PRINT AT 0,0; PAPER 4
; INK 7; FLASH 1;"          REP
RODUCAO          "
 310 PRINT AT 10,0;"Endereco ini
cial do bloco?      (ENTER= o ul
timo usado)          ";: INPUT a$
: IF a$="" THEN GOTO 330
 320 LET e=VAL a$: LET b=INT (e/
256): LET c=e-256*b: POKE 32773,
c: POKE 32774,b: IF e<33000 THEN
 GOTO 300
 330 PRINT a$;AT 14,0;"Numero de
 bytes do bloco?       (ENTER= o
 ultimo usado)          ";: INPUT
 a$: IF a$="" THEN GOTO 350
 340 LET n=VAL a$: LET b=INT (n/
256): LET c=n-256*b: POKE 32773,
c: POKE 32774,b
 350 PRINT a$: IF n+e>65000 THEN
 GOTO 300
 360 BORDER 4: RAND USR 32805: B
ORDER 5: GOTO 100
 500 SAVE CHR$ 18+CHR$ 1+"SINTEV
OX" LINE 1
```

Sintevox

Paolo Fabrizio Pugno cursa atualmente o primeiro ano da Faculdade de Engenharia Industrial (SP) e possui um ZX Spectrum, com o qual desenvolve programas.

Esta é uma boa hora para se aprender a armazenar e recuperar telas em alta resolução no TK90X com algumas rotinas em Assembler.

Super telas

Nelson N. S. Santos

Com duas rotinas simples e pequenas em linguagem de máquina, será possível produzir efeitos de tela impossíveis de serem obtidos em BASIC.

A primeira das rotinas arquiva uma tela completa em alta resolução (incluindo os atributos de cor), e a segunda a recupera, reimprimindo-a de maneira aparentemente instantânea.

Para arquivar a tela de alta resolução, são necessários quase 7 Kb. Assim sendo, se você tem um TK90X de 16 Kb de RAM, vão lhe sobrar apenas 2 Kb para o programa BASIC, pois 14 Kb de sua RAM serão ocupados pelos arquivos de tela – o usual e o seu novo arquivo. Se seu TK90X tem 48 Kb, então não há grandes problemas, pois vão lhe sobrar 34 Kb para seu programa BASIC.

Super telas está escrito para micros com 48 Kb de RAM. Se você tem apenas 16 Kb, consulte a lista de alterações no final desta matéria.

Começaremos o programa BASIC com **CLEAR 56999**, para reservar a área de endereço 57000 em diante para a rotina em si e para o arquivo.

A rotina de armazenamento da tela é:

```
LD HL, 16384      21 00 40      33 0 64
LD DE, 57088      11 00 DF      17 0 223
LD BC, 6912       01 00 1B       1 0 27
LDIR              ED B0        237 176
RET               C9           201
```

Vamos entender. Para usar LDIR, que é uma instrução de transferência de blocos, HL deve conter o endereço de partida. Este é 16384, o início do arquivo de tela do micro. DE deve conter o endereço de destino (impossível esquecer: DEstino); escolhemos 57088 como início do nosso arquivo (223 x 256 = 57088). BC deve conter o número de bytes a serem transferidos (fácil de guardar): BC = Byte Counter). Este número é 24 x 32 x 8 (tela de alta resolução) + 24 x 32 (atributos de tela) = 24 x 32 x 9 = 27 x 256 = 6912.

LDIR é uma das mais poderosas instruções do Z80. Ela realiza LD (DE), (HL), seguido de INC HL, INC DE e DEC BC; e repete este ciclo até que BC seja zero.

A rotina de recuperação da tela é quase igual. Veja:

```
LD HL, 57088      21 00 DF      33 0 223
LD DE, 16384      11 00 40      17 0  64
LD BC, 6912       01 00 1B       1 0  27
LDIR              ED B0        237 176
RET               C9           201
```

Observe como apenas foram invertidos os endereços de partida e de destino.

Para colocar estas rotinas na memória do computador, digite:

```
10 CLEAR 56999
20 FOR f = 57000 TO 57023
30 READ a: POKE f, a
40 NEXT f
50 DATA 33,0,64,17,0,223,1,0,27,237,176,201,33,0,
   223,17,0,64,1,0,27,237,176,201
```

Rode o programa. Nada observável acontece, mas o código de máquina está na memória do micro.

Para verificar, imprima qualquer coisa na tela e digite o comando direto **RAND USR 57000**. Limpe a tela e digite

RAND USR 57012. Você deve ver a sua tela anterior instantaneamente de volta.

Isto pode ter várias aplicações. Uma delas pode ser armazenar a tela de um jogo quando for necessário interromper para uma tela de help ou para um menu (bastante comum em adventures). Outra possibilidade é movimentar algo através de um fundo detalhado, sem alterar este fundo. A idéia é a seguinte: crie o fundo, armazene-o, imprima o seu "algo", recupere a tela de fundo (apagando o "algo"), imprima o "algo" na nova posição, e assim por diante. O efeito é veloz e literalmente impossível de ser obtido em BASIC. Como ilustração, complemente o nosso programa:

```
 60 BORDER 6: PAPER 0: CLS
 70 LET r = 1
 80 FOR f = 1 TO 181 STEP 5
 90 CIRCLE INK (RND * 6) + 1; f, r, r: LET r = r + 2
100 NEXT f
```

As linhas anteriores criam um fundo de círculos coloridos, produzindo a ilusão de um cone em terceira dimensão.

110 RAND USR 57000

Isto armazena a figura criada.

120 PAUSE 90: CLS: PAUSE 90

Dá uma pausa, limpa a tela, e a mantém limpa por alguns instantes.

```
130 FOR f = 0 TO 19
140 RAND USR 57012
150 PRINT AT 20-f, 6 + f; PAPER 6; INK 0; "TK90X"
160 NEXT f
```

O loop anterior recupera a tela com o cone, imprime TK90X, recupera a tela (apagando os caracteres), imprime TK90X mais acima e mais à direita etc.

A ferramenta que apresentamos poderá incrementar bastante seus programas.

ALTERAÇÕES PARA 16 KB DE RAM

Linha 10 – CLEAR 24999
Linha 20 – FOR f = 25000 TO 25023
Linha 50 – O número 223 deve ser trocado por 98, e o endereço de início do nosso arquivo de tela passa a ser 25088 (98 x 256 = 25088)

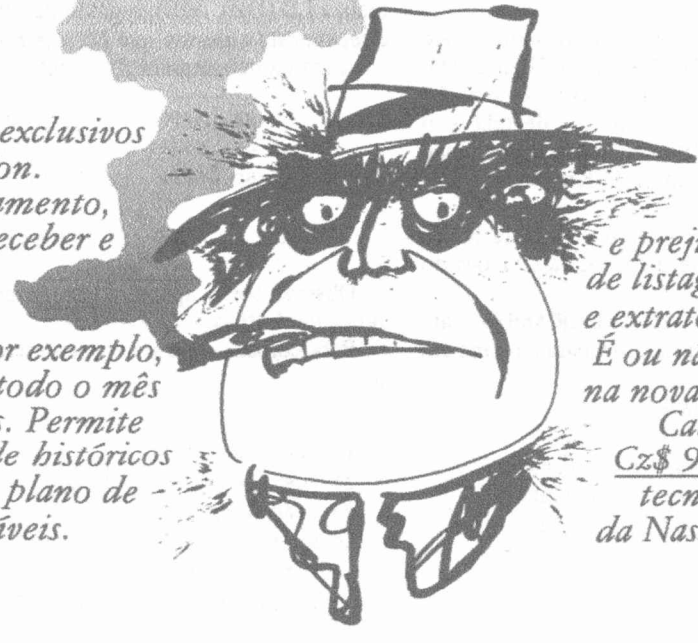
A rotina de armazenamento deve ser chamada por **RAND USR 25000**, e a de recuperar por **RAND USR 25012**.

Linha 110 – RAND USR 25000
Linha 120 – RAND USR 25012

Nelson N. S. Santos desempenha atualmente a função de Editor na editora Campus. Usuário da linha Sinclair, ele já trabalhou como Consultor Técnico na área de linguagens, sendo também autor do livro "Além do BASIC", publicado pela Campus.

É melhor você entrar na linha.

Software Nasajon compatível com a linha IBM-PC.



Chegaram os exclusivos sistemas da Nasajon.

Folha de Pagamento, Contas a Pagar/Receber e outros.

O sistema de Contabilidade, por exemplo, faz o trabalho de todo o mês em apenas 2 horas. Permite o cadastramento de históricos padronizados e de plano de contas com até 5 níveis.

Emite diário, razão, balancete, balanço, demonstração de lucros e prejuízos acumulados, além de listagem por centro de custo e extrato de contas.

É ou não é melhor você entrar na nova linha?

Cada sistema custa apenas Cz$ 9.300, incluídas aí a alta tecnologia e toda a tradição da Nasajon.

Spectrum

nasajon sistemas 5 anos de Tecnologia em Software

Av. Rio Branco, 45 Gr. 1311 - RJ
Tels.: (021) 263-1241 e 233-0615

Breve Filial em S. Paulo:
R. Xavier de Toledo, 161 - conj. 106

Rotulador de fitas

Gelson Dias Santos

O programa Rotulador de fitas cassete foi desenvolvido em um TK90X, equipado com uma impressora Alphacon 32, com o objetivo de por em ordem a minha fitoteca. Tudo começou quando o número de cassetes da mesma passou a aumentar progressivamente, dificultando a identificação do conteúdo de cada uma. Ocorre que eu tenho por hábito comprar fitas de uma única marca, conseqüentemente, por fora todas as fitas são iguais.

Além disso, eu não tinha controle sobre o que gravar em determinada fita, resultando em uma *salada* na qual cada uma continha ao mesmo tempo jogos, aplicativos, programas em desenvolvimento etc., tornando ainda mais difícil a localização dos programas.

A fim de resolver estes problemas, elaborei o programa Rotulador.

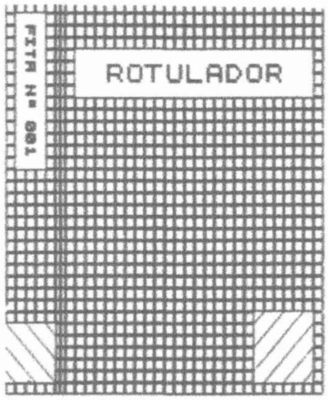


Figura 1 – capa

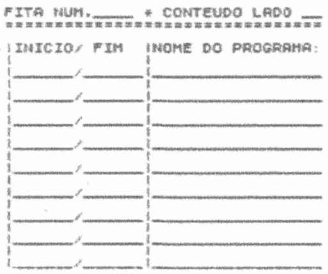


Figura 2 – lista

O PROGRAMA

Rotulador foi desenvolvido para ser usado com as impressoras Timex Sinclair 2040 ou Alphacon 32. Outras impressoras não poderão ser usadas (como a ZX Printer) devido as diferenças no formato de impressão.

A função do programa é, como o nome diz, imprimir rótulos para os cassetes. Além disso, ele gera também uma lista para facilitar o registro de cada programa gravado.

Ao ser executado, ele inicia mostrando o menu de opções. Digitando 1, o programa solicita o nome da fita, que poderá ter até 12 caracteres quaisquer, compreendidos entre o espaço (código 32) até o símbolo delta (código 127), conforme mostra a página D1 do manual do TK90X. A seguir, ele solicita o número de série da fita na fitoteca, o qual pode ir de 1 a 999. Informados estes dados, será impressa uma capa na forma da figura 1.

A opção 2 imprime uma lista padronizada para anotação dos conteúdos das fitas cassete (figura 2).

Na listagem do programa, os caracteres das linhas 260, 290, 830, 850 e 860, que estão entre aspas, são caracteres gráficos (UDGs).

Escrito inteiramente em BASIC, o programa porém utiliza alguns comandos PEEK estratégicos.

Na linha 700, inicia-se uma rotina de ampliação dos caracteres do micro, a fim de criar o título da fita. Esta rotina usa como referência a tabela de caracteres do micro, que está na ROM a partir do endereço 15616.

Na linha 160, foi utilizado endereçamento indireto via USR "a" e USR "u"-1 para que o programa possa rodar tanto em máquinas com 16 Kb quanto nas de 48 Kb de RAM.

```
   1 REM
               ATENCAO
 Todos os CHR$ GRIFADOS devem
ser digitados no modo GRAPHICS
   5 LET lp=135: DIM r$(3)
  10 CLS
  11 PRINT "********************************"
  12 PRINT "*                              *"
  13 PRINT "* ROTULADOR DE FITAS CASSETTE *"
  14 PRINT "*                              *"
  15 PRINT "********************************"
  16 PRINT "** GELSON DIAS SANTOS - 1985 **"
  17 PRINT "********************************"
 100 PRINT AT 10,2;"ESCOLHA OPCAO:"
 110 BRIGHT 1: FLASH 1: PRINT AT 13,3;"1";AT 16,3;"2";AT 19,3;"3": BRIGHT 0: FLASH 0
 120 PRINT AT 13,5;"- CAPA PARA CASSETTE";AT 16,5;"- LISTA DE CONTEUDO";AT 19,5;"- FIM DO PROGRAMA"
 130 LET o$=INKEY$: IF o$<"1" OR o$>"3" THEN GOTO 130
 140 IF o$="3" THEN STOP
 150 IF o$="2" THEN GOTO 900
 160 IF PEEK USR "a"<>255 THEN FOR r=USR "a" TO USR "u"-1: READ p: POKE r,p: NEXT r
 170 INPUT "NOME DA FITA ? (max. 12 caract.)";n$
 180 IF LEN n$>12 THEN PRINT #0;"NOME INVALIDO": SOUND 1,1: GOTO 170
 190 INPUT "NUMERO DA FITA ? (max.3 digitos)";num
 200 IF num<1 OR num>999 THEN PRINT #0;"NUMERO INVALIDO": SOUND 1,1: GOTO 190
 210 LET s$=STR$ num
 220 IF LEN s$<3 THEN LET s$="0"+s$: GOTO 220
 230 FOR r=1 TO 3: LET r$(r)=CHR$ (CODE s$(r)+104): NEXT r
 240 PRINT : PRINT "LIGUE A IMPRESSORA E TECLE ENTER": INPUT o$
 250 CLS
 260 FOR n=0 TO 21: PRINT AT n,0;"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
 270 NEXT n
 290 PRINT AT 2,1;" C ";AT 3,1;" D ";AT 4,1;" E ";AT 5,1;" F ";AT 6,1;"   ";AT 7,1;" G ";AT 8,1;" H ";AT 9,1;"   ";AT 10,1;" ";r$(1);" ";AT 11,1;" ";r$(2);" ";AT 12,1;" ";r$(3);" "
 300 PRINT AT 1,1;"   ";AT 13,1;"   "
 310 FOR q=4 TO 7: PRINT AT q,7;"                        ": NEXT q
 320 LET cp=167-(LEN n$/2)*16
 700 FOR v=1 TO LEN n$
 710 LET e=(CODE n$(v)-32)*8+15616
 720 FOR c=e TO (e+7
 730 LET l=PEEK c
 740 FOR b=1 TO 16 STEP 2
 750 IF (l/2)<>INT (l/2) THEN PLOT (cp-b),lp: PLOT (cp-b)+1,lp: PLOT (cp-b),lp-1: PLOT (cp-b)+1,lp-1
 760 LET l=INT (l/2)
 770 NEXT b
 775 LET lp=lp-2
 780 NEXT c
 790 LET cp=cp+16: LET lp=135
 800 NEXT v
 810 COPY : CLS
 830 FOR n=1 TO 4: LPRINT TAB 0;"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA": NEXT n
 840 NEXT n
 850 LPRINT "AAAAEAAAAAAAAAAAAAAAAAAAAAS S AT I TEAAAAAAAAAAAAAAAAAAAAS S AT I I BAAAAAAAAAAAAAAAAAAAS S AT I I BAAAAAAAAAAAAAAAAAAAS S AT I I BAAAAAAAAAAAAAAAAAAAS S AT I I BAAAAAAAAAAAAAAAAAAAS S S "
 860 LPRINT "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
 870 LPRINT : LPRINT
 890 RUN
 900 PRINT : PRINT "LIGUE A IMPRESSORA E TECLE ENTER": INPUT q$
 910 CLS
 920 PRINT "FITA NUM._____ * CONTEUDO LADO __"
 930 PRINT "================================": PRINT : PRINT "!INICIO/ FIM   !NOME DO PROGRAMA:"
 940 FOR x=1 TO 9
 950 PRINT "!            !"
 960 PRINT "!_____/_____!__________________"
 970 NEXT x
 980 COPY
 990 INPUT "OUTRA LISTA ? (S/N)";o$: IF o$="S" OR o$="s" THEN LPRINT : LPRINT : LPRINT : GOTO 980
1000 RUN
1020 DATA 255,129,129,129,129,129,129,255,255,153,153,153,153,153,153,255,255,255,27,27,27,27,3,0,195,195,255,255,195,195,0,0,3,3,255,255,3,3,0,0,255,255,27,27,27,255,255,0,255,255,12,24,48,255,255,0,0,0,31,31,27,31,31,0
1030 DATA 126,255,243,219,207,255,126,0,0,198,198,255,255,192,192,0,198,231,243,251,223,207,198,0,102,231,195,219,219,255,102,0,31,31,24,24,24,255,255,0,95,223,219,219,219,251,115,0,126,255,219,219,219,251,114,0,3,3,243,251,31,15,7,0,102,255,219,219,219,255,102,0,78,223,219,219,219,255,126,0
1040 DATA 1,2,4,8,16,32,64,128,128,64,32,16,8,4,2,1
```

Rotulador

APLICAÇÃO

Acredito que este programa lhe será muito útil. Sugiro que você utilize nomes de modo a estabelecer o tipo de conteúdo que cada fita possui. Eu utilizo nomes como Jogos 1, Dados 2 ou Trabalho 1. Trabalho é o nome que dou aos cassetes destinados a armazenar programas incompletos ou em desenvolvimento (também poderia ser Rascunho).

***Gelson Dias Santos** possui conhecimentos de eletrônica e é atualmente operador do computador Burroughs B6700, da Universidade Federal do Rio Grande do Sul. Ele possui um TK90X e suas áreas de interesse são a computação gráfica, teleprocessamento e lazer eletrônico.*

Modificando caracteres

Felipe Oriente

Este programa serve para modificar o banco de caracteres de microcomputadores compatíveis com o ZX Spectrum (TS 2068 e TK90X).

Para usá-lo, basta juntá-lo ao seu programa com o comando Merge ou então digitar seu programa entre as linhas 1 e 9679.

Se, entretanto, o seu micro só possuir 16 Kb de memória, então faça as seguintes alterações:

linha 1: substitua os dois 249 por 124
linha 9680: substitua 48384 por 16384
linha 9685: substitua 63744 por 31744

Felipe Oriente cursa atualmente o segundo grau no Colégio Pio XII, em São Paulo. Ele possui um ZX Spectrum há um ano, com o qual desenvolve programas.

```
   1 POKE 23607,249: GOSUB 9680:
 POKE 23607,249
9679 STOP
9680 LET b=48384: FOR i=15616 TO
 16383: POKE (i+b),PEEK i: NEXT
i:
9685 FOR i=1 TO 62: READ a$: LET
 adr=63744+8*CODE a$
9690 FOR j=0 TO 7: READ a: POKE
adr+j,a: NEXT j: NEXT i
9695 DATA "a",0,0,56,8,56,72,52,
0
9700 DATA "b",0,16,32,60,36,34,6
0,0
9705 DATA "c",0,0,24,36,32,48,12
,0
9710 DATA "d",0,8,4,60,36,68,60,
0
9715 DATA "e",0,0,48,76,88,96,60
,0
9720 DATA "f",0,28,16,24,16,16,8
,0
9725 DATA "g",0,0,56,44,68,60,4,
60
9727 DATA "h",0,32,64,88,100,66,
0,0
9729 DATA "i",0,16,0,56,16,16,8,
0
9730 DATA "j",0,8,0,4,4,4,52,24
9735 DATA "k",0,64,32,40,48,40,3
6,0
9740 DATA "l",0,32,16,16,16,20,2
4,0
9745 DATA "m",0,0,40,84,84,84,80
,0
9750 DATA "n",0,0,56,76,68,68,36
,0
9755 DATA "o",0,0,48,76,68,100,2
4,0
9760 DATA "p",0,0,120,68,72,120,
64,32
9765 DATA "q",0,0,56,44,68,60,4,
6
9770 DATA "r",0,0,12,48,32,32,16
,0
9775 DATA "s",0,0,60,64,56,8,120
,0
9780 DATA "t",0,16,56,16,16,20,2
4,0
9785 DATA "u",0,0,36,68,68,100,2
6,0
9790 DATA "v",0,0,72,68,40,40,16
,0
9795 DATA "x",0,0,72,40,16,40,36
,0
9800 DATA "y",0,0,36,36,68,56,4,
56
9805 DATA "w",0,0,8,68,84,84,40,
0
9810 DATA "z",0,0,60,72,16,36,12
0,0
9820 DATA "A",0,56,36,66,126,66,
34,0
9825 DATA "B",0,14,34,124,66,70,
28,0
9830 DATA "C",0,48,76,64,64,50,1
2,0
9835 DATA "D",0,48,76,66,66,100,
24,0
9840 DATA "E",0,60,64,120,64,66,
60,0
9845 DATA "F",0,62,64,72,112,64,
32,0
9850 DATA "G",0,56,64,76,66,50,1
2,0
9855 DATA "H",0,68,66,122,70,66,
34,0
9860 DATA "I",0,28,10,10,40,40,2
0,0
9865 DATA "J",0,4,4,2,66,66,60,0
9870 DATA "K",0,4,72,112,72,68,3
4,0
9875 DATA "L",0,64,32,64,64,66,6
0,0
9880 DATA "M",0,66,102,90,66,66,
0,0
9885 DATA "N",0,72,100,82,74,38,
16,0
9890 DATA "O",0,48,76,66,66,50,1
2,0
9895 DATA "P",0,60,66,70,120,64,
32,0
9900 DATA "Q",0,48,76,66,74,50,1
4,0
9905 DATA "R",0,60,66,70,120,68,
34,0
9910 DATA "S",0,56,64,60,2,50,12
,0
9915 DATA "T",0,124,144,144,16,1
6,8,0
9920 DATA "U",0,4,68,66,66,66,60
,0
9925 DATA "V",0,4,100,66,66,36,2
4,0
9930 DATA "X",0,72,36,24,24,36,1
8,0
9935 DATA "Y",0,2,68,40,16,16,8,
0
9940 DATA "W",0,8,68,66,66,90,36
,0
9945 DATA "Z",0,30,36,72,18,36,1
20,0
9950 DATA "1",0,24,40,40,8,10,60
,0
9955 DATA "2",0,60,66,6,56,66,12
4,0
9960 DATA "3",0,60,2,12,68,66,60
,0
9965 DATA "4",0,8,24,40,74,60,8,
0
9970 DATA "5",0,126,64,60,2,66,6
0,0
9975 DATA "6",0,60,64,120,70,66,
60,0
9980 DATA "7",0,60,66,4,8,16,16,
0
9985 DATA "8",0,60,66,60,68,34,6
0,0
9990 DATA "9",0,60,66,114,14,2,6
0,0
9995 DATA "0",0,28,100,74,82,102
,56,0
9996 RETURN
```

Caracteres

mensagem de erro

■ Em MS nº 55, na matéria "CBBS: a informação via computador", pág. 11, o telefone do ABBS Atari saiu errado. O número correto é: (011) 454-7557.

■ Em MS nº 52, no programa "Micro-golf", pág. 48, faltou a linha 125 da listagem. A linha é a seguinte:

125 GOSUB 1010

■ Em MS nº 55, no artigo "Mensagem de Erro no Color", pág. 48, na quinta linha do quarto parágrafo, houve um erro de impressão. O certo é: ". . .será exibida. Uma outra opção, ao invés de limpar a tela e listar a linha de erro, seria acionar a rotina EDIT, colocando automaticamente tal linha em modo de edição. . .".

Incremente os seus programas no TK90X, incorporando a ele algumas rotinas em BASIC e linguagem de máquina que simulam efeitos SCROLL.

SCROLL no TK90X

Nelson N. S. Santos

Um dos primeiros fatos que intriga o usuário que migrou da família TK85 para o TK90X é a ausência do comando **SCROLL**. Assim, é interessante ver as diversas maneiras de simular esta operação no TK90X. Este artigo, sem ter a pretensão de esgotar o tema, apresenta várias idéias em BASIC e linguagem de máquina.

É possível criar o efeito de **SCROLL** em BASIC, armazenando a tela numa string dimensionada para conter os 704 (22 x 32) caracteres.

Vamos usar um truque: criar uma tela randômica na linha 1, arquivando-a em t$, e não mexer mais nela. Assim, digite a linha 1 abaixo, rode-a com **RUN**, e nunca mais use **RUN** ou **CLEAR**! Para rodar os programas seguintes digite GOTO 5.

```
1 DIM t$ (704): FOR f = 1 TO 704: LET t$ (f) = CHR$ (INT (RND*133) +
32): NEXT f
```

SCROLL PARA CIMA

```
 5 BORDER 6: PAPER 6: INK 1: CLS
10 LET a$ = t$
20 FOR f = 1 TO 23
30 PRINT AT 0, 0; a$
40 LET a$ = a$ (33 TO) + "32 espaços"
50 NEXT f
```

SCROLL PARA BAIXO

Para se obter um **SCROLL** para baixo, basta alterar a linha 40 no programa acima para: 40 LET a$ = "32 espaços" + a$(TO 672).

SCROLL PARA A ESQUERDA

Os **SCROLL** para cima e para baixo são relativamente rápidos. Os laterais são possíveis, mas bastante mais lentos. Altere as linhas abaixo:

```
20 FOR f = 1 TO 33
40 FOR n = 1 TO 673 STEP 32
50 LET a$ (n TO n + 31) = a$ (n + 1 TO n + 31) + "espaço"
60 NEXT n
70 NEXT f
```

SCROLL PARA A DIREITA

Em relação ao programa anterior, altere apenas a linha 50 para: 50 LET a$(n TO n+31) = "espaço" + a$(n TO n+30).

SCROLL PARA CIMA USANDO A ROM

Podemos melhorar muito o efeito do **SCROLL** para cima usando a rotina da ROM cujo endereço é 3582. Ela desloca a tela de uma linha de caracteres para cima (ou seja, desloca a tela de baixa resolução), não afetando a posição do PRINT nem a variável do sistema SCRINC(SCR CT no ZX Spectrum), localizada no endereço 23692.

RAND USR 3582 ou LET z = USR 3582 substitui perfeitamente o esquema usando em jogos que requerem **SCROLL** contínuo: POKE 23692,255: PRINT AT 21,31.

Portanto, para efetuar um **SCROLL** para cima utilizando a rotina da ROM, mantenha as linhas 1 e 5, digitadas inicialmente; apague todas as outras linhas; e entre com o seguinte programa:

```
10 PRINT t$ ; PAUSE 90
20 FOR f = 1 TO 22
30 RAND USR 3582
40 NEXT f
```

Para rodá-lo, digite GOTO 5 e veja como melhorou em relação ao primeiro programa.

SUBSTITUINDO O CLS

É possível substituir o comando **CLS** por uma rotina em linguagem de máquina que execute um **SCROLL** para cima. Vamos explicar a rotina que executa esta tarefa, observando que os mnemônicos têm ao seu lado dois conjuntos de números. O primeiro conjunto são os códigos em hexadecimal correspondentes ao próprio mnemônico e o segundo são os valores destes códigos convertidos para o sistema decimal.

A nossa primeira tarefa é preparar o loop FOR f = 1 TO 22 . . .NEXT f.

```
LD B,22   6   16   6 22
```

A seguir, preservamos este contador colocando-o na pilha e rotulamos este ponto do programa de **A**.

```
PUSH BC   C5   197
```

Chamamos a rotina que faz o **SCROLL**.

```
CALL 3582   CD   FE   0D   205   254   13
```

Recuperamos o contador.

```
POP BC   C1   193
```

Fazemos o loop.

DJNZ A 10 F9 16 249

Vamos agora colocar a posição do PRINT em 0,0. Antes de chamar a rotina que faz isto, precisamos dar os parâmetros necessários através de BC.

Observe:

B = 24 – número da linha

C = 33 – número da coluna

Assim B = 24 e C = 33.

LD BC,6177 01 21 18 1 33 24

Chamamos agora a rotina que faz o PRINT AT. Seu endereço é 3545.

CALL 3545 CD D9 0D 205 217 13

E voltamos ao BASIC.

RET C9 201

Digite NEW e introduza este programa no micro.

```
  1 CLEAR 31999
  5 BORDER 6: PAPER 6 : INK 1 : CLS
 10 RAND : FOR f = 1 TO 22 : LET a$ = CHR$ (INT  (RND * 133) +
32) : FOR n = 1 TO 5 : LET a$ = a$ + a$ : NEXT n : PRINT a$:
NEXT f : PAUSE 90
 20 FOR f = 32000 TO 32015
 30 READ a : POKE f, a
 40 NEXT f
 50 DATA 6, 22, 197, 205, 254, 13, 193, 16, 249, 1, 33, 24,
205, 217, 13, 201
 60 RAND USR 32000
 70  PRINT "Tela limpa"
```

OUTRO SCROLL PARA A ESQUERDA

Vamos agora criar um **SCROLL** para a esquerda em alta resolução e linguagem de máquina. Primeiro, digite RAND USR 0 para limpar tudo que está na memória do seu micro e a seguir, vamos à rotina.

HL conterá o endereço do último byte da tela.

LD HL,22527 21 FF 57 33 255 87

C conterá o número de linhas da tela em alta resolução: 24 x 8 = 192.

LD C,192 0E 14 192

B será o contador para colunas: 32. Rotularemos este ponto do programa de B.

LD B,32 06 20 6 32

Usaremos OR A para zerar a *Carry Flag*.

OR A B7 183

Nesta etapa, rodaremos o conteúdo da tela para a esquerda, sendo que, através de RL(HL), cada bit é deslocado de uma posição à esquerda, o conteúdo do bit 7 é levado para o *Carry* e o conteúdo deste é colocado no bit 0. Assim, as posições à direita vão sendo sucessivamente zeradas. Rotularemos este ponto do programa de A.

RL(HL) CB 16 203 22

Voltamos para trás na tela, decrementamos os contadores e fechamos o programa, retornando ao BASIC.

DEC HL	2B	43
DJNZ A	10 FB	16 251
DEC C	0D	13
JRNZ,B	20 F5	32 245
RET	C9	201

Para demonstrar o efeito obtido, escrevemos o programa abaixo, que minhas filhas batizaram de "Passeio em noite estrelada".

```
  1 CLEAR 31999
 10 BORDER 0 : PAPER 0 : INK 6 : CLS
 20 FOR f = 1 TO 100 : PLOT INT (256 * RND), INT (176 * RND) :
NEXT f
 30 FOR f = 32000 TO 32016
 40 READ a : POKE f, a
 50 NEXT f
 60 DATA 33, 255, 87, 14, 192, 6, 32, 183, 203, 22, 43, 16, 251,
13, 32, 245, 201
 70  LET z = USR 32000
 80 PLOT 255, INT (176 * RND)
 90 GOTO 70
```

Se você desejar, poderá facilmente alterar este programa para fazer um **SCROLL** à direita, bastando modificar as linhas 60 e 80 para:

```
 60 DATA 33, 0, 64, 14, 192, 6, 32, 183, 203, 30, 35, 16, 251,
13, 32, 245, 201
 80 PLOT 0, (176 * RND)
```

A explicação para o programa gerado pela linha 60 é dada no próximo item.

OUTRO SCROLL PARA A DIREITA

Existe outro modo de obter um **SCROLL** à direita, que possibilita inclusive efetuar um **SCROLL** parcial. Para isto, utilize a linha 60 do último programa, elimine a linha 20 e digite as seguintes linhas:

```
 65 LET a = 0
 80 PRINT AT a, 0; "■"
 90 LET a = a + 1 : IF a = 22 THEN LET a = 0
100 GOTO 70
```

Vamos desmontar este programa e comentá-lo resumidamente.

LD HL,16384 21 00 40 33 0 64

Na instrução anterior, observe que HL contém o endereço do primeiro byte da tela.

LD C,192	14 C0	14 192
LD B,32	06 20	6 32
OR A	B7	183
RR(HL)	CB 1E	203 30
INC HL	23	35
DJNZ A	10 FB	16 251
DEC C	0D	13
JRNZ,B	20 F5	32 245
RET	C9	201

Não há necessidade de fazer **SCROLL** de toda a tela. Se você já reparou numa tela sendo recuperada a partir da gravação em fita, percebeu que é composta em três blocos de oito linhas cada. Altere a linha 60 para obter **SCROLL** apenas do terço inferior da tela.

60 DATA 33,0,80,14,64,6,32,183,203,30,35,16,251,13,22, 245,201

Rode o programa anterior e veja o efeito. Note que o terceiro número foi trocado de 64 para 80, e o quinto de 192 para 64.

Deixamos para você as possíveis combinações. Observe:

TERCEIRO NÚMERO	QUINTO NÚMERO
64 → início do terço superior	192 → tela completa (3/3):24 x 8
72 → início do terço médio	128 → 2/3 da tela : 16 x 8
80 → início do terço inferior	64 → 1/3 da tela : 8 x 8

Raciocine e conclua que são possíveis um total de seis combinações diferentes, incluindo as duas que apresentamos. Trabalhe e encontre-as.

Temos a certeza de que suas idéias em combinação com as ferramentas que apresentamos poderão incrementar bastante os seus programas.

■

Nelson N. S. Santos desempenha atualmente a função de Editor na Editora Campus. Usuário da linha Sinclair, já trabalhou como Consultor Técnico na área de linguagens, sendo também autor do livro "Além do BASIC", publicado pela Campus.

Uma vantagem adicional do TK90X é possuir uma tela continuamente em alta resolução. Estes programas geram interessantes efeitos na tela.

Efeitos na tela do TK90X

Paolo F. Pugno

Acredito que todos os usuários de micros compatíveis com o ZX Spectrum já perceberam que é uma grande vantagem o fato da tela do equipamento ser gerenciada permanentemente em alta resolução gráfica.

Na verdade, não existe um modo texto, pois as rotinas de impressão de um caráter no vídeo, contidas na ROM, praticamente o desenham no arquivo de tela, *pokeando* os bytes relativos ao seu desenho. Essa é uma vantagem que não é encontrada em muitos equipamentos de maior porte. Alguém aí já viu algum outro micro desenhando o gráfico da função seno sobre a listagem, e, ainda por cima, em modo OVER, fazer um OR-EXCLUSIVO por onde passa?

A memória de vídeo é mapeada de uma forma que pode parecer no mínimo estranha aos iniciantes. A tela está dividida em três blocos de oito linhas de texto cada. Um bloco tem 2048 bytes e está montado da seguinte maneira: primeiro, os bytes relativos à primeira linha de pixels de todos os caracteres do bloco; depois, aqueles relativos à segunda linha de pixels de todos os caracteres; a seguir, os da terceira linha etc. até chegar à oitava linha (um caráter é definido por oito bytes). Essa formatação é percebida facilmente quando carrega-se uma tela sob a forma LOAD "nome" SCREEN$, ou então rodando o seguinte programinha:

```
10 FOR F=0 TO 2
20 FOR G=0 TO 2047
30 POKE (16384+F*2048+G),255
40 NEXT G
50 NEXT F
```

Concluímos, então, que a memória de vídeo tem 6 Kb (três blocos de 2 Kb).

Posicionado logo após o arquivo de tela, encontra-se o arquivo de atributos, o qual contém 768 bytes (32 x 24 caracteres). Cada byte contém o código de atributo para cada caráter da tela. Podemos pesquisar o conteúdo desses bytes através da função ATTR.

O comando **PRINT ATTR** (linha, coluna) retorna um número que pode variar de 0 a 255. Esse número é constituído assim: Cor do INK + 8 x cor do PAPER + 64 se BRIGHT 1 ou + 128 se FLASH 1.

Desta forma, para INK 7, PAPER 1, BRIGHT 0 e FLASH 1, temos: 7 + 8 x 1 + 0 + 128 = 143.

Ao contrário do que muitos possam pensar, existe a possibilidade de se aumentar a resolução colorida do micro, expandindo o arquivo de cores para 6 Kb e definindo-se um atributo para cada byte da tela gráfica, e não para um conjunto de 8 bytes. Desta forma, o ZX Spectrum quase alcança a resolução gráfica de um MSX (ficam faltando algumas cores)! Mas isto é assunto para mais tarde...

Preparei algumas rotinas em linguagem de máquina que agem diretamente sobre a tela e que, acredito, farão a felicidade dos ainda iniciantes na programação Assembler.

SUPER-CLS

Este programa (listagem 1) produz um interessante efeito audiovisual toda vez que é chamado. Ele realiza um CLS por rotação de bits; apanhando cada byte do arquivo de tela e fazendo uma rotação à direita (pode ser também à esquerda), enviando a seguir o resultado à tela e à porta I/O 254. Esta porta controla, entre outras coisas:

Bits de 0 a 3 – cor do BORDER

Bit 4 – liga/desliga o alto-falante

Além disso, ela também controla a saída MIC e é utilizada com o microdrive.

O efeito do Super-CLS depende do que há na tela. Se esta estiver vazia, não haverá muita graça. No fim, antes de devolver o controle ao BASIC, ela chama a rotina de CLS da ROM, para atualizar **PAPER**, **INK**, **BRIGHT** e **FLASH**, além de posicionar a próxima impressão em (0,0).

Sua execução é no endereço 30000, mas o programa é realocável (veja a listagem).

COLOR

Este interessante programa (listagem 2) atua somente sobre os atributos, inclusive sobre as duas últimas linhas da tela.

Para ilustrar sua utilidade, vou dar um exemplo: tenho um certo desenho todo colorido na tela e, em certo momento, desejaria trocar tudo o que for PAPER 5 por PAPER 4, sem, porém, apagar o desenho, deixando intactos os outros atributos. Ou então gostaria de ver como ficaria este mesmo desenho em BRIGHT 1, sem apagá-lo e fazê-lo de novo. Bem, Color faz exatamente isto.

Para que o programa saiba o que tem de fazer, ele precisa saber primeiro quais atributos ele vai procurar. Isso poderia ser feito através de POKEs, o que tornaria o programa, porém, de certa forma deselegante. Escolhi, então, um meio alternativo: as informações são passadas através de uma string, no caso Z$. Vou explicar a sintaxe: LET Z$="lista de comandos": RAND USR 32200.

A lista de comandos é montada de forma que se quisermos mexer com INK ou PAPER a sintaxe é: INK vn ou PAPER vn; onde v=número da cor que será procurada (a cor velha) e n=número da nova cor.

Assim, se quisermos que tudo o que tenha INK 3 seja mudado para INK 7, a sintaxe será: INK 37. Para o PAPER vale o mesmo formato.

Para BRIGHT e FLASH, a sintaxe é: BRIGHT n ou FLASH n; onde n pode ser 0 ou 1 (lógico).

Vamos então montar a string, supondo que eu queira que tudo vá para FLASH 1: LET Z$="FLASH 1": RAND USR 32200

Para FLASH 0: LET Z$="FLASH 0": RAND USR 32200.

O que tiver PAPER 6 deve mudar para PAPER 2: LET Z$ ="PAPER 62": RAND USR 32200.

Você não deve digitar os comandos **PAPER** e **BRIGHT** caráter a caráter, e sim **PAPER, BRIGHT, INK** e **FLASH**, ou seja, symbol shift em modo extendido. Além disso, mais uma surpresa: você não precisa chamar a rotina para cada mudança que vai fazer, pois se for necessário fazer várias mudanças seguidamente, escreva os comandos um atrás do outro, da seguinte forma: LET Z$="PAPER 23 INK 45 BRIGHT 0 - FLASH 1": RAND USR 32200.

Desta forma, você vai fazer todas essas mudanças de uma só vez. Caso não tenha sido definida a variável Z$, acontecerá um erro de variável inexistente e se algum parâmetro foi escrito na forma errada, ocorrerá um erro do tipo Argumento Inválido, do mesmo modo que aconteceria em BASIC.

Digite a listagem 2 e a seguir o comando **RUN**. Se tudo estiver correto, você poderá então salvá-lo em fita, e depois dar um **NEW**, pois ele encontra-se protegido, acima do Ramtop. Experimente então digitar e rodar o seguinte programa:

```
 1 REM exemplo
10 FOR i = 0 TO 21
20 PRINT AT i,0;PAPER (RND*8); INK (RND*8);"ABCDEFGHIJKLMNOPQ
   RSTUVWXYZ012345"
30 NEXT i
40 LET Z$= "PAPER 01 PAPER 12 PAPER 23 PAPER 34 PAPER 45
   PAPER 56 PAPER 67"
50 RAND USR 32200
60 LET Z$= "INK 76 INK 65 INK 54 INK 43 INK 32 INK 21 INK 10"
70 RAND USR 32200
80 LET Z$= "BRIGHT 1" : RAND USR  32200
90 PAUSE 0: LET Z$= "FLASH 1" : RAND USR  32200
100 PAUSE 0: LET Z$= "BRIGHT 0 FLASH 0" : RAND USR  32200
```

Você poderá descobrir muitas utilidades para este programa, é só usar a imaginação!

INVERTE TELA

A listagem 3 apresenta o programa Inverte Tela. Seu funcionamento é muito simples: cada vez que é chamado, ele realiza uma inversão da tela no sentido horizontal, isto é, o que estava à esquerda vai para a direita, e vice-versa. Essa inversão é feita a nível de pixels, portanto, se, por exemplo, listarmos algum

COMPUTAÇÃO É COM A CAMPUS

LANÇAMENTOS

1. DICIONÁRIO ENCICLOPÉDICO DE INFORMÁTICA. *Fragomeni, A.H.* — **Cz$ 498,00**
2. C — A LINGUAGEM DE PROGRAMAÇÃO. *Kernighan, B.W. e RITCHIE, D.M.* — **Cz$ 165,00**
3. PLANILHAS ELETRÔNICAS: COMO USÁ-LAS. *Berry, T.* — **Cz$ 145,00**
4. VIDEO GAMES. *Ehrlich, M.S.* — **Cz$ 95,00**
5. SUB-ROTINAS BASIC PARA CP500 E COMPATÍVEIS (TRS-80). *Sinclair, I.* — **Cz$ 69,00**
6. PC ASSEMBLER. *Quadros, D.G.* — **Cz$ 79,00**
7. MANUTENÇÃO DE MICROS (INCLUI PROJETOS). *Costa, C.* — **Cz$ 119,00**
8. LISP PARA MICROS. *Oakey, S.* — **Cz$ 129,00**
9. CIRCUITOS SINCLAIR. *Santos Lima, D.* — **Cz$ 75,00**
10. BANCO DE DADOS PARA TK90X. *Rodrigues, M.S.* — **Cz$ 65,00**
11. BASIC PARA MICROS. *Watt, S. e Mangada, M.* — **Cz$ 59,90**

NOVIDADES 86

12. VISICALC: Guia do Usuário Brasileiro. *Alcantara, R.B. e Alcantara, P.M.* — **Cz$ 86,00**
13. VISITREND/VISIPLOT: Guia do Usuário Brasileiro. *Christmann, R.U.* — **Cz$ 67,00**
14. COMO PROGRAMAR SEU PC. *Hartnell, T.* — **Cz$ 61,00**
15. MUMPS. *Lobo, M.P.C.* — **Cz$ 73,00**
16. PASCAL PARA MICROS. *James, M.* — **Cz$ 92,00**
17. TÉCNICA DE GERENCIAMENTO DE ARQUIVOS. *Claybrook, B.Y.* — **Cz$ 155,00**
18. LOGO: INTRODUÇÃO AO PODER DO ENSINO ATRAVÉS DA PROGRAMAÇÃO. *Goodyear, P.* — **Cz$ 98,00**
19. PROJETO ESTRUTURADO DE SISTEMAS. *Stevens, W.P.* — **Cz$ 110,00**
20. INTELIGÊNCIA ARTIFICIAL EM BASIC. *James, M.* — **Cz$ 112,00**
21. MANUAL DE LINGUAGEM C. *Hancock, L. e Krieger, M.* — **Cz$ 109,00**
22. INFORMÁTICA: UMA INTRODUÇÃO. *Velloso, F.C.* — **Cz$ 95,00**

Editora Campus

Rua Barão de Itapagipe 55 Rio Comprido CEP 20261
Telefone: (021) 284 8443 · Telex: (00038) 021-32606 EDCP

☐ Anexo Cheque nº..... do Banco..... nominal à Editora Campus Ltda., livre de despesas postais.

☐ Desejo receber o maior e melhor catálogo de computação da América Latina — CAMPUSWARE.

Desejo receber os livros assinalados abaixo:

1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22

Nome:.....

Endereço:.....

CEP .....Cidade .....Estado .....

Data...../...../.....

MS/SET/86

Listagem 1

```
   1 REM ************************
   2 REM **                    **
   3 REM **     SUPER CLS      **
   4 REM **      PFP 1986      **
   5 REM ************************
   6 REM
  10 DATA 58,72,92,245,6,8,17,0,
24,33,0,64,126,211,254,203,30,35
,27,122,179,32,245,16,237,241,31
,31,31,211,254,205,107,13,201
  20 CLEAR 29999: RESTORE : FOR
f=30000 TO 30034: READ a: POKE f
,a: NEXT f
  30 REM
  40 REM Esta rotina e completa-
         mente relocavel. Para
         alterar seu endereco
         inicial, basta colocar
         seu novo valor na linha
         20, alterando tambem o
         CLEAR de forma adequada
  50 REM
  60 REM Para executar a rotina,
         basta executar um RAND
         USR xxxx , onde xxxx e
         o endereco inicial.
```

Listagem 2

```
   1 REM ************************
   2 REM **                    **
   3 REM **       COLOR        **
   4 REM **                    **
   5 REM ************************
  10 DEF FN M(A$)=CODE A$-48-7*(
A$>"9"): DEF FN N(A$)=16*FN M(A$
(1))+FN M(A$(2))
  20 CLEAR 32199: LET E=32200: L
ET T=0
  30 FOR F=120 TO 140 STEP 10
  40 READ A$: LET T=VAL A$( TO 5
): LET A$=A$(6 TO ): IF LEN A$/2
<>INT (LEN A$/2) THEN  PRINT AT
10,0;"ERRO NO NUMERO DE CARACTER
ES""LINHA ";F: STOP
  50 POKE E,FN N(A$): LET T=T-PE
EK E: LET E=E+1
  60 LET A$=A$(3 TO ): IF A$<>""
 THEN  GOTO 50
  70 IF T THEN  PRINT AT 10,0;"E
RRO NA LINHA ";F: STOP
  80 PRINT AT 10,0;"LINHA ";F;"
OK"
  90 NEXT F
 100 SAVE "COLOR"CODE 32200,311
 113 REM
 114 REM
 115 REM          DADOS
             LETRAS EM MAIUSCULO
 116 REM
 117 REM
 120 DATA "120742A485C7EFE802002
CF01FE5A2806CD8819EB18EF23462323
E5C57EFED8CAFE7DFEDCCA377EFEDACA
707EFED9CA847ECF09E11910E3C9237E
FE302807FE312819C3F87E1100032100
58C80E23187AB320F8C10511020OC3F9
7D1100032100580C8FE23187AB3"
 130 DATA "0960520F8C10511020OC3
F97D237EFE302807FE312819C3F87E11
00032100580C8B6231B7AB320F8C10511
0200C3F97D11000321005BCBF6231B7A
B320F8C105110200C3F97D237ECDEC7E
CB17CB17CB1732005B237ECDEC7ECB17
CB17CB1732015B110003"
 140 DATA "116442100587EE638473A
005B8020097EE6C7473A015BB0771B23
7AB320E7C10505110300C3F97D237ECD
EC7E32005B237ECDEC7E32015B110003
2100587EE607473A005B8020097EE6F8
473A015BB0771B237AB320E7C1050511
0300C3F97DFE383803C3F87EFE30DAFB
7ED630C9C1E1CF19"
```

Listagem 3

```
   1 REM ************************
   2 REM **   INVERTE TELA     **
   3 REM **      PFP 1986      **
   4 REM ************************
  10 DEF FN M(A$)=CODE A$-48-7*(
A$>"9"): DEF FN N(A$)=16*FN M(A$
(1))+FN M(A$(2))
  20 CLEAR 32199: LET E=32200: L
ET T=0
  30 READ A$: LET T=VAL A$( TO 5
): LET A$=A$(6 TO ): IF LEN A$/2
<>INT (LEN A$/2) THEN  PRINT AT
10,0;"ERRO NO NUMERO DE CARACTER
ES    DA LINHA  DATA ": STOP
  50 POKE E,FN N(A$): LET T=T-PE
EK E: LET E=E+1
  60 LET A$=A$(3 TO ): IF A$<>""
 THEN  GOTO 50
  70 IF T THEN  PRINT AT 10,0;"E
RRO NA LINHA  DATA ": STOP
  80 REM
  90 REM
 100 REM           L/M
 110 REM
 120 REM
 130 DATA "0B110F321004006C0C511
1F5B0620E5CD007E121B2310F8D1E521
005B01200OED80E1C110E3FB21005B06
10C5E506201 11F507E12231B10FAD1E5
21005B012000ED80E1C110E5C93E00C5
06084ECB111F10FBC1C9"
```

programa no vídeo e chamarmos a rotina, o resultado será um programa que só poderá ser lido se colocarmos um televisor na frente de um espelho (muito trabalhoso! Chame a rotina novamente e restabeleça a ordem normal das coisas). Concordo que esta rotina não apresenta tanta utilidade como o Color, mas será extremamente útil caso você deseje fazer o seu próprio programa gráfico.

Seu endereço de chamada também é 32200. Aliás, esse é também o endereço de chamada do Color (Super-CLS é chamado em 30000, mas é realocável). Escolhi esses endereços para dar oportunidade a quem tem um micro de 16 Kb. Para realocar o Inverte Tela, basta alterar o endereço de chamada de um call e nada mais. Quem tiver uma certa prática em Assembler poderá fazê-lo sem maiores problemas. Já o programa Color apresentará uma dificuldade maior, devido à presença de vários jumps absolutos e calls, os quais não puderam ser evitados. Mesmo aqui, a tarefa não é impossível, porém mais trabalhosa.

Bem, por enquanto é só... Use e abuse dos programas do modo que quiser e bom divertimento!

Paolo Fabrizio Pugno cursa atualmente o primeiro ano da Faculdade de Engenharia Industrial, em São Paulo, e possui um ZX Spectrum, com o qual desenvolve programas.

Quem tem tradição em software, tem tudo.

Faz a Folha de Pagamento de sua empresa, emitindo relatórios como Guia de IAPAS, Guia de FGTS, Relação de Empregados, Relação para I.R., Relação para Banco, Informe de Rendimentos, Acumulados Anuais, RAIS e Recibo de Pagamento. A folha pode ser semanal ou mensal. As tabelas são modificadas pelo próprio usuário.
Permite também, adiantamentos de salário, reajuste salarial, alterações de acumulados e outras funções que agilizem o processamento da Folha de Pagamento da empresa.

A Contabilidade de um mês em apenas 2 horas! Este Sistema permite o cadastramento de históricos padronizados e de plano de contas com até 5 níveis.
Emite Diário, Razão, Balancete, Balanço, Demonstração de Resultados, Demonstração de Lucros e Prejuízos acumulados, Listagem por centro de custo e extrato de contas, entre outras funções.

Controla o estoque de itens com Especificação, Estoque Mínimo, Unidade, Fornecedor, Localização e outras informações relacionadas no item como Custo Médio, Entradas e Saídas no período, etc.
Fornece Listagens Geral e Parcial dos produtos, Listagem Físico-Financeira, Listagem dos produtos abaixo do estoque mínimo, Lista de Preços e Etiquetas, entre outras.
Admite também, Reajuste de Preços, Alteração de Dados e Exclusão de Produtos.

A NASAJON oferece assistência técnica total, garantia permanente e mantém à sua disposição programadores e analistas para desenvolver sistemas específicos sob encomenda. Conte com a NASAJON SISTEMAS

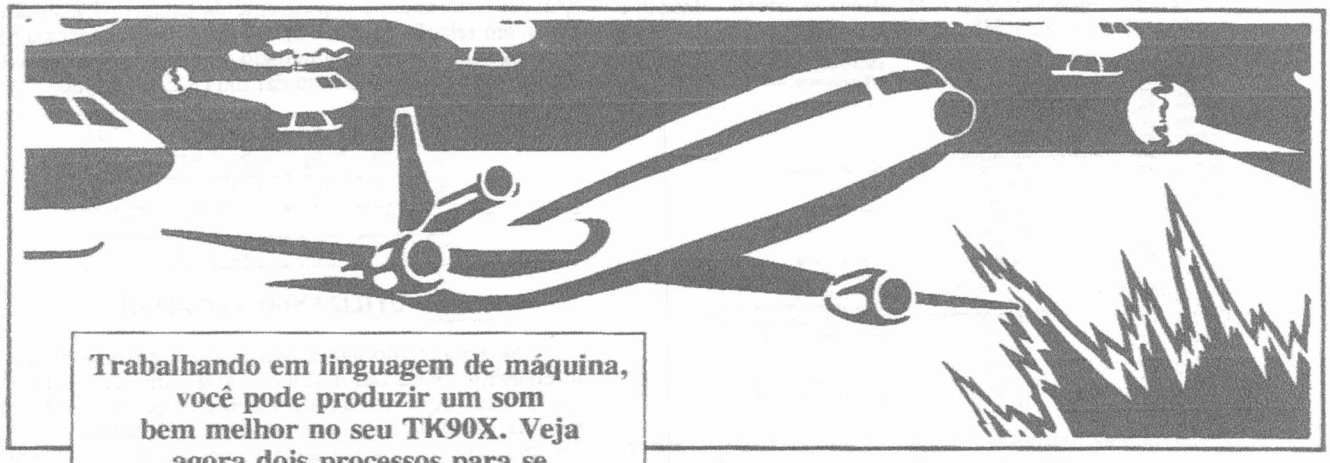
nasajon sistemas

Av. Rio Branco, 45 - Grupo 1.311
Rio de Janeiro - CEP 20.090
Tels.: (021) 263-1241 e 233-0615

Compatíveis com as linhas TRS-80 e Apple. Também disponíveis para IBM-PC. Procure-nos para maiores informações.

Empresa filiada à ASSESPRO.

speroni



Trabalhando em linguagem de máquina, você pode produzir um som bem melhor no seu TK90X. Veja agora dois processos para se criar som em Assembler e ponha sua criatividade em ação.

Som no Spectrum

Álvaro Ferreira de Freitas Borja

No TK90X podemos obter sons de duas maneiras: em programação BASIC ou em linguagem de máquina. Em BASIC, obtemos sons usando o comando **SOUND**, sendo que devemos informar a freqüência e a duração da nota desejada (isto vem explicado no manual do fabricante e já foi amplamente discutido em **MS** nº 49, no artigo "Componha no TK90X). O inconveniente é que este método gasta muita memória, além de produzir um ruído de fundo. Experimente o exemplo a seguir (listagem 1):

```
200 DATA -12,-11,-10,-9,-8,-7,-
6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7
,8,9,10,11
210 FOR I=1 TO 24
220 READ X: SOUND .25,X
230 NEXT I: RESTORE
```

Todavia, podemos contornar esses problemas acessando a rotina de som sem intermédio do BASIC (o causador de tudo) e utilizando a linguagem de máquina. Contudo, mesmo com esse recurso, deve ficar claro que não iremos conseguir os mesmos efeitos de som que obtemos com o TRS-Color, porém é imprescindível alcançarmos resultados satisfatórios, o que vai depender da criatividade de cada um.

Para produzirmos sons em linguagem de máquinas podemos utilizar dois processos: 1) usar a própria rotina da ROM ou 2) usar as portas diretamente.

UTILIZANDO A ROTINA DA ROM

Esta rotina inicia no endereço 949 (03B5h), sendo chamada por meio da instrução CALL 03B5h (CD B5 03). Antes de chamarmos a rotina, os pares de registradores HL e DE devem ser carregados com valores relacionados à freqüência e duração do som, respectivamente. A equação para achar o valor de DE é a seguinte:

DE = INT (FREQÜÊNCIA X DURAÇÃO EM SEGUNDOS)

A equação para determinar o valor de HL é:

HL = INT ((437500/FREQÜÊNCIA) – 30.125),

onde a freqüência deve ser em Hertz.

Ex.: Supondo que queremos produzir um som de freqüência igual a 500 Hz durante dois segundos.

DE = INT (500 x 2) ⟶ DE = 1000

HL = INT ((437500/500) – 30.125 ⟶ HL = 844

Em Assembler ficaria:

```
23300   21 4C 03   LD HL,844     ; Carrega HL com frequencia
23303   11 E8 03   LD DE,1000    ; Carrega DE com duracao
23306   CD B5 03   CALL 949      ; Chama rotina da ROM
23309   C9         RET
```

Para testar, usaremos o seguinte programa (listagem 2):

```
  10 DATA 33,76,3,17,232,3,205,1
81,3,201
  20 FOR I=1 TO 10
  30 READ X: POKE 23299+I,X
  40 NEXT I
  50 RAND USR 23300
```

Naturalmente que para obter apenas uma nota não haveria necessidade de recorrermos a esse método. Entretanto, quando desejamos executar algo mais completo, já se torna interessante, pois, como veremos mais adiante, é extremamente econômico em termos de tempo e memória.

A tabela a seguir indica os valores das freqüências em Hertz, bem como os valores de HL e DE para obter-se o som de cada nota durante um segundo.

Frequencia Hz	Nota	HL	DE
261.63	DO	1642	261
277.18	DO#	1548	277
293.66	RE	1459	293
311.13	RE#	1406	311
329.63	MI	1297	329
349.23	FA	1222	349
369.99	FA#	1152	369
392.00	SOL	1085	392
415.30	SOL#	1023	415
440.00	LA	964	440
466.16	LA#	908	466
493.88	SI	855	493

Obs.: Para se conseguir uma nota em escala acima e abaixo, basta multiplicar ou dividir, respectivamente, por dois, o valor da freqüência da referida nota.

O próximo programa (listagem 3) possibilita tocarmos uma seqüência de notas que estejam previamente armazenadas na memória (sendo aconselhável não executar notas muito agudas, pois estas causam um estalo no auto-falante que chega a incomodar, o que já não ocorre com as notas mais graves. Este

fenômeno é conseqüência do tipo de onda gerada pelo micro, como veremos mais adiante). Para carregá-lo, utilize o Monitor Assembler publicado em MS nº 51, página 10.

```
23300  01 00 00      LD BC,0000h     ;BC=end. da primeira nota
23303  21 00 00      LD HL,0000h     ;HL=numero de notas
23306  E5        TOCA PUSH HL
23307  0A             LD A,(BC)
23308  6F             LD L,A
23309  03             INC BC
23310  0A             LD A,(BC)
23311  67             LD H,A          ;HL=frequencia
23312  03             INC BC
23313  0A             LD A,(BC)
23314  5F             LD E,A
23315  03             INC BC
23316  0A             LD A,(BC)
23317  57             LD D,A          ;DE=duracao
23318  C5             PUSH BC         ;guarda end. da tabela
23319  CD B5 03       CALL 03B5h      ;toca nota
23322  C1             POP BC          ;recupera endereco
23323  03             INC BC          ;end. da proxima nota
23324  E1             POP HL          ;recupera cont. de notas
23325  2B             DEC HL
23326  7D             LD A,L
23327  B4             OR H            ;testa se chegou ao fim
23328  20 E8          JR NZ,TOCA      ;toca notas ate o fim
23330  C9             RET
```

Os valores das notas deverão ficar armazenados a partir do endereço apontado pelo par de registradores BC. Para carregar estes valores, utilize o programa abaixo (listagem 4). Tecle N para encerrar a entrada de dados e S para continuar.

```
  10 CLS : INPUT "ENDERECO ? ";E
NDI: POKE 23301,ENDI-256*INT (EN
DI/256): POKE 23302,INT (ENDI/25
6): LET NOTAS=0: LET END=ENDI
  20 INPUT "HL ";X$: IF X$="C" O
R X$="I" THEN  GOTO 70
  25 LET HL=VAL X$: RAND USR 328
0: PRINT AT 21,0;"HL = ";HL,"<C>
ORRIGIR"
  30 POKE END,(HL-256*INT (HL/25
6)): POKE END+1,INT (HL/256)
  40 INPUT "DE ";X$: IF X$="C" O
R X$="I" THEN  GOTO 70
  45 LET DE=VAL X$: RAND USR 328
0: PRINT AT 21,0;"DE = ";DE,"<I>
ERMINAR"
  50 POKE END+2,(DE-256*INT (DE/
256)): POKE END+3,INT (DE/256)
  60 LET NOTAS=NOTAS+1: LET END=
END+4: GOTO 20
  70 IF X$="C" THEN  LET NOTAS=N
OTAS-(NOTAS>0): LET END=END-4: I
F END<=ENDI THEN  CLS
  75 IF X$="C" THEN  GOTO 20
  80 POKE 23304,NOTAS-256*INT (N
OTAS/256): POKE 23305,INT (NOTAS
/256)
  90 CLS : PRINT "END.INICIAL=";
ENDI,NOTAS;" NOTAS"
 100 PRINT AT 10,2; FLASH 1;" AP
ERTE UMA TECLA PARA TOCAR "
 110 PAUSE 0
 120 RAND USR 23300
 130 STOP
```

Agora que você já está com a rotina em Assembler carregada, use novamente o programa da listagem 1 e sinta a diferença em relação ao BASIC; utilize os dados da tabela fornecida a seguir. Quando for solicitado o endereço, responda 40100, sendo que antes você deverá dar um CLEAR 40000.

HL	3314	3125	2949	2758	2604
DE	32	34	36	38	41
HL	2475	2334	2202	2070	1955
DE	43	46	49	52	55
HL	1846	1741	1642	1548	1459
DE	58	61	65	69	73
HL	1406	1297	1226	1157	1092
DE	77	82	87	92	98
HL	1030	964	909	855	
DE	103	110	116	123	

UTILIZANDO AS PORTAS

Será mostrado agora como obter sons usando as portas diretamente. Para tal, devemos enviar um sinal para a porta apropriada – que no caso do TK90X é a porta 254 (FEh) – sinal este que deverá agir somente no bit-4. Sabendo em qual bit devemos agir, temos que ter em mente que setando esse bit ligamos o auto-falante; e ressetando-o, desligamos. Fazendo o valor deste bit alternar entre 0 e 1 obteremos um ruído. Veja o exemplo:

```
10 OUT 254,16: OUT 254,0: GOTO 10
```

16 = 00010000 (bit 4 = 1)

0 = 00000000 (bit 4 = 0)

Contudo, este modo é pouco eficiente para obtermos o som de uma explosão, por exemplo. Para contornarmos esse problema devemos recorrer novamente à linguagem de máquina, utilizando a instrução OUT que tem o mesmo funcionamento da instrução em BASIC. Este programa o ajudará a compreender melhor:

A Micro's garante a garantia da Microtec, o maior fabricante de IBM/Compatible do Brasil. Só mesmo a maior revenda Microtec do país pode garantir a tranqüilidade e a satisfação que você exige.

Porque a Micro's tem o melhor departamento de assistência técnica. É a única que conta com equipes capazes de atender a qualquer chamado em até 3 horas.

E o melhor: a Micro's firma um novo contrato de garantia para seu micro quando terminar o prazo do fabricante. Mesmo que você não tenha comprado seu Microtec na Micro's.

Micro's. A melhor assistência técnica. O melhor atendimento. A melhor revenda.

Isso é garantido.

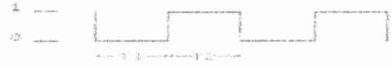
MICRO'S
Informática e Tecnologia Ltda.

RIO (021) 221-3654/S. PAULO (011) 813-9140/BRASÍLIA (061) 273-0888

```
23300  F3        DI          ;desabilita as interrupcoes
23301  0E FF     LD C,FF     ;numero de ciclos
23303  3E 00     LD A,00
23305  D3 FE     OUT (FE),A ;desl. autofalante (bit 4=0)
23307  06 C8     LD B,C8
23309  10 FE     DJNZ 23309 ;mantem desligado (T1)
23311  3E 10     LD A,10
23313  D3 FE     OUT (FE),A ;liga autofalante (bit 4=1)
23315  06 C8     LD B,C8
23317  10 FE     DJNZ 23317 ;mantem ligado (T2)
23319  0D        DEC C
23320  20 ED     JR NZ,23300
23322  FB        EI
23323  C9        RET
```

A instrução DI serve para garantir que a nossa rotina não seja interrompida a cada 1/50 de segundo, o que comprometeria o resultado.

Os dois laços de tempo que utilizam o registro **B** serve para determinar o tempo em que o auto-falante ficará ligado/desligado. Modificando esses valores, podemos determinar a tonalidade do som. Observe a figura a seguir:



Este tipo de onda se chama onda quadrada; por ela ser assim é que o som gerado pelo micro é um tanto duro, uma vez que os sons que nos parecem agradáveis (de um violão, por exemplo), têm uma onda mais suave (tipo senoidal).

Tendo compreendido como se comportam os dois ciclos, ficará fácil para o leitor perceber o comportamento de um ruído, tipo o causado pela turbina de um avião. O ruído é formado por pulsos espaçados de forma randômica, logo, é fácil de ser simulado. Abaixo temos a onda de um ruído qualquer.

Para demonstrar o que foi explicado, vamos simular o som de uma explosão (programa a seguir — listagem 6). Para tal, devemos fazer o ruído evoluir de uma tonalidade aguda até uma mais grave (esta tonalidade é obtida, diminuindo-se ou aumentando-se o tempo dos ciclos, conforme o caso), tornando a pausa entre os ciclos cada vez maior.

```
23300    F3          DI
23301    21 00 00    LD HL,0      ;base da tab. de dados
23304    0E 00       LD C,0
23306    16 1B       LD D,1Bh     ;tempo da explosao
23308    7E          LD A,(HL)    ;carrega som
23309    E6 10       AND 10h      ;afeta apenas bit 4
23311    D3 FE       OUT (FE),A   ;lig/desl.conforme o
23313    41          LD B,C       ;estado do bit 4
23314    10 FE       DJNZ 23314
23316    23          INC HL
23317    15          DEC D
23318    20 F4       JR NZ,23308
23320    0C          INC C        ;usando DEC C obtemos um
23321    20 EF       JR NZ,23306  ;som cada vez mais agudo
23323    FB          EI
23324    C9          RET
```

O registrador C é quem controlará o tempo dos ciclos, fazendo com que fiquem cada vez mais demorados, possibilitando o efeito desejado. Fazendo variar o valor inicial do registrador **D**, podemos aumentar ou diminuir o tempo da explosão. A instrução AND 10h tem como objetivo evitar que a borda da tela fique alternando de cor. O par de registradores HL é inicializado com o endereço da tabela de dados (aleatórios) para a obtenção do som. No nosso caso foi escolhida a própria ROM, podendo ser qualquer endereço, o que permite ao leitor criar sua própria tabela.

Álvaro Ferreira de Freitas Borja está cursando engenharia civil, na UFPA. Atualmente, ele trabalha como Desenhista-Projetista na Telepará, sendo também usuário dos micros TK83 e TK90X.

Com a Centraldata a entrega é imediata

253-1120 253-1120

NÃO PONHA EM RISCO O SEU COMPUTADOR, ADQUIRINDO PRODUTOS DE QUALIDADE CONSAGRADA.

MÍDIA MAGNÉTICA

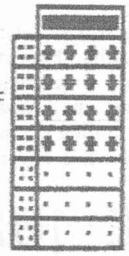
- Disketes e fitas magnéticas, marca DATALIFE VERBATIM, com 5 (cinco) anos de garantia
- Discos magnéticos, marca IMPELCO, com 1 (um) ano de garantia
- Disketes de 5 1/4" para limpeza do cabeçote de leitura e/ou gravação
- FITAS p/impressoras em geral, marca CARBOFITAS, com garantia total contra defeitos de fabricação
- Etiquetas PIMACO - PIMATAB
- Formulários contínuos e pastas
- Arquivos p/disketes com capacidade para 10 (dez) ou 100 (cem) disketes

CONDIÇÕES ESPECIAIS PARA REVENDEDORES

Suprimento é coisa séria

CENTRALDATA
Com. e Representações Ltda.

Distribuidor Autorizado:
CARBOFITAS • PIMACO • VERBATIM
Av. Presidente Vargas, 482 - Gr. 201/203
Tel.: KS (021) 253-1120 - Telex (021) 34318



Senha

Francisco Pires N. de Souza e Marcio Henrique A.-Costa

Para os aficcionados por jogos de adivinhação, apresentamos a seguir dois programas do tipo *Senha*, adequados aos micros compatíveis com o ZX Spectrum (TK90X).

Observe que, durante a fase de digitação, os caracteres grifados devem ser digitados no modo GRAPHICS.

SENHA I

Esta é uma adaptação do famoso jogo **Senha** da Grow. Neste caso, quem esconde a seqüência de cores é o micro e o usuário tentará adivinhar (jogando com inteligência e sorte) a sucessão escondida.

Para ocultar a seqüência ou para tentar adivinhá-la, é possível usar sete cores (já que a preta é a cor de fundo).

Depois da aparição do nome do jogo, o micro esperará que você entre com a sua seqüência — não vale cor repetida. Logo em seguida, o equipamento irá verificá-la emitindo o seguinte código:

S — Cor certa na hora errada
S com FLASH — Cor certa na ordem certa

Nota: A ordem em que aparecem as letras "S" não tem relação com a ordem de sua seqüência.

Se você conseguir acertar a sucessão de quatro cores escondida pelo micro, ganhará o jogo. Mas, se o computador conseguir vencê-lo, não se preocupe, pois será possível tentar de novo. Caso você ganhe ou perca, o micro lhe mostrará a seqüência correta.

Depois de digitado o programa, grave-o dando o comando direto: SAVE "senha" LINE 1, e bom divertimento!

SENHA II

Aqui o objetivo é descobrir uma seqüência de cores ou letras formada pelo computador. Existem ao todo oito cores e letras diferentes, sendo que apenas cinco irão compor o código por escolha puramente aleatória. Você terá 12 chances para descobrir a senha, tendo inclusive as suas tentativas para entrar com a possível combinação limitadas pelo tempo estipulado no início do jogo; o qual é determinado em segundos (no mínimo cinco e no máximo, 120).

Caso o tempo se esgote e você tenha respondido com uma seqüência perderá »

```
  1 REM +++FRAPI  SOFT+++
  2 REM    +++SENHA+++
  5 REM -FRAncisco PIres-1986
 10 PAPER 0: BORDER 0: INK 7: C
LS
 15 REM DESENHA AS LETRAS
 20 INK 6: BRIGHT 1: PLOT 10,17
5: DRAW -10,0: DRAW 0,-10: DRAW
10,0: DRAW 0,-10: DRAW -10,0
 22 SOUND 0,1,12
 25 INK 2: PLOT 30,175: DRAW -1
0,0: DRAW 0,-20: DRAW 10,0: PLOT
 25,165: DRAW -5,0
 27 SOUND 0,1,13
 30 INK 4: PLOT 40,175: DRAW 0,
-20: PLOT 40,175: DRAW 13,-21: D
RAW 0,21
 32 SOUND 0,1,14
 35 INK 7: PLOT 64,175: DRAW 0,
-20: PLOT 78,175: DRAW 0,-20: PL
OT 64,165: DRAW 13,0
 37 SOUND 0,1,15
 40 INK 5: PLOT 86,155: DRAW 9,
20: DRAW 9,-20: PLOT 90,165: DRA
W 10,0
 42 SOUND 0,1,16
 45 BRIGHT 0
 46 DIM J(5): DIM K(5): DIM A(5
): DIM L(5)
 47 GOSUB 2000
 48 REM ESCONDE SEQUENCIA
 50 LET L(1)=INT (RND*7)+1
 60 LET L(2)=INT (RND*7)+1: IF
L(2)=L(1) THEN GOTO 60
 70 LET L(3)=INT (RND*7)+1: IF
L(3)=L(2) OR L(3)=L(1) THEN GOTO
 70
 80 LET L(4)=INT (RND*7)+1: IF
L(4)=L(1) OR L(4)=L(2) OR L(4)=L
(3) THEN GOTO 80
 90 LET a=9
 100 LET i=a: LET i=9
 110 PRINT #0;AT 1,4; INK 5;"ENT
RE COM SUA SEQUENCIA"
 120 REM ENTRADA DA SEQUENCIA
 150 FOR F=1 TO 4
 160 IF INKEY$="" THEN GOTO 160
 162 LET a$=INKEY$
 165 IF a$<"1" OR a$>"7" THEN GO
TO 160
 170 LET A(F)=VAL a$
 175 IF F=2 THEN IF A(F)=A(F-1)
THEN GOTO 160
 175 IF F=3 THEN IF A(F)=A(F-1)
OR A(F)=A(F-2) THEN GOTO 160
 177 IF F=4 THEN IF A(F)=A(F-1)
OR A(F)=A(F-2) OR A(F)=A(F-3) TH
EN GOTO 160
 180 PRINT AT M,I; INK A(F);"■"
 GOSUB 1500
 200 LET I=I+2
 400 NEXT F
 500 REM VERIFICA SEQUENCIA
 501 OVER 1: PRINT #0;AT 1,4;"
                "
 505 LET S5=0
 507 LET J=M-1: LET K=18
 510 FOR N=1 TO 4
 530 IF A(N)=L(N) THEN PRINT AT
J,K; FLASH 1; INK 7; PAPER 2;"S"
: LET K=K+1: LET S5=S5+1: IF S5=
4 THEN GOTO 3000
 535 IF K=20 THEN LET J=J+1: LET
 K=18
 537 NEXT N
 540 IF A(1)=L(2) OR A(1)=L(3) O
R A(1)=L(4) THEN PRINT AT J,K; I
NK 7; PAPER 2;"S": GOSUB 700
 550 IF A(2)=L(1) OR A(2)=L(3) O
R A(2)=L(4) THEN PRINT AT J,K; I
NK 7; PAPER 2;"S": GOSUB 700
 560 IF A(3)=L(1) OR A(3)=L(2) O
R A(3)=L(4) THEN PRINT AT J,K; I
NK 7; PAPER 2;"S": GOSUB 700
 570 IF A(4)=L(1) OR A(4)=L(2) O
R A(4)=L(3) THEN PRINT AT J,K; I
NK 7; PAPER 2;"S": GOSUB 700
 650 LET M=M-2: LET I=9: OVER 0
 660 IF M=7 THEN GOTO 3000
 670 GOTO 110
 700 REM POSICAO DA RESPOSTA
 701 LET K=K+1: IF K=20 THEN LET
 J=J+1: LET K=18
 710 RETURN
1500 SOUND 0.004,-15: SOUND 0.00
4,-20: SOUND 0.004,20: SOUND 0.0
04,25
1510 RETURN
2000 REM DESENHA TABULEIRO
2010 PRINT INK 5;AT 5,6;"■■■■■■
■■■■"
2020 FOR F=8 TO 19: PRINT AT F,6
; INK 5;"■";AT F,17;"■": NEXT F
2030 PRINT INK 5;AT 20,8;"■■■■■
■■■■"
2040 PRINT INK 5; BRIGHT 1;AT 7,
16;"■■■": FOR F=8 TO 19: PRINT A
T F,20; BRIGHT 1; INK 5;"■": NEX
T F: PRINT INK 5; BRIGHT 1;AT 20
,18;"■■■"
2050 FOR F=8 TO 18 STEP 2: PRINT
 AT F,9; INK 4; OVER 1;"■■■■■■■■
"; PAPER 5; INK 6;"■■■■": PAPER 0;
 INK 4;"■": NEXT F
2100 PRINT INK 7; PAPER 7; BRIGH
T 1;AT 5,8;"■■■■■■■■■■": PRINT P
APER 7; INK 7; BRIGHT 1;AT 6,8;"
■■■■■■■■■■";AT 8,17;"■■";AT 7,8;"■■■■■■■■■■"
2400 REM CARACTER DA "U.D.S."
2500 FOR F=0 TO 7: READ A: POKE
USR "a"+F,A: NEXT F
2510 DATA BIN 00000000,BIN 00111
100,BIN 01111110,BIN 01111110,BI
N 01111110,BIN 01111110,BIN 0011
1100,BIN 00000000
2520 RETURN
3000 REM ROTINA DE PARABENS
3010 IF S5=4 THEN FOR F=0 TO 12:
 SOUND 0.04,F+5: NEXT F: FOR F=1
2 TO 0 STEP -1: SOUND 0.04,F+5:
NEXT F
3015 IF S5=4 THEN FOR f=1 TO 7:
SOUND 0.1,f+INT (RND*4): PRINT A
T 2,19; INK f;"Voce venceu!": NE
XT f
3020 IF M=7 THEN FOR F=20 TO 30:
 SOUND 0.04,-F: NEXT F
3030 PRINT AT 6,a; INK L(1);"■";
AT 6,A+2; INK L(2);"■";AT 6,A+4;
 INK L(3);"■";AT 6,A+6; INK L(4)
;"■"
3040 PRINT AT 21,6; BRIGHT 1; IN
K 4;"Quer jogar de novo ? (S/N)"
3050 IF INKEY$="" THEN GOTO 3050
3060 IF INKEY$="S" OR INKEY$="s"
 THEN RUN
3070 IF INKEY$="N" OR INKEY$="n"
 THEN STOP
3080 GOTO 3050
```

Senha I

```
   5 RAND : UDG 0
   7 BORDER 4: PAPER 7: INK 0
  10 CLS : DIM C(5): DIM S(5): D
IM S$(5): DIM K(5): DIM A$(5): D
IM P(5)
  20 GOSUB 3500
  30 PRINT AT 5,9;"C D E K I  MM
M"
  40 PRINT AT 8,17;"By MErcio"
  50 PRINT AT 12,2;"[": FLASH 1;
"1"; FLASH 0;"] - Cores      [";
FLASH 1;"2"; FLASH 0;"] - Letras
"
  53 PLOT 0,0: DRAW 255,0: DRAW
0,175: DRAW -255,0: DRAW 0,-175
  55 SOUND .25,15: SOUND .25,15
  60 LET Z$=INKEY$: IF Z$="" THE
N GOTO 60
  70 IF Z$<"1" OR Z$>"2" THEN GO
TO 60
  75 PAUSE 100
  80 LET OP=VAL Z$: PRINT AT 18,
10;"Boa Sorte !"
  90 PAUSE 100
 110 LET R=8: GOSUB 1000: FOR I=
1 TO 5: LET C(I)=K(I)
 120 RESTORE 600: FOR J=1 TO C(I
)
 130 READ Z$: NEXT J
 140 LET S$(I)=Z$: NEXT I
 170 GOSUB 1500: LET TT=0
 180 IF OP=1 THEN GOSUB 2000: GO
TO 200
 190 CLS : PRINT AT 7,0;"[P] Pre
to""[V] Vermelho""[M] Marrom""
"[L] Lilas""[R] Rosa""[A] Azul
""[D] Verde""[E] Amarelo"
 200 PRINT AT 2,0; FLASH 1;"TEMP
O :";AT 2,20;"TOTAL :"; FLASH 0;
" ";TT;AT 4,0; FLASH 1;"LIMITE:"
; FLASH 0;" ";TM;#0;TAB 8;"C D E
K T  MMM"
 210 FOR L=5 TO 16: LET M=0: LET
MM=0: LET T=0: LET C=14: LET I=
1: LET X$=""
 220 PRINT AT 0,0;"Digite as 5 c
ores do codigo :";AT 2,8;" "
 230 LET Z$=INKEY$: IF INT T>=TM
THEN LET TT=TT+TM: PRINT AT 2,2
8;TT: SOUND .25,15: GOTO 470
 240 IF Z$="" THEN LET T=T+.043:
PRINT AT 2,8;INT T: GOTO 230
 245 SOUND .025,30: IF X$=Z$ THE
N LET Z$="": LET T=T+.05: GOTO 2
40
 250 LET X$=Z$: LET A$(I)=Z$
 260 IF OP=1 THEN IF Z$<"0" OR Z
$>"7" THEN GOSUB 5500: GOTO 230
 265 IF OP=1 THEN GOTO 280
 270 IF OP=2 THEN GOSUB 5000
 275 IF M=1 THEN GOTO 230
 277 GOTO 300
 280 LET G$="A": LET Z=VAL Z$: G
OSUB 2500: IF Z=7 THEN LET Z=0:
LET G$="B"
 290 PRINT AT L,C; INK Z;G$: GOT
O 310
 300 PRINT AT L,C;A$(I)
 310 LET C=C+2: LET I=I+1
 320 IF C<>24 THEN GOTO 230
 330 LET TT=TT+INT T: PRINT AT 2
,28;TT
 340 LET I=0: LET M=0
 350 FOR J=1 TO 5: LET I=I+1
 360 FOR K=1 TO 5
 370 IF A$(J)=S$(K) THEN GOTO 39
0
 380 LET M=M+1: GOTO 410
 390 IF J=K THEN LET P(I)=2: GOT
O 410
 400 LET P(I)=1: LET K=5
 410 NEXT K
 420 IF M=5 THEN LET P(I)=0
 430 LET M=0: NEXT J
 440 LET R=5: GOSUB 1000
 450 PRINT AT L,26;P(K(1));P(K(2
));P(K(3));P(K(4));P(K(5))
 460 IF P(1)=2 AND P(2)=2 AND P(
3)=2 AND P(4)=2 AND P(5)=2 THEN
GOTO 560
 470 NEXT L: FOR I=20 TO 1 STEP
-1: SOUND .125,I-15: NEXT I
 480 PRINT AT 18,0;"Sinto muito,
voce perdeu."
 490 IF OP=1 THEN LET Z$="A comb
inaURo era : ": GOSUB 3000: GOTO
 510
 500 PRINT "A combinaURo era : "
;S$(1);" ";S$(2);" ";S$(3);" ";S
$(4);" ";S$(5)
 510 PRINT AT 21,0;"Deseja jogar
 novamente (S/N) ?"
 520 LET Z$=INKEY$: IF Z$="" THE
N GOTO 520
 530 IF Z$="S" THEN RUN 10
 540 IF Z$<>"N" THEN GOTO 520
 550 BORDER 7: CLS : STOP
 560 PRINT AT 18,8; FLASH 1;"VOC
E VENCEU !!!"
 570 FOR J=1 TO 4: FOR I=1 TO 20
: SOUND .025,I+5: NEXT I: NEXT J
 580 IF OP=1 THEN LET Z$="Eis a
combinaURo : ": GOSUB 3000: GOTO
 510
 590 PRINT AT 20,0;"Eis a combin
aURo : ";S$(1);" ";S$(2);" ";S$(
3);" ";S$(4);" ";S$(5): GOTO 510
 600 DATA "P","V","M","L","R","A
","D","E"
 610 DATA "P0","V1","M2","L3","R
4","A5","D6","E7"
 620 DATA 0,60,126,255,255,126,6
0,0
 630 DATA 0,60,66,129,129,66,60,
0
 640 DATA 128,129,128,252,126,1,
129,126
 650 DATA 126,129,128,248,252,12
8,129,126
 660 DATA 130,193,225,177,153,14
1,135,65
 670 DATA 66,129,129,255,255,129
,129,66
 680 DATA 126,24,24,24,24,24,24,
126
 690 DATA 126,129,129,255,255,12
9,129,66
1000 LET K(1)=INT (RND*R+1)
1010 LET K(2)=INT (RND*R+1)
1020 IF K(2)=K(1) THEN GOTO 1010
1030 LET K(3)=INT (RND*R+1)
1040 IF K(3)=K(2) OR K(3)=K(1) T
HEN GOTO 1030
1050 LET K(4)=INT (RND*R+1)
1060 IF K(4)=K(3) OR K(4)=K(2) O
R K(4)=K(1) THEN GOTO 1050
1070 LET K(5)=INT (RND*R+1)
1080 IF K(5)=K(4) OR K(5)=K(3) O
R K(5)=K(2) OR K(5)=K(1) THEN GO
TO 1070
1090 RETURN
1500 SOUND .25,15: INPUT "Tempo
MEximo de Resposta : ";TM: LET T
M=INT TM
1510 IF TM<5 OR TM>120 THEN GOTO
 1500
1515 POKE 23658,8: PAUSE 100
1520 RETURN
2000 CLS : PRINT AT 7,0;"[0] Pre
to""[1] Azul""[2] Vermelho""[
3] Magenta""[4] Verde""[5] Cia
no""[6] Amarelo""[7] Branco"
2010 RETURN
2500 RESTORE 600: FOR J=0 TO Z
2510 READ Z$: NEXT J
2520 LET A$(I)=Z$: RETURN
3000 FOR I=1 TO 5: RESTORE 610
3010 FOR J=1 TO 8: READ X$
3020 IF X$(1)=S$(I) THEN LET S(I
)=VAL (X$(2)): GOTO 3040
3030 NEXT J
3040 NEXT I
3050 PRINT AT 20,0;Z$;: FOR I=1
TO 5: IF S(I)=7 THEN LET G$="B":
 LET S(I)=0: GOTO 3070
3060 LET G$="A"
3070 PRINT INK S(I);G$;" ";: NEX
T I
3080 RETURN
3500 RESTORE 620: LET Z$="a": GO
SUB 3530
3510 LET Z$="b": GOSUB 3530
3511 LET Z$="c": GOSUB 3530
3512 LET Z$="d": GOSUB 3530
3513 LET Z$="e": GOSUB 3530
3514 LET Z$="k": GOSUB 3530
3515 LET Z$="m": GOSUB 3530
3516 LET Z$="t": GOSUB 3530
3520 RETURN
3530 FOR I=0 TO 7: READ Z
3540 POKE USR Z$+I,Z: NEXT I
3550 RETURN
4000 SAVE "SENHA III" LINE 1
4010 STOP
5000 RESTORE 600: FOR Z=1 TO 8
5010 READ X$: IF X$=Z$ THEN LET
M=0: RETURN
5020 NEXT Z: LET M=1: SOUND .25,
0: LET MM=MM+1: IF MM=3 THEN RUN
 10
5030 RETURN
5500 SOUND .25,0: LET MM=MM+1: I
F MM=3 THEN RUN 10
5510 RETURN
```

Senha II

esta chance, passando para a próxima que é inteiramente independente da anterior.

Após a seleção da sua seqüência, será apresentado ao lado desta última, uma formação de números (compreendidos entre 0 e 2) referente ao resultado de sua tentativa, onde:

Arte no meio do texto

- 0 significa cor ou letra inexistente;
- 1 significa cor ou letra certa em posição errada;
- 2 significa cor ou letra certa em posição certa.

Essa formação é impressa de forma aleatória, isto é, não significa que o primeiro número (pista) se refira a primeira posição da senha. Por exemplo, na seqüência abaixo, o primeiro número (1) poderá se referir a qualquer uma das letras (como indica o tracejado), ou seja P, L, V, R ou E e não apenas a letra P.

```
PLVRE 10202
```

Caso você pressione por três vezes, uma tecla que não esteja no menu, o computador reiniciará o jogo.

O placar tempo irá indicar o tempo gasto em segundos para descobrir a senha.

Francisco Pires Nestor de Souza estuda na Escola Técnica Estadual Visconde de Mauá - RJ. Autodidata em BASIC, utiliza o seu TK90X para desenvolver programas educacionais relacionados a eletrônica, e em jogos de inteligência e ação. Francisco é o autor de Senha I.

Márcio Henrique Alexandre Costa é formado em Administração de Empresas e usuário dos micros CP500 e TK90X. Trabalha atualmente como Instrutor de Programação no Curso de Processamento de Dados Data-Rio. Márcio é o autor de Senha II.

Peek & Poke

SOFTWARE PARA CP 400 E COMPATÍVEIS

APLICATIVOS: Processador de Textos PPTEXTO c/ acentuação em português, Planilhas Eletrônicas, Bancos de Dados, Pacotes Integrados.
UTILITÁRIOS: Editor/Assembler, Expansor de Tela, Supertec, Compiladores, Copiadores, Etc.
GRÁFICOS: DUMP P-500, Esquemas Eletrônicos, Bjork Blocks, Comerciais, Graphicom e outros.
LINGUAGENS: LOGO em português, PILOT para professores e uso didático, PASCAL, CBasic.
JOGOS: Muitos títulos para mencionar aqui! Nossos catálogos mostram as telas dos jogos. Temos jogos exclusivos. Escolha melhor na Peek & Poke!

SOFTWARE PARA CP 500 / CP 300

— Processador de textos PPTEXT 500 DISCO c/ acentuação.
— Planilha Eletrônica CALC-300. Única em Z-80 para versão K-7.
— Utilitários copiadores para fita e disco.
— Editor/Assembler, monitor MACMON e tutor de linguagem de máquina. Em fita ou disco. Aprenda Z-80!
— Compilador ZBasic. Produz código objeto com rapidez.

SOFTWARE PARA LINHA MSX

Revendemos cartuchos e fitas c/ os melhores preços. Desenvolvemos títulos próprios. Entre em contato.

A PEEK & POKE demonstra, garante e dá suporte permanente a seus programas. Venha conhecê-los ou solicite catálogo completo para o seu equipamento.

PEEK & POKE Microcomputação e Comércio Ltda.
Av. Brig. Faria Lima, 1564 - Conj. 1102
01452 - São Paulo - SP - Fone: (011) 813-3277

Saiba como é possível fazer o seu TK90X executar cálculos avançados em Assembler e aumente a sua eficiência no processamento de dados.

Cálculos avançados no TK-90X

Álvaro Ferreira de Freitas Borja

Todos os que conhecem linguagem de máquina, sabem que ela apresenta uma certa restrição no que diz respeito a cálculos matemáticos, ficando limitada às quatro operações básicas (adição, subtração, multiplicação e divisão), sendo que mesmo estas são pouco flexíveis, já que são restritas a valores inteiros e variando de 0 a 255 (quando utilizamos um registrador simples) ou de 0 a 65535 (quando utilizamos um par de registradores).

Porém, sabendo utilizar a imaginação e tendo um bom conhecimento de Assembler e matemática, é possível simularmos qualquer cálculo científico. Todavia, isto tomaria tempo até conseguirmos calcular por exemplo SQR (9).

Entretanto, existe um modo mais fácil e rápido que consiste em utilizarmos as rotinas do calculador.

Como podemos fazer isto? Primeiramente devemos ter em mente o seguinte procedimento, que é padrão para todos os cálculos feitos usando o calculador:

a) Transferir os valores para a pilha (stack) do calculador;
b) Chamar o calculador;
c) Executar os cálculos;
d) Encerrar os cálculos;
e) Retirar o resultado da pilha;
f) Continuar o programa.

TRANSFERIR OS VALORES

Existem na ROM do ZX Spectrum (TK90X) duas rotinas que fazem a transferência dos valores para a pilha do calculador. Elas porém ficam restritas aos registradores A (valores entre 0 e 255) e BC (valores entre 0 e 65535).

Para transferir um valor que esteja no registrador A, devemos chamar a rotina que fica no endereço 2D28h (11560); ela coloca o valor sob a forma de ponto flutuante no topo da pilha do calculador.

Para transferir um valor que esteja no par BC, devemos chamar a rotina que fica no endereço 2D2Bh (11563).

A primeira rotina simplesmente faz o seguinte: C = A e B = 0, continuando normalmente como se fosse transferir BC.

Exemplos de uso:

```
LD A,   23      ;carrega A com 23
CALL    2D28h   ;transf para o topo da pilha do calc.

LD BC,  3000    ;carrega BC com 3000
CALL    2D2Bh   ;transf para o topo da pilha do calc.
```

CHAMAR O CALCULADOR

Para acessarmos o calculador lançamos mão de um dos RESTARTS do Z80, que foram inteligentemente explorados pela Sinclair Research. Observe que o RESTART que chama o calculador é o 28h (40). Assim, para entrarmos no calculador devemos fazer RST 28h, sendo que este comando simplesmente executa um JP 335Bh (13147). A vantagem de usarmos RST 28h é que será gasto apenas um byte, além de ser mais fácil de memorizar.

Função	Cod.	Função	Cod.	Função	Cod.
troca (SWAP)	01	LN	25	CODE	1C
delete	02	EXP	26	VAL	1D
subtração	03	INT	27	LEN	1E
multiplicação	04	SQR	28	SIN	1F
divisão	05	SGN	29	COS	20
potenciação	06	ABS	2A	PEEK	21
adição	0F	OR	07	IN	22
TAN	21	VAL$	18	USR	20
ASN	22	USR$	19	STR$	2E
ACS	23	negate	1B	CHR$	2F
NOT	30	duplicar	31	No. AND No.	08
ATN	24	truncar	3A	val AND val$	1D

Figura 1

Valor	Cod.
Empilha o valor 0 (zero)	A0
Empilha o valor 1 (um)	A1
Empilha o valor 1/2	A2
Empilha o valor PI/2	A3
Empilha o valor 10 (dez)	A4

Figura 2

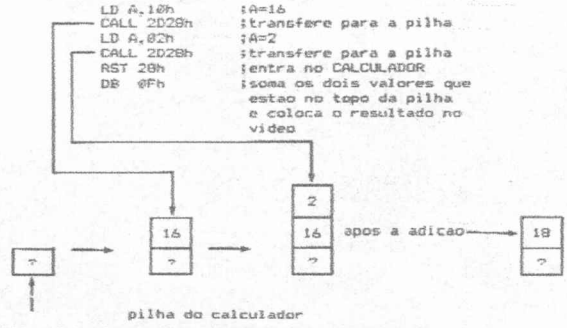
INSTRUÇÕES DE CÁLCULOS

Após executarmos um RST 28h passamos a contar com as funções matemáticas do ZX Spectrum (TK90X), sendo cada função definida por um código específico, na figura 1 estão os códigos de cada uma das funções.

A figura 2 fornece os códigos que permitem usarmos as constantes do calculador. Após os referidos códigos, o valor correspondente será colocado no topo da pilha do calculador.

Exemplificando:

```
LD A,10h        ;A=16
CALL 2D28h      ;transfere para a pilha
LD A,02h        ;A=2
CALL 2D28h      ;transfere para a pilha
RST 28h         ;entra no CALCULADOR
DB  0Fh         ;soma os dois valores que
                 estao no topo da pilha
                 e coloca o resultado no
                 video
```



ENCERRAR OS CÁLCULOS

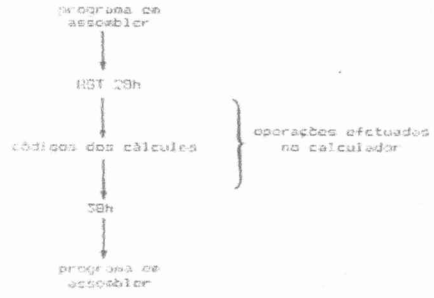
Antes de retornarmos do calculador, devemos tomar o cuidado de encerrar a operação, caso contrário, os códigos continuarão a ser in-

terpretados como sendo do calculador e não códigos do Z80. Para tal devemos utilizar o código 38h (56).

Exemplificando:

```
LD A,   10h         ;A=16
CALL    2D28h       ;transfere para a pilha
LD A,   02h         ;A=2
CALL    2D28h       ;transfere para a pilha
RST     28h         ;entra no CALCULADOR
DB      0Fh         ;soma os dois valores que
                     estao no topo da pilha
DB      38h         ;encerra calculo e retorna
    .
    .
    .
continuacao do programa
em Assembler
```

Esquematicamente, podemos representar estes procedimentos da seguinte maneira:



RETIRAR OS VALORES DA PILHA

Para retirarmos os valores que estão no topo da pilha do calculador dispomos de duas rotinas da ROM, que são as seguintes:

a) 2307h (8967) – transfere os dois números em ponto flutuante que estão no topo do calculador para o par de registradores BC, sendo que o valor retornado será um número inteiro na faixa de 0 a 65535.

b) 2314h (8980) – transfere um número em ponto flutuante que esteja no topo da pilha do calculador para o registrador A. O valor retornado será um número inteiro na faixa de 0 a 255.

Você já deve ter notado que se o valor do resultado for um número inteiro e positivo dentro das citadas faixas, basta colocarmos o valor do resultado no par BC e efetuarmos um RETorno ao BASIC (ao entrarmos, deveremos usar um PRINT USR endereço da rotina), que teremos o valor corretamente impresso na tela.

Entretanto, quando tivermos um resultado que não atenda estas solicitações deveremos lançar mão de outro recurso, que é o seguinte: encerrar os cálculos com o código 38h (56), não retirar o resultado do topo da pilha do calculador; e chamar a rotina da ROM que imprime um valor em ponto flutuante, localizada no endereço 2DE3h (11747). Desta forma, poderemos imprimir qualquer que seja o resultado da(s) operação(ões) realizada(s) no calculador.

Exemplo:

```
LD A,   10h         ;A=16
CALL    2D28h       ;transfere para a pilha
LD A,   02h         ;A=2
CALL    2D28h       ;transfere para a pilha
RST     28h         ;entra no CALCULADOR
DB      0Fh         ;soma os dois valores
DB      38h         ;encerra calculo e retorna
CALL    2DE3h       ;imprime o valor que esta no topo
```

Todavia, nem sempre desejamos imprimir os resultados das operações efetuadas no calculador, para isto devemos recorrer às memórias do calculador ou qualquer outra área designada para trabalhar como memória auxiliar.

Ao utilizarmos as memórias do calculador, não precisamos sair do calculador para armazenarmos o resultado. A única restrição é a disponibilidade de apenas 30 bytes, o que equivale a 6 memórias de 5 bytes (ponto flutuante).

O valor que é armazenado é sempre o que está no topo da pilha do calculador. A figura 3 apresenta o endereço de cada uma das memórias, bem como o código para guardar e recuperar os dados.

Um outro problema em guardarmos valores nestas memórias é que as mesmas são controladas pelo sistema, isto é, são variáveis do sistema e, portanto, estão sujeitas a alterações, principalmente se estivermos utilizando estas rotinas de cálculo em programas híbridos (BASIC & Assembler).

O melhor é evitarmos transtornos, para tal devemos reservar uma área da memória destinada ao armazenamento dos resultados. Assim,

podemos ter certeza que quando quisermos o resultado de uma determinada operação feita logo no início dos trabalhos, o mesmo ainda está guardado, bastando transferi-los para a área das memórias do calculador e efetuar novos cálculos ou simplesmente imprimi-los.

A seguir, temos dois programas de demonstração das facilidades oferecidas pelo calculador. Eles podem ser carregados em qualquer posição de memória, devendo a entrada dos mesmos ser feita por meio de um comando RAND USR endereço inicial.

Exemplo 1: $\sqrt{5*LN(4)} = 2.6327688$

```
23300  3E 02        LD A, 02
23302  CD 01 16     CALL  1601h    ;abre o canal de video
23305  3E 04        LD A, 04
23307  CD 28 2D     CALL  2D28h    ;empilha 4
23310  EF           RST   28h      ;chama o CALCULADOR
23311  25           DB    25h      ;LN(4)
23312  C0           DB    C0h      ;guarda na memoria 0
23313  38           DB    38h      ;encerrar
23314  3E 05        LD A, 05
23316  CD 28 2D     CALL  2D28h    ;empilha 5
23319  EF           RST   28h
23320  E0           DB    E0h      ;chama da memoria 0
23321  04           DB    04h      ;multiplicacao -> LN(4)*5
23322  28           DB    28h      ;SQR -> SQR(L
23323  38           DB    38h      ;encerrar
23324  CD E3 2D     CALL  2DE3h    ;imprime resultado
23327  C9           RET            ;retorna ao BASIC
```

Exemplo 2: $(5)^3 * (PI/2) = 196.34954$

```
23300  3E 02        LD A, 02h
23302  CD 01 16     CALL  1601h    ;abre o canal de video
23305  3E 05        LD A, 05h
23307  CD 28 2D     CALL  2D28h    ;empilha 5
23310  3E 03        LD A, 03h
23312  CD 28 2D     CALL  2D28h    ;empilha 3
23315  EF           RST   28h      ;entra no CALCULADOR
23316  06           DB    06h      ;potencia -> 5↑3
23317  C0           DB    C0h      ;guarda na memoria 0
23318  A3           DB    A3h      ;empilha PI/2
23319  E0           DB    E0h      ;chama memoria 0
23320  04           DB    04h      ;multiplica ->(5↑3)*(PI/2)
23321  38           DB    38h      ;encerrar
23322  CD E3 2D     CALL  2DE3h    ;imprime resultado
23325  C9           RET            ;retorna ao BASIC
```

Caso você não queira imprimir o resultado (pode ser um resultado intermediário), devemos utilizar o seguinte artifício:

a) Proceder os cálculos normalmente;

b) Antes de retornar, salvar o resultado em uma das memórias do calculador;

Figura 3

memória	endereço	guardar	recuperar
00	23698	C0	E0
01	23703	C1	E1
02	23708	C2	E2
03	23713	C3	E3
04	23718	C4	E4
05	23723	C5	E5

c) Retornar (código 38h);

d) Transferir o conteúdo da memória do calculador para uma área reservada (memória auxiliar).

O programa a seguir executa esta transferência:

```
LD BC, 05h     ;BC=numero de bytes
LD DE, xxxx    ;DE=end. da mem. auxiliar
LD HL, yyyy    ;HL=end. da mem. calcul.(figura 3)
LDIR           ;executa a transferencia
```

Toda vez que precisar utilizar (ou imprimir) o valor armazenado nesta memória, devemos transferir o valor para a memória do calculador novamente, para tal devemos proceder da seguinte forma:

```
LD BC, 05h     ;BC=numero de bytes
LD DE, xxxx    ;DE=end. da mem. do calcul.(figura 3)
LD HL, yyyy    ;HL=end. da mem. auxiliar
LDIR           ;executa a transferencia
```

Isto que foi explicado e demonstrado é apenas uma pequena parte do que é possível fazer com o calculador. O bom aproveitamento dos poderosos recursos apresentados aqui, vai depender de cada um.

Não basta, portanto, ler este artigo e já achar que sabe tudo; é preciso ficar algumas noites sobre o micro para conseguir bons resultados. Todavia, espero que isto não seja um desestímulo, mas sim uma advertência contra os possíveis fracassos iniciais que venham a ocorrer.

Álvaro Ferreira de Freitas Borja está cursando engenharia civil na UFPA. Atualmente, trabalha como Desenhista-Projetista na Telepará, sendo também usuário dos micros TK83 e TK90X.

MSX·SOFTWARE

CIBERTRON

Conheça as novidades CIBERTRON para MSX em fita e diskette: são sete títulos incluindo aplicações domésticas, profissionais e entretenimento. Os programas CIBERTRON são totalmente em português e são acompanhados por manuais de operação detalhados e completos. Venha ver em primeira mão porque CIBERTRON é muito mais software.

ASSEMBLY & DESASSEMBLY

Assembly - Totalmente relocável, editor incluso macros, assembiagem condicional, extrema rapidez de compilação. Desassembly - Execução passo a passo, múltiplos pontos de interrupção cópia inteligente. Disponível em fita e diskette.

BANCO DE DADOS

Sistema profissional de arquivo e recuperacão de informações encontrando aplicações no arquivamento de fichas de clientes, informações bibliográficas, mala direta e outros. Disponível em fita K-7.

CONTROLE DE ESTOQUE

Permite processar de maneira eficiente uma quantidade indeterminada de registros limitados apenas pela capacidade do diskette. Inclui código e nome do produto, fornecedor, unidade, quantidade, quantidade mínima, preço de compra preço de venda e data de validade. Disponível em diskette.

MSX-WORD

Poderoso processador de textos destinado ao uso doméstico ou profissional permitindo o armazenamento de até 480 linhas. Inclui busca de palavras movimentação de blocos, reformulação de parágrafos, brocagem, definição de margens, duas páginas de auxílio ao usuário e muito mais. Disponível em fita e diskette.

PITFALL II, THEZEUS & GALAGA

Disco contendo três emocionantes jogos totalmente em liguagem de máquina. Incríveis imagens de alta resolução gráfica a cores. Som fantástico. Somente em diskette.

SIMULADOR DE VÔO 737

Escrito por um piloto profissional de Boeing 737. Simula a pilotagem de um moderno avião a jato. Controle por teclado ou "joystick". Disponível em fita K-7.

Procure SOFTWARE CIBERTRON nos bons magazines em todo o Brasil. Não encontrando o porgrama desejado, escreva-nos - Caixa Postal 17.005 - CEP 02399 - São Paulo/SP.

CIBERTRON SOFTWARE

Manipule os recursos gráficos do seu microcomputador TK90X, utilizando as simples rotinas em Assembler aqui apresentadas.

Rotinas gráficas no TK90X

Claudio Bittencourt

Pretendemos aqui desmistificar para os usuários da linguagem Assembler a tela do TK90X, cuja estrutura, à primeira vista, parece um bicho-de-sete-cabeças. E apresentaremos também algumas rotinas de aplicação geral.

Primeiro destaquemos o artigo "A tela do TK90X, de Álvaro Ferreira de Freitas Borja, publicado na edição de MS nº 56, que descreve com grande propriedade a estrutura da tela. O conhecimento dessa estrutura é pré-requisito para o perfeito entendimento deste artigo. Álvaro Borja definiu e numerou os seguintes elementos da tela, que serão respeitados por nós.

Setor – a tela divide-se em três setores, numerados de 0 a 2;

Linha – cada setor contém oito linhas numeradas de 0 a 7;

Sub-linha – cada linha contém oito sub-linhas numeradas de 0 a 7.

A ordem de numeração desses elementos é de cima para baixo. Por exemplo: o setor 0 é o de cima e o 2 é o de baixo da tela. Independentemente desse desmembramento, a tela pode ser dividida em 32 colunas, numeradas de 0 a 31, em ordem crescente da esquerda para a direita.

Nós definiremos, ainda, mais um elemento, que chamaremos de quadrante, (área ocupada por um caráter normal do BASIC Sinclair). Cada linha contém 32 quadrantes, por isso iremos numerá-los da esquerda para a direita, de 0 a 31. Como o quadrante é o resultado do cruzamento de uma linha com uma coluna, existem 24 quadrantes em cada coluna.

Cada quadrante é constituído, na tela, por um empilhamento de 8 bytes, sendo portanto uma matriz de 8 x 8 bits. A cada quadrante corresponde um byte na área de "atributos", que armazena informações sobre cores, BRIGHT e FLASH.

A confusão relativa à estrutura da te-

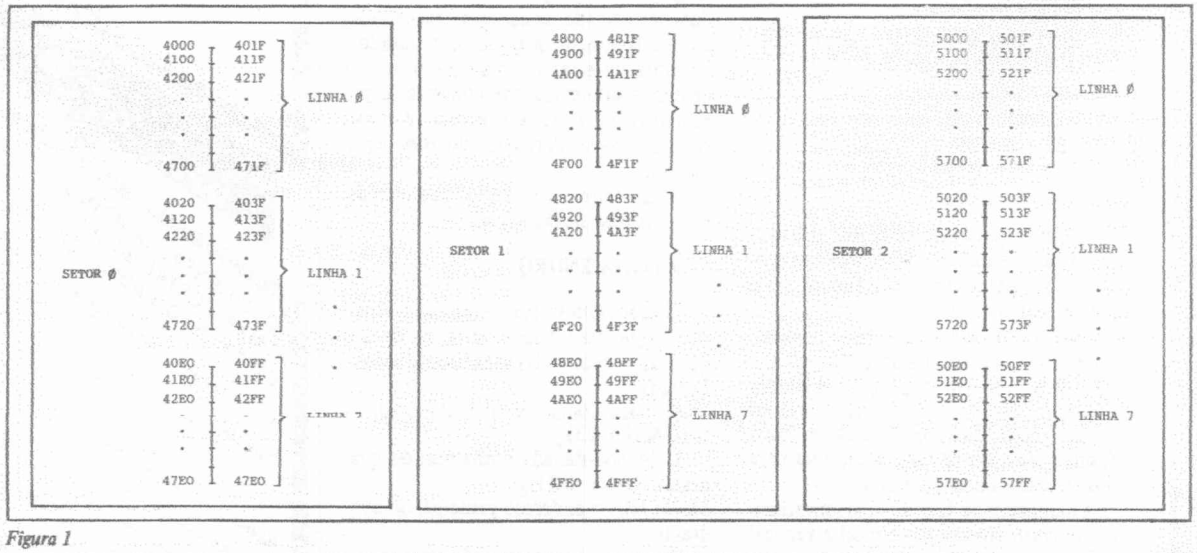


Figura 1

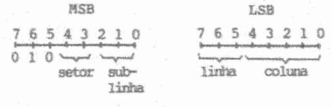
la começa a se desfazer quando analisamos os endereços na base hexadecimal (não é a toa que ela é mais usada do que a base decimal). Veja a figura 1, onde a tela é representada esquematicamente. Cada pequeno traço horizontal corresponde a uma sublinha, com os dois endereços extremos, à esquerda e à direita.

Observe, medite e entenda o seguinte:

- O byte mais significativo (MSB) de cada endereço identifica o respectivo setor e a sublinha;
- O byte menos significativo (LSB) identifica a linha e a coluna;
- Os oito endereços dos bytes de um quadrante têm o mesmo LSB, que, não por coincidência, é igual ao LSB do endereço do atributo correspondente;
- Dentro de um quadrante, para passarmos do endereço de um byte para o endereço do byte imediatamente abaixo, basta somar uma unidade ao MSB.

É ou não é uma estrututa porreta?

Na base binária, as coisas ficam mais lógicas ainda. De acordo com a numeração estabelecida para cada elemento em que subdividimos a tela, são necessários 2 bits para identificar o setor, 3 para a linha, 3 para a sublinha e 5 para a coluna de um determinado endereço da tela. Veja então como fica a estrutura desse endereço em binário:



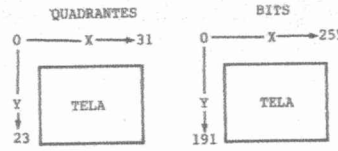
Quando dizemos, por exemplo, que os bits 3 e 4 do MSB identificam o setor, queremos dizer que eles, tomados isoladamente, são iguais ao número do setor (aquele número definido no início do artigo). Assim, se o bit 3 for 0 e o bits 4 for 1, trata-se de uma posição no setor 2, porque $10_b = 2_d$ (observe que b = binário e d = decimal). Isso é válido também para os demais elementos da tela.

Agora veja a estrutura, em binário, de um endereço na área de atributos:

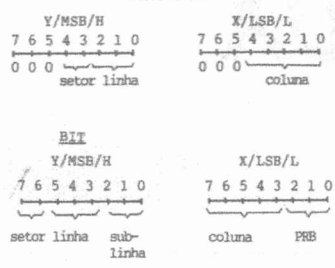
Observe que alguns bits são invariantes: 3 na área da tela e 6 na área dos atributos (sempre os mais significativos, ou seja os mais à esquerda).

Isto posto, vamos às coisas práticas. Façamos o que eu chamo de parametrizar a tela, que nada mais é do que identificar as posições por suas coordenadas e não pelos endereços. Existem duas formas úteis de se fazer isso, uma para quadrantes e outra para bits.

Chamemos de X e Y, respectivamente, as coordenadas horizontal e vertical da tela. Colocando a origem no canto superior esquerdo, teremos, em cada caso, os seguintes esquemas de numeração para X e Y:



Se armazenarmos X e Y em um par de registradores (em HL, por exemplo), sendo X colocado no LSB(L) e Y no MSB(H), teremos as seguintes estruturas em binário:



Por PRB entenda-se Posição Relativa ao Bit no seu byte. Aqui há uma ligeira complicação, porquanto X, sendo crescente da esquerda para a direita, inverte a numeração normal dos bits, que é crescente da direita para a esquerda. Assim, PRB=0 aponta o bit 7, enquanto PRB=7 aponta o bit 0.

Agora, para passarmos de um sistema para o outro, é só comparar a distribuição nos endereços e fazer as devidas trocas. Claro que só há sentido em passar de coordenadas para endereço; creio que nunca será o caso de caminhar no sentido contrário, motivo pelo qual apresentaremos as rotinas em mão única.

As rotinas que se seguem permitem a universalização de outras rotinas. Por exemplo, se eu quero fazer uma rotina que trace um círculo, irei desenvolvê-la em termos de parâmetros (coordenadas) e não de endereços, tornando-a compatível com qualquer computador. Na hora de setar ou resetar um bit, eu chamo a sub-rotina que converte os parâmetros em endereço, e esta se encarrega das particularidades do meu equipamento.

Nota: todos os valores, nas rotinas, estão na base hexadecimal.

ROTINA ENDRQ

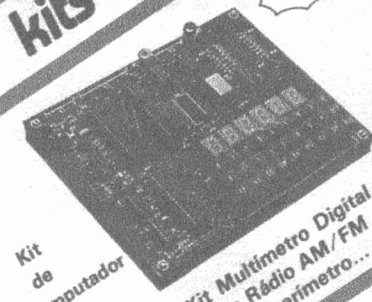
Função: determina o endereço principal (o de mais baixo número) de um quadrante dado pelas suas coordenadas.
Entrada:

H = coordenada vertical do quadrante $(0 \leqslant H \leqslant 23)_d$.

L = coordenada horizontal do quadrante $(0 \leqslant L \leqslant 31)_d$.

Saída: HL = endereço principal do quadrante.

Demais registradores: não afetados.

cursos técnicos!
• eletrônica básica • eletrônica digital
• áudio e rádio • televisão pb/cores
• programação basic • programação cobol
• análise de sistemas • microprocessadores
• eletrotécnica • instalações elétricas
• refrigeração e ar condicionado
kits exclusivos!
Z-80

Kit de Microcomputador e mais
• Kit de Televisão
• Kit de Refrigeração
• Kit Digital Avançado
• Kit Multímetro Digital
• Kit de Rádio AM/FM
• Volt Amperímetro...
cursos por correspondência
intensivos! dinâmicos!
OCCIDENTAL SCHOOLS
cursos técnicos especializados
Alameda Ribeiro da Silva, 700
01217 São Paulo SP
Fone: (011) 826-2700
SOLICITE MAIORES INFORMAÇÕES SEM COMPROMISSO!
OCCIDENTAL SCHOOLS
CAIXA POSTAL 30.663
01051 SÃO PAULO SP
Desejo receber, gratuitamente, o catálogo ilustrado do
Curso de: indicar o curso desejado
nº
Nome
Endereço
Bairro
Cidade
Estado
CEP

PRB	A		B	BIT
	(BINÁRIO)	(DECIMAL)		
0	10000000	128	1	7
1	01000000	64	2	6
2	00100000	32	3	5
3	00010000	16	4	4
4	00001000	8	5	3
5	00000100	4	6	2
6	00000010	2	7	1
7	00000001	1	8	0

Figura 2

```
ENDRQ PUSH AF
      LD A,H
      RRCA
      RRCA
      RRCA
      AND E0
      OR L
      LD L,A
      LD A,H
      AND 18
      OR 40
      LD H,A
      POP AF
      RET
```

ROTINA ENDRB

Função; determina o endereço de um bit dado pelas suas coordenadas e indica a posição relativa desse bit dentro do byte que o contém.

Entrada:

H = coordenada vertical do bit $(0 \leqslant H \leqslant 191)_d$.

L = coordenada horizontal do bit $(0 \leqslant L \leqslant 255)_d$.

Saída:

HL = endereço do byte que contém o bit.

A = todos os bits do registrador A são zerados, com exceção de um, que corresponde ao bit em atribuição.

B = PRB + 1 (ou seja, "Posição Relativa do Bit" + 1).

Demais registradores: não afetados.

Sub-rotina utilizada: ENDRQ.

Obs.: a figura 2 indica os valores de A e B, conforme a posição do bit em atribuição dentro do byte.

```
ENDRB PUSH AF
      LD A,L
      AND 7
      INC A
      LD B,A
      POP AF
      PUSH BC
      PUSH AF
      LD C,1
      RRC C
      DJNZ -4
      LD A,7
      AND H
      LD B,3
      SRL H
      SRL L
      DJNZ -6
      CALL ENDRQ
      OR H
      LD H,A
      POP AF
      LD A,C
      POP BC
      RET
```

ROTINAS SETB, RESB E INVB

Função: seta, reseta ou inverte um bit da tela dado por suas coordenadas.

Entrada: igual a ENDRB.

Saída: além dos flags, são afetados os registradores A, B e HL.

Sub-rotina utilizada: ENDRB.

```
SETB  CALL ENDRB
      OR (HL)
      LD (HL),A
      RET

RESB  CALL ENDRB
      CPL
      AND (HL)
      LD (HL),A
      RET

INVB  CALL ENDRB
      XOR (HL)
      LD (HL),A
      RET
```

ROTINA PROAT

Função: encontra o endereço, na área de atributos, correspondente a um dado endereço do arquivo de imagens.

Entrada: HL = endereço na área de arquivo de imagens.

$16384_d = 4000_h \leqslant HL \leqslant 57FF_h = 22527_d$

Saída: HL =endereço do atributo.

Demais registradores: não afetados.

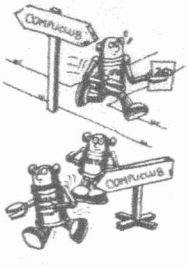
```
PROAT PUSH AF
      LD A,H
      AND 18
      RRCA
      RRCA
      RRCA
      OR 58
      LD H,A
      POP AF
      RET
```

Em um próximo artigo, apresentaremos rotinas que permitam colocar na tela figuras previamente definidas. Até lá !

Cláudio de Freitas B. Bittencourt é formado em engenharia metalúrgica e Professor de Pós-Graduação em engenharia nuclear do IME, Instituto Militar de Engenharia, no Rio de Janeiro.

MSX COMPUCLUB

Finalmente, os usuários da linha MSX já podem desfrutar do melhor clube de soft do Brasil.



Veja tudo o que você tem a ganhar:

- Edições bimestrais do Compuclub News, com notícias do mundo da Informática, programas de jogos, aplicativos e dicas especiais para o seu MSX.
- A cada 30 dias, programas amplamente documentados, com seus manuais de instrução.
- Livre escolha de softs.
- Sorteios mensais de micros e outros prêmios.

ATENÇÃO: No COMPUBLUB não há mensalidades.

Não perca tempo! Solicite, ainda hoje, informações detalhadas acerca do COMPUCLUB. Não se esqueça, porém, de mencionar o tipo de equipamento que você possui.

MICROS BRASILEIROS DA LINHA MSX
HOTBIT, EXPERT, ETC.

COMPUCLUB — Caixa Postal 46 — CEP 36570 — Viçosa, MG.

Aprenda um pouco mais sobre o seu TK90X e linguagem de máquina, escrevendo um programa que enche a tela com caracteres randômicos.

Caracteres randômicos no TK90X

Nelson N. S. Santos

Nosso objetivo será escrever um programa em linguagem de máquina que produza o efeito de encher a tela com caracteres randômicos, usando **BORDER** amarelo, **PAPER** amarelo e **INK** azul. Ou seja, obter o código de máquina equivalente a:

```
10 BORDER 6
20 PAPER 6 : INK 1
30 CLS
40 RAND
50 FOR f = 1 TO 704
60 PRINT CHR$ (INT (RND * 133) + 32);
70 NEXT f
```

Rode este programa BASIC e observe o tempo que a tela leva para ser preenchida. Cronometramos 19,9 segundos até o aparecimento da mensagem do sistema operacional 0 Executado 70 : 1.

Nosso programa em linguagem de máquina será fantasticamente mais rápido, e usaremos um pequeno truque para tornar invisível a mensagem do sistema operacional. Quando terminarmos seu estudo, você certamente conhecerá melhor as variáveis do sistema, a ROM e a maneira do seu TK90X funcionar.

BORDER 6

O manual não é particularmente esclarecedor em relação às funções IN e OUT. OUT pela porta 254 altera a cor de BORDER, que é controlada pelos primeiros três bits (0, 1 e 2) do valor que você enviar. Os outros bits controlam a produção de som. Como não é isto o que desejamos, nos concentraremos nesses três bits.

bit 2	bit 1	bit 0	valor decimal	cor de border
0	0	0	0	preto
0	0	1	1	azul
0	1	0	2	vermelho
0	1	1	3	magenta
1	0	0	4	verde
1	0	1	5	ciano
1	1	0	6	amarelo
1	1	1	7	branco

Comprove tais fatos através deste pequeno programa:

```
10 FOR f = 0 TO 7
20 OUT 254, f
30 PRINT AT 0,0; f
40 PAUSE 0
50 NEXT f
```

Como fazer OUT 254,6 em código de máquina? Simples, pois OUT é uma das instruções do Z80. Usaremos a instrução OUT N, A; cujo código hexa é D3. Assim:

ASSEMBLY	HEXA	DECIMAL
LD A, 6	3E 06	62 6
OUT 254, A	D3 FE	211 254

Agora, vamos informar ao sistema operacional que a cor do BORDER é amarela e aproveitar para implementar o truque de tornar invisíveis as mensagens do sistema operacional. No endereço 23624 (5C48) reside a variável do sistema BORCLR (BORDCR no Spectrum), que "contém o número da cor definida com BORDER vezes 8, mais os atributos da parte baixa da tela". Vamos entender isto.

bit →	7	6	5	4	3	2	1	0
	FLASH da parte baixa	BRIGHT da parte baixa	BORDER parte	(c PAPER baixa)	da	IN	K (da par baix	te a)

Faremos FLASH = 0, BRIGHT = 0, BORDER = 6 e INK = 6 para tornarmos as mensagens do sistema operacional invisíveis, pois PAPER = INK!

bit →	7	6	5	4	3	2	1	0
	0	0	1	1	0	1	1	0
			6 d			6 d		

Tal número em decimal é 32 + 16 + 4 + 2 = 54. Confirme o que dissemos, digite este programinha e comprove que a mensagem 0 Executado 30 : 1 é invisível. Tente digitar alguma coisa. Tudo é invisível, embora faça efeito.

```
10 OUT 254,6
20 POKE 23624,54
30 PRINT "Micro Sistemas"
```

Passando para linguagem de máquina:

LD A, 54	3E 36	62 54
LD (23624), A	32 48 5C	50 72 92

PAPER 6 E INK 1

Isto é fácil de executar. No endereço 23693 (5C8D) está a variável do sistema ATCLR P (ATTR P no Spectrum) que dá a "situação de posição permanente da tela quanto à cor etc.". Veja:

bit →	7	6	5	4	3	2	1	0
	FLASH	BRIGHT		PAPER			INK	

Faremos FLASH = 0, BRIGHT = 0, PAPER = 6 e INK = 1. Logo.

bit →	7	6	5	4	3	2	1	0
	0	0	1	1	0	0	0	1
			6d			1d		

Tal número em decimal é 32 + 16 + 1 = 49. Confirme nossos passos. Digite o programa a seguir e verifique: BORDER amarelo, PAPER amarelo, INK azul e mensagem do sistema operacional invisível.

```
10 OUT 254,6
20 POKE 23624,54
25 POKE 23693,49
30 PRINT "Micro Sistemas"
```

Em linguagem de máquina:

LD A, 49	3E 31	62 49
LD (23693), A	32 8D 5C	50 141 92

CLS

A maneira mais prática de executar um CLS em linguagem de máquina é dar um CALL na rotina da ROM que faz isto. CALL 3435 (0D6B) limpa toda a tela e restaura os atributos para aqueles contidos em ATCLR P para a parte principal da tela em BORCLR para a parte baixa. Assim, LET £ = USR 3435 é equivalente a CLS. Comrpove:

```
10 OUT 254,6
20 POKE 23624,54
25 POKE 23693,49
27 LET £ = USR 3435
30 PRINT "Micro Sistemas"
```

Em linguagem de máquina:

CALL 3435	CD 6B 0D	205 107 13

Agora, para ser possível a impressão na tela é necessário "abrir" o canal S. O manual não explica, logo... vamos lá.

Sem periféricos ligados, temos acesso a três canais sem que apareça a mensagem de erro "O Canal inválido" Observe:

# 0 ou # 1	parte baixa da tela: canal K
# 2	parte principal da tela: canal S
# 3	impressora: canal P

A rotina CLS, chamada através de CALL 3435, abre o canal # 0. Assim, após ela, qualquer impressão seria deslocada para a parte baixa da tela, e não para a tela. A rotina da ROM que abre canal está no endereço 5633 (1601h) Antes de chamá-la é preciso carregar o acumulador com o número necessário – no caso 2, para abrir o canal S. Em linguagem de máquina:

LD A, 2	3E 02	62 2
CALL 5633	CD 01 16	205 1 22

O LOOP.

Veremos agora como colocar caracteres randômicos na tela com um loop. A tela tem 704 posições (22 linhas x 32 colunas). Usaremos BC como contador de loop.

LD BC, 704	01 C0 02	1 192 2

Agora vamos gerar um número randômico entre 0 e 65535. O processo usado foi extraído do célebre livro de Toni Baker, *Mastering Machine Code on your ZX81*. Modificamos levemente a técnica, porque a variável do sistema TVCOUNT (FRAMES no Spectrum) é de três bytes, ao contrário do Sinclair tradicional, onde é de dois bytes.

Esta variável de três bytes tem seu endereço inicial 23672 (5C78) e é inicializada com 0 quando o micro é ligado, sendo incrementada em um cada vez que a rotina normal de interrupção é chamada, aproximadamente a cada 20 ms.

Vamos colocar em HL o primeiro e o segundo bytes de TVCOUNT; e em DE, o segundo e o terceiro bytes. Rotulemos este ponto do programa de Ⓐ

LD HL, (TVCOUNT-baixa)	2A 78 50	42 120 92
LD DE, (TVCOUNT-alta)	ED 5B 79 5C	237 91 121 92

Agora vamos gerar um número pseudo-randômico através de sucessivas somas com realimentação.

ADD HL, HL	29	41
ADD HL, HL	29	41
ADD HL, DE	19	25
ADD HL, HL	29	41
ADD HL, HL	29	41
ADD HL, HL	29	41
ADD HL, DE	19	25

O número gerado, o recolocaremos em TVCOUNT.

LD (TVCOUNT-baixa), HI	22 78 5C	34 120 92

Transferimos a parte baixa do número gerado (L) para A.

LD A, L	7D	125

A contém agora um número entre 0 e 255. Se A estiver entre 0 e 31, não é um caráter imprimível, e voltaremos ao ponto Ⓐ

CP 32	FE 20	254 32
JR C, Ⓐ	38 EA	56 234

Se A estiver entre 144 e 255, também não é imprimível (exceto entre 144 e 164, que correspondem a UDG), e também voltaremos ao ponto Ⓐ

CP 144	FE 90	254 144
JR NC, Ⓐ	30 E6	48 230

Teremos agora a certeza de que o conteúdo de A está entre 32 e 143. Logo, pode ser colocado na tela através de RST 16.

RST 16	D7	215

Agora, decrementa-se o contador e verifica-se se atingiu a 0. Se não, volta-se ao ponto Ⓐ . RET volta ao BASIC.

DEC BC	0B	11
LD A, B	78	120
OR G	B1	177
JRNZ, A	20 E0	32 224
RET	C9	201

O programa está pronto, e vamos carregá-lo no micro. Digite:

```
10 CLEAR 31999 : LET s = 0
20 FOR f = 32000 TO 32057
30 READ a = POKE f, a
40 LET s = s + a : NEXT f
50 IF s<>5722 THEN PRINT "Erro nos DATA" : STOP
60 PRINT "Aperte uma tecla para rodar" : PAUSE 0
70 RAND USR 32000
80 DATA 62, 6, 211, 254, 62, 54, 50, 72, 92, 62, 49, 50, 141, 92
90 DATA 205, 107, 13, 62, 2, 205, 1, 22, 1, 192, 2, 42, 120, 92,
   237, 91, 121, 92, 41, 41, 25, 41, 41, 41, 25, 34, 120, 92,
   125, 254, 32, 56, 234, 254, 144, 48, 230, 215, 11, 120, 177,
   32, 224, 201.
```

Cronometramos 0,6 segundos para o preenchimento da tela, ou seja, aproximadamente 33 vezes mais rápido que o programa BASIC. Que tal o efeito?

Se você nos seguiu até aqui, certamente conhece o seu TK90X melhor do que antes.

Nelson N. S. Santos desempenha atualmente a função de Editor na Editora Campus. Usuário da linha Sinclair, já trabalhou como Consultor Técnico na área de linguagens, sendo também autor do livro "Além do BASIC", publicado pela Campus.

MSX

INÉDITO NO BRASIL

A MAC SOFT traz até você os últimos lançamentos em jogos para linhas msx. São 4 pacotes cheios de emoções e aventuras.

PACOTE 1
- THE HEIST
- ZAIL WOOD
- BOOGABOO
- BOSCOVIAN
- MANIC MINER
- BLOCKADE RUNNER II
- ZAXXON II
- EDDIE KID
- BANK PANIC
- POKER II

PACOTE 2
- DAMBUSTERS
- FOT VOLEY
- MOLE-MOLE
- FRONT LINE
- ATHLETIC LANDER
- STEP UP
- SPARTA X
- FRUIT FRANK
- MAGICAL KIDWIZ
- DRAGON SLAYER

PACOTE 3
- TIME CURB
- VALKIR
- GAMON
- THUNDERBALL
- MOPIRANGER
- MAZER KILLER
- SPELUNKER
- ZOOM 909
- TIME PILOT
- CAPTAIN

PACOTE 4
- SPIT FIRE 40
- CUBOS
- SPEED KING
- GIRL DISASTER
- FLIPER SLIPER
- MAC JOE
- HYDWE
- ANIMAL
- JET BOMBER
- SWEET A CORN

• O preço de cada pacote é Cz$ 350,00. Envie seu pedido acompanhado de cheque nominal e cruzado à NEY GUIMARÃES SYLVESTRE Cx. Postal 33161 CEP: 22442 - RIO/RJ.
• Os jogos também poderão ser adquiridos **SEPARADAMENTE** (de qualquer pacote) ao preço de Cz$ 50,00 por programa (pedido mínimo de cinco jogos).
• Para maiores informações sobre outros jogos, aplicativos e/ou utilitários, além de drives p/msx, escreva-nos solicitando o nosso catálogo.
• Nossos produtos são **GARANTIDOS**.

MAC SOFT

Pac-Man

Márcio José de Carvalho

Pac-Man foi adaptado de um original trancês, de *Jean J. Jacquet*, da linha ZX81 para ZX Spectrum. Sendo de fácil digitação – pois é totalmente em BASIC –, o jogo inicia com uma sugestiva apresentação de bom efeito visual. Em seguida, outra tela é apresentada com as informações sobre o programa.

Apesar de os fantasminhas se movimentarem aleatoriamente, uma grande dificuldade para o jogador é a de que as paredes do labirinto estão eletrificadas. Você não pode se encostar pois é morte certa.

Após você haver comido todas as pastilhas do labirinto ou perdido sua última vida, são apresentados os resultados e a pergunta se deseja outra partida. Na primeira vez que você jogar, qualquer que seja o resultado, o computador dirá que o jogador venceu, já que em sua memória não há registro de resultados anteriores.

MELHORIAS DO JOGO

Aqui faremos algumas sugestões para um melhor aperfeiçoamento de *Pac-Man:*

- Alterar o movimento aleatório dos fantasmas para o movimento de perseguição;
- Acrescentar uma variável que faça você mudar de cor, após comer um determinado número de pastilhas, e possa eliminar os fantasmas;
- Acrescentar variável de gravação dos resultados obtidos e/ou variável de nível de dificuldade.

Márcio José de Carvalho é formado em Administração de Empresas e pós-graduado em Administração Financeira, trabalhando atualmente na Themag Engenharia, Belém — PA. Ele, como hobby, programa na linguagem BASIC nos equipamentos ZX81 e TK90X.

```
10 REM
15 CLS : PAPER 0: BORDER 0: INK 4
20 REM
25 REM
80 LET ms=0: LET mt=1000
90 CLS : LET A=0: PRINT AT 1,10;"PAC-MAN";AT 4,0;" VOCE ESTA PRESO EM UM LABIRINTOCOM DOIS FANTASMAS ANSIOSOS POR DEVORA-LO, COM GRANDE APETITE. OOBJETIVO DO JOGO E COMER O MAIORNUMERO DE PASTILHAS(o), NO MENORESPACO DE TEMPO POSSIVEL. DEPOISQUE COMER TODAS ELAS, VOCE PASSAPARA OUTRA FASE.";AT 13,0;" CUIDADO ! AS PAREDES DO LABIRINTO ESTAO ELETRIFICADAS.";AT 15,6;"...BOA SORTE..."
91 FOR I=1 TO 1000: SOUND .1,10: LET A=A+1: IF A=99 THEN GOTO 97
92 IF I=18 THEN LET I=0: PRINT AT 21,18;"
93 INK INT (RND*8)+1: PRINT AT 19,I;"J. C. JACQUET";AT 20,I;"MARCIO CARVALHO"
94 NEXT I
96 REM
97 REM
98 LET s=0: LET at=0: LET fase=1: LET bonus=0
99 LET zx=0
100 LET v=19: LET h=16
110 LET a=0
111 FOR x=0 TO 200
112 READ z,x,g,m
113 IF m=1000 THEN GOTO 115
114 NEXT x
115 GOSUB 5000
116 RESTORE
117 REM
118 REM
120 DATA 15,20,15,13,16,20,16,13,17,20,17,13,17,19,16,13,17,19,19,13,17,17,19,12,17,16,19,11,17,15,19,10,17,14,19,9,17,13,18,9,18,13,17,9,19,13,16,9,19,15,16,8,19,16,16,7,19,17,16,6,19,18,16,5,19,20,16,4,19,21,16,3,19,22,16,2,19,23,16,1,18,23,15,1,17,22,14,1,16,22,13,1
122 DATA 15,22,12,1,14,22,12,2,14,23,12,3,14,24,12,4,14,25,12,5,14,26,12,6,14,25,12,7,16,25,12,8,16,25,11,8,16,24,10,8,16,23,10,7,16,22,10,6,17,22,10,5,18,23,10,4,19,23,10,3,19,24,9,3,19,25,8,3,19,26,7,3,19,27,7,4,19,28,7,5,18,28,7,6,17,28,7,7,16,28,7,8,15,28,8,8,14,28,9,8,13,28,9,7,12,28,9,6,12,29,9,5,12,30,10,5,11,30,10,4,10,30,10,3,9,30,9,3,8,30,8,3,7,30,7,3,6,30,7,4,5,30,7,5
125 DATA 5,30,7,5,5,29,7,6,5,28,7,7,6,28,7,8,7,28,7,9,8,28,7,10,9,28,7,11,10,28,7,12,10,27,7,13,10,26,7,14,9,26,8,14,8,26,9,14,7,26,10,4,7,25,10,13,7,24,10,12,7,23,10,11,7,22,10,10,7,21,9,10,9,21,8,10,10,21,7,10,10,20,7,9,11,19,7,8,11,18,6,8,11,7,5,8,11,16,5,7,10,16,5,6,9,16,5,5,8,16,5,4,8,17,5,3,8,18,5,2,8,19,5,1,7,19,5,1,6,19,5,1,5,19,5,2,5,18,5,3,5,17,5,4,5,16,5,5,5,15,5,6,5,16,5,7,6,16,5,8,6,17,5,9,6,18,5,10,6,19,5,11,7,19,5,12,8,19,5,13,9,19,5,14,11,19,5,15,12,19,5,16,12,20,5,17,13,20,5,18,14,20,6,18,15,20,7,18,15,20,8,18,5,20,10,18,15,20,11,18,15,20,12,18,15,20,113,1000
131 REM
132 REM
140 BORDER 0: PAPER 0: CLS
141 LET vida=3
151 LET t=0
199 PAPER 0: BORDER 0
200 CLS : INK 6: PLOT 25,120: DRAW 48,0,-2: PLOT 73,119: DRAW 0,-66: PLOT 24,120: DRAW 0,-66: FOR f=1 TO 3: DRAW 7,0,4: DRAW 7,0,-4: NEXT f: DRAW 7,0,4
210 PRINT ; FLASH 1;AT 8,4;"5";AT 8,7;"5";AT 10,5; FLASH 0: INK 6;AT 11,4;"(__)"
211 PLOT 110,0: DRAW 0,173
212 FOR i=1 TO 20: PRINT ; INK 1;AT i,15;"████████": NEXT i
213 INK 6: PRINT ; FLASH 1;AT 17,3;"PAC-MAN"
214 PRINT ; FLASH 1;AT 1,3;"PAC-MAN"
215 INK 7: PRINT ; FLASH 1;AT 2,19;"COMANDOS"
216 PRINT AT 6,16;"5 -->ESQUERDA"
217 PRINT AT 8,16;"8 -->DIREITA"
218 PRINT AT 10,16;"7 -->PARA CIMA"
219 PRINT AT 12,16;"6 -->PARA BAIXO"
220 PRINT AT 16,21;"TECLE";AT 17,21;"**0**";AT 20,0;" PROGRAMADOR:"; FLASH 1;AT 21,0;"J. C. JACQUET"; FLASH 0;AT 20,15;" TRAD/ADAPTACAO :"; FLASH 1;AT 21,16;"MARCIO CARVALHO"; FLASH 0
221 FOR e=1 TO 4000
222 IF INKEY$="0" THEN GOTO 230
223 FOR x=1 TO 3: SOUND .3,7: SOUND .15,4: SOUND .15,4: SOUND .3,7: NEXT x: SOUND .6,0:
224 NEXT e
225 REM
226 REM
999 GOSUB 4000
1000 FOR n=1 TO 9e9
1001 LET t=t+1
1002 IF s=3130 THEN GOTO 8000
1300 READ z,x,g,m
1400 IF m=1000 THEN RESTORE : READ z,x,g,m
1401 IF ATTR (z,x)=8 THEN GOTO 3000
1402 PRINT AT z,x; INK 7; OVER 1;"O": PRINT ; FLASH 1; INK 7;AT 2,26;s: PRINT AT z,x; INK 1; OVER 1;"O"
1403 IF ATTR (g,m)=6 THEN GOTO 3000
1404 PRINT AT g,m; INK 7: OVER 1;"Q": SOUND .001,50: PRINT AT g,m; INK 1; OVER 1;"Q"
1405 IF INKEY$="5" THEN LET a=1: LET h=h-1: LET zx=5
1406 IF INKEY$="8" THEN LET a=2: LET h=h+1: LET zx=5
1407 IF INKEY$="6" THEN LET a=3: LET v=v+1: LET zx=5
1408 IF INKEY$="7" THEN LET a=4: LET v=v-1: LET zx=5
1409 IF zx=5 AND SCREEN$ (v,h)="o" THEN LET s=s+10
1420 IF ATTR (v,h)=4 OR ATTR (v,h)=5 THEN GOTO 3000
1500 IF a=1 THEN PRINT AT v,h+1; PAPER 0; INK 0;" ";AT v,h; INK 6;"P"
1510 IF a=2 THEN PRINT AT v,h-1; PAPER 0; INK 0;" ";AT v,h; INK 6;"C"
1520 IF a=3 THEN PRINT AT v-1,h; PAPER 0; INK 0;" ";AT v,h; INK 6;"C"
1530 IF a=4 THEN PRINT AT v+1,h; PAPER 0; INK 0;" ";AT v,h; INK 6;"P"
1560 NEXT n
1561 REM
1562 REM
1563 REM
1564 REM
3000 IF ATTR (z,x)=6 OR ATTR (g,m)=6 THEN PRINT AT v,h; INK 0; PAPER 0;" ": LET at=1: GOTO 3011
3001 IF a=1 THEN PRINT AT v,h+1; PAPER 0; INK 0;" "
3002 IF a=2 THEN PRINT AT v,h-1; PAPER 0; INK 0;" "
3003 IF a=3 THEN PRINT AT v-1,h; PAPER 0; INK 0;" "
3004 IF a=4 THEN PRINT AT v+1,h; PAPER 0; INK 0;" "
3011 LET vida=vida-1: PRINT ; INK 4;AT 21,19-vida;"G"
3012 LET g=.6: SOUND g,0: SOUND g*2/3,0: SOUND g/3,0: SOUND g,0: SOUND g*2/3,3: SOUND g/3,2: SOUND g*2/3,2: SOUND g/3,0: SOUND g*2/3,0: SOUND g/3,-1: SOUND g,0
3013 IF vida=0 THEN GOTO 6000
3014 IF at=1 THEN LET at=0: GOTO 3020
3016 IF ATTR (v,h)=5 THEN PRINT ; INK 5;AT v,h;"B"
3017 IF ATTR (z,x)=6 THEN PRINT ; PAPER 0;AT v,h;" "
3018 IF ATTR (g,m)=6 THEN PRINT ; PAPER 0;AT v,h;" "
3019 IF ATTR (v,h)=4 THEN PRINT ; INK 4;AT v,h;"D"
3020 LET v=20: LET h=16
3021 PRINT ; INK 6;AT v,h;"C"
3022 GOTO 1000
3023 REM
3024 REM
3025 REM
4000 CLS : BORDER 0: PAPER 0
4001 FOR i=5 TO 20
4002 INK 1: PRINT AT i,2;"oooooooooooooooooooooooooooooo"
4003 NEXT i
4010 INK 4: PRINT AT 0,0;"DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD"
4020 INK 4: PRINT AT 4,0;"DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD"
4030 IF vida=3 THEN PRINT ; INK 4;AT 21,0;"DDDDDDDDDDDDDVIDA CCCDDDDDDDDDDD"
4031 IF vida=2 THEN PRINT ; INK 4;AT 21,0;"DDDDDDDDDDDDDVIDA DCCDDDDDDDDDDD"
4032 IF vida=1 THEN PRINT ; INK 4;AT 21,0;"DDDDDDDDDDDDDVIDA DDCDDDDDDDDDDD"
4033 PRINT ; INK 2; FLASH 1;AT 14,15;"5555"
4034 PRINT ; INK 2; FLASH 1;AT 15,15;"5555"
4040 FOR i=0 TO 21
4041 INK 4: PRINT AT i,0;"D";AT i,31;"D"
4043 INK 5: PRINT AT 2,20; FLASH 1;"PLACAR: ";s;AT 2,3;"BONUS: ";bonus
4050 NEXT i
4061 INK 5: PRINT AT 6,2;"BBBB";AT 7,2;"B";AT 8,2;"B";AT 8,4;"BBB";AT 9,2;"B";AT 9,4;"B";AT 10,2;"B";AT 11,2;"BBBB";AT 13,2;"BBBBBB";AT 14,2;"B";AT 14,7;"B";AT 15,2;"B";AT 15,7;"B";AT 15,4;"BB";AT 17,2;"B";AT 17,7;"B";AT 18,2;"BBBBBB"
4063 PRINT AT 6,9;"BBBBBBB";AT 7,15;"BBB";AT 8,9;"B";AT 8,15;"B";AT 9,9;"B";AT 9,11;"BBB";AT 9,15;"B";AT 10,9;"B";AT 10,15;"B";AT 11,9;"BBBBBBB";AT 13,10;"BB";AT 14,11;"B";AT 15,10;"BB";AT 16,11;"BB";AT 13,14;"BBBBBB";AT 14,14;"B";AT 14,19;"B";AT 15,14;"B";AT 15,19;"B";AT 16,14;"BBBBBB";AT 18,10;"BBB"
4065 PRINT AT 18,14;"BBBBBB";AT 19,14;"B";AT 19,19;"B"
4067 PRINT AT 9,18;"B";AT 10,19;"B";AT 6,20;"BBBBBBBB";AT 7,20;"B";AT 7,27;"B";AT 7,29;"B";AT 8,20;"B";AT 8,22;"BBBB";AT 8,27;"B";AT 8,29;"B";AT 9,20;"B";AT 9,27;"B";AT 10,23;"BB";AT 9,29;"B";AT 10,29;"B";AT 11,20;"BBB";AT 11,25;"BBB";AT 11,29;"B"
4070 PRINT AT 13,21;"BBBBBBB";AT 14,21;"B";AT 14,27;"B";AT 14,29;"B";AT 15,21;"B";AT 15,27;"B";AT 15,29;"B";AT 15,23;"BBB";AT 16,21;"B";AT 16,27;"B";AT 16,29;"B";AT 17,21;"B";AT 17,27;"B";AT 17,29;"B";AT 17,24;"B";AT 18,21;"BB";AT 18,26;"BB";AT 18,29;"B"
4088 PRINT #1; FLASH 1;"RECORDE ";a$; FLASH 0;"                "; FLASH 1;"MTEMPO ";mt
4089 PRINT ; INK 1;AT 5,1;"o"
4090 PRINT ; INK 6;AT v,h;"C"
4092 RETURN
4093 REM
4094 REM
4095 REM
4096 REM
5020 FOR x=0 TO 7: READ rn: POKE USR "d"+x,rn: NEXT x
5021 DATA 60,255,129,255,255,129,255,60
5025 FOR x=0 TO 7: READ rn: POKE USR "c"+x,rn: NEXT x
5026 DATA 62,102,255,240,240,255,126,62
5030 FOR x=0 TO 7: READ rn: POKE USR "b"+x,rn: NEXT x
5031 DATA 204,204,204,204,204,204,204,204
5035 FOR x=0 TO 7: READ rn: POKE [illegible] NEXT x
5036 DATA 124,102,255,15,15,255,126,124
5040 FOR x=0 TO 7: READ rn: POKE USR "q"+x,rn: NEXT x
5041 DATA 60,126,219,219,255,255,153,153
5045 FOR x=0 TO 7: READ rn: POKE USR "s"+x,rn: NEXT x
5046 DATA 24,60,126,231,231,126,60,24
5047 RETURN
5048 REM
5049 REM
5050 REM
5051 REM
6000 LET s=bonus+s: CLS : INK 6: PLOT 25,120: DRAW 48,0,-2: PLOT 73,119: DRAW 0,-66: PLOT 24,120: DRAW 0,-66: FOR f=1 TO 3: DRAW 7,0,4: DRAW 7,0,-4: NEXT f: DRAW 7,0,4
6002 LET VA=0
6010 PRINT ; FLASH 1;AT 8,4;"5";AT 8,7;"5";AT 10,5; FLASH 0; INK 6;AT 11,4;"(__)"
6011 PLOT 110,0: DRAW 0,173
6014 FOR i=1 TO 21: PRINT ; INK 1;AT i,15;"████████": NEXT i
6015 PRINT ; INK 7; FLASH 1;AT 2,18;"CRESULTADOP";
6016 INK 7: PRINT ; FLASH 1;AT 17,3;"PAC-MAN"
6017 INK 7: PRINT ; FLASH 1;AT 1,3;"PAC-MAN"
6018 IF ms>s THEN GOTO 6050
6019 IF ms<s THEN GOTO 6100
6020 IF ms=s THEN GOTO 6150
6051 PRINT AT 4,16; PAPER 1;"COITADO ! VOCE "
6052 PRINT AT 5,16; PAPER 1;"PERDEU. SEU "
6053 PRINT AT 6,16; PAPER 1;" PLACAR FOI  "
6057 PRINT AT 7,16; PAPER 1;s;" PTS. EM "
6058 PRINT AT 8,16; PAPER 1;t;" SEGUNDOS "
6059 PRINT AT 9,16; PAPER 1;"  O MELHOR  "
6060 PRINT AT 10,16; PAPER 1;" PLACAR  "
6061 PRINT AT 11,16; PAPER 1;"PERTENCE AO  "
6062 PRINT AT 12,16; PAPER 1;a$;" EM "
6063 PRINT AT 13,16; PAPER 1;mt;" SEGUNDOS "
6064 PRINT AT 14,16; PAPER 1;"COM ";ms;" PTS "
6065 PRINT AT 16,17; PAPER 1;" VOCE QUER "
6066 PRINT AT 17,16; PAPER 1;" REPETIR(s/n) ": IF VA=2 THEN RETURN
6067 FOR y=1 TO 5e9
6068 IF INKEY$="n" THEN STOP
6069 IF INKEY$="s" THEN GOTO 90
6070 LET g=.6: SOUND g,0: SOUND g*2/3,0: SOUND g/3,0: SOUND g,0: SOUND g*2/3,3: SOUND g/3,2: SOUND g*2/3,2: SOUND g/3,0: SOUND g*2/3,0: SOUND g/3,-1: SOUND g,0
6080 NEXT y
6100 INPUT "SEU NOME ?",A$
6101 LET ms=s: LET mt=t
6110 PRINT AT 4,16; PAPER 1;"MUITO BEM! VOCE"
6111 PRINT AT 5,16; PAPER 1;"GANHOU. SEU ": PRINT AT 6,16; PAPER 1;"PLACAR E DE: "
6112 PRINT AT 7,16; PAPER 1;s;" PTS EM "
6113 LET VA=2: GOSUB 6057
6114 FOR r=1 TO 10000
6115 IF INKEY$="s" THEN GOTO 90
6116 IF INKEY$="n" THEN STOP
6117 FOR x=1 TO 3: SOUND .3,7: SOUND .15,4: SOUND .15,4: SOUND .3,7: NEXT x: SOUND .6,0:
6118 IF INKEY$="n" THEN STOP
6119 NEXT r
6150 IF mt<t THEN GOTO 6100
6155 IF mt>t THEN GOTO 6050
6156 REM
6157 REM
6158 REM
6159 REM
8000 CLS : BORDER 6: PAPER 1: LET v=19: LET h=16
8001 LET fase=fase+1: LET bonus=bonus+s: LET s=0
8002 PRINT AT [illegible]; FLASH 1;fase;" FASE"
8003 FOR x=1 TO 3: SOUND .3,7: SOUND .15,4: SOUND .15,4: SOUND .3,7: NEXT x: SOUND .6,0:
8004 CLS : BORDER 0: PAPER 0
8005 GOSUB 4000
8006 PRINT AT v,h; INK 6;"C"
8007 GOTO 1000
8010 SAVE "pacman"
8020 RUN
```

Pac-Man

Aliens

Maurício Ribeiro Palermo

Os *Aliens*, após fugirem de seu planeta numa nave espacial terrestre, preparam uma total invasão à Terra. Apesar de serem medonhos, são inteligentes e aprendem a usar alguns recursos da nave, inclusive uma tripla barreira protetora e uma pequena nave para invasões! Cabe a nós, terrestres, evitar isto com nosso canhão laser e nossos mísseis de duplo impacto.

Como a nave com os *Aliens* está situada numa altura muito grande, o laser não a alcança, só sendo eficaz para destruir partes da barreira. Os mísseis, que suportam dois choques, são as únicas armas capazes de eliminar os invasores! Observe que o raio tem ação imediata, enquanto que os mísseis levam algum tempo até chegar ao alvo, sendo necessário *prever* o momento do disparo. Somente um míssil pode estar sendo usado a cada instante.

A nave terrestre onde estão os *Aliens*, no entanto, é bastante forte agüentando muitos impactos de bombas. O único modo de se evitar uma total invasão alienígena é acertar um míssil exatamente sob as estranhas criaturas na nave, isso após ter destruído com outro míssil a parte da nave abaixo do respectivo *Alien*...

ALIENS GERADORES

Existem três *Aliens* na nave invasora que quando se multiplicam dão origem a uma invasão. Como estes são os geradores dos outros, a eliminação desses *Aliens* é o suficiente para acabar com a invasão, mesmo que outros já tenham tomado alguns pontos da Terra.

O jogo termina ou com a conclusão da invasão alienígena ou com uma vitória dos defensores! A vitória terrestre se dará com a eliminação dos três *Aliens* da nave; e a invasão dos inimigos completar-se-á quando os cinco prédios de sua cidade são destruídos e tomados por eles. Para destruir a barreira e os alienígenas, você tem um estoque limitado e não renovável de bombas e laser.

As bombas só são disparadas na direção vertical e o laser tem três posições de disparo: esquerda, acima e direita, sendo este último limitado por não alcançar a barreira superior. A escolha das teclas se dará no jogo e todas as informações são claramente indicadas.

No caso de vitória, várias informações são dadas, inclusive um analisador da sua pontuação. No mais, vale a pena digitar este jogo (são 14 Kb de programa – junto com a tela e variáveis que ocupam mais ou menos 7 Kb, dá 21 Kb –, por isso incompatível com micros de 16 Kb), que é recheado de efeitos especiais de som, imagem e cores, dando mais ação ao jogo (uma grande falha da maioria dos programas em BASIC).

COMO JOGAR

Para jogar, tem que se responder obrigatoriamente à pergunta treino ou realidade, sendo que a diferença é que no primeiro o laser é ilimitado e a nave dos *Aliens* segue uma trajetória fixa, ao contrário da opção realidade onde a nave às vezes encosta na lateral da tela e outras não (isto dificulta muito pois não se pode quase *prever* a hora certa de disparar um míssil para chegar exatamente sobre o invasor. Porém, nas duas opções as bombas são limitadas!).

Em seguida, responde-se à pergunta opcional (se não quiser, aperte uma tecla que não uma das indicadas) sobre a utilização do joystick ou teclado; em ambos os casos todas as informações são claramente indicadas.

O programa foi montado num conjunto de sub-rotinas que executam uma função cada, todas indicadas na listagem, sendo o seu mecanismo somente a parte indicada como LOOP PRINCIPAL, daí a agilidade do mesmo.

Os caracteres definíveis pelo usuário são usados ao máximo, pois durante o decorrer do jogo ocorre uma rápida e imperceptível redefinição de caracteres devido à falta de outros mais. Todas as variáveis utilizadas estão inicializadas de forma clara no início do programa, sendo seus nomes indicadores de sua função. Atenção: para gravar, digite SAVE "ALIENS" LINE 10 (ENTER). E boa sorte!

Maurício Ribeiro Palermo estuda engenharia eletrônica na UFRJ e desenvolve programas nas linguagens BASIC, FORTRAN, Pascal, FORTH e Assembler Z80, em seu TK90X.

```
  10 REM ========================
  20 REM =  Jogo  ALIENS 48K     =
  30 REM =por Mauricio Palermo =
  40 REM =   em    12/86         =
  50 REM = TK 90X & ZX Spectrum=
  60 REM ========================
  70 REM
  80 BORDER 6: PAPER 5: INK 1
  90 CLEAR 49999
  99 REM [illegible]
 100 LET bombas=32
 110 LET laser=171
 120 LET cnave=10
 130 LET hbomba=20
 140 LET psbomba=0
 150 LET pontos=0
 160 LET alimov=599
 170 LET psnave=-1
 180 LET aliens=3
 190 LET e$="QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ"
 200 LET a$="          "
 210 LET b$="          "
 220 LET c$="          "
 230 LET d$="          "
 240 DIM m$(3,33)
 250 LET m$(1)="                              "
 260 LET m$(2)=m$(1)
 270 LET m$(3)=m$(1)
 280 LET p$="
"
 290 LET byte=PEEK 50000
 300 DIM k$(4)
 310 LET facil=0
 320 LET dificil=0
 330 LET cini=0
 499 REM [illegible]
 500 GOSUB 9500: GOSUB 3700
 510 IF byte THEN GOTO 550
 520 PRINT FLASH 1;AT 15,2;"ALIENIGENAS SE APROXIMANDO"
 530 GOSUB 3000
 540 PRINT AT 15,0;p$
 550 PRINT AT 20,20;"A": GOSUB 3000
 560 GOSUB 5800
 570 PRINT FLASH 1;AT 0,0;"  A  L  I  E  N  S"
 580 PAUSE 60: GOSUB 6000
 590 PRINT AT 0,0;p$
 600 GOSUB 7300
 999 REM [illegible]
1000 PRINT PAPER 4;AT 5,0;m$(1); PAPER 5;m$(2); PAPER 7;m$(3)
1010 PRINT INK 3;AT 3,cnave;a$;AT 4,cnave;b$;AT 5,cnave;c$;AT 6,cnave;d$
1020 IF bombas<>0 THEN PRINT INK 3; PAPER 8;AT hbomba,15;"Q": IF psbomba<>0 THEN PRINT PAPER 8;AT hbomba-psbomba,15;" "
1030 SOUND 0,202,0
1040 LET i$=INKEY$
1050 IF i$=k$(1) AND psbomba=0 AND bombas>=0 THEN LET psbomba=-1: GOSUB 6100
1060 IF laser=0 THEN GOTO 1100
1070 IF i$=k$(2) THEN GOSUB 3000
1080 IF i$=k$(3) THEN GOSUB 8050
1090 IF i$=k$(4) THEN GOSUB 8100
1100 IF hbomba=12 OR hbomba=10 OR hbomba=8 THEN GOSUB 6300
1110 IF hbomba=6 THEN GOSUB 6200
1120 LET alimov=alimov-1: IF alimov/100=INT (alimov/100) THEN GOSUB 5000
1130 LET m$(1)=m$(1,2 TO )+m$(1,1): LET m$(2)=m$(2,31)+m$(2, TO 30): LET m$(3)=m$(3,2 TO )+m$(3,1)
1140 LET hbomba=hbomba+psbomba
1150 LET cnave=cnave+psnave
1160 IF cnave=cfin THEN LET psnave=-1: LET cini=(INT (RND*4))+dificil
1170 IF cnave=cini THEN LET psnave=1: LET cfin=15-facil+(12+INT (RND*4))-dificil
1180 GOTO 1000
2999 REM [illegible]
3000 PRINT FLASH 1;AT 15,4;"Opcao Treino ou Realidade": GOSUB 3500: GOSUB 3600: IF i$<>"t" AND i$<>"r" THEN GOTO 3000
3010 LET facil=i$="t": LET dificil=i$="r": PRINT AT 15,0;p$
3020 PRINT FLASH 1;AT 15,5;"TECLE J PARA TECLAS": GOSUB 3500: GOSUB 3600: PRINT AT 15,0;p$
3030 IF i$<>"1" THEN RETURN
3040 PRINT AT 15,4;"Joystick ou Teclado?"
3050 GOSUB 3600: IF i$="" OR (i$<>"j" AND i$<>"t") THEN GOTO 3050
3060 IF i$="j" THEN GOTO 3300
3070 PRINT INVERSE 1;AT 8,0;"  BOMBAS           LASER ESQUERDA     LASER ACIMA        LASER DIREITA "
3080 PRINT FLASH 1;AT 15,4;"Toque em CINCO teclas";AT 8,0;: SOUND 1,30
3090 FOR i=1 TO 4
3100 GOSUB 3600
3110 LET k$(i)=i$: POKE 40000+i,CODE k$(i)
3120 PRINT INVERSE 1;i$
3130 SOUND 0.5,30
3140 NEXT i
3150 FOR i=8 TO 11
3160 PRINT AT i,0;p$
3170 SOUND 0.1,i
3180 NEXT i
3190 PRINT AT 15,0;p$
3200 RETURN
3300 PRINT INVERSE 1;AT 8,0;"Tiro BOMBA        <- LASER ESQUERDA   ↑  LASER ACIMA        -> LASER DIREITA "
3310 DATA "0","6","9","7"
3320 RESTORE 3310
3330 FOR i=1 TO 4
3340 READ k$(i): POKE 40000+i,CODE k$(i)
3350 NEXT i
3360 PRINT AT 15,4;"TECLA para comecar o jogo"
3370 SOUND 1,30: GOSUB 3600
3380 FOR i=8 TO 11
3400 SOUND 0.1,i
3410 NEXT i
3420 PRINT AT 15,0;p$
3430 RETURN
3499 REM [illegible]
3500 FOR i=24 TO 12 STEP -2
3510 SOUND 0.01,i
3520 NEXT i
3530 FOR i=12 TO 26 STEP 2
3540 SOUND 0.01,i
3550 NEXT i
3560 RETURN
3599 REM [illegible]
3600 LET i$=INKEY$: IF i$="" THEN GOTO 3600
3610 RETURN
3699 REM [illegible]
3700 FOR i=1 TO 4
3710 LET k$(i)=CHR$ PEEK (40000+i)
3720 NEXT i
3730 RETURN
3999 REM [illegible]
4000 POKE 50047,71
4010 LET u=USR 50041
4020 PAUSE 50: POKE 50047,22
4030 LET u=USR 50041
4040 POKE 50014,255: POKE 50017,20: LET u=USR 50000: POKE 50017,01: FOR i=1 TO 100: BORDER 2: POKE 50014,i: LET u=USR 50000: BORDER 6: NEXT i
4050 POKE 50047,41: LET u=USR 50041: FOR i=3 TO 6: PRINT AT i,0;p$: PAUSE 10: NEXT i: PAUSE 100
4060 BORDER 5: PAPER 5: CLS
4070 FOR j=1 TO 3: FOR i=1 TO 20
4080 PRINT BRIGHT 1; PAPER INT (RND*7)+1; INK 9;AT 3,1;"PARABENS INVASORES ELIMINADOS!"
4090 SOUND 0.002,i+30
4100 NEXT i: NEXT j
4110 PRINT PAPER 2; INK 7;AT 5,0;"Movimentos dos ALIENS ";600-alimov: SOUND 0.5,11
4120 LET pontos=(alimov+bombas*5+laser)*10: PRINT '' PAPER 4; INK 7;"SEUS pontos sao ";pontos;'' PAPER 3; INK 7;"Sua PERFORMANCE foi ";("SOFRIVEL" AND pontos<2000);("RAZOAVEL" AND pontos>=2000 AND pontos<3000);("BOA" AND pontos>=3000 AND pontos<4000);("MUITA SORTE" AND pontos>=4000): SOUND 0.5,20
4130 PRINT '' PAPER 7; INK 0; FLASH 1;"TECLA PARA DESTRUI-LOS NOVAMENTE"
4140 PAUSE 60: SOUND 0.2,20: SOUND 0.5,16
4150 GOTO 4500
4499 REM [illegible]
4500 PAUSE 0
4510 BORDER 6: PAPER 5: INK 1: CLS
```

ALIENS

```
4520 CLEAR
4530 GOTO 100
4999 REM [illegible]
5000 PRINT FLASH 1;AT 15,5;"INVA
SAO ALIENIGENA!"
5010 FOR i=1 TO 5: SOUND 0.3,24:
 SOUND 0.6,12: NEXT i
5020 DATA 0,5,10,25,28
5030 RESTORE 5020
5040 FOR i=1 TO aliмov/100: READ
 col: NEXT i
5050 RESTORE 9990: FOR i=USR "s"
 TO USR "t"+7: READ byte: POKE i
,byte: NEXT i
5060 PRINT AT 8,0;p$;AT 10,0;p$;
AT 12,0;p$;AT 15,0;p$
5070 LET pssinv=1
5080 LET colinv=INT (RND*29)
5090 PRINT FLASH 1; INK 2;AT 13,
colinv;"□": LET u=USR 50058: PR
INT AT 13,colinv;" "
5100 LET colinv=colinv+pssinv
5110 IF colinv=29 THEN LET pssin
v=-1
5120 IF colinv=0 THEN LET pssinv
=1
5130 IF colinv<>col THEN GOTO 50
90
5150 FOR i=16 TO 20
5160 FOR j=-2 TO 0
5170 PRINT INK 3;AT i+j,col;"[graphic]"
;AT i+j+1,col;"[graphic]": PAUSE 20
5180 PRINT AT i+j,col;" ";AT i+
j+1,col;" "
5190 NEXT j: GOSUB 7500: NEXT i
5200 PRINT INK 3;AT 20,col;"[graphic]";
AT 21,col;"[graphic]"
5210 RESTORE 9997
5220 FOR i=USR "s" TO USR "t"+7:
 READ byte: POKE i,byte: NEXT i
5230 IF alimov<>100 THEN RETURN
5235 REM [illegible]
5240 FOR i=0 TO -30 STEP -3: SOU
ND 0.15,i: NEXT i: SOUND 2,-30:
POKE 50047,25: LET u=USR 50041
5250 FOR i=1 TO 150: LET l=2+INT
 (RND*10): LET c=2+INT (RND*15)
5260 PRINT INK INT (RND*7)+1; PA
PER 3;AT l,c;"[graphic]";AT l+1,c;"[graphic]"
5270 SOUND 0.01,RND*30
5280 NEXT i
5300 PRINT INVERSE 1;AT 2,0;"Voc
e se revelou um fracasso como
   ": FLASH 1;"DEFENSOR": FLA
SH 0;"...             ": FOR i=1
TO 3: SOUND 0.1,20: NEXT i: SOUN
D 0.5,20
5310 PAUSE 100: PRINT INVERSE 1;
"Agora nosso Mundo ficou ": FLA
SH 1;"MAGENTA!": SOUND 1,-30
5320 PRINT FLASH 1;AT 15,3;"TECL
A PARA TENTAR DE NOVO"
5330 GOTO 4500
5999 REM [illegible]
6000 POKE 50017,02
6010 FOR i=1 TO 20
6020 POKE 50014,i
6030 LET u=USR 50000
6040 NEXT i
6050 FOR i=20 TO 1 STEP -1
6060 POKE 50014,i
6070 LET u=USR 50000
6080 NEXT i
6090 RETURN
6099 REM [illegible]
6100 POKE 50017,20: POKE 50014,2
0: LET u=USR 50000: RETURN
6199 REM [illegible]
6200 LET hbomba=20
6210 LET pssbomba=0
6220 GOSUB 7300
6230 IF cnave>12 OR cnave<3 THEN
 RETURN
6240 GOSUB 7100
6250 IF d$(16-cnave)<>" " THEN L
ET d$(16-cnave)=" ": RETURN
6260 IF z$(16-cnave)<>" " THEN G
OSUB 6800
6270 RETURN
6299 REM [illegible]
6300 LET barr=(hbomba-6)/2
6310 IF m$(barr,16)=" " THEN RET
URN
6320 LET m$(barr,16)=" "
6330 GOSUB 7000
6340 GOSUB 7100
6350 RETURN
6599 REM [illegible]
6600 LET laser=laser-3+dif:c:1:
PLOT 252,175: DRAW INK 0; PAPER
7;0,laser-175: RETURN
6699 REM [illegible]
6700 POKE 50014,i: POKE 50017,07
: LET u=USR 50000: RETURN
6799 REM [illegible]
6800 PRINT FLASH 1;AT 15,6;"MORR
AM ALIENIGENAS!": PAUSE 120
6810 FOR i=15 TO 17
6820 LET c$(i-cnave)=" ": LET b$
(i-cnave)=" "
6830 NEXT i
6840 LET aliens=aliens-1
6850 PRINT AT 15,0;p$
6860 FOR i=0 TO 8: PRINT INK 8;A
T 11-i,16-i;"[graphic]";AT 11-i,17+i;"[graphic]"
;AT 12+i/2,16-i;"[graphic]";AT 12+i/2,17
+i;"[graphic]": GOSUB 6700
6870 PRINT INK 8;AT 11-i,16-i;"
";AT 11-i,17+i;" ";AT 12+i/2,16-
i;" ";AT 12+i/2,17+i;" ": NEXT i
6880 IF aliens=0 THEN GOTO 4000
6890 RETURN
6899 REM [illegible]
6900 POKE 50014,01: POKE 50017,1
0: FOR i=1 TO 4: LET u=USR 50000
: NEXT i: RETURN
6999 REM [illegible]
7000 LET pssbomba=pssbomba+0.5
7010 IF pssbomba=0 THEN LET hbom
ba=20: GOSUB 7300
7020 RETURN
7099 REM [illegible]
7100 POKE 50017,01: FOR i=1 TO 2
0: BORDER 2: POKE 50014,i: LET u
=USR 50000: BORDER 6: NEXT i: RE
TURN
7299 REM [illegible]
7300 LET bombas=bombas-1: IF bom
bas<0 THEN RETURN
7310 PRINT #0; PAPER 7;AT 1,0;p$
;AT 1,0,e$; TO bombas)
7320 RETURN
7499 REM [illegible]
7500 POKE 50017,02: POKE 50014,2
55: LET u=USR 50000: RETURN
7999 REM [illegible]
8000 PLOT INK 2;163,16
8010 IF m$(3,14)<>" " THEN DRAW
INK 2;-55,55: PRINT AT 12,13; PA
PER 2;" ": LET m$(3,14)=" ": GOS
UB 6900: DRAW OVER 1;55,-55: GOS
UB 7100: GOTO 8040
8020 IF m$(2,12)<>" " THEN DRAW
INK 2;-75,75: PRINT AT 10,11; PA
PER 2;" ": LET m$(2,12)=" ": GOS
UB 6900: DRAW OVER 1;75,-75: GOS
UB 7100: GOTO 8040
8030 IF m$(1,10)<>" " THEN DRAW
INK 2;-90,90: PRINT AT 8,9; PAPE
R 2;" ": LET m$(1,10)=" ": GOSUB
 6900: DRAW OVER 1;90,-90: GOSUB
 7100: GOTO 8040
8035 RETURN
8040 GOSUB 6600: RETURN
8049 REM [illegible]
8050 PLOT INK 2;163,16
8060 IF m$(3,21)<>" " THEN DRAW
INK 2;0,60: PRINT AT 12,20; PAPE
R 2;" ": LET m$(3,21)=" ": GOSUB
 6900: DRAW OVER 1;0,-60: GOSUB
7100: GOTO 8090
8070 IF m$(2,21)<>" " THEN DRAW
INK 2;0,75: PRINT AT 10,20; PAPE
R 2;" ": LET m$(2,21)=" ": GOSUB
 6900: DRAW OVER 1;0,-75: GOSUB
7100: GOTO 8090
8080 IF m$(1,21)<>" " THEN DRAW
INK 2;0,90: PRINT AT 8,20; PAPER
 2;" ": LET m$(1,21)=" ": GOSUB
6900: DRAW OVER 1;0,-90: GOSUB 7
100: GOTO 8090
8085 RETURN
8090 GOSUB 6600: RETURN
8099 REM [illegible]
8100 PLOT INK 2;163,16
8110 IF m$(3,28)<>" " THEN DRAW
INK 2;55,55: PRINT AT 12,27; PAP
ER 2;" ": LET m$(3,28)=" ": GOSU
B 6900: DRAW OVER 1;-55,-55: GOS
UB 7100: GOTO 8140
8120 IF m$(2,30)<>" " THEN DRAW
INK 2;75,75: PRINT AT 10,29; PAP
ER 2;" ": LET m$(2,30)=" ": GOSU
B 6900: DRAW OVER 1;-75,-75: GOS
UB 7100: GOTO 8140
8130 RETURN
8140 GOSUB 6600: RETURN
8799 REM [illegible]
8800 PRINT INK 8;AT 0,10;a$: GOS
UB 8900: PRINT INK 8;AT 0,10;c$;
AT 1,10;d$: GOSUB 8900: PRINT IN
K 8;AT 0,10;b$;AT 1,10;c$;AT 2,1
0;d$: GOSUB 8900
8810 PRINT INK 8;AT 0,10;a$;AT 1
,10;b$;AT 2,10;c$;AT 3,10;d$: GO
SUB 8900: PRINT INK 8;AT 0,0;p$;
AT 1,10;a$;AT 2,10;b$;AT 3,10;c$
;AT 4,10;d$: GOSUB 8900: PRINT I
NK 8;AT 1,0;p$;AT 2,10;a$;AT 3,1
0;b$;AT 4,10;c$;AT 5,10;d$
8820 GOSUB 8900: PRINT INK 8;AT
2,0;p$;AT 3,10;a$;AT 4,10;b$;AT
5,10;c$;AT 6,10;d$
8830 FOR i=1 TO 3: GOSUB 8900: N
EXT i: RETURN
8900 REM [illegible]
8910 FOR i=30 TO 0 STEP -1: SOUN
D 0.002,i: NEXT i: RETURN
8998 REM [illegible]
9000 RESTORE 9980: FOR e=50000 T
O 50080: READ byte: POKE e,byte:
 SOUND 0.002,RND*30+20: NEXT e
9010 RESTORE 9994: FOR e=USR "a"
 TO USR "u"+7: READ byte: POKE e
,byte: SOUND 0.002,RND*30+20: NE
XT e
9020 RETURN
9500 REM [illegible]
9510 FOR i=17 TO 21: PRINT INK 7
;AT i,0;"[graphic]": INK 6;AT i,5;"[graphic]";
 INK 4;AT i,10;"[graphic]": INK 2;AT i,
15;"[graphic]": INK 1;AT i,25;"[graphic]": NEX
T i
9520 PRINT INK 6;AT 21,20;"[graphic]"
9530 PLOT 112,0: DRAW 20,0: DRAW
 0,7: DRAW -1,0: DRAW 0,-7: DRA
W -1,7: DRAW 7,0: DRAW -7,-7: DR
AW 0,7: DRAW -1,0: DRAW -7,-7
9540 PRINT INK 7;AT 0,0;p$: INK
6;p$;p$: INK 4;p$: INK 3;p$;p$:
INK 2;p$: INK 1;p$
9550 PRINT #0; PAPER 7;AT 0,0;p$
;p$
9560 FOR i=0 TO 21: PRINT INK 0;
AT i,31;"[graphic]": NEXT i: PLOT 252,0:
 DRAW OVER 1; INK 0; PAPER 7;0,1
70
9570 PRINT #0; PAPER 7;AT 0,0;"B
ombas": INK 2;AT 0,27;"Laser"
9580 RETURN
9979 REM [illegible]
9980 DATA 58,72,92,203,63,203,63
,203,63,17,01,00,01,254,01,33,00
,03,197,70,203,120,06,00,40,02,2
03,224,168,237,121,193,197,16,25
4,237,82,32,235,193,201,33,00,00
,22,64,62,00,06,32,119,35,16,252
,21,32,247,201,22,01,62,22,211,2
54,66,16,254,62,06,211,254,66,16
,254,20,62,255,186,32,238,201
9989 REM [illegible]
9990 DATA 3,7,31,19,115,127,255,
55,192,224,248,200,206,254,255,8
4
9993 REM [illegible]
9994 DATA 7,15,24,63,96,127,192,
255,255,255,0,255,0,255,0,255,22
4,240,24,252,6,254,3,255,63,127,
255,255,63,63,255,255,255,248,24
0,240,240,240,248,255,4,12,2,12,
51,1,12,16,44,32,252,176,192,19
2,137,255,31,15,15,15,15,31,255,
252,254,255,255,252,252,255,255,
63,63,255,255,63,63,255,255,67,7
,13,2,4,4,2,13,240,229,224,32,16
,0,0,0,0,255,0,255,128,0,0,0,0,0
,0,0,0,0,255,0,255,255,127,121,181
,255,255,153,153,153,153,153,153
,255,255,144,152,152,156,158,159
,255,56,56,56,56,56,16,56
9996 REM [illegible]
9997 DATA 255,255,192,192,192,19
2,255,255,254,254,6,6,6,6,254,25
4
9998 REM [illegible]
9999 DATA 24,24,60,36,231,189,12
9,189
```

Aliens

Softnew Informática

TUDO PARA O COLOR E MSX!!!

Tradicional em softwares para o CP-400 e MSX.
Imensa variedade de softwares, 2800 programas para o CP-400
e 450 programas para o MSX.

CP-400

JOGOS - Cz$ 9,50

Aplicativos e Utilitários — Cz$ 95,00

- cocomax I e II — Cz$ 180,00
- vip-library — Cz$ 600,00
- vizidraw — Cz$ 180,00
- OS9-sist. operacional — Cz$ 789,00
- livro 500 peeks e pokes e exec's - traduzido — Cz$ 140,00
- adventures em português — Cz$ 9,50
- minimax — Cz$ 180,00
- deskmate — Cz$ 450,00
- pen-pal — Cz$ 450,00
- copiladores — Cz$ 250,00
- joysticks — Cz$ 299,00

MSX

JOGOS - Cz$ 29,00

- Aplicativos e Utilitários — Cz$ 110,00
- Copiadores — Cz$ 350,00
- controle de estoque (disco) — Cz$ 250,00
- Master Voice (sintetizador de voz) — Cz$ 250,00

PROMOÇÃO POR TEMPO LIMITADO!!!

Nosso sistema de trabalho:

Cobramos uma taxa de Cz$ 40,00 correspondente a fita cassete, sendo que nessa fita podem ser colocados aproximadamente 30 programas para CP-400 e 10 programas para MSX. Caso deseje que sejam divididos os programas em várias fitas, nos informe o número de fitas.
OBS.: - Encomenda mínima Cz$ 290,00 - Taxa de correio Cz$ 45,00.
Despachamos para qualquer lugar do Brasil (Via Cedex). A fita e/ou disco será entregue em sua residência.
Caso você, deseje que seus programas sejam colocados em disketes, o valor do mesmo é de Cz$ 85,00

Atendemos todos os dias no horário comercial e aos sábados das 8 às 14 hs.

Solicite nosso catálogo hoje mesmo, e quando recebê-lo envie seu pedido e a quantia através de cheque, dinheiro ou Vale Postal para a agência Casa Verde, no valor correspondente ao seu pedido.

SOFTNEW

Rua Miguel Maldonado, 173 - Bairro Jardim São Bento - São Paulo - SP - Tel.: (011) 266-2902/266-5307 CEP 02524

Sofistique os programas do seu ZX Spectrum (TK90X) com a exibição de menus, mensagens de aviso, diretórios etc., utilizando-se dos frames.

Frame para o TK90X

Álvaro Ferreira de Freitas Borja

Como usuário de um micro da linha IBM-PC, convivo diariamente com softwares do tipo Open Access, PCPAINT, Turbo PROLOG etc., e uma das coisas que mais acho interessante na interação destes programas com o usuário é a presença de frames para apresentar menus, mensagens de aviso, diretório etc.

Para os que ainda não tiveram contato com esses programas, procurarei explicar brevemente o que são frames.

São quadros (frame significa quadro em inglês) que são colocados *sobre* a imagem que está na tela, como por exemplo um menu de opções, o qual após o usuário fazer a escolha é retirado, sendo restabelecido o conteúdo da tela como se nada tivesse acontecido.

Isto representa uma grande sofisticação para os softwares, pois, para apresentarmos um menu(ou qualquer outra coisa), não é necessários limpar a tela (a não ser que se queira isto), além de ter um grande impacto visual sobre quem está operando o programa. Até hoje não vi quem não achasse interessante este efeito.

Muito bem, agora que todos já têm uma noção sobre o que é um frame e para o que serve, vamos conhecer o frame para o ZX Spectrum (TK90X).

Embora ele esteja escrito totalmente em linguagem de máquina, o programador não é obrigado a saber Assembler para poder desfrutar de suas vantagens em seus programas em BASIC, uma vez que a chamada e a entrada dos parâmetros do frame são feitas todas em BASIC.

Pensando nisto é que procurei fazer um programa que fosse praticamente à prova de erros, digo praticamente pois, segundo as "Leis de Murph", se alguma coisa tem a mais remota chance de dar errado, pode ter certeza que dará. Todavia, não se assustem, visto que durante os testes feitos por mim, nunca houve problemas.

```
-----------------------------------------------------------------+
LIVRE| CARFRM |  PROG.  | CONTADOR | TABELAS |  ATI  |  UDG's   |
-----------------------------------------------------------------+
     ↑        ↑         ↑          ↑         ↑       ↑          ↑
   53925    54000     54860      54861     55086   65365     65535
```

Figura 1

Na figura 1, temos o posicionamento do programa e suas variáveis na RAM, sendo reservados 10279 bytes para o Arquivo Temporário de Imagens (ATI) e ficando livres 30170 bytes para programas.

Poderão ser colocados frames sucessivos, isto é, não é necessário retirar um frame que esteja na tela para colocar outro, bastando para tal que haja espaço disponível no ATI. Sendo que por problemas de poupar memória, ficou limitado o número de frames em 25, o que é bastante razoável, pois dificilmente, em condições normais, alguém irá colocar mais de 25 quadros ao mesmo tempo na tela, mesmo porque certamente não terá espaço disponível.

Os frames obedecem o sistema de pilha, ou seja: o último que é colocado é o primeiro a ser retirado. Entretanto, caso você tenha colocado cinco frames, por exemplo, e resolva dar um CLS na tela, haverá duas alternativas: retirar todos os frames e depois executar o CLS ou então zerar o contador de frames e executar o CLS, operação esta feita por meio de um comando **POKE 54860,0.**

É aconselhável colocar este comando na primeira linha de execução do programa, pois assim toda vez que este for rodado você terá certeza de que não há um frame armazenado, tendo portanto toda a área do ATI livre. Quando não houver espaço no ATI, aparecerá a mensagem de erro "MEMÓRIA INSUFICIENTE".

Entretanto se for utilizado de modo racional, dificilmente haverá um problema deste tipo.

A chamada do comando é feita por meio de um RAND USR 54000, cujo parâmetro (54000) sugiro seja armazenado em uma variável, logo no início do programa (podendo ficar na mesma linha do POKE 54860,0), exemplo: 10 FRAME=54000: POKE 54860, 0. Assim a chamada será feita por meio de RAND USR FRAME, o que paralelamente é uma boa maneira de saber qual rotina estamos acessando.

Para criarmos um frame deveremos usar o CFRAME(Criar Frame), cuja sintaxe é a seguinte: CFRAME L1,C1,L2,C2,COR

Onde:

L1,C1 – são as coordenadas do canto superior esquerdo;

L2,C2 – são as coordenadas do canto inferior direito;

COR – é o parâmetro de cor do frame.

A representação no vídeo destas coordenadas e parâmetros pode ser vista na figura 2, sendo que estes podem variar dentro dos seguintes intervalos: L1(0 a 20), L2(1 a 21), C1 (0 a 30) e C2(1 a 31).

Caso L2(C2) seja menor ou igual a L1(C1), aparecerá a mensagem "ERRO DE PARAMETRO".

Os parâmetros de cor estão expressos na

```
+--------------------------------------------+
| (L1,C1)                                    |
|        +=====================+             |
|        |                     |             |
|        |                     |             |
|        +=====================+             |
|                               (L2,C2)     |
+--------------------------------------------+
```

Figura 2

tabela da figura 3. Para obter um frame com BRIGHT ON, basta adicionar 64 aos valores dessa tabela e para obter um frame com FLASH ON, deveremos adicionar 128.

Obs.: a cor da borda do frame obedece à cor do INK.

Observe também que o espaço após a palavra-chave CFRAME é obrigatório, caso não seja colocado será acusado "ERRO DE SINTAXE".

O TK90X não aceita linhas com palavras-chave fora do padrão, assim, duvido que alguém consiga digitar uma linha do tipo 200 CFRAME 0,0,10,8,56(se alguém quiser tentar). Para superar esta limitação, devemos colocar o CFRAME em uma linha REM da seguinte forma: 200 REM CFRAME 0,0,10,8, 56. Aqui vale fazer uma ressalva: esta linha deverá ser inserida logo após a linha de chamada, pois qualquer variação nesta imposição provocará um "ERRO DE SINTAXE". A seguir temos um exemplo de uso:

```
100 RAND USR FRAME (ou RAND USR 54000)
200 REM CFRAME 0,0,8,20,40
```

Agora que já sabemos como criar um frame, vamos ver como fazer para retirar (ou remover) um frame (não esquecer a teoria da pilha).

Basicamente o procedimento é o mesmo, sendo que a palavra-chave é RFRAME(retira frame), não sendo necessário especificar parâmetros. A seguir temos um exemplo de uso:

```
300 RAND USR FRAME
400 REM RFRAME
```

Aqui vale fazer mais uma ressalva: tanto CFRAME como RFRAME devem ser digitados com letras maiúsculas, caso contrário teremos um "ERRO DE SINTAXE". Além disso, se for dado um RFRAME sem haver um frame no ATI, será impressa a mensagem de erro "FIM DE DADOS".

Como já deu para perceber, a rotina procura checar o comando antes de executá-lo, visando, assim, evitar um *crash* do sistema.

PAPER	INK: PRETO	AZUL	VERM.	MAGEN	VERDE	CIANO	AMARE	BRANC
PRETO	00	01	02	03	04	05	06	07
AZUL	08	09	10	11	12	13	14	15
VERM.	16	17	18	19	20	21	22	23
MAGEN	24	25	26	27	28	29	30	31
VERDE	32	33	34	35	36	37	38	39
CIANO	40	41	42	43	44	45	46	47
AMARE	48	49	50	51	52	53	54	55
BRANC	56	57	58	59	60	61	62	63

Figura 3

Agora vamos ver como proceder para carregar a rotina na memória do micro.

CARREGANDO A ROTINA

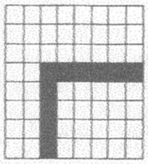
Inicialmente, devemos pokear a memória com os códigos hexadecimais da LISTAGEM 1, utilizando para tal tarefa o "Monitor Assembler", publicado em MS nº 51, ou qualquer outro. Para protegermos os códigos contra possíveis alterações, devemos antes de mais nada baixar o topo da RAM através de um CLEAR 53925.

Após digitar e conferir tudo, é aconselhável salvar antes de utilizar (pois pode acontecer alguma falha na energia), para tanto, digite SAVE "FRAME"CODE 53925,940. Feito isto, toda vez que quisermos carregá-lo bastará dar o CLEAR 53925 e LOAD""CODE.

Caso você deseje mudar o desenho da borda do frame, apresento a seguir a estrutura da tabela de caracteres adotada pelo programa. Cada caractere é representado por uma matriz 8x8, que no nosso caso corresponde a oito bytes, sendo que o endereço de cada uma das partes que compõem o frame está representado abaixo:

```
Canto Superior Esquerdo     = 53936 a 53943
Borda Horizontal Superior   = 53944 a 53951
Borda Horizontal Inferior   = 53952 a 53959
Canto Superior Direito      = 53960 a 53967
Borda Vertical Esquerda     = 53968 a 53975
Borda Vertical Direita      = 53976 a 53983
Canto Inferior Esquerdo     = 53984 a 53991
Canto Inferior Direito      = 53992 a 53999
```

Para saber qual o valor a ser atribuído a cada byte, você deverá usar um procedimento igual ao adotado para redefinir caracteres, isto é, cada ponto aceso na tela representa 1 bit setado, exemplo:



```
00000000 = 0
00000000 = 0
00000000 = 0
00111111 = 63
00100000 = 32
00100000 = 32
00100000 = 32
00100000 = 32
```

Para finalizar, a LISTAGEM 2 apresenta um programa que demonstra como usar os comandos CFRAME e RFRAME.

Álvaro Ferreira de Freitas Borja está cursando engenharia civil, na UFPA. Atualmente, trabalha como Desenhista-Projetista na Telepará, sendo também usuário dos micros PC 2001 e TK 95.

Listagem 1

```
53928 00 00 00 00 00 00 00 00
53936 00 00 7f 40 5f 50 50 50
53944 00 00 ff 00 ff 00 00 00
53952 00 00 00 ff 00 ff 00 00
53960 00 00 fe 02 fa 0a 0a 0a
53968 50 50 50 50 50 50 50 50
53976 0a 0a 0a 0a 0a 0a 0a 0a
53984 50 50 50 5f 40 7f 00 00
53992 0a 0a 0a fa 02 fe 00 00
54000 2a 55 5c 7e 32 46 5c 23
54008 7e 32 45 5c 23 23 23 7e
54016 fe ea 28 23 cd 21 d6 cf
54024 0b 11 22 d3 06 05 1a be
54032 20 f8 23 13 10 f8 c9 3e
54040 20 be 20 e8 23 be 28 fc
54048 2b c9 46 52 41 4d 45 23
54056 7e fe 52 23 ca d7 d4 fe
54064 43 20 d1 cd 09 d3 cd 17
54072 d3 cd dc d5 ed 53 04 5b
54080 cd c7 d5 cd dc d5 ed 53
54088 08 5b cd cf d5 cd dc d5
54096 ed 53 06 5b cd c7 d5 cd
54104 dc d5 ed 53 0a 5b cd cf
54112 d5 cd dc d5 7b 32 18 5b
54120 21 04 5b cd bd d5 fe 15
54128 d2 09 d6 32 0c 5b 21 08
54136 5b cd bd d5 fe 1f 30 f0
54144 32 0a 5b 21 e0 57 11 20
54152 00 3a 04 5b 47 c5 04 19
54160 10 fd ed 5b 08 5b 19 22
54168 16 5b c1 a7 fe 08 30 0c
54176 21 e0 3f 04 11 20 00 19
54184 10 fd 18 13 fe 10 30 08
54192 21 e0 47 d6 08 47 18 eb
54200 21 e0 4f d6 10 18 f6 ed
54208 5b 08 5b 19 22 0e 5b 22
54216 10 5b eb 21 4d d6 cd 0e
54224 d6 73 23 72 ed 5b 16 5b
54232 21 fc d6 cd 0e d6 73 23
54240 72 21 b1 d6 cd 0e d6 3a
54248 04 5b 77 23 3a 0c 5b 77
54256 21 e3 d6 cd 16 d6 3a 0d
54264 5b 77 3a 0d 5b 26 00 6f
54272 3a 0c 5b 23 23 3c 44 4d
54280 09 3d 20 fc e5 06 07 54
54288 5d 19 10 fd d1 19 eb 21
54296 7f d5 cd 0e d6 23 23 e5
54304 3a 4c d6 b7 20 09 01 2e
54312 d7 2b 2b 71 23 70 23 2b
54320 46 2b 4e ed 43 12 5b eb
54328 a7 ed 4a 01 55 ff e5 38
54336 04 ed 42 38 05 cd 21 d6
54344 cf 0f d1 21 4c d6 34 e1
54352 73 23 72 cd 33 d6 3a 0d
54360 5b 57 14 14 7e 02 3a 18
54368 5b 77 03 23 15 20 f5 1d
54376 28 05 cd 26 d6 18 e7 ed
54384 43 12 5b 11 b0 d2 2a 0e
54392 5b cd 6b d5 11 b8 d2 ed
54400 53 14 5b cd 82 d5 11 c8
54408 d2 cd 6b d5 3a 0c 5b 47
54416 3a 04 5b 3c cd 53 d5 22
54424 10 5b 11 d0 d2 c5 f5 cd
54432 6b d5 11 a8 d2 ed 53 14
54440 5b cd 82 d5 11 d8 d2 cd
54448 6b d5 f1 c1 10 dd 3c cd
54456 53 d5 11 e0 d2 2a 0e 5b
54464 22 10 5b cd 6b d5 11 c0
54472 d2 ed 53 14 5b cd 82 d5
54480 11 e8 d2 cd 6b d5 c9 cd
54488 09 d3 21 4c d6 7e b7 20
54496 05 cd 21 d6 cf 07 35 21
54504 4d d6 cd 41 d6 ed 53 0e
54512 5b 21 7f d6 cd 41 d6 ed
54520 53 12 5b 21 fc d6 cd 41
54528 d6 ed 53 16 5b 21 b1 d6
54536 cd 0e d6 7e 32 04 5b 23
54544 7e 32 0c 5b 21 e3 d6 cd
54552 18 d6 7e 32 0d 5b cd 33
54560 d6 3a 0d 5b 57 14 14 0a
54568 77 03 23 15 20 f9 1d 28
54576 05 cd 26 d6 18 eb ed 43
54584 12 5b cd 98 d5 3a 0c 5b
54592 47 04 3a 04 5b 3c cd 53
54600 d5 c5 f5 cd 98 d5 f1 c1
54608 10 f3 c9 fe 08 28 0f fe
54616 10 28 0b 11 20 00 2a 0e
54624 5b 19 22 0e 5b c9 11 20
54632 07 18 f3 3e 08 ed 4b 12
54640 5b f5 7e 02 1a 77 24 13
54648 03 f1 3d 20 f4 ed 43 12
54656 5b c9 3a 0d 5b 47 cd d4
54664 d5 ed 5b 14 5b c5 cd 6b
54672 d5 c1 10 f2 cd d4 d5 c9
54680 3a 0d 5b 47 04 04 2a 0e
54688 5b 22 10 5b ed 5b 12 5b
54696 3e 08 f5 1a 77 24 13 f1
54704 3d 20 f7 cd d4 d5 10 f0
54712 ed 53 12 5b c9 46 23 23
54720 7e 3d a7 98 38 43 c9 7b
54728 1e 16 a7 9b 30 3b c9 7b
54736 1e 20 18 f6 2a 10 5b 23
54744 22 10 5b c9 11 00 00 23
54752 7e fe 0d c8 fe 20 c8 fe
54760 2c c8 d6 30 38 1b fe 0a
54768 30 17 e5 f5 21 00 00 3e
54776 0a 06 08 29 17 30 01 19
54784 10 f9 f1 4f 09 eb e1 18
54792 d6 cd 21 d6 cf 19 3a 4c
54800 d6 b7 c8 23 23 3d 18 fa
54808 3a 4c d6 b7 c8 23 3d 18
54816 fb af 32 4c d6 c9 d5 11
54824 20 00 2a 16 5b 19 d1 22
54832 16 5b c9 3a 0c 5b 5f 1c
54840 1c 2a 16 5b ed 4b 12 5b
54848 c9 cd 0e d6 5e 23 56 c9
54856 00 00 00
```

Listagem 2

```
  10 POKE 54860,0: LET FRAME=540
00
  20 LIST 40
  30 RAND USR FRAME
  40 REM CFRAME 0,0,10,8,56
  50 PRINT AT 0,1;"Menu 1";AT 2,
1;"EXEMPLO";AT 3,1;"DE USO"
  60 PAUSE 100
  70 RAND USR FRAME
  80 REM CFRAME 3,3,8,31,40
  90 PRINT AT 5,4;"PODE SER USAD
O PARA MOSTRAR";AT 6,4;"MENSAGEN
S P/ O USUARIO"
 100 PAUSE 100
 110 RAND USR FRAME
 120 REM CFRAME 4,1,13,30,48
 130 PRINT AT 8,4; PAPER 6; INK
0;" VAMOS RETIRAR OS FRAMES": PA
USE 50
 140 FOR I=1 TO 3: PAUSE 150
 150 RAND USR FRAME
 160 REM RFRAME
 170 NEXT I
 180 PAUSE 150: GOTO 30
```

Com este software, o usuário do TK90X poderá redefinir até cinco conjuntos de caracteres e melhorar visualmente os seus programas.

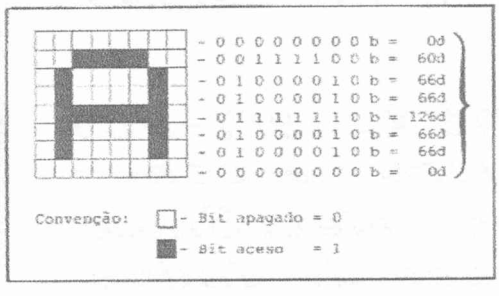
Redefinidor de caracteres

Márcio Henrique Alexandre Costa

O *Redefinidor de caracteres* é um simples programa que lhe permitirá redefinir, de forma fácil e eficiente, até cinco bancos de caracteres para o seu TK90X. Desta forma, seus programas poderão apresentar um melhor visual quanto à parte de texto, oferecendo maior sofisticação em sua elaboração.

O SISTEMA

A ROM do ZX Spectrum possui uma área onde estão armazenadas as matrizes de pontos, que representam a forma gráfica de todos os caracteres padrão ASCII. Essas matrizes podem ser visualizadas, como o exemplo abaixo, acionando-se a função UDG 2. Repare que a letra **A** está contida em um quadrado composto de 8 x 8 pontos, onde cada ponto, aceso ou apagado (bit – representação binária), determinará o desenho do caractere; enquanto que uma linha horizontal, formada por oito destes pontos, representa um dos 8 bytes do conjunto:



Conjunto de oito bytes em representação binária, na forma do caractere.

Existem 128 caracteres, incluindo os 32 primeiros que são de controle, na configuração básica do padrão ASCII. Portanto, a área da ROM reservada para as matrizes ocupa exatamente 1 Kb, ou seja, 1024 Kb: 128 caracteres x 8 bytes = 1024 Kb. Esta área está situada a partir do endereço 15360d (4C00h).

Para controle do microcomputador, existe uma variável de sistema chamada **PTBL CHR**, contida nos endereços 23606d e 23607d (5C36h e 5C37h), cuja função é apontar para o endereço base do conjunto de caracteres. Sempre que o sistema é inicializado ou o comando **NEW** é executado, esta variável é atualizada para apontar o endereço 15360d, com a seguinte fórmula:

```
LET H=INT (15360/256)
LET L=15360-256*H
POKE 23606,L: POKE 23607,H
```

Onde:

H – parte alta de 15360d, em hexadecimal;
L – parte baixa de 15360d, em hexadecimal;
23606 – deve conter a parte baixa do endereço base; e
23607 – deve conter a parte alta do endereço base.

Portanto, supondo-se que exista um novo banco de caracteres a partir do endereço 50000d, por exemplo, para acessá-lo basta executar a pequena rotina:

```
LET H=INT (50000/256):LET L=50000-256*H
POKE 23606,L: POKE 23607,H
```

e pronto!

- **Observação 1**: caso você seja uma pessoa prevenida e não esteja disposta a perder a área reservada para o novo banco, execute um **CLEAR RAMTOP**. Daí então o pgm BASIC não poderá modificá-lo. Como exemplo, **CLEAR 49999** fará com que, a partir do endereço seguinte (50000), esta área não seja violada.

O PROGRAMA

Quando executado, na forma **RUN**, o microcomputador reservará os endereços a partir de 60230d (áreas para trabalho e bancos) e ativará a sub-rotina 9000 **Inicialização**, a qual deverá ser executada antes de qualquer outra operação, pois ela encarrega-se de transferir para a RAM, em cinco cópias, o banco de caracteres *standard*, para posterior modificação.

- **Observação 2**: esta sub-rotina cancela qualquer outro banco de caracteres existente em memória. Caso queira reiniciar o programa, sem afetar aquela área, digite **RUN 20**. Será apresentado, então, um menu de operação:

[1] – Redefinição de caracteres: esta opção permite redefinir um novo banco, cancelando qualquer outro em memória;
[2] – Alteração: possibilita alterar qualquer banco, sem can-

Redefinidor de caracteres

```
    1 REM ************************
    2 REM *  REDEFINIDOR  II  *
    3 REM *  MARCIO COSTA 1987  *
    4 REM ************************
   10 BORDER 0: PAPER 0: INK 7: C
LEAR 60000229: GOSUB 9000
   20 REM ** MENU DE OPERACAO **
   30 CLS : PRINT AT 5,5;"[ Menu
de Operacao ]"
   40 PRINT AT 10,0;"[1] - Redefi
nicao de Caracteres"
   50 PRINT "[2] - Alteracao"
   60 PRINT "[3] - Gravacao de Ba
nco"
   70 PRINT "[4] - Leitura de Ban
co"
   80 PRINT "[5] - Visualizacao d
os Bancos"
   90 PRINT "[6] - Fim"
  100 INPUT "Escolha a opcao ";
LINE N$
  110 IF N$<"1" OR N$>"6" THEN GO
TO 100
  120 IF N$="6" THEN CLS : STOP
  130 GOSUB 1000+VAL N$: GOTO 30
 1000 REM ** REDEFINICAO CHR$ **
 1010 PRINT AT 19,0; INVERSE 1;"R
edefinicao"
 1020 GOSUB 9000: GOSUB 5000: IF
 N$="0" THEN RETURN
 1030 GOSUB 6500: RETURN
 2000 REM ** ALTERACAO **
 2010 PRINT AT 19,0; INVERSE 1;"A
lteracao"
 2020 GOSUB 6000: IF N$="0" THEN
RETURN
 2030 GOSUB 6500: RETURN
 3000 REM ** GRAVACAO **
 3010 PRINT AT 19,0; INVERSE 1;"G
ravacao"
 3020 GOSUB 6000: IF N$="0" THEN
RETURN
 3030 INPUT "Nome do Bloco ?";LI
NE N$: IF N$="" OR LEN N$>10 THE
N GOTO 3030
 3040 PRINT "Banco #";BANCO;"
-";N$: SAVE N$CODE ADD,1024
 3050 CLS : PRINT "OK - Volte a
 fita para verifica-cao, Em caso
 de ERRO digite      GOTO 3000 pa
ra regravar ou      GOTO 3060 pa
ra nova tentativa."
 3060 VERIFY N$CODE ADD
 3070 PRINT "Arquivo OK": PAUSE
 300: RETURN
 4000 REM ** LEITURA **
 4010 PRINT AT 19,0; INVERSE 1;"L
eitura"
 4020 GOSUB 6000: IF N$="0" THEN
RETURN
 4030 INPUT "Carregar Bloco :"; L
INE N$
 4040 IF LEN N$>10 THEN GOTO 4030
 4050 CLS : PRINT "Em caso de ER
RO digite GOTO 4060"
 4060 LOAD N$CODE ADD: RETURN
 5000 REM ** VISUALIZACAO **
 5010 LET BANCO=1
 5020 FOR I=60245 TO 64341 STEP 1
024: CLS : PRINT AT 2,9; INVERSE
 1;" Visualizacao "
 5030 LET H=INT (I/256): LET L=I-
256*H: POKE 23606,L: POKE 23607,
H: PRINT AT 5,0;"Banco #";BANCO
 5040 PRINT AT 9,0;: FOR J=32 TO
 127: PRINT CHR$ J;" ";: NEXT J
 5050 PRINT AT 20,0;"[ENTER] Con
tinua" AND BANCO<6;"[SPACE] Ret
orna ao Menu"
 5060 LET I$=INKEY$: IF I$="" THE
N GOTO 5060
 5062 IF I$=CHR$ 32 THEN GOTO 508
0
 5064 IF I$=CHR$ 13 AND BANCO<6 T
HEN GOTO 5070
 5066 GOTO 5060
 5070 LET BANCO=BANCO+1: NEXT I
 5080 POKE 23606,0: POKE 23607,60
: RETURN
 6000 REM ** SELECAO DE BANCO **
 6010 INPUT "Selecione Banco (1/5
)"; LINE X$: IF X$<"0" OR X$>"5"
 THEN GOTO 6010
 6020 LET BANCO=VAL X$: LET ADD=(
BANCO-1)*1024+60245
 6030 RETURN
 6500 REM ** REDEFINICAO UDG 2 **
 6520 LET P=0: FOR I=ADD+256 TO A
DD+1023 STEP 168: LET P=P+1
 6530 LET H=INT (I/256): LET L=I-
256*H
 6540 POKE 23675,L: POKE 23676,H
 6550 CLS : PRINT TAB 20;"Banco
 #";BANCO'TAB 20;"Pagina #";P
 6560 UDG 2
 6570 PRINT AT 19,0;"[1] - Proxi
ma Pagina" AND P<85;"[2] - Reto
rcede Pagina" AND P>1;"[3] - Me
nu"
 6580 IF INKEY$<>"" THEN GOTO 658
0
 6590 LET Z$=INKEY$: IF Z$="" THE
N GOTO 6590
 6600 IF Z$="2" AND P>1 THEN LET
 P=P-1: LET I=I-(168 AND I+ADD>25
6): GOTO 6530
 6610 IF Z$="3" THEN RETURN
 6620 IF Z$="1" AND P<85 THEN GOTO
 6540
 6630 GOTO 6550
 6640 NEXT I
 9000 REM ** INICIALIZACAO **
 9010 RESTORE : POKE 23606,0: POK
E 23607,60
 9020 FOR I=0 TO 11: READ A: POKE
 I+60230,A: NEXT I
 9025 DATA 33,0,60,17,00,00,1,0,4
,237,176,201
 9030 FOR I=60245 TO 64341 STEP 1
024
 9040 LET H=INT (I/256): LET L=I-
256*H
 9050 POKE 60234,L: POKE 60235,H
 9060 RAND USR 60230: NEXT I
 9070 RETURN
```

celar os que já foram redefinidos;

[3] – Gravação de banco: permite gravar em fita um dos cinco bancos de caracteres;

[4] – Leitura de banco: permite carregar um banco de caracteres redefinido da fita;

[5] – Visualização dos bancos: fornece uma listagem completa dos bancos de caracteres; e

[6] – Fim: finaliza a execução.

CHEGA DE SOLIDÃO !!!

Videotexto, SAMPA, Cirandão, SAMPA Sul, Aruanda, Forum # 80 e mais o mundo inteiro no teclado do seu micro.

Temos kits (Apple, MSX, IBM-PC) que habilita o seu micro a conectar qualquer correio eletrônico ou base de dados com comunicação assíncrona.

SISTEMA SAMPA®

FONE (011) 35-2750 (VOZ)
35-2729 (MODEM)

ESTAREMOS NO "INFORMÁTICA 87" NO STAND DA RHEDE TECNOLOGIA

REDEFINIDOR DE CARACTERES

```
MICRO SISTEMAS
A primeira revista brasileira
de microcomputadores.

MICRO SISTEMAS
A primeira revista brasileira
de microcomputadores.

MICRO SISTEMAS
A primeira revista brasileira
de microcomputadores.

MICRO SISTEMAS
A primeira revista brasileira
de microcomputadores.

MICRO SISTEMAS
A PRIMEIRA REVISTA BRASILEIRA
DE MICROCOMPUTADORES.
```

Figura 1 – Exemplos de bancos redefinidos.

```
[ LINHAS ]        [ FUNCAO ]

  10              Preliminares para inicializacao
  20 - 130        Menu de operacao
1000 - 1030       Redefinicao de caracteres
2000 - 2030       Alteracao
3000 - 3070       Gravacao de banco
4000 - 4060       Leitura de banco
5000 - 5090       Visualizacao dos bancos
6000 - 6030       Selecao do banco
6500 - 6640       Redefinicao em UDG 2
9000 - 9070       Inicializacao dos bancos
```

Figura 2 – Estrutura do programa BASIC.

```
[ ENDERECO ]      [ FUNCAO ]

60228             Transferencia dos caracteres Standard
60245             Banco CHR I
61269             Banco CHR II
62293             Banco CHR III
63317             Banco CHR IV
64341             Banco CHR V
```

Figura 3 – Áreas de memória utilizadas.

PARTICULARIDADES

Nas opções de [1] a [4], será questionado o número do banco a operar. A resposta zero (0) permitirá voltar ao menu; já na opção [3], o nome do bloco a gravar não poderá ter mais de dez caracteres; e na opção [4], caso não conheça o nome do bloco a carregar, pressione **ENTER** à pergunta **Carregar o Bloco:**. Será carregado então o primeiro bloco encontrado.

- **Observações gerais:** como já foi mencionado, se por algum motivo você quiser reiniciar o programa sem alterar os bancos em memória, digite **RUN 20**; e durante a redefinição dos caracteres, ao final de um bloco surgirão "caracteres estranhos" a partir da letra **M**, na página de redefinição # 5. Estes caracteres são os 32 primeiros do bloco seguinte, que representam os caracteres de controle, portanto, não se deve alterá-los, limitando-se até o caractere Δ.

Márcio Henrique A. Costa é Técnico em Administração de Empresas, Programador e Instrutor de Programação. Atualmente trabalha no CPD de MICRO SISTEMAS, onde desenvolve projetos em equipamentos da linha TRS-80 mod. III e ZX Spectrum, nas linguagens BASIC, Assembler e Pascal.

CIÊNCIA MODERNA PROGRAMAS PARA MSX
(HOT BIT E EXPERT) EM FITA CASSETE

ITEM	TÍTULO	DESCRIÇÃO	PREÇO
217	River Raid	Sua missão é destruir os inimigos no Rio Raid. Igual ao do Fliperama	70,00 *
224	Galaga	Sensacional jogo onde sua perícia de piloto será testada a todo momento	70,00 *
228	Editor de Texto	Em português	170,00 *
230	Xadrez	Tradicional jogo de xadrez em 6 níveis diferentes	70,00 *
231	Road Fighter	Sensacional corrida de Fórmula 1 em 6 autódromos diferentes	70,00 *
232	Pascal	Compilador Pascal. (Manual em espanhol)	300,00
236	Controle de Estoque	Controle o seu estoque até 100 itens por arquivo	150,00 *
240	Tennis	Sensacional jogo de Tenis com o micro	70,00 *
243	Lazy Jones	Jogue Fliperama num prédio maluco cheio de monstros com diversos jogos. Exitente!	70,00 *
244	Elevator	Ajude o porquinho a escapar dos bandidos	70,00 *
246	Ping-Pong	Jogue Ping-Pong com seu amigo ou com o micro	70,00 *
247	Ultra Chess	Última versão de xadrez internacional	70,00 *
248	Beamrider	Batalha Inter-Estrelar	70,00 *
250	Buck Roger	Guerra Espacial em 3 dimensões. Muito Bom. Igual ao do Fliperama	70,00 *
251	The Goonies	Liberte os 7 Goonies presos na caverna	80,00 *
252	Volei-Ball	Sensacional jogo de Volei com o micro	80,00 *
256	Kings Valley	Pegue o tesouro do Faraó. Diversas fases	80,00 *
257	Flight Deck	Combate nas Malvinas. Decole do Porta-Aviões, fotografe a ilha, bombardeia-a e trave combate aéreo com os Caças inimigos. O maior jogo já feito para o sistema MSX	120,00 *
258	F-16	Combate aéreo entre o F-16 e os Migs-25 soviéticos. Com tiros e etc... Bom!	80,00 *
262	Football	Jogue futebol com o micro ou com um amigo	80,00 *
263	Hyper Rally	Sensacional corrida de rally de dia de noite e na neve	90,00 *
266	Super Bilhar	Jogue sinuca. Dizem até que o Rui Chapeu tem este programa para treinar	80,00 *
267	Knightmare	Ajude o Viking a destruir a Medusa do Mal (programa sensação da Feira de Informática 1986)	90,00 *
269	Roller Ball	Sensacional jogo de Flipper com quatro tabuleiros	80,00 *
271	Box	Luta de Boxe contra o micro ou com um amigo	80,00 *
274	Rambo	Baseado no famoso filme "Rambo" com Silvester Stalone	90,00 *
276	Jet Bomber	Tipo Zaxxon com muito mais emoção	90,00 *
		Helicópteros iguais ao seu	90,00 *
282	Exerion II	Continuação do Exerion I. Muito bom	90,00 *
283	Twinbee	Defenda-se dos invasores. Tipo Knightmare	90,00 *
287	Rockmiker	Entre na mina. Cace o tesouro evitando ser pego pelos estranhos habitantes da mina	90,00 *
289	Mopiranger	Salve os prisioneiros da prisão no labirinto. Muito Bom!	90,00 *
290	Boulder Dash	Recupere os diamentes da mina. Inúmeras dificuldades. Muito interessante	90,00 *
291	Speed King	Participe de um Rally/corrida de moto em diversos autódromos	90,00 *
293	Bosconian	Sensacional jogo tipo Star-War. Sensacional	90,00 *
294	Front Line	Você é um valente soldado que busca destruir a base dos tanques inimigos. Excelente	90,00 *
296	The Way of The Tiger	Jogo de luta de karatê em 3 níveis (um programa como se fosse 3 em 1) o melhor jogo de luta já feito para o MSX	150,00 *
298	Spitifire 40	Caça avião na Segunda Guerra Mundial	120,00 *
299	Bosconian	Uma nave que dá tiro em 4 direções	90,00 *
300	Senjyo	Tiro ao alvo	90,00 *
301	Dam Busters	Destrua a represa inimiga pilotando o seu Caça	120,00 *
302	Monipirangar	Salve os uruinhos do labirinto	90,00 *
303	Time Curb	Pilote um avião Caça e destrua os inimigos	120,00 *
304	Samantha Fox	Sensacional jogo de Strip Poker	100,00 *
306	Batman	Transforme-se no Batman e salve o Robin do bandido	
		Pingolim	100,00 *
307	Avenger	Luta de karatê continuação do The Way of The Tiger	100,00 *
310	Guardians	Você terá que sair de um labirinto espacial para isso terá que usar toda a sua habilidade de piloto. Sensacional!	120,00 *
311	Fine Applins	Você será o maior caçador de abacaxis do mundo, porém terá que escapar de diversas dificuldades	120,00 *
312	Sorcery	Você está em um calabouço de um castelo encantado e sua missão é descobrir o mago que o ajudará	120,00 *
313	Luta Livre	Participe de um dos mais famosos torneios de Luta Livre	120,00 *
314	Internacional Karatê	Você é um grande mestre de Karatê participe de um concurso internacional com os melores mestres	120,00
		Linguagem Forth	320,00
315	Kuma Forth		
316	Camelot Warriors	Desvende o mistério do castelo de Camelot. Programa sensação de 1987. (Qualidade turbo soft)	150,00
317	Star Soldier	Defenda a terra pilotando a mais fantástica nave existente até então. Sensacional. (Qualidade turbo soft)	150,00
318	Desoletor	Defenda o planeta Zhor pilotando uma nave de guerra super equipada com armas poderosas. Ótimo! (Qualidade turbo soft)	150,00
319	Green Beret	Sinta-se o verdadeiro Rambo participando de um regate num território inimigo. Excelente! (Qualidade turbo soft)	150,00
320	Gyrodine	Pilote o helicóptero mais poderoso já desenvolvido no mundo. O seu objetivo é com ele resgatar os líderes governamentais, das grandes potências que estão presos num determinado país de América latina. Ótimo! (Qualidade turbo)	150,00
321	GP World	Melhor jogo já desenvolvido em termos de corrida de Fórmula 1. Participe de um treino passe marchas, trabalhe seu próprio circuito. Simplesmente o melhor. (Qualidade turbo soft)	150,00
322	Trailblazer	Controle em três dimensões a bola mais maluca que você já viu. Jogo que requer coordenação motora e inteligência ao mesmo tempo. Muito bom! (Qualidade turbo soft)	150,00
323	Future Knight	Você é um bravo guerreiro e terá a missão de salvar uma linda princesa das garras de um feiticeiro hostil Excelente!	120,00
324	Super Packman	Igual ao tradicional Packman, com som	100,00 *
325	Conflito Mágico	Sensacional duelo entre os dois maiores feiticeiros da terra (turbo soft)	150,00
326	Lutador Nippon	Ajude o poderoso Ninja a recuperar a pedra milagrosa, enfrentando os melhores lutadores do oriente, (turbo soft)	150,00
327	Baskett	Sensacional jogo de Baskett com os maiores craques do mundo. Muito bom	100,00 *
328	Ski-Command	Batalha de Ski na neve entre russos e americanos	100,00 *
329	Kaleidoscope Special	Destrua as bases inimigas com a sua poderosa nave de guerra. Muito bom	100,00 *
330	Cybenus	Participe desta sensacional batalha espacial em busca do poderoso cristal de Cybe afim de tornar-se o mais poderoso ser do universo (turbo-soft)	150,00
331	Zenec	Estamos no século XXV seja o comandante de uma poderosa nave e combata os rebeldes da lua Lo (Qualidade turbo-soft)	150,00
332	Thexder	Ajude ao robor mutante (tipo transformer) salvar o planeta Mechworld dos robos invasores (Qualidade turbo-soft)	150,00
333	Star Force	Pilote a melhor nave do universo e destrua os piratas rebeldes que ameaçam a segurança da terra (qualidade turbo-soft)	150,00
334	Choplifter	Salve os sobreviventes de uma guerra no deserto com o seu helicóptero (qualidade turbo soft)	150,00
335	Alpha Blaster	Tipo invader, só que 300 vezes melhor com any tipos de fases. Muito bom	100,00 *

DBS: * JOGOS DISPONÍVEIS EM DISKETTE:
1 JOGO NUM DISCO – 200,00
2 JOGOS NUM DISCO – 320,00
SOLICITE GRATUITAMENTE UMA LISTA COMPLETA DE ESTOQUE E CONHEÇA TAMBÉM O MAIOR ACERVO BIBLIOGRÁFICO DO PAÍS

Desejo receber os programas abaixo relacionados pelo(s) qual(is) estou remetendo anexo um cheque nominal à Ciência Moderna Computação Ltda., Av. Rio Branco, 156 - Loja 127 - Centro - RJ - CEP 20043 no valor de Cz$

PROGRAMAS Nº _____

NOME: _____

END: _____

CIDADE _____ UF _____ CEP _____

ZX SPECTRUM

Supermonitor UDG 2

Francisco P. Nestor de Souza

Nós, usuários da linha ZX Spectrum, somos beneficiados com o monitor UDG 2, o que dá água na boca em muitos programadores de outros micros. Mas, muitas vezes deparamo-nos com problemas na definição de caracteres especiais. Apesar dos compatíveis nacionais serem dotados deste monitor, ele só permite a definição de um caractere numa matriz de 8x8 pixels.

Como todos sabem, porém, nem sempre este tamanho é suficiente para satisfazer a nossa imaginação. Temos que recorrer, então, a uma matriz maior; e a mais comum e satisfatória é a de 16x16 pontos. Aí, vem à cabeça do usuário três alternativas:

Na primeira, ele teria que utilizar uma folha de papel quadriculado, marcando-a com o tamanho da matriz desejada. Neste método, a parte mais chata é a de ter que marcar a folha do tamanho desejado. Outra coisa que dá nos nervos é ter que pintar cada quadradinho daqueles com um lápis. No entanto, muito cuidado, porque se você borrar, para apagar é uma tarefa que exige muita habilidade (a não ser que você vá apagar todo o desenho).

A segunda opção consiste em você ter que dividir o seu gráfico em quatro partes – duas para cima e duas para baixo, e formá-las uma-a-uma no monitor residente no equipamento (UDG 2). A deficiência desta opção está no fato de que você não poderá ver o seu desenho completo, mas só duas partes unidas de cada vez (cima/cima; baixo/baixo), a não ser que o programador a coloque em seu software.

Já a terceira opção nasceu da minha cabeça, pois se o monitor que está na ROM do microcomputador deixa a desejar, por que não inventar um *Supermonitor UDG 2?*

Assim sendo, esse programa permite que você crie uma figura numa matriz de 16x16 pontos, direto na tela do seu micro, com muitas facilidades; entre elas, a possibilidade de escolher a cor da tinta, a cor do fundo (monitor), limpeza de todos os pontos, uso do joystick, transformação dos gráficos em valores decimais para posterior uso em seus programas), visualização direta da figura que está sendo criada etc..

Antes de iniciar o programa, escolha

```
  10 REM ++++++++++++++++++++++++
  20 REM +      FRAPI  SOFT      +
  30 REM +  FRANCISCO  PIRES     +
  40 REM +                       +
  50 REM + SUPER MONITOR UDG2 +
  60 REM ++++++++++++++++++++++++
  70 PAPER 0: BORDER 0: INK 7: C
LS : CLS
  72 DIM S$(32,8): POKE 23658,8
  75 PRINT AT 0,6; INK 6; BRIGHT
 1;"SUPER MONITOR UDG2"
  77 PRINT AT 3,0;"TECLAS : "
  79 PRINT AT 5,2; INVERSE 1;"1"
; INVERSE 0;" - LIMPA O MONITOR"
  80 PRINT AT 7,2; INVERSE 1;"2"
; INVERSE 0;" - CODIGOS DECIMAIS
"
  82 PRINT AT 9,2; INVERSE 1;"3"
; INVERSE 0;" - FAZ OS 4 GRAFICO
S NO TAMANHO ORIGINAL"
  85 PRINT AT 17,0; INK 7; BRIGH
T 1;"Selecione a cor da tinta
"
  90 LET G=3: FOR N=1 TO 7: PRIN
T AT 19,G; INK N;"■";AT 20,G+1
; BRIGHT 1;N: LET G=G+3: NEXT N
 100 IF INKEY$="" THEN GOTO 100
 102 LET A$=INKEY$
 105 IF A$<"1" OR A$>"7" THEN GO
TO 100
 106 SOUND 0,01,35: SOUND 0,01,3
9
 108 LET COR=VAL A$: INK 7
 109 GOSUB 2500
 110 CLS : OVER 0: PRINT AT 0,1;
 INK 6; BRIGHT 1;"SUPER MONITOR
 UDG2"
 115 INK MON: PLOT 15,31: DRAW 1
28,0: DRAW 0,128: DRAW -128,0: D
RAW 0,-128
 120 SOUND 0,1,10
 130 FOR F=39 TO 159 STEP 8: PLO
T 15,F: DRAW 127,0: NEXT F
 135 SOUND 0,1,10
 140 FOR F=23 TO 136 STEP 8: PLO
T F,32: DRAW 0,127: NEXT F
 145 SOUND 0,1,10
 170 LET A$="/ FRAPI SOFT"
 180 FOR F=1 TO LEN A$: PRINT AT
 0,F+19; INK 7; BRIGHT 1;A$(F)
NEXT F
 185 PRINT AT 20,0; PAPER 6; INK
 8;"1-LIMPA / 2-CODIGOS / 3-REDE
FINE"
 190 RETURN
 210 LET G=0: LET P=180: LET A=1
6: LET B=155: LET T=150
 220 PLOT A,B: PLOT A+1,B: PLOT
A,B+1: PLOT A+1,B+1
 225 LET P=POINT (A,B+2)
 240 IF P=1 THEN OVER 1: INK COR
 245 LET S=IN 61438-40
 250 IF S>12 THEN GOSUB 1000: LE
T A=A-(8 AND A>20): LET B=B-(1 A
ND T>135)
 260 IF S=4 THEN GOSUB 1000: LET
 A=A-(8 AND A>20): LET B=B-(1 AN
D B>180)
 270 IF S=16 THEN GOSUB 1000: LE
T B=B-(8 AND B<38): LET T=T-(1 A
ND T>135)
 280 IF S=8 THEN GOSUB 1000: LE
T B=B+(8 AND B<150): LET T=T+(1
AND T<150)
 290 IF S=11 THEN GOSUB 1000: GO
SUB 1500: LET A=A+(8 AND A<135):
 GOSUB 2000: LET R=R+(1 AND R<19
5)
 300 IF S=3 THEN GOSUB 1000: GOS
UB 1500: LET A=A-(8 AND A>20): G
OSUB 2000: LET R=R-(1 AND R>150)
 310 IF S=17 THEN GOSUB 1000: GO
SUB 1500: LET B=B+(8 AND B<150):
 GOSUB 2000: LET T=T+(1 AND T<15
0)
 320 IF S=15 THEN GOSUB 1000: GO
SUB 1500: LET B=B-(8 AND B>38):
GOSUB 2000: LET T=T-(1 AND T>135
)
 322 IF S=19 THEN GOSUB 1000: GO
SUB 1500: GOSUB 2000
 325 LET L$=INKEY$
 330 IF L$="1" THEN SOUND 0.1,20
: GOSUB 110
 340 IF L$="2" THEN PRINT OVER 0
;AT 20,10; INK 2; PAPER 6; FLASH
 1;"2": SOUND 0,1,20: GOSUB 3000
 350 IF L$="3" THEN PRINT OVER 0
;AT 20,22; INK 2; PAPER 6; FLASH
 1;"3": SOUND 0,1,20: GOSUB 3000
 500 GOTO 220
1000 REM
1010 OVER 1: PLOT A,B: PLOT A+1,
B: PLOT A,B+1: PLOT A+1,B+1
1020 OVER 0
1030 IF P=1 THEN INK COR: PLOT A
,B: PLOT A+1,B: PLOT A,B+1: PLOT
 A+1,B+1
1040 INK MON
1100 RETURN
1500 REM
1510 LET C=(170-B)/8: LET D=(A-5
)/8
1520 IF POINT (A,T)=1 THEN LET O
=1
1550 RETURN
2000 OVER 1
2001 PLOT INK 7,A,T: OVER 0
2010 IF P=1 THEN GOTO 2040
2020 PRINT AT C,D; INK COR; PAPE
R 0; BRIGHT 1; OVER 1;"■"
2030 RETURN
2040 PRINT AT C,D; INK MON; BRIG
HT 0; OVER 1;"■"
2050 RETURN
2500 PRINT AT 19,3;"

2505 PRINT AT 17,0; BRIGHT 1;"Se
lecione a cor do monitor : "
2510 LET G=3: FOR N=1 TO 7: PRIN
T AT 19,G; INK N;"■": LET G=G+
3: NEXT N
2520 IF INKEY$="" THEN GOTO 2520
2530 LET A$=INKEY$
2532 IF A$<"1" OR A$>"7" THEN GO
TO 2520
2535 SOUND 0.01,35: SOUND 0.01,3
9
2540 LET MON=VAL A$: INK 7: CLS
: GOSUB 110: GOSUB 210
3000 REM [illegible]
3002 OVER 0: DIM Z(16,16)
3005 PRINT AT 10,22; INK 7; BRIG
HT 1;"AGUARDE"
3010 FOR L=2 TO 17: FOR M=2 TO 1
7
3030 IF ATTR (L,M)=COR+64 THEN L
ET Z(L-1,M-1)=1
3040 NEXT M: NEXT L
3050 INK 7
3090 DIM S$(32,8): DIM E(32)
3100 LET CONT=1: LET O=1: LET P=
1
3105 FOR M=1 TO 2
3110 FOR L=1 TO 8
3120 FOR K=1 TO 2
3130 FOR F=CONT TO CONT+7
3200 LET S$(F)=STR$ Z(O,P)+STR$
Z(O,P+1)+STR$ Z(O,P+2)+STR$ Z(O,
P+3)+STR$ Z(O,P+4)+STR$ Z(O,P+5)
+STR$ Z(O,P+6)+STR$ Z(O,P+7)
3250 LET O=O+1: NEXT F: LET O=1:
 LET P=9: LET CONT=9: NEXT M: LE
T O=9: LET P=1: LET CONT=17: NEX
T L: LET O=9: LET P=9: LET CONT=
25: NEXT K
3300 FOR F=1 TO 32: LET BE=0: LE
T NU=0: FOR J=8 TO 1 STEP -1: LE
T E(F)=(2tNU)+VAL S$(F,J): LET E
(F)=BE+E(F): LET BE=E(F): LET NU
=NU+1: NEXT J: NEXT F
3310 IF L$="3" THEN GOTO 4000
3350 SOUND ,,6: CLS
3400 PRINT AT 1,1; INK 6;"CODIGO
S DECIMAIS";AT 2,1;"PARA FORMACA
O DE UDG 2"
3410 PRINT AT 5,1; INK 5;"1ª CARA
CTER ";: FOR F=1 TO 8: PRINT I
NK 7;E(F);",";" AND F<8);: NEXT F
3420 PRINT AT 8,1; INK 5;"2ª CARA
CTER ";: FOR F=9 TO 16: PRINT
INK 7;E(F);",";" AND F<16);: NEXT
 F
3430 PRINT AT 11,1; INK 5;"3ª CAR
ACTER ";: FOR F=17 TO 24: PRIN
T INK 7;E(F);",";" AND F<24);: NE
XT F
3440 PRINT AT 14,1; INK 5;"4ª CAR
ACTER ";: FOR F=25 TO 32: PRIN
T INK 7;E(F);",";" AND F<32);: NE
XT F
3500 PRINT AT 18,0;"SELECIONE:";
AT 20,0;"1-VOLTA AO DESENHO ANTE
RIOR";AT 21,0;"2-NOVO DESENHO"
3510 IF INKEY$="" THEN GOTO 3510
3520 IF INKEY$="1" OR INKEY$="2"
 THEN GOTO 3500
3530 IF INKEY$="1" THEN GOTO 500
0
3540 IF INKEY$="2" THEN RUN
4000 REM [illegible]
4005 OVER 0: PRINT AT 20,22; INK
 8; PAPER 6;"3"
4007 SOUND 0,1,12
4010 PRINT AT 8,22; INK 7;"INICI
AR";AT 9,20;"REDEFINICAO"
4030 PRINT INK 7;AT 11,21;"CARAC
TER: ";AT 12,22;"(A - R)"
4040 LET C$=INKEY$
4050 IF INKEY$="" THEN GOTO 4050
4060 IF C$<"A" OR INKEY$>"S" THE
N GOTO 4040
4070 SOUND 0,1,15
4080 FOR F=0 TO 31: POKE USR C$+
F,E(F+1): NEXT F
4090 FOR F=8 TO 12: PRINT AT F,2
0;"                ": NEXT F
4100 PRINT AT 8,23; INK 7;"GRAVA
";AT 9,20;"CARACTERES?";AT 10,23
;"(S/N)"
4110 IF INKEY$="" THEN GOTO 4110
4120 IF INKEY$="N" OR INKEY$="n"
 THEN FOR F=8 TO 10: PRINT AT F,
20;"                ": NEXT F: INK M
ON: GOTO 240
4130 INPUT "NOME DA PARTE: "; LI
NE N$: IF LEN N$<1 OR LEN N$>10
THEN GOTO 4130
4140 SAVE N$CODE USR "A",168
4150 PRINT AT 8,23;"     ";AT 9,
20; INK 7;" VERIFICA? ";AT 10,23
;"(S/N)"
4160 IF INKEY$="" THEN GOTO 4160
4170 IF INKEY$="N" OR INKEY$="n"
 THEN PRINT AT 9,20;"
";AT 10,23;"   ": INK MON: GOT
O 240
4180 PRINT AT 9,21; INK 7;"REBOB
INE";AT 10,22;"A FITA"
4185 INK 0: PRINT AT 20,0
4190 VERIFY N$CODE USR "A",168
4195 PRINT AT 12,22; INK 7; INVE
RSE 1;" O.K. ": FOR F=0 TO 100: N
EXT F: PAUSE 200: FOR F=9 TO 12:
 PRINT AT F,21;"          ": NEX
T F
4200 INK MON: GOTO 240
5000 REM [illegible]
5010 GOSUB 110
5012 PRINT AT 10,20; INK 7; BRIG
HT 1;"AGUARDE"
5015 LET R=180: LET T=150: INK 7
5020 FOR F=1 TO 8: FOR G=1 TO 8:
 IF S$(F,G)="1" THEN PRINT AT P+
1,G+1; BRIGHT 1; INK COR; OVER 1
;"■": PLOT R,T
5025 LET R=R+1
5030 NEXT G: LET T=T-1: LET R=R
-8: NEXT F: LET R=R+8: LET T=T+8
5040 FOR F=9 TO 16: FOR G=1 TO 8
: IF S$(F,G)="1" THEN PRINT AT F
-7,G+9; BRIGHT 1; INK COR; OVER
1;"■": PLOT R,T
5045 LET R=R+1
5050 NEXT G: LET T=T-1: LET R=R-
8: NEXT F: LET R=R-8
5060 FOR F=17 TO 24: FOR G=1 TO
8: IF S$(F,G)="1" THEN PRINT AT
F-15,G+1; BRIGHT 1; INK COR; OVE
R 1;"■": PLOT R,T
5065 LET R=R+1
5070 NEXT G: LET T=T-1: LET R=R-
8: NEXT F: LET R=R+8: LET T=T+8
5080 FOR F=25 TO 32: FOR G=1 TO
8: IF S$(F,G)="1" THEN PRINT AT
F-15,G+9; BRIGHT 1; INK COR; OVE
R 1;"■": PLOT R,T
5085 LET R=R+1
5090 NEXT G: LET T=T-1: LET R=R-
8: NEXT F
5095 PRINT AT 10,20;"
5100 SOUND 0,1,10: GOTO 210
```

Supermonitor UDG 2

a cor da tinta e a cor do fundo do monitor. Em seguida, você verá a matriz se formando com a cor que foi escolhida. Observe um pequeno ponto no canto superior esquerdo do monitor, pois é o cursor com o qual você irá desenhar e apagar o seu desenho.

MOVENDO O CURSOR

Utilize um joystick ou as teclas correspondentes a este, que são as seguintes: 6 – Esquerda; 7 – Direita; 8 – Desce; 9 – Sobe; 0 – Apaga/Pinta.

- **Pintando** – pressione o botão de disparo do joystick (tecla 0), juntamente com a direção desejada, assim será pintado o local onde estava o cursor;
- **Apagando** – posicione o cursor sobre o local que você deseja apagar. Pressione o botão do joystick (tecla 0), e o local estará novamente limpo;
- **Figura no tamanho original** – observe no canto superior direito do vídeo uma figura que vai se formando conforme você monta o seu desenho no monitor. Esta figura tem o tamanho original dos caracteres gráficos que irão se formar na UDG 2. Inicialmente, a cor desta figura é branca, porém, se você quiser que ela fique da cor da tinta, altere o **INK** 7 da linha 2001 para **INK COR**;
- **Função 1** (limpa monitor) – se você quiser recomeçar o seu desenho, pressione a tecla 1 e então o monitor será limpo;
- **Função 2** (códigos UDG 2) – uma das mais atraentes funções do programa, que permite ver o seu desenho sob a forma de códigos decimais. Para usar o seu desenho no programa, você deverá transformá-lo em códigos, pois só assim ele poderá ser utilizado com a forma de caracteres especiais (UDG 2). Como estamos utilizando um monitor de 16x16 pontos, teremos que transformá-lo em quatro de 8x8, o que, logicamente, consistiria em formar quatro figuras que deverão ser agrupadas da seguinte forma:

1º caractere	2º caractere
3º caractere	4º caractere

Depois de pressionar a tecla 2, você terá que aguardar durante cerca de um minuto. Esta demora deve-se ao fato de que o programa transforma cada linha de oito colunas em números binários, para então convertê-lo à base decimal. Logo após, você poderá voltar ao desenho anterior ou começar um novo; e

- **Função 3** (redefinição) – esta função faz a redefinição dos quatro caracteres no monitor do micro, utilizando quatro caracteres no formato acima. Esta opção inclui ainda a opção de gravá-los e também verificar a gravação.

Francisco Pires Nestor de Souza estuda na Escola Técnica Barão de Mauá-RJ. Autodidata em BASIC, utiliza o seu TK90X para desenvolver programas educacionais relacionados a eletrônica, e em jogos de inteligência e ação.

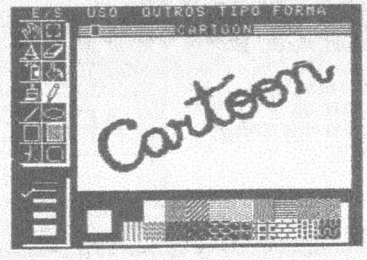
MICROMAQ — MSX

dBASE II PLUS MSX

Produzido por: DATALOGICA

Distribuído por: PRINCESSWARE

dBASE II é Marca Registrada ASHTON-TATE



EDITOR GRÁFICO

O melhor e de mais fácil utilização para a linha MSX. Possibilita a criação de desenhos de alta resolução e qualidade.

Aplicações no campo do ensino, desenho profissional, programação visual, terapia ocupacional, suporte no desenvolvimento mental da criança além de 1001 outras aplicações ilimitadas como sua imaginação.

Versões Fita ou Disco ..... Consulte-nos

Um best-seller mundial

Com seus poderosos recursos o dBASE II tornou-se o software para gerenciamento de dados mais difundido no mundo.
É indicado para o desenvolvimento rápido e eficiente de programas, bem como para consulta sem necessidade de uma pré-programação. Por exemplo: contabilidade, custo de serviços, gerenciamento de mala direta, controle de estoque, etc.

Somente em Disco..... 11 OTN

CONTROLE DE ESTOQUE E PROJEÇÃO DE CUSTOS

Potente gerenciador de estoques com capacidade para até 1800 artigos por disco simples.
Controla estoques e projeta custos industriais, permitindo a completa manipulação de produtos acabados e/ou matérias-primas. Gera 10 relatórios diferentes, entre eles: tabela de preços, lista de pedidos, custo de produtos acabados, etc.

Somente em Disco ..... Consulte-nos

e ainda: • DIETAS • COPYMAQ • SIGA • CONTROLE DE AÇÕES • ZAPPER • CONTROLE BANCÁRIO • FLUXO DE CAIXA • CADASTRO DE CLIENTES • TEORIA DOS CONJUNTOS, ETC.
Próximos lançamentos: 60 MINUTOS (adventure) e MATEMÁTICA AVANÇADA. **E MUITO, MUITO MAIS!**

Escreva-nos solicitando catálogo completo, inteiramente Grátis, para as linhas MSX, Color e TK-90. Não esqueça de indicar o equipamento.

CONDIÇÕES ESPECIAIS PARA REVENDA, SOLICITE INFORMAÇÕES.

Dispomos de Equipe Especializada em MSX e Color apta a implantar Sistemas que possam resolver o seu problema específico. Entre em contato conosco.

Comércio de Aparelhos Eletrônicos MICROMAQ Ltda.
Rua Sete de Setembro, 92 Loja 106 - Centro
RIO-RJ CEP: 20050 Tel.: (021) 222-6088

Mizael

ZX SPECTRUM

Abrindo janelas

João Carlos Mendes Luis

O objetivo deste aplicativo, desenvolvido para os micros compatíveis com a linha ZX Spectrum, é inovar o sistema de leitura de cassete. Com este programa, em linguagem de máquina, você poderá abrir janelas no vídeo de seu micro, as quais servirão para indicar o carregamento do programa. Esta mesma rotina também lhe permitirá modificar as cores das faixas de sincronismo, ou simplesmente retirá-las. As janelas inseridas no vídeo são de dimensão e posicionamento definíveis, assim como as cores das faixas.

Não pretendemos aqui ensinar Assembler a ninguém, por isso não nos prenderemos a detalhes de programação. Para aqueles que não possuem um assemblador, a listagem 1 exibe os códigos hexadecimais para se entrar com o *BUG90*. Para aqueles que possuem um assemblador, a listagem 2 mostra o programa fonte, ligeiramente comentado, para maior entendimento.

PROCEDIMENTO

O procedimento básico deste programa é o seguinte: 1 — esperar um sinal de *leader* da fita; 2 — esperar um pulso de sincronismo; 3 — verificar o *markbyte*; 4 — proceder à leitura dos bytes; e 5 — testar, ao final, a paridade dos bytes lidos.

Mas, como se usa este programa? Bem, vamos começar definindo as janelas.

Primeiro escolha qual o formato da janela e desenhe-a por meio de pontos (INK). Depois procure saber quais são os atributos que preenchem tal janela (no máximo 24 atributos) e anote os endereços. Caso o número de endereços não chegue a 24, substitua os que sobrarem por 00000. Agora modifique no programa fonte onde está indicado. Obs.: Para definir a cor do fundo da janela, modifique onde está indicado "máscara de atributos", definindo-a com os bits de atributos que deverão ser mantidos em 1. Em resumo, os pontos de INK serão definidos pelas cores das faixas, os pontos de paper, o BRIGHT e o FLASH pela "máscara de atributos".

```
MASCARA DE ATRIBUTOS - X X X X X X X X
                       ¦ ¦ \ ¦ / \ ¦ /
                       ¦ ¦   ¦     ¦
                       ¦ ¦   ¦     +-Cor do INK (manter em 0)
                       ¦ ¦   +-Cor do PAPER
                       ¦ +-BRIGHT
                       +-FLASH
```

Como nós dissemos anteriormente, nosso objetivo não é ensinar Assembler, por isso, para maiores detalhes sobre o programa, procure literatura especializada sobre linguagem de máquina.

Convém aqui lembrar também que a área de atributos começa em 22528 e termina em 23295. A fórmula para saber o endereço de um determinado atributo a partir das coordenadas X e Y de um **PRINT** é a seguinte: ATTR = 22528 + X * 32 + Y, onde X é a linha e Y a coluna da posição de **PRINT**.

Além das características das janelas, podemos definir também as cores das faixas. A primeira cor a definir está no endereço 65036, que é a primeira cor das faixas. A relação de alternância das faixas está definida por uma operação XOR no endereço 65180. Este valor deve ter os BITs 3 a 7 com valor 1 devido a outras funções deste mesmo BYTE. Mais do que a alternância normal das faixas, percebe-se também no **LOAD** e **SAVE** do BASIC uma troca de cores quando o sinal de sincronismo é encontrado. O BYTE que determina esta troca é o que está localizado no endereço 65091. Também devido a outras funções deste BYTE, deve-se manter os BITs 3 a 7 agora com o valor 0.

Ambos os valores de alternância são determinados por operações XOR, por isso, caso você não saiba usar esta instrução a nível de BITs, não tente modificar os valores pré-definidos.

Em alguns casos é interessante desativar o BREAK da rotina de **LOAD**. Para isto, deve-se pokear os endereços 65039 e 65172 com 0.

Finalmente, devemos ainda ajustar os registradores de endereço inicial (65002) e de comprimento (65005), ambos com o formato de 2 bytes (LSB e MSB), e o *markbyte* (65008). Caso a operação a ser executada seja um **VERIFY** da memória, deve-se pokear o endereço 65009 com o valor 0.

Depois de tudo definido, conforme a vontade do programador, pode-se gravar o programa com o comando BASIC: SAVE "LOADER" CODE 65000, 261.

Obs.: este programa acusa erro sob a forma das flag CARRY e ZERO, sendo necessário um outro programa em LM para identificá-lo (CARRY em 0 para erro, ZERO em 0 se erro foi BREAK). Caso contrário, o BASIC simplesmente deverá ignorar o erro.

Agora que você já sabe como usar este sistema de leitura, eis como gravar os programas a serem lidos.

- Opção 1: grave o programa desejado com "SAVE "n"CODE (endereço inicial), (comprimento)", sendo que você só deve ligar o gravador logo após o *header* (obs.: *markbyte* = 255).
- Opção 2: use o programa em Assembler logo abaixo:

```
LD IX, endereço inicial
LD DE, comprimento
LD A, markbyte
JP 1218
```

Para quem não tem assemblador, eis os códigos HEXA: DD 21 ee ee 11 cc cc 3E mm C3 C2 04.

```
FDE8  DD 21 00 40 11 00 1B 3E  FE70  B8 C8 15 06 C9 D2 6A FE
FDF0  FF 37 CD 00 FE 3A 48 5C  FE78  7C AD 67 7A B3 20 CA 7C
FDF8  0F 0F 0F CD 9E FE FB C9  FE80  FE 01 C9 CD B7 FE D0 3E
FE00  14 08 15 F3 3E 0F D3 FE  FE88  02 3D 20 FD A7 04 C8 3E
FE08  CD 9E FE 3E 02 4F 8F C0  FE88  02 3D 20 FD A7 04 C8 3E
FE10  CD B7 FE 30 FA 26 14 10  FE90  7F DB FE 1F D0 A9 E6 20
FE18  FE 25 20 FB CD B3 FE 30  FE98  28 F3 79 EE FF 4F E6 07
FE20  EE 06 9C CD B3 FE 30 E7  FEA0  D3 FE F6 38 32 21 5B 32
FE28  3E C6 88 30 E3 24 20 F1  FEA8  22 5B 32 23 5B 32 24 5B
FE30  FE D4 30 F4 CD B7 FE 30  FEB8  5B 32 28 5B 32 41 5B 32
FE40  CE 79 EE 03 4F 06 C9 18  FEC0  42 5B 32 43 5B 32 44 5B
FE48  1F 08 20 07 30 0F DD 75  FEC8  32 45 5B 32 46 5B 32 47
FE50  00 18 0F CB 11 AD C0 79  FED0  5B 32 48 5B 32 61 5B 32
FE58  1F 4F 13 18 07 DD 7E 00  FED8  62 5B 32 63 5B 32 64 5B
FE60  AD C0 DD 23 1B 08 06 CB  FEE0  32 65 5B 32 66 5B 32 67
FE68  2E 01 CD B3 FE D0 3E E4  FEE8  5B 32 68 5B 37 C9 00 00
```

Listagem 1

```
                ORG 65000
65000           LD IX,16384     ;Endereco inicial da tela
65004           LD DE,6912      ;Comprimento da tela
65007           LD A,255        ;Markbyte igual ao do BASIC
65009           SCF             ;Ler, nao verificar
65010           CALL LOAD
65013           LD A,(23624)    ;Define cor da borda e da(s)
65016           RRCA            ;janelas apos o load
65017           RRCA
65018           RRCA
65019           CALL OUTE
65022           EI              ;Retorna ao BASIC
65023           RET
65024 LOAD      INC D
65025           EX AF,AF'
65026           DEC D
65027           DI
65028           LD A,#0F        ;Cor da borda
65030           OUT (#FE),A
65032           CALL OUTE
65035           LD A,2          ;Primeira cor das faixas
65037           LD C,A
65038           CP A
65039 BREAK     RET NZ          ;Retorna se BREAK
65040 START     CALL EDGE1      ;Espera primeiro pulso da fita
65043           JR NC,BREAK
65045           LD H,20         ;Define uma pequena pausa
65047 WAIT      DJNZ WAIT
65049           DEC H
65050           JR NZ,WAIT
65052           CALL EDGE2
65055           JR NC,BREAK
65057 LEADER    LD B,#9C        ;Detecta um sinal de leader
65059           CALL EDGE2
65062           JR NC,BREAK
65064           LD A,#C6
65066           CP B
65067           JR NC,START
65069           INC H
65070           JR NZ,LEADER
65072 SYNC      LD B,#C9        ;Espera pulsos de sincronismo
65074           CALL EDGE1
65077           JR NC,BREAK
65079           LD A,B
65080           CP #D4
65082           JR NC,SYNC
65084           CALL EDGE1
65087           JR NC,BREAK
65089           LD A,C          ;Troca as cores das faixas
65090           XOR #03         ;Obs:nunca usar um numero
65092           LD C,A          ;maior que 7 neste comando
65093           LD B,#C9
65095           JR MARKER
65097 LOOP      EX AF,AF'       ;Depois de lido um BYTE da fita,
65098           JR NZ,FLAG      ;prossegue-se a determinacao da
65100           JR NC,VERIFY    ;funcao deste BYTE
65102           LD (IX+#00),L
65105           JR NEXT
65107 FLAG      RL C            ;Obs: funcao MARKBYTE
65109           XOR L           ;primeiro BYTE da fita
65110           RET NZ
65111           LD A,C
65112           RRA
65113           LD C,A
65114           INC DE
65115           JR DEC
65117 VERIFY    LD A,(IX+#00)   ;Obs:funcao VERIFY
65120           XOR L           ;Para quando em verificacao
65121           RET NZ          ;da memoria
65122 NEXT      INC IX          ;Incrementa endereco
65124 DEC       DEC DE          ;Decrementa contador
65125           EX AF,AF'
65126           LD B,#CB
65128 MARKER    LD L,#01
65130 BYTE      CALL EDGE2      ;Le um BYTE completo da fita
65133           RET NC
65134           LD A,#E4
65136           CP B            ;BIT 0 ou 1
65137           RL L
65139           LD B,#C9
65141           JP NC,BYTE      ;Executa ate 8 BITs
65144           LD A,H          ;Atualiza PARITY BYTE
65145           XOR L
65146           LD H,A
65147           LD A,D          ;Continua se nao chegou ao final
65148           OR E
65149           JR NZ,LOOP
65151           LD A,H          ;Testa paridade
65152           CP #01
65154           RET
65155 EDGE2     CALL EDGE1      ;Ler 2 ciclos
65158           RET NC
65159 EDGE1     LD A,2          ;Ler 1 ciclo
65161 DELAY     DEC A
65162           JR NZ,DELAY
65164           AND A
65165 SAMPLE    INC B           ;Testa porta de entrada EAR
65166           RET Z           ;e BREAK (porta #FE)
65167           LD A,#7F
65169           IN A,(#FE)
65171           RRA
65172           RET NC          ;Retorna se BREAK pressionado
65173           XOR C
65174           AND #20
65176           JR Z,SAMPLE     ;Prossegue teste de porta
65178           LD A,C          ;Alterna as cores das faixas
65179           XOR #FF         ;Obs: manter BITs 3 a 7 em 1
65181           LD C,A
65182 OUTE      AND #07
65184           OUT (#FE),A     ;Manda cor para a borda
65186           OR #38          ;Mascara de atributos da janela
65188           LD (22561),A    ;Estas instrucoes definem
65191           LD (22562),A    ;os enderecos dos atributos
65194           LD (22563),A    ;das janelas a serem usadas
65197           LD (22564),A    ;Caso hajam janelas demais,
65200           LD (22565),A    ;use o endereco 00000 para
65203           LD (22566),A    ;anular as que restarem
65206           LD (22567),A    ;Obs: nao e possivel usar
65209           LD (22568),A    ;mais atributos para se
65212           LD (22593),A    ;definir uma janela devido
65215           LD (22594),A    ;ao sincronismo de leitura
65218           LD (22595),A
65221           LD (22596),A
65224           LD (22597),A
65527           LD (22598),A
65230           LD (22599),A
65233           LD (22600),A
65236           LD (22625),A
65239           LD (22626),A
65242           LD (22627),A
65245           LD (22628),A
65248           LD (22629),A
65251           LD (22630),A
65254           LD (22631),A
65257           LD (22632),A    ;Fim da definicao de atributos
65260           SCF
65261           RET
```

Listagem 2

Para melhor entendimento destas rotinas em LM, sugiro o livro *"The Complete Spectrum ROM Disassembly"*, de Ian Logan e Frank O'Hara. Devo agradecer a este livro todo o meu conhecimento sobre a ROM desse computador maravilhoso que é o Spectrum.

Finalmente, desejo boa sorte e bons programas a todos, especialmente àqueles que forem usar este novo e criativo sistema de leitura cassete.

João Carlos Mendes Luis é estudante de eletrônica do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, no Rio de Janeiro, sendo autodidata em BASIC e Assembler.

LAZAROSOFT MSX

Os Magos do software para MSX

ALIENS, O RESGATE • **NEMESIS** • FREDDY HARDEST I e II • PEGASUS • SUPER HOCKEY • LEONIDAS • JACK THE NIPPER • INDI 500 • WONDER BOY • PANEL • BOGY 84 • STAR FEEKER (Astrológico) • FLASH GORDON • SMALL JONES • EL CID • E OUTROS...

Você avalia um serviço pela sua eficiência:

- Nossa qualidade ..... indiscutível
- Nossa experiência ..... inigualável
- Nosso prazo/entrega ..... 24 horas + correio
- Nosso acervo ..... + 2000 títulos
- Nossa documentação ..... farta
- Nosso super-catálogo ..... grátis (peça o seu!)
- Nossos lançamentos ..... semanalmente
- Nossas promoções ..... diversas

Como vê, ninguém tem mais a oferecer do que o Mago da Lazzarosoft. Escreva-nos! Você só tem a ganhar!

- Jogo (qualquer um) ..... Cz$ 90,00
- Jogo Nemesis + Disco ..... Cz$ 700,00
- Aplicativos/Utilitários ..... Cz$ 250,00
- Copiadores ..... Cz$ 400,00
- Linguagens ..... Cz$ 400,00
- Disco ou Fita de qualidade (em média comportam 12 programas) ..... Cz$ 250,00
- Custo postal (enc. registrada) ..... Cz$ 100,00

Enviar cheque nominal cruzado à **Carlos Henrique B. Magalhães**, fornecendo o máximo de informações sobre seu equipamento e telefone para um eventual contacto.

Caixa Postal 1955 — CEP 20001 — Rio de Janeiro — RJ
Tel.: (021) 248-1575

ZX SPECTRUM

Rotscreen

Carlos Eduardo C. de Matos

Os usuários do ZX Spectrum (TK90X) que gostam de elaborar jogos devem estar sentindo uma dificuldade: a falta de rotinas de efeitos de vídeo para esta linha. A partir de agora, não precisam chorar mais, pois o *Rotscreen* chegou para ajudá-los.

Assim, digite o programa da listagem 1 e grave-o com **SAVE "ROTSCREEN" LINE 10.** Entre depois com **NEW** e **CLEAR 31999.** Carregue o "Monitor Assembler no TK90X", publicado em **MS** nº 51, e digite os códigos hexa da listagem 2 a partir do endereço 32000.

Terminada a digitação, se seu micro tiver 16 Kb, grave o bloco Assembler logo depois da listagem 1 com **SAVE "RS" CODE 32000, 456.** Caso seu micro tiver 48 Kb, apague o monitor com **NEW** (o bloco Assembler não será afetado) e digite o programa da listagem 3. Ao ser rodado, ele renumerará a parte Assembler e executará um **SAVE.** Grave logo depois da primeira listagem.

LA AROSOFT MSX

HORAS

Os Magos do software para MSX

TURBO MODULA II (Ling.) • KENDO • MSX TOOLS • POLICE ACADEMY 2 • HE MAN • ULTRA MAN • MOVPAC • TEMPTATION • GLASS • TIMEBOMB • GULKAVE • INCA • BOILED • DRÁCULA • STARBYTE • KNIGHTGHOST • NEMESIS E OUTROS...

Você avalia um serviço pela sua eficiência:

• Nossa qualidade	indiscutível
• Nossa experiência	inigualável
• Nosso prazo/entrega	24 horas + correio
• Nosso acervo	+ 2000 títulos
• Nossa documentação	farta
• Nosso super-catálogo	grátis (peça o seu!)
• Nossos lançamentos	semanalmente
• Nossas promoções	diversas

Como vê, ninguém tem mais a oferecer do que o Mago da Lazzarosoft.

Escreva-nos e receba um brinde! Você só tem a ganhar!

• Jogo (qualquer um)	Cz$ 100,00
• Jogo Nemesis + Disco	Cz$ 700,00
• Aplicativos/Utilitários	Cz$ 300,00
• Copiadores	Cz$ 450,00
• Linguagens	Cz$ 450,00
• Disco ou Fita de Qualidade (em média comportam 12 programas)	Cz$ 360,00
• Custo postal (encomenda registrada)	Cz$ 150,00

Enviar cheque nominal cruzado à **Carlos Henrique B. Magalhães**, fornecendo o máximo de informações sobre seu equipamento e telefone para um eventual contacto.

Caixa Postal 1955 — CEP 20001 — Rio de Janeiro — RJ
Tel.: (021) 248-1575

USANDO O ROTSCREEN

O *Rotscreen* é um conjunto de oito rotinas independentes. Todas elas utilizam os endereços 23296-23299 para ler os seguintes dados:

23296 — coluna inicial
23297 — linha inicial
23298 — número de colunas
23299 — número de linhas

Esta é a melhor característica do programa, pois possibilita a criação de inúmeros efeitos especiais. A listagem 1 dá uma demonstração de alguns desses efeitos. Veja as rotinas de que o *Rotscreen* dispõe e seus endereços:

Scroll esquerda	32037	64805	roda a tela para a esquerda
Scroll direita	32096	64864	roda a tela p/ a direita
Scroll p/ cima	32164	64932	roda a tela p/ cima
Scroll p/ baixo	32171	64939	roda a tela p/ baixo
Block	32284	65052	desenha um bloco na cor corrente
Moldura	32317	65085	desenha uma moldura na cor corrente
Inverse	32372	65140	inverte a tela
Explosão	32412	65180	gera uma explosão. Precisa ser executada 20 vezes seguidas.
	-16K-	-48K-	

Lembre-se de definir **POKEs** nos endereços 23296-23299, senão poderá provocar um *crash*. A linha inicial não pode ser maior que 23, enquanto a coluna inicial não deve ser maior que 31. O número de linhas e de colunas não pode ser menor que um, nem exceder os limites da tela. As duas linhas inferiores podem ser usadas.

No caso do Scroll para os lados, o número de colunas não deve ser menor que dois. Já no caso do Scroll para cima ou para baixo, o número de linhas é que não pode ser menor que dois. Preste atenção nestas regras para evitar surpresas desagradáveis.

Quando quiser usar o *Rotscreen*, digite a listagem 1, que ele carregará o bloco Assembler, fará uma demonstração e se autodestruirá, porém esse bloco permanecerá na memória.

Para utilizá-lo com outro programa, proceda da seguinte forma: faça com que seu programa, quando carregado, execute um **CLEAR 31999** (ou **CLEAR 64767** para 48 Kb) e digite o

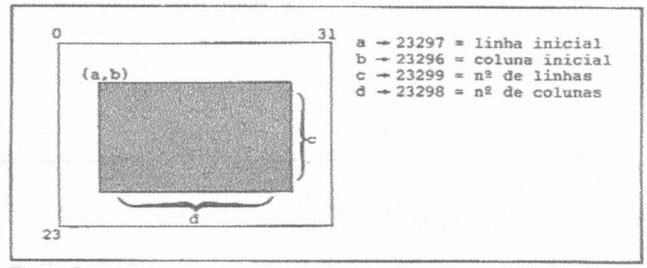


Figura 1

bloco Assembler, que deverá estar gravado depois, com **SAVE "RS" CODE 32000** (ou 64768, para 48 Kb), **456.** Sugiro ainda que analise a listagem 1 e tente modificar os efeitos, pois assim aprenderá a usar os recursos que este programa lhe oferece.

Agora, carregue o *Rotscreen*. Gostou da demonstração? Vamos conhecê-lo melhor.

Primeiro, encha a tela com caracteres quaisquer, para servir de teste. Depois, digite **POKE 23296,5:POKE 23297,5:POKE 23298,22:POKE 23299,12.** Isso quer dizer que as rotinas agirão apenas sobre o bloco de tela da posição (5,5) até a (17,27), formando um retângulo. Veja a figura 1.

Se você entrar com **RAND USR 32037** (ou 64805, para 48 Kb), verá esse bloco de tela rodar para a esquerda. Percebe agora por que esta é a melhor característica do *Rotscreen*? Basta usar as rotinas em conjunto, "pôr os miolos para funcionar" e conseguirá inventar 1E38 efeitos especiais.

O funcionamento de Block e moldura é o seguinte: ele preenche um bloco de tela com o atributo corrente, sem alterar os caracteres. Caso, seguindo o exemplo anterior, você digitasse **RAND USR 32284** (ou 65052, para 48 Kb), veria que os caracteres daquele bloco de tela iriam ficar na cor corrente. Se estes estivessem pretos sobre o fundo branco e você entrasse com **PAPER 6: FLASH 1**, passariam a estar sobre fundo amarelo e em flash, depois de executar a rotina.

Também a rotina de moldura age de modo similar, apenas que em vez de preencher todo o bloco, ocupa somente sua borda. Já a rotina de explosão desfaz os carateres do bloco aos poucos, sem alterar as cores. Para a dissolução completa, precisa ser executada 20 vezes. No caso da rotina de **INVERSE**, ela é semelhante à função **INVERSE** do micro, porém inverte o bloco diretamente e não através do **PRINT**. Agora, deixo por conta de vocês. Bom trabalho!

Carlos Eduardo Carvalho de Matos cursa atualmente o segundo grau e tem como hobby o desenvolvimento de programas para o seu TK90X.

```
  10 BORDER 0: PAPER 0: INK 0: C
LEAR 31999: LET M=0
  20 POKE 32768,M: IF NOT PEEK 3
2768 THEN CLEAR 64767: LET M=327
68
  30 LOAD ""CODE : INK 4: CLS
  40 NEXT f: STOP
  50 PLOT 70,20: DRAW 120,120,75
7*PI: PRINT AT 10,12; INVERSE 1;
"ROTSCREEN"
  60 POKE 23296,4: POKE 23297,0:
 POKE 23298,24: POKE 23299,22
  70 FOR N=.15 TO .01 STEP -.01:
 FOR F=1 TO 6
  80 INK F: RAND USR (32284+M):
SOUND N,F: NEXT F: NEXT N
  90 FOR F=0 TO 20: RAND USR (32
412+M): SOUND .02,20-F: NEXT F
 100 RAND : FOR N=0 TO 14: FOR F
=10 TO 0 STEP -1
 110 POKE 23296,F: POKE 23297,F:
 POKE 23298,32-F-F: POKE 23299,2
2-F-F
 120 PAPER RND*6+1: LET L=USR (3
2317+M): NEXT F: NEXT N
 130 PAPER 0: FOR F=0 TO 15
 140 POKE 23296,0: POKE 23297,0:
 POKE 23298,32: POKE 23299,9
 150 RAND USR (32171+M): POKE 23
297,15: RAND USR (32164+M)
 160 POKE 23297,9: POKE 23298,16
: POKE 23299,5
 170 RAND USR (32037+M): POKE 23
296,16: RAND USR (32096+M)
 180 SOUND .1,F: NEXT F: NEW
```

Listagem 1

```
32000 c5 21 00 40 11 00 58 78
32008 fe 08 38 0b d6 08 f5 7c
32016 c6 08 67 f1 14 18 f1 47
32024 79 04 05 28 04 c6 20 10
32032 fc 6f 5f c1 c9 ed 4b 00
32040 5b 3a 03 5b f5 cd 00 7d
32048 c5 d5 06 00 3e 06 f5 e5
32056 54 5d 23 3a 02 5b 4f 0d
32064 ed b0 af 12 e1 24 f1 3d
32072 20 ec d1 62 6b 23 3a 02
32080 5b 4f 0d ed b0 3a 8d 5c
32088 12 c1 04 f1 3d 20 cd c9
32096 ed 4b 00 5b 3a 03 5b f5
32104 cd 00 7d c5 3a 02 5b 3d
32112 47 7d 80 6f 5f d5 06 00
32120 3e 08 f5 e5 54 5d 2b 3a
32128 02 5b 4f 0d ed b8 af 12
32136 e1 24 f1 3d 20 ec d1 62
32144 6b 2b 3a 02 5b 4f 0d ed
32152 b8 3a 8d 5c 12 c1 04 f1
32160 3d 20 c4 c9 3e 04 32 bd
32168 fd 18 05 af 32 bd fd 3e
32176 05 32 cb fd ed 4b 00 5b
32184 3a 03 5b 3d 18 04 57 80
32192 47 7a f5 cd 00 7d dd 62
32200 dd 6b e5 04 cd 00 7d fd
32208 62 fd 6b d1 c5 06 00 3a
32216 02 5b 4f 3e 08 e5 d5 c5
32224 ed b0 c1 d1 14 e1 24 3d
32232 20 f3 fd 54 fd 5d eb dd
32240 54 dd 5d ed b0 c1 f1 3d
32248 20 c8 cd 00 7d 3e 08 f5
32256 e5 3a 02 5b 36 00 23 3d
32264 20 fa e1 24 f1 3d 20 ef
32272 3a 8d 5c ed 4b 01 5b 12
32280 13 10 fc c9 ed 4b 00 5b
32288 cd 00 7d ed 4b 02 5b 3a
32296 8d 5c 4f 62 6b 3a 02 5b
32304 71 23 3d 20 fb 21 20 00
32312 19 eb 10 ef c9 ed 4b 00
32320 5b cd 00 7d ed 4b 02 5b
32328 3a 8d 5c 4f 62 5b 3a 02
32336 5b 71 23 3d 20 fb 05 c8
32344 21 20 00 19 05 28 0c 54
32352 5d 71 3a 02 5b 3d 85 6f
32360 71 18 ed 3a 02 5b 71 23
32368 3d 20 fb c9 ed 4b 00 5b
32376 3a 03 5b f5 cd 00 7d ed
32384 5b 02 5b 16 08 dd 63 e5
32392 7e 2f 77 23 1d 20 f9 e1
32400 24 dd 5c 15 20 ef 04 f1
32408 3d 20 e0 c9 ed 4b 00 5b
32416 3a 03 5b f5 cd 00 7d ed
32424 5b 02 5b 16 08 dd 63 e5
32432 7e d6 0d 30 01 af 77 23
32440 1d 20 f5 e1 24 dd 5c 15
32448 20 eb 04 f1 3d 20 dc c9
```

Listagem 2

```
  10 FOR f=32000 TO 32455
  20 IF PEEK f=125 AND f<>32113
THEN POKE f+32768,253: GOTO 40
  30 POKE f+32768,PEEK f
  40 NEXT f: SAVE "RS"CODE 64768
,456: STOP
```

Listagem 3

MSX

ECTRON

ECTRON - ELETRÔNICA LTDA.

Jogos de Mesa... $ 100
063 Bilhar
149 Vídeo-Poker
162 Otello
281 Sete-e-meio
284 Super Chess
433 Tarot

Paredão etc....... $ 100
122 Arkanoide
350 Break-in
418 Chick Fighter
549 super Snake
569 Frogger
573 Ilusion

Quebra-cabeças.. $ 100
294 Torres
330 Genogram's
566 Puzzle
598 Memory-Games

Bola e Balões..... $ 100
001 Bounder
059 Mister Chin
356 Chost Ball
370 King & Ballon
410 Cub'Hert
545 Ball Blazer
547 Stop The Ball
637 Trail Blazer

Ferrovia........... $ 100
395 Panic Junction
432 Traffic(disco)
568 Craze Train
624 Railway

Corridas........... $ 100
079 Formula One
087 Corrida Maluca
088 American Truck
241 Hang-on(moto)
538 G.P.Rider(moto)
572 G.P.World
677 Car Fighter
681 Car Jamboree

Na cozinha....... $ 100
026 Mister Wong's
219 Macattack
322 Capitain Chef
365 Atack Tomatoes
420 Jump Land

Raid's Aereos.... $ 150
019 River Raid
080 Time Bandits
218 Alfa Squadron
450 Sky Hawk
605 Battle Chopper
647 Ultra Man
661 Gulkave
555 Aqua-Polis
558 Sea King
682 SASA

Guerra............. $ 150
008 Army Moves I
020 Rambo I
076 Green Beret
169 Disc Warrior
258 Front Line
387 Lonesone Tank
595 Rambo II
651 El-Cid
678 Game-Over I
688 Game-Over II

Espaciais.......... $ 100
077 Desolator
177 Zexas Ltd. II
295 Alfa Roid
435 Phantis I

Espaciais.......... $ 150
297 Last Mission
439 Final Justice
542 Scarlet 7
556 Sky Galdo
650 Zaider
672 Star Fighter
685 Zone-Out
687 Droid's

Ficção.............. $ 100
348 Soul of Robot
364 Outroid
372 Colony

Ficção.............. $ 150
363 Mobile Planet
403 Science Fiction
411 Martianoids
436 Phantis II
676 Planet Glass
683 Octagon Squad
684 Gundam

Simuladores...... $ 100
007 Uboat
046 Spitfire 40
110 Boeing 737
152 F16 Simulator
331 The Sprinter
649 Sea Harrier

Esportes........... $ 100
073 Tenis
101 Basquete
108 Ping-Pong
313 Golfe
646 Trialski

Esportes........... $ 150
585 Soccer(futebol)
662 Ice Hockey
640 Alpine Sky

Arte Marcial..... $ 150
016 Avenger
085 Champion Box
606 Uchimata(judo)
660 Kendo

Labirinto.......... $ 100
026 Mister Woong's
106 Ogro
114 Eggerland
140 Globi Blood
248 Mole-Mole
291 Kinasai
293 Spooks Ladders
314 Super Door's
449 Mole-Mole II
459 Godzila

Labirinto.......... $ 150
644 Bouder Dash
668 Pippols
673 Time Bomb
674 Move Pacman
190 Oh Shit!
261 MJ-05
288 Chima-Chima
308 Eric & Floaters

Infantis............ $ 100
524 MSXFIG
614 Mago Voador I
615 Mago Voador II
616 Missão resgate
641 Abelha sábia

Aventuras......... $ 100
018 Padeiro Maluco
032 Goonies
045 Batman
049 Knigh Mare
054 King's Valley
060 The Wreck
067 Feud (halloween)
074 Pitfall
083 Theseus
103 Mister Do
121 Wizzard Lair
125 Demon Cristal
132 Head Over Hell
148 Ghostbuster
151 Camelot Warrior
163 Jumping Rabbit
165 Monkey Academy
166 Elevator Action
198 Load Runner II
221 Circus Charlie
274 Hid Lid
286 B.C.Quest
323 Cruzader

Aventuras......... $ 150
317 Abu & Simbel
336 Terminus
352 Demonia
354 Las Tres Luces
540 Star Byte
599 Freddy Hardest-1
600 Freddy Hardest-2
607 Apeman(D.Kong)
648 He-Man
652 Jack in Nipper-2
653 Nuclear Bowls
657 Pegasus
658 Espadachim
663 Wonder Boy
664 Leon
667 Hard Boyle
680 Temptations

Educativos........ $ 150
617 Star Seeker
631 O Corpo Humano
634 Química
635 Eletricidade
636 Geometria

Programas em Disco..................................
611 DOS Tool's (1 disco) $ 1200
654 Edimus c/ 50 músicas......(1 disco) $ 1000
655 Curso de Exatas............(2 discos) $ 1500
529 Nemesis......................(1 disco) $ 500

Utilitários/aplicativos em disco....................
474 COPY3 (cópias entre disco/fita.... $ 500
473 RKP (cópias entre fitas)............ $ 500
479 Zapper de arquivos...................... $ 300
480 Zapper de setores........................ $ 300
471 Classificador de diretórios........... $ 700
481 Para-disco, após carga de jogo..... $ 200
482 Printer (imprime telas/jogos)....... $ 300
130 Postal (mala-direta)..................... $ 300

Utilitários/aplicativos (disco/fita)..................
486 Mala-direta em fita...................... $ 300
670 Sony-Calc (planilha de cálculo)..... $ 300
490 MSXWRITE (Editor/texto)........... $ 500
489 MSX Banc (Mini Data/Base)......... $ 300
530 Music Studio 67 (edit.musical)...... $ 300
630 Calendar (imprime calendÁrios)... $ 200
491 Game-Design (cria jogos)............ $ 200
528 Master-Voice (digital. som)......... $ 150
632 Condomínios em fita.................. $ 150

Outros produtos...
81811 Fita K7 C60...... $ 400
81821 Disco de 5 e 1/4 pol.............. $ 400
81822 Disco de 3 e 1/2 pol.............. $ 1.200
81311 Driver (5 e 1/4, 360K)...........(consulte)
81312 Driver (5 e 1/4, 720K) (consulte)
81313 Driver (3 e 1/2, 720K) (consulte)
81911 Livro aprofundand. no MSX $ 2.000
81912 Livro 100 dicas para o MSX $ 2.100
81913 Livro Program. Avançada MSX $ 2.000
Preços válidos até 20/08/88. Após esse periodo, calcular 20% a mais

Pedidos via Correio
* Por carta, o pedido mínimo será de 7 programas.
* Use os números dos programas como referência.
* O preço das fitas (ou discos) para gravação serão cobrados a parte: cabem, no máximo, 9 programas em cada uma.
* Não deixe de colocar, no envelope, seus dados completos.
* Pague com cheque nominal ou Vale-Postal.
* As despesas de retorno, correrão por nossa conta.
* Nosso catálogo é completo e grátis: é só pedir.
* Pedidos acima de Cz$ 1.500, dão direito a 2 jogos ou um copiador inteiramente grátis (mencione seus números).
* Peça também por telefone ou venha pessoalmente.

Ectron Eletrônica Ltda.
Rua Leite de Morais, 126 (Caixa Postal 12005)
Metrô Santana — São Paulo — SP
Fone (011) 290-7266 — CEP 02098

ZX SPECTRUM

Screen Loader

Carlos Alexandre da Silva Cardoso

Ao adquirir um TK90X o usuário encontra no manual as referências e exemplos de utilização do comando SCREEN$, que serve para a manipulação de telas gráficas. Ali se aprende que para salvar uma tela gráfica basta usar o comando SAVE "nome" SCREEN$, sem maiores problemas. O inverso é igualmente simples: para carregar uma tela da fita para a memória é só dar o comando LOAD "" SCREEN$ e o micro faz o resto.

Mas há um porém que a gente logo descobre ao tentar carregar a tela de apresentação de um jogo como Deathchase, por exemplo. Para início de conversa, a tela vem gravada junto com o programa, num único bloco; e só para atrapalhar, a gravação ainda é feita sem o header, o que obviamente a torna inacessível para o comando SCREEN$. Desse modo, os usuários que possuem algum aplicativo gráfico como o Paint Plus ou o excelente Art Studio ficam impossibilitados de expressar suas habilidades artísticas, pois tais programas não conseguirão carregar essas telas. E para piorar as coisas, é cada vez maior o número de programas que vêm sendo gravados desta maneira.

O que fazer, então?

Para a salvação da lavoura, desenvolvi um utilitário capaz de carregar uma tela gravada com ou sem header, mesmo que o bloco tenha mais de 6912 bytes e esteja com os ponteiros trocados. Esta é mais uma pequena proteção de que se valem alguns softwares para dificultar a ação de programas copiadores: modificar o primeiro byte dos blocos em fita, para dar a impressão de que o bloco acabou. No intuito de permitir a recuperação das telas, desenvolvi uma rotina de carregamento que ignora a maior parte dessas proteções.

```
100 INK 4: PAPER 0: BORDER 0: BR
IGHT 1: CLS
110 PRINT "SCREEN LOADER V1.0"
120 PLOT 0,166: DRAW INK 3; BRIG
HT 1;150,0
130 PRINT "1 - Ver tela""2 - LO
AD c/ header"
140 PRINT "3 - LOAD s/ header""
4 - Salvar tela""5 - Mudar mark
: "; INK 7;MARK
150 LET X$=INKEY$: IF X$<"1" OR
X$>"5" THEN GOTO 150
160 GOSUB RUIDO
170 GOSUB VAL X$*1000
180 GOTO 100
1000 REM inicio das rotinas
1010 RAND USR MOVTELA
1020 PAUSE 140
1030 GOSUB RUIDO
1040 GOSUB ESPERA
1050 RETURN
2000 CLS : PRINT "Acione o grava
dor"
2010 IF X$="2" THEN RAND USR LOD
TELA
2020 RAND USR LODTELA
2030 PRINT "Trabalho feito..."
2040 GOSUB ESPERA
2050 RETURN
3000 GOTO 2000
4000 CLS : PRINT "Ligue o gravad
or e aperte ENTER"
4010 IF INKEY$<>CHR$ 13 THEN GOT
O 4010
4020 RAND USR MOVTELA
4030 RAND USR SAVTELA
4040 GOSUB RUIDO: GOSUB ESPERA
4050 RETURN
5000 IF MARK=255 THEN LET MARK=0
: GOTO 5020
5010 IF MARK=0 THEN LET MARK=255
5020 POKE 65028,MARK
5030 RETURN
6000 PRINT #0; FLASH 1; "Qualque
r tecla para menu"
6010 IF INKEY$="" THEN GOTO 6010
6020 RETURN
7000 FOR X=1 TO 10: SOUND 5E-03,
15: SOUND .02,X: NEXT X
7010 RETURN
8000 CLEAR 55999
8020 LET SAVTELA=65000
8030 LET LODTELA=65026
8040 LET MOVTELA=65040
8050 LET RUIDO=7000
8060 LET ESPERA=6000
8070 LET MARK=255
8080 DATA 62,0,221,33,28,254,17,
17,0,205,194,4,0,62,255,221,33,0
,64,17,0,27,205,194,4,201,55,62,
255,221,33,192,218,17,0,27,205,8
6,5,201,17,0,64,33,192,218,1,0,2
7,237,176,201
8090 DATA 3,170,76,79,65,68,69,8
2,32,32,32,0,27,0,64,0,0
8100 FOR X=65000 TO 65068: READ
A: POKE X,A: NEXT X
8100 GOTO 100
```

OPERAÇÃO

Para operar o Screen Loader, digite a listagem 1 (cuidado com as linhas DATA) e grave-a com o seguinte comando direto:

SAVE "S-LOADER" LINE 8000

O programa se executará automaticamente após o carregamento, exibindo na tela um menu com cinco opções. Posicione a fita no gravador e aperte a tecla correspondente à opção desejada. Caso você queira carregar uma tela sem header e o programa não o esteja fazendo corretamente, altere o *mark* usando a opção 5 e repita a tentativa.

Por final, é interessante observar que algumas telas que apresentam defeito durante o carregamento são recuperadas pelas rotinas do Screen Loader.

Finalmente MSX em Videocassete

Tenha um professor exclusivo em casa, orientando-o de uma forma didática como usufruir de todos os recursos de seu equipamento MSX.
DOMINANDO O MSX
Apresentação:
PIERLUIGI PIAZZI

À VENDA:
* Brenno Rossi * Bruno Blois
* Computer Technics * Ectron
* Filcril * Lima * Mappin
* Paulisoft * Princessware
* Softnew * Universoft
Distribuição para magazines:
Orionsoft (011) 228-9598

Produção:
MPO SOFT VIDEO
Av. Paulista, 2001
19º andar – conj. 1923
Tel.: (011) 285-3875

ZX SPECTRUM

Compressor de telas

Francisco Antonio Nascimento

Se você é o feliz proprietário de um micro TK90X ou TK95, o número 6912 deve lhe parecer dolorosamente familiar. Esta nada desprezível quantidade de bytes, gasta no mapeamento do vídeo dos ZX Spectrum-compatíveis (256 pontos horizontais X 192 pontos verticais + atributos), representa a principal causa da calvície precoce normalmente observada nos usuários de tais máquinas que tentam desenvolver programas com um visual mais apurado — sem falar no desespero de alguns outros que cultivam o estranho hábito de esperar um programa com uma ou mais telas carregar do gravador cassete. Assim, meu prezado amigo, antes que você também se torne mais um candidato ao controle de pressão ou a um implante capilar em tão tenra idade (se é que ainda não chegou lá), aceite este conselho: está na hora de começar a compactar as suas telas.

Caso você ainda não tenha parado para pensar nisso, tal feito pode ser conseguido mediante o uso de uma técnica bastante simples. Repare que o byte que aparece com mais freqüência numa tela é o zero, ou seja, o espaço em branco. Desse modo, o que temos a fazer é criar uma tabela apontando a posição e a quantidade de espaços encontrados em seqüência num trecho da tela. Os bytes diferentes de zero serão agrupados à parte, numa outra posição da tabela.

Tecnicamente, podemos resumir o procedimento da seguinte forma: se o byte que retiramos da tela for zero, incrementa-se um contador a ele associado. Caso este atinja 255 (maior valor possível para um byte num micro de 8 bits), transferimos o contador para a tabela de zeros e o ressetamos em seguida. Para recuperar uma tela comprimida por este processo, basta seguir o caminho inverso: verificamos a memória e, se acharmos um zero, preenchemos a tela com os "n" espaços indicados pela tabela. Se o byte encontrado for diferente de zero, nós o transferimos normalmente para a tela.

UTILIZANDO O PROGRAMA

Rode o programa da listagem 1 e grave a rotina com o comando SAVE "ROTRED" CODE 57000,200. Para carregar é só dar o comando CLEAR 50000: LOAD " " CODE. Agora, com a rotina na memória, prepare uma tela e introduza o programa da listagem 2; ele irá gravar a tela no formato comprimido. Para recuperar uma tela reduzida, digite o seguinte comando direto:

CLEAR 50000: LOAD " " CODE: RAND USR 57100

A estrutura da memória durante a utilização do programa ficará como ilustrado na figura 1.

Para encerrar, e a fim de que você não perca os últimos cabelos que lhe restam, não custa lembrar o óbvio: a eficiência deste programa é diretamente proporcional à quantidade de bytes zero existentes na tela. Desse modo, a aplicação da rotina em telas com poucos espaços resultará em pequena economia de memória.

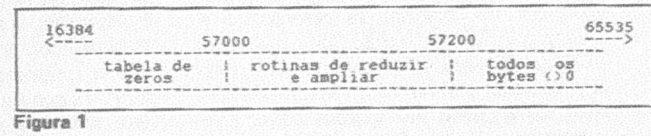


Figura 1

```
10 LOAD "" SCREEN$
20 RAND USR 57000
30 LET INI=PEEK 57195+256*PEEK 5
7196
40 LET FIM=PEEK 57197+256*PEEK 5
7198
50 INPUT "NOME DA TELA:"; LINE N
$
60 SAVE N$ CODE INI,FIM-INI
```

Listagem 1

```
5 CLEAR 55000
10 DATA 33,0,64,17,112,223,221,3
3,167,222,253,33,111,223,175,253
,119,0,126,254,1,56,35,229,43,12
6,254,1,225,48,9,253
20 DATA 126,0,205,237,222,253,11
9,0,126,18,19,35,124,254,91,56,2
25,221,34,107,223,237,83,109,223
,201,253,126,0,60,254
30 DATA 255,212,237,222,253,119,
0,35,24,227,221,119,0,221,43,175
,18,19,201,33,0,64,17,112,223,22
1,33,167,222,26,254,1
40 DATA 56,10,26,119,35,19,124,2
54,91,56,242,201,221,70,0,175,11
9,35,16,252,19,221,43,24,237
50 FOR I=57000 TO 57081:READ A:P
OKE I,A:NEXT I
60 FOR I=57100 TO 57137:READ A:P
OKE I,A:NEXT I
```

Listagem 2

* DRIVE 5.1/4 SLIN COMPLETO
* EXPANSOR DE SLOT (C/4 SLOTS)
* PLACA 80 COLUNAS
* GABINETE P/DRIVE COM FONTE FRIA
* MODEN DE COMUNICAÇÃO
* INTERFACE DUPLA P/DRIVE

* **PACOTÃO EM DISCO:** 100 JOGOS (ESCOLHER) + 5 APLICATIVOS + 10 DISCOS = Cz$ 40.000,00
* **PACOTÃO EM FITA:** 100 JOGOS (ESCOLHER) + 5 APLICATIVOS + 7 FITAS = Cz$ 40.000,00

SOLICITE NOSSO CATÁLOGO DE PROGRAMAS PARA FAZER A SUA ESCOLHA. ATENDEMOS TODOS OS ESTADOS EM 24 HORAS VIA SEDEX.
PARA FAZER SEU PEDIDO ENVIE CHEQUE NOMINAL COM CARTA DETALHADA PARA **A. NASSER.**
RIO DE JANEIRO: RUA GONZAGA BASTOS 411/203 — VILA ISABEL — RJ CEP 20541 — TEL (021) 234-0775
FILIAL CURITIBA: AV. 7 DE SETEMBRO, 3146 LOJA 20 — SHOPPING SETE — CURITIBA — PR — CEP 80010 — TEL 233-0046

Você possui um TK90X e quer desenvolver o seu próprio adventure? Se a resposta for sim, este editor poderá auxiliá-lo nesta missão.

Editor de Adventures

Paolo Fabrizio Pugno

Antes de mais nada, vamos botar os pingos nos devidos iis: os programas "Editor" e "Adventure" foram elaborados e adaptados para a linha ZX Spectrum a partir de uma idéia semelhante publicada na revista italiana *"Microcomputer"*, para a linha Apple.

O assunto é adventure, este fascinante tipo de jogo que põe a dura prova nosso raciocínio, estratégia, paciência, e, às vezes, até mesmo sorte. Jogá-lo é uma coisa, elaborá-lo é outra totalmente diferente, onde devemos levar em consideração todas as possíveis situações que poderão ser enfrentadas pelo programa. Em resumo, não é nada fácil, embora não seja impossível. E tem mais, para cada adventure que escrevermos, teremos que bolar um programa gerenciador específico, que manipule os dados necessários. Isto faz com que horas e mais horas sejam perdidas na depuração de um maldito programa. Às vezes, só de pensar em enfrentar tais dificuldades, desistimos logo de saída. Mas, que tal um programa universal, isto é, que sirva para jogar todos os adventures que escrevermos? Tudo o que temos a fazer é "ensinar-lhe" as regras do jogo. Porém, é necessário que os dados lhe sejam fornecidos de uma maneira padronizada, e é aqui que entra em cena o "Editor".

Na realidade, são dois programas: com o "Editor" ensinamos as regras do jogo à máquina, que vai organizá-las de uma forma que o "Adventure" possa manipulá-las. Os dados são salvos em fita, através da gravação de matrizes numéricas e alfanuméricas, que serão posteriormente carregadas pelo "Adventure". Desta forma, podemos inventar o adventure que quisermos, sem nos preocuparmos com o programa gerenciador. Como jogar um adventure feito por você mesmo perde logo a graça, a sugestão é realizar trocas entre amigos que possuam um ZX Spectrum ou TK90X; essa troca consistiria nas matrizes salvas em fita pelo "Editor".

EDITOR

Antes de rodar o "Editor", é obrigatório anotar tudo o que for necessário; fazer um mapa; dar números aos locais (começando por 1) e aos objetos; anotar os verbos e seus complementos; enfim, tudo o que depois será pedido pelo programa.

Dado o **RUN**, o "Editor" irá pedir o nome do jogo, o qual não deverá ter mais de dez caracteres, pois será utilizado como nome de uma matriz na hora de gravar os dados em fita. O programa então perguntará se o adventure é por pontos. Teclando S será pedido o escore a ser atingido para que o jogador alcance a vitória. Logo a seguir, devemos introduzir o número total de locais de jogo, após o que será pedida a descrição de cada local. Essa descrição não deve ultrapassar três linhas (96 caracteres); o que, no entanto, pode ser alterado na linha 60, mas cuidado para não desperdiçar memória. Após introduzirmos cada descrição, devemos indicar que local é atingido quando nos movimentamos nas seis direções possíveis (Norte, Sul, Leste, Oeste, Suba e Desça). Introduzindo o zero para uma direção qualquer, iremos provocar o aparecimento da mensagem "não posso ir nessa direção", quando a mesma for executada nesse local; introduzindo o número do próprio local, não haverá deslocamento. Note-se que não é obrigatório nos locomovermos para locais vizinhos (no mapa). Aliás, quanto mais complicado for o mapa, mais interessante o adventure se tornará.

Na segunda parte do programa, é pedido o número de objetos existentes no jogo e, depois, o nome de cada um (sem utilizar artigo) e sua localização inicial. Aqui, há um truque: o local zero não existe no jogo, e serve para colocarmos temporariamente objetos que devem desaparecer ou para guardarmos outros que irão surgir posteriormente. Portanto, se uma chave só aparecer quando quebrarmos um vaso (por exemplo), seu local inicial será o zero e quando a ação "quebre o vaso" for executada, seu vetor posição será igual à posição do jogador. Não há limites de objetos por local. O nome do objeto deve ter, no máximo, 14 caracteres.

Agora vem a parte mais importante, os verbos: Pegue; Apanhe; Solte e Largue, para a manipulação de objetos (alguns verbos já estão programados no adventure por serem os mais utilizados, portanto, não é necessário defini-los).

Além dos verbos, podemos utilizar os seguintes comandos: Inventário (I), que mostra quais objetos estamos carregando; Pontos, informa sobre o escore; e Desisto, faz você recomeçar o jogo ou terminá-lo. E, é claro, Norte (N), Sul (S), Leste (L), Oeste (O), Suba e Desça, para os movimentos.

Para definir um verbo (e, portanto, uma ação), é necessário que tenhamos uma série de informações. Suponha que a descrição de um determinado local nos diz que há uma porta fechada ali. Estabelecemos que essa porta só pode ser aberta com uma chave, que deve estar com o jogador, e que ao abri-la, perdemos a chave e somos transportados para outro local qualquer. A ação será "abra a porta", e ela só pode ser executada nesse local onde há a porta e que, além disso, ao executá-la ganhamos alguns pontos.

Como codificar este exemplo? O micro vai perguntar:
1) Imperativo do verbo:
Abra.
2) Complemento:
Porta (sem utilizar artigo).
3) Resposta do micro:
"Ok. Puxa que lugar estranho!" (se quisermos que a resposta seja somente "Ok.", basta teclarmos ENTER).
4) Para que local sou transportado?
(Aqui deverá ser indicado o número do local de destino, sendo que 0 indicará a permanência no local).
5) Deslocamento de objetos:
Nesta pergunta, ao introduzirmos um número inteiro positivo, indicamos que o objeto que tem esse número irá aparecer no local. Um número inteiro negativo indica que o objeto que tem esse número (em módulo) irá desaparecer, ou melhor, sua posição será a sala zero, de que falamos há pouco. No nosso caso, se a chave for o objeto número 3, teclaremos: -3. Se o número indicado for zero, não haverá deslocamento de objetos.
6) Variação de escore (só se o jogo for por pontos):
Se o adventure for por pontos, introduziremos a variação no escore do jogador (positiva, negativa e nula) quando for executada esta ação.
7) Onde executar o comando?
Introduzindo um número inteiro positivo, indicamos que esta ação só pode ser executada no local que tenha este número. Um número inteiro negativo indica que esta ação só não pode ser executada no local que tenha este número (em módulo). Zero indica que a ação pode ser executada em qualquer lugar.
8) Objeto necessário:
Aqui, indicamos o número do objeto necessário para a execução desta ação. O objeto deve estar em poder do jogador para que esta ação seja executada. Zero indica que não é necessário objeto algum. No nosso caso, teclaremos 3 (o número da chave).
9) Mais de uma execução?
Se a ação puder ser executada várias vezes, teclaremos S, caso contrário, teclaremos N. Se teclarmos N, o micro somente executará esta ação uma vez, recusando-se a repeti-la. Por exemplo, não podemos quebrar o mesmo vaso duas vezes.

Note que se você quiser incluir o verbo "examine", deverá programá-lo para cada objeto que puder ser examinado. O programa permite que tenhamos diversos imperativos iguais, desde que os complementos sejam diferentes; assim como permite complementos iguais, se os imperativos forem diferentes. Então, podemos ter um mesmo adventure: abra porta, abra janela, abra garrafa, quebre garrafa, tampe garrafa, examine chave, examine vaso, examine garrafa etc., desde que forneçamos as informações adequadas para cada caso.

Observações importantes: o local -1 é o que indica que o jogador morreu ou foi eliminado do jogo. Portanto, se ao executar alguma ação o jogador levar a pior, devendo ser eliminado (por exemplo: salte despenhadeiro), basta transportá-lo para o local -1. Analogamente, o local -2 é o da vitória. Quando o adventure for resolvido, basta transportar o jogador para o local -2, que automaticamente o programa desvia para uma rotina de felicitações. A passagem para estes locais pode ser feita normalmente, seja através de conseqüência de uma ação (ou ações) como por deslocamento voluntário (N, S, L, O, Suba ou Desça).

Além disso, quando um objeto está na posição -1 significa que está em poder do jogador, o qual poderá carregar até cinco objetos. As posições dos objetos e do jogador não interferem entre si, pois são guardadas em variáveis distintas.

Por fim, será perguntado o local inicial de jogo, e, então, serão gravadas em fita as 14 matrizes que contêm os dados introduzidos. Devido às características do micro, é necessário que apertemos uma tecla qualquer antes do SAVE, portanto, cuidado para não se distrair. Se o VERIFY acusar erros, rebobine a fita e dê um GO TO 400.

Quanto à digitação, esta não apresenta maiores problemas; as escritas em INVERSE são obtidas teclando-se INVERSE VIDEO, antes do texto, e TRUE VIDEO, depois. Cuidado máximo nas li-

CP500/M,® uma grande decisão para homens que tomam decisões.

O CP 500/M^{80} agora também é compatível com o CP/M, o Sistema mais difundido no mundo. Isso significa que, somando o tradicional DOS 500 com o CP/M, você passa a dispor da maior biblioteca de programas aplicativos do mercado. E você poderá tirar vantagens de programas como Calc Star*, dBase II*, Word Star* e muitos outros, que vão agilizar tanto sua vida quanto a de sua empresa.
Para você, executivo, o CP 500/M^{80} é uma poderosa ferramenta de auxílio a decisões gerenciais, com programas específicos em sua área de atuação, como análise de fluxo de caixa, controles de compras/custos/estoques e administração financeira.
Como resultado final, sua empresa terá grandes ganhos de produtividade e eficiência.
CP 500/M^{80}, um grande passo na informática com a Qualidade CP e a Tecnologia Prológica.

Características	Sistemas Operacionais Inclusos	
Memória RAM	SO-08	DOS 500
Memória ROM	64 Kbytes	48 Kbytes
Compatibilidade	2 Kbytes	16 Kbytes
Vídeo	CP/M*	TRS-DOS
	80 colunas por 24 linhas	64 ou 32 colunas por 16 linhas

Vídeo 12" fósforo verde com controle de intensidade
Teclado profissional capacitivo com numérico reduzido
Interface para impressora paralela
Opera com 1 ou 2 drives de face simples ou dupla
Acesso a toda a Rede Nacional de Telemática, através de Interface serial padrão RS-232 C (opcional)

* Marcas Registradas

CP
COMPUTADORES PESSOAIS
TECNOLOGIA PROLÓGICA

nhas 410 e 430; elas devem ser iguais, exceto pela substituição dos **SAVE** pelos **VERIFY**. Uma dica é escrever a linha 410, editá-la e substituir o número de linha e os comandos **SAVE** por **VERIFY**.

ADVENTURE

Ao rodar o segundo programa, ele irá simplesmente perguntar o nome do adventure que desejamos carregar, e, logo após o término do carregamento, entrar automaticamente em execução.

Algumas recomendações: para movimentar-se, tecle "va para o norte" ou simplesmente N. Ao executar uma ação, você pode usar o artigo que quiser, tendo em mente que o programa reconhece somente a primeira e a última palavra da frase introduzida. Podemos então digitar: pegue martelo, pegue o martelo, pegue esse martelo ou pegue essa droga de martelo; que o efeito será o mesmo em todos os casos. Esse é o motivo pelo qual o Editor não aceita os artigos.

OUTROS MACETES

Se há uma sala escura que só revela seu conteúdo se acendermos uma vela (por exemplo), então, é conveniente termos duas salas; sendo uma com a descrição da escuridão, sem outras saídas ou objetos; e outra, que é o seu equivalente, caso tenhamos luz. Definimos assim uma ação, "acenda vela", de tal forma que só poderá ser executada na sala escura e se tivermos a vela, e que além disso, nos transporte para a outra sala, com a outra descrição. E um verbo, "apague vela", que faça o contrário.

Experimentalmente, conseguiu-se dimensionar matrizes referentes a 100 salas, 80 objetos e 150 verbos, o que acreditamos serem suficientes para a maioria dos casos. O comando **POKE 23658,8** força o cursor para o modo C (maiúsculas). Se você quiser incrementar um pouco mais, experimente um **POKE 23609,30**.

UM EXEMPLO

Aqui está um exemplo de adventure com cinco salas. Tente rodá-lo. Os números estão na ordem em que o micro os pede.

Nome – "Teste"
É por pontos? – Não
Quantos locais? – 5

LOCAL 1

Descrição: "Estamos numa velha sala de estar. Há muito pó, aqui!"

NORTE	SUL	LESTE	OESTE	SUBA	DESÇA
0	0	2	0	0	0

LOCAL 2

Descrição: "Estamos num corredor. Há uma escada que desce e uma porta".

NORTE	SUL	LESTE	OESTE	SUBA	DESÇA
0	0	0	1	0	4

LOCAL 3

Descrição: "Que lugar estranho! Há um botão na parede".

NORTE	SUL	LESTE	OESTE	SUBA	DESÇA
0	0	0	2	0	0

LOCAL 4

Descrição: "Está muito escuro, mas percebo uma tocha apagada na parede".

NORTE	SUL	LESTE	OESTE	SUBA	DESÇA
0	0	0	0	2	0

Adventure

```
   1)REM **********************
   2 REM **      ADVENTURE     **
   3 REM **********************
   4 REM
  10 POKE 23658,8: BORDER 1: PAP
ER 5: INPUT "QUAL O NOME DO ADVE
NTURE QUE    VOCE QUER JOGAR?";
A$: CLS
  15 LOAD A$ DATA Z(): LOAD "LOC
AIS" DATA L$(): LOAD "DESLOC." D
ATA M(): LOAD "OBJETOS" DATA O$(
): LOAD "POS.OBJ." DATA P(): LOA
D "VERBOS" DATA V$(): LOAD "COMP
L." DATA C$(): LOAD "MENSAGENS"
DATA M$(): LOAD "VOUPARA" DATA D
(): LOAD "DES.OBJ." DATA F(): LO
AD "AUM.SCORE" DATA A(): LOAD "P
ODE?" DATA U(): LOAD "OBJ.NECES.
" DATA B(): LOAD "REP.VER." DATA
 K()
  20 CLS : LET SCORE=0: LET POS=
Z(1): LET SPON=Z(2): LET PONMAX=
Z(3): LET NL=Z(4): LET NO=Z(5):
LET NV=Z(6): DIM E(NV)
  25 LET J=0: LET POSI=POS: DIM
Q(NO): FOR I=1 TO NO: LET Q(I)=P
(I): NEXT I: CLS
  30 PRINT L$(POS)''
  35 IF POS=-1 THEN GOTO 470
  40 IF POS=-2 THEN GOTO 495
  45 LET QW=0
  50 FOR I=1 TO NO
  55 IF P(I)<>POS THEN GOTO 75
  60 IF QW THEN GOTO 70
  65 PRINT '"OS OBJETOS VISIVEIS
SAO:": LET QW=1
  70 PRINT O$(I, TO 14),
  75 NEXT I
  80 INPUT FLASH 1;"O QUE FAREI
AGORA? "'"H$: PRINT ''">";H$''
  85 IF H$="" THEN CLS : GOTO 30
  90 LET A$="": FOR F=1 TO LEN H
$
  95 IF H$(F)=" " THEN GOTO 105
 100 LET A$=A$+H$(F): NEXT F
 105 LET B$="": FOR F=LEN H$ TO
LEN A$+1 STEP -1
 110 IF H$(F)=" " THEN GOTO 120
 115 LET B$=H$(F)+B$: NEXT F
 120 LET QW=0: FOR I=1 TO NV
 125 IF A$()V$(I, TO CODE V$(I,1
5)) THEN GOTO 195
 130 IF B$()C$(I, TO CODE C$(I,1
5)) THEN GOTO 195
 135 IF K(I) AND E(I) THEN PRINT
 FLASH 1;"Mas ja fiz isso!!": GO
TO 190
 140 IF B(I) THEN IF P(B(I))()-1
 THEN PRINT "Ainda nao posso faz
e-lo!!": GOTO 190
 145 IF U(I)=0 THEN GOTO 160
 150 IF SGN U(I)=-1 THEN IF POS=
ABS U(I) THEN PRINT "Nao posso f
aze-lo aqui!!": GOTO 190
 155 IF U(I)>0 AND U(I)()POS THE
N PRINT "Nao posso faze-lo aqui!
!": GOTO 190
 160 PRINT ''M$(I)
 165 IF D(I) THEN LET POS=D(I):
IF POS>=1 THEN PRINT ''L$(POS)''
 170 IF F(I)=0 THEN GOTO 185
 175 IF SGN F(I)=1 THEN LET P(I)
=POS: GOTO 185
 180 LET P(ABS F(I))=0
 185 LET E(I)=1: IF SPON THEN LE
T SCORE=SCORE+A(I)
 190 LET I=NV: LET QW=1
 195 NEXT I
 200 IF SCORE)=PONMAX THEN GOTO
495
 205 IF QW THEN GOTO 35
 210 IF NOT A$="PONTOS" THEN GOT
O 220
 212 IF NOT SPON THEN PRINT "O A
DVENTURE NAO E POR PONTOS!": GOT
O 80
 215 PRINT "TEMOS ATE AGORA ";SC
ORE;" PONTOS''': GOTO 80
 220 IF NOT (A$="PEGUE" OR A$="A
PANHE") THEN GOTO 280
 225 IF B$="" THEN PRINT "DIGA-M
E PELO MENOS O QUE!"': GOTO 80
 230 IF J=5 THEN PRINT "NAO POSS
O CARREGAR MAIS NADA!": GOTO 80
 235 LET QW=0: FOR I=1 TO NO
 240 IF NOT (B$=O$(I, TO CODE O$
(I,15)) AND P(I)=POS) THEN GOTO
265
 245 PRINT "OK, ";A$;"I ";B$
 250 LET QW=1
 255 LET P(I)=-1
 260 LET J=J+1
 265 NEXT I
 270 IF NOT QW THEN PRINT INVERS
E 1;B$;"?"'"NAO VEJO ISSO POR AQ
UI."
 275 GOTO 80
 280 IF NOT (A$="SOLTE" OR A$="L
ARGUE") THEN GOTO 335
 285 LET QW=0
 290 FOR I=1 TO NO
 295 IF NOT (B$=O$(I, TO CODE O$
(I,15)) AND P(I)=-1) THEN GOTO 3
20
 300 PRINT "OK. ";A$;"I ";B$
 305 LET QW=1
 310 LET P(I)=POS
 315 LET J=J-1
 320 NEXT I
 325 IF QW THEN GOTO 35
 330 PRINT "NAO TEMOS ";B$;" CON
OSCO!": GOTO 80
 335 IF A$()"DESISTO" THEN GOTO
345
 340 CLS : PRINT "NESTE ADVENTUR
E VOCE FEZ ";SCORE'"PONTOS, ISTO
 E',";SCORE/PONMAX*100;"%"'"DO T
OTAL"'': GOTO 475
 345 IF NOT (A$="I" OR A$="INVEN
TARIO") THEN GOTO 400
 350 LET QW=0
 355 FOR I=1 TO NO
 360 IF P(I)()-1 THEN GOTO 385
 365 IF QW THEN GOTO 380
 370 PRINT "ESTOU CARREGANDO:"'
 375 LET QW=1
 380 PRINT TAB 10; BRIGHT 1;O$(I
 TO 14)
 385 NEXT I
 390 IF QW THEN GOTO 80
 395 PRINT "NAO ESTOU CARREGANDO
 NADA...": GOTO 80
 400 IF NOT ((A$="N" AND M(POS,1
)) OR (A$="VA" AND B$="NORTE" AN
D M(POS,1))) THEN GOTO 410
 405 LET POS=M(POS,1): GOTO 30
 410 IF NOT ((A$="S" AND M(POS,2
)) OR (A$="VA" AND B$="SUL" AND
M(POS,2))) THEN GOTO 420
 415 LET POS=M(POS,2): GOTO 30
 420 IF NOT ((A$="L" AND M(POS,3
)) OR (A$="VA" AND B$="LESTE" AN
D M(POS,3))) THEN GOTO 430
 425 LET POS=M(POS,3): GOTO 30
 430 IF NOT ((A$="O" AND M(POS,4
)) OR (A$="VA" AND B$="OESTE" AN
D M(POS,4))) THEN GOTO 440
 435 LET POS=M(POS,4): GOTO 30
 440 IF NOT (A$="SUBA" AND M(POS
,5)) THEN GOTO 450
 445 LET POS=M(POS,5): GOTO 30
 450 IF NOT (A$="DESCA" AND M(PO
S,6)) THEN GOTO 460
 455 LET POS=M(POS,6): GOTO 30
 460 IF LEN A$=1 OR A$="SUBA" OR
 A$="DESCA" OR A$="VA" THEN PRIN
T "NAO POSSO IR NESTA DIRECAO!":
 GOTO 80
 465 PRINT FLASH 1; INK 2; PAPER
 6;"NAO ENTENDI A SUA ORDEM!": G
OTO 80
 470 PAUSE 0: BORDER 0: PAPER 0:
 INK 6: CLS : PRINT AT 10,0; FLA
SH 1; BRIGHT 1;"SINTO MUITO, MAS
 VOCE PERDEU...  MELHOR SORTE NA
 PROXIMA VEZ... ": FOR F=40 TO 0
 STEP -1: SOUND .01,F: NEXT F
 475 INPUT INK 9;"QUER JOGAR DE
NOVO? (S/N) ";H$
 480 IF H$="N" THEN STOP
 485 FOR I=1 TO NO: LET P(I)=Q(I
): NEXT I: LET POS=POSI: LET SCO
RE=0: FOR I=1 TO NV: LET E(I)=0:
 NEXT I: LET J=0
 490 BORDER 1: PAPER 5: INK 0: C
LS : GOTO 30
 495 PAUSE 0: BORDER 4: PAPER 6:
 INK 1: CLS : PRINT AT 5,0; FLAS
H 1;"MEUS PARABENS! VOCE CONSEGU
IU  COMPLETAR O ADVENTURE!!!";
FOR F=0 TO 40: SOUND .01,F: NEXT
 F: IF SPON THEN PRINT ''"VOCE
FEZ ";SCORE;" PONTOS,"'"DE UM TO
TAL DE ";PONMAX;"."'"ISTO E', ";
SCORE/PONMAX*100;"% DO TOTAL!"
 500 GOTO 475
```

Editor

```
  1)REM **********************
  2 REM **      EDITOR      **
  3 REM **********************
  10 BRIGHT 0: PAPER 7: INK 1: B
ORDER 1: CLS : POKE 23658,8: CLS
 : INPUT "QUAL E O NOME DO ADVEN
TURE ?  ";A$: IF A$="" THEN GO
TO 10
  20 LET SPON=0: LET PONMAX=1
  30 PAPER 6: CLS : PRINT "O ADV
ENTURE E POR PONTOS ? (S/N)": I
NPUT H$: PRINT H$: IF H$()"N" AN
D H$()"S" THEN GOTO 30
  40 IF H$="S" THEN LET SPON=1:
PRINT "QUAL E O SCORE MAXIMO ?":
 INPUT PONMAX: PRINT PONMAX
  50 PAPER 4: BRIGHT 1: CLS : IN
PUT "QUANTOS LOCAIS TEM O ADVENT
URE ?";NL
  60 DIM L$(NL,96): DIM M(NL,6)
  70 FOR I=1 TO NL
  80 CLS : PRINT "DESCRICAO DO L
OCAL ";: INVERSE 1;I: INPUT LINE
 L$(I, TO ): PRINT L$(I, TO )''
  90 PRINT "QUE LOCAIS ALCANCO E
XECUTANDO...""
 100 PRINT "NORTE     (N) ";: INP
UT H: LET M(I,1)=H: PRINT H
 110 PRINT "SUL       (S) ";: INP
UT H: LET M(I,2)=H: PRINT H
 120 PRINT "LESTE     (L) ";: INP
UT H: LET M(I,3)=H: PRINT H
 130 PRINT "OESTE     (O) ";: INP
UT H: LET M(I,4)=H: PRINT H
 140 PRINT "SUBA          ";: INP
UT H: LET M(I,5)=H: PRINT H
 150 PRINT "DESCA         ";: INP
UT H: LET M(I,6)=H: PRINT H
 160 INPUT "TUDO OK? ";H$: IF H$
="N" THEN GOTO 80
 170 NEXT I
 180 PAPER 1: BRIGHT 1: INK 7: C
LS : INPUT "QUANTOS OBJETOS HA N
O JOGO ? ";NO: DIM O$(NO,15): DI
M P(NO)
 190 FOR I=1 TO NO
 200 CLS : PRINT "QUAL O OBJETO
";: INVERSE 1;I": INPUT H$: LET O
$(I, TO 14)=H$: LET O$(I,15)=CHR
$ LEN H$: PRINT H$''
 210 PRINT "EM QUE LOCAL SE ENCO
NTRA? ";: INPUT P(I): PRINT P(I)
 220 INPUT "TUDO OK? ";H$: IF H$
="N" THEN GOTO 200
 230 NEXT I
 240 PAPER 2: INPUT "QUANTOS VER
BOS? ";NV: DIM V$(NV,15): DIM C$
(NV,15): DIM M$(NV,64): DIM D(NV
): DIM F(NV): DIM A(NV): DIM U(N
V): DIM B(NV): DIM K(NV): FOR I=
1 TO NV
 250 CLS : PRINT "QUAL O VERBO "
;I": INPUT H$: LET V$(I, TO 14)=
H$: LET V$(I,15)=CHR$ LEN H$: PR
INT H$
 260 PRINT "QUAL SEU COMPLEMENT
O ?": INPUT H$: LET C$(I, TO 14
)=H$: LET C$(I,15)=CHR$ LEN H$:
PRINT H$
 270 PRINT "QUAL A RESPOSTA DO
MICRO ?": INPUT LINE H$: IF H$="
" THEN LET H$="OK."
 280 LET M$(I, TO )=H$: PRINT M$
(I, TO )
 290 PRINT "PARA QUE LOCAL SOU
TRANSPORTADO (0=NAO SOU TRASNSPO
RTADO      ": INPUT D(I): PRIN
T D(I)
 300 PRINT "DESLOCAMENTO DE OBJ
ETO?          (+n,-n,0)": INPUT F
(I): PRINT F(I)
 310 IF SPON THEN PRINT "VARIAC
AO DE SCORE?   (+n,-n,0)  ": INP
UT A(I): PRINT A(I)
 320 PRINT "ONDE POSSO EXECUTAR
 O COMANDO?  ": INPUT U(I): PRIN
T "SALA ";U(I)
 330 PRINT "QUE OBJETO DEVO TER
?  (0=NENHUM)": INPUT B(I): IF B
(I) THEN PRINT O$(B(I), TO CODE
O$(B(I),15))
 340 PRINT "POSSO EXECUTAR ESTE
VERBO MAIS  DE UMA VEZ? (S/N)":
 INPUT H$: IF H$="N" THEN LET K(
I)=1: GOTO 360
 350 LET K(I)=0
 360 INPUT FLASH 1;"TUDO OK? ";H
$: IF H$="N" THEN GOTO 250
 370 NEXT I
 380 PAPER 4: INK 0: BRIGHT 0: C
LS : PRINT "QUAL O LOCAL DE INIC
IO? ";: INPUT POS: PRINT POS
 390 DIM Z(6): LET Z(1)=POS: LET
 Z(2)=SPON: LET Z(3)=PONMAX: LET
 Z(4)=NL: LET Z(5)=NO: LET Z(6)=
NV
 400 PAPER 7: INK 2: CLS : PRINT
 AT 10,5;"VOU GRAVAR AS MATRIZES
            APERTE UMA TECLA": P
AUSE 0
 410 SAVE A$ DATA Z(): SAVE "LOC
AIS" DATA L$(): SAVE "DESLOC." D
ATA M(): SAVE "OBJETOS" DATA O$(
): SAVE "POS.OBJ." DATA P(): SAV
E "VERBOS" DATA V$(): SAVE "COMP
L." DATA C$(): SAVE "MENSAGENS"
DATA M$(): SAVE "VOUPARA" DATA D
(): SAVE "DES.OBJ." DATA F(): SA
VE "AUM.SCORE" DATA A(): SAVE "P
ODE?" DATA U(): SAVE "OBJ.NECES.
" DATA B(): SAVE "REP.VER." DATA
 K()
 420 CLS : PRINT AT 10,0;"RLBOBI
NE A FITA PARA O VERIFY"
 430 VERIFY A$ DATA Z(): VERIFY
"LOCAIS" DATA L$(): VERIFY "DESL
OC." DATA M(): VERIFY "OBJETOS"
DATA O$(): VERIFY "POS.OBJ." DAT
A P(): VERIFY "VERBOS" DATA V$()
: VERIFY "COMPL." DATA C$(): VER
IFY "MENSAGENS" DATA M$(): VERIF
Y "VOUPARA" DATA D(): VERIFY "DE
S.OBJ." DATA F(): VERIFY "AUM.SC
ORE" DATA A(): VERIFY "PODE?" DA
TA U(): VERIFY "OBJ.NECES." DATA
 B(): VERIFY "REP.VER." DATA K()
 440 CLS : PRINT AT 10,0; FLASH
1;"OK OK OK OK OK OK OK OK OK OK
 OK";AT 20,0; FLASH 0;"BOM JOGO
COM...""ADVENTURE!"""
```

LOCAL 5

Descrição: "Estamos num porão. Veja uma saída ao Sul e uma caixa no chão."

NORTE	SUL	LESTE	OESTE	SUBA	DESÇA
0	1	0	0	2	0

Quantos objetos? – 7

OBJETO	DESCRIÇÃO	LOCAL
1	CHAVE	0
2	ISQUEIRO	1
3	MAPA	1
4	RELOGIO	2
5	MICRO	3
6	MARTELO	5
7	SERROTE	3

Número de verbos – 6

Para exemplificar a entrada dos verbos, complementos e outras informações sobre o adventure, descreveremos apenas o primeiro exemplo. Portanto, valerão para todos os exemplos, após o primeiro, as mesmas perguntas.

Verbo 1 – Abra
Complemento – Caixa
Resposta do micro – Veja o que achamos!
Transportado para – 0
Deslocamento de objetos – 1
Onde pode executar – 5
Objeto que deve ter – 0
Pode-se executar mais de uma vez – N

Verbo 2
Aperte
Botão
Ok (tecle apenas Enter)
-2, 0, 3, 0, N

Verbo 3
Abra
Porta
Ok
3, 0, 2, 1, N

Verbo 4
Examine
Mapa
É um mapa de Hong Kong!
0, 0, -1, 3, S

Verbo 5
Acenda
Tocha
Assim é melhor
5, 0, 4, 2, S

Verbo 6
Apague
Tocha
Ok
4, 0, 5, 0, S

Local de início: -1

VARIÁVEIS UTILIZADAS PELOS PROGRAMAS

L$ (matriz alfanumérica) – Contém as descrições dos vários locais.

M (matriz numérica) – Contém os vetores de deslocamento nas seis direções (N, S, L, O, Suba e Desça).

O$ (matriz alfanumérica) – Contém os nomes dos objetos, sem artigo.

P (matriz numérica) – Contém a posição de cada objeto. Há posições particulares!

-1 indica que o objeto está com o jogador

0 indica que o objeto está no local zero (guarda-objetos)

V$ (matriz alfanumérica) – Contém o imperativo dos verbos do jogo.

C$ (matriz alfanumérica) – Contém o complemento de cada verbo.

M$ (matriz alfanumérica) – Contém a mensagem a ser dada ao jogador quando for executado o verbo correspondente.

D (matriz numérica) – Contém o eventual deslocamento do jogador relativo ao verbo correspondente.

F(matriz numérica) – Contém o eventual deslocamento de objetos (aparecimento ou desaparecimento) relativo à execução do verbo correspondente.

A (matriz numérica) – Contém os pontos relativos à execução do verbo correspondente.

U (matriz numérica) – Contém o flag que indica onde o verbo correspondente pode ser executado (0 = em qualquer lugar).

B (matriz numérica) – Contém o número do objeto necessário à execução do verbo correspondente.

K (matriz numérica) – Contém o flag que indica se determinado verbo pode ser executado mais de uma vez.

Z (matriz numérica) – Contém os seguintes flags:

Z(1) = posição inicial
Z(2) = flag que indica se o adventure é por pontos
Z(3) = escore a ser alcançado
Z(4) = número de locais
Z(5) = número de objetos
Z(6) = número de verbos

J – Número de objetos em poder do jogador
A$ – Primeira palavra introduzida
B$ – Última palavra introduzida
PONMAX – Escore a ser alcançado
SPON – Indica se o adventure é por pontos
SCORE – Escore do jogador
POS – Posição do jogador (ver texto)
H$ – Ordem do jogador

Paolo Fabrizio Pugno cursa atualmente o primeiro ano da Faculdade de Engenharia Industrial (SP) e possui um ZX Spectrum, com o qual desenvolve programas.

Metodologia de criação

A elaboração de um adventure está associada a uma série de fatores não relacionados à programação propriamente dita, que podem determinar o sucesso ou fracasso de um empreendimento. Tais fatores são geralmente fruto da experiência pessoal de cada indivíduo, porém eles guardam certas semelhanças que os tornam codificáveis em um conjunto de procedimentos, o qual denominamos de Metodologia de Desenvolvimento de Projeto.

Em essência, a metodologia nada mais é do que a racionalização dos processos envolvidos, num trabalho, para que os objetivos sejam atingidos no menor espaço de tempo possível e os resultados sejam os mais satisfatórios. O mais importante para a compreensão da metodologia é a observação e a análise das atividades que compõem um projeto.

Na elaboração de um adventure comercial, o intervalo que separa a idéia inicial do produto à venda nas lojas está repleto de atividades que podem ter um relacionamento, umas com as outras, bastante complexo. Para se ter um jogo pronto é preciso: pesquisar o assunto tema; definir uma estrutura lógica; avaliar o mercado potencial; determinar a funcionalidade do jogo; estabelecer os objetivos da narrativa; escrever as rotinas funcionais; criar os desenhos e elementos gráficos; criar os textos explicativos; montar o jogo; e produzir testes de utilização setorizada e plena.

Cada uma dessas atividades encerra um outro tanto de ações ou trabalhos que concorrem para a produção global do produto. A função da metodologia é a manutenção de uma ordem prioritária onde a importância de cada ação é quantificada em relação ao todo, produzindo assim uma espécie de roteiro de execução do projeto. Uma vez definidas as etapas e fases de execução e a sua importância na ordem organizacional do projeto, a criação do jogo assume características quase automáticas.

FASES DO PROJETO DE UM ADVENTURE

A elaboração de um determinado adventure pode ser dividida em diversas fases, e mesmo esta divisão não é de uma rigidez absoluta. Ela vai depender bastante de um trabalho consciencioso do autor e das características que se quer imprimir ao jogo.

A idéia inicial – o ponto de partida da maioria dos produtos comerciais é a definição de um espectro mais ou menos homogêneo de objetivos. Isso não exclui a idéia ocasional surgida de improviso, pelo contrário, permite uma espécie de *educação* dos processos de raciocínio. Vejamos uma breve comparação.

Nós poderíamos acordar um dia e dizer "tive uma idéia para fazer um jogo sobre o mercado de ações", ou então partimos de uma definição abrangente do tipo "vou fazer um jogo que: tenha uma dinâmica de valorização do raciocínio empresarial; que utilize valores monetários e conceitos financeiros; que tenha momentos de disputa; que permita situações ambíguas ou que possua componentes emotivos fortes; que crie um relacionamento operacional próximo a uma realidade atual".

Nesses dois casos a idéia seria fazer um jogo chamado "BOLSA DE VALORES", por exemplo, porém a forma como se chegou a esta conclusão foi diferente. No primeiro caso, a inspiração dependeu de fatores que fogem ao controle do autor; e no segundo a idéia foi o resultado de uma planificação muito mais racional.

A inspiração espontânea não permite uma avaliação mais profunda do mercdo consumidor, pois ela freqüentemente está dissociada da definição dos elementos de empatia do tema. Em contrapartida, o grau de identificação do tema com as aspirações particulares do autor pode ser tão alto que provoque avaliações errôneas ou emocionais. Isso não implica em que o processo ideal seja caracterizado por uma frieza de relacionamento, porém há que se dosar com cuidado estes valores empíricos.

Nós tenderíamos a ficar num jogo de adivinhações se quiséssemos "ter uma idéia instantânea, e boa, para um jogo" a todo instante. No caso da racionalização, nós poderíamos ainda concluir pelo acréscimo de outros elementos de atualidade. No exemplo, hoje um jogo sobre Bolsa de Valores ou Mercado Financeiro seria incompleto se não previsse coisas do tipo quebra de bancos e corretoras, maxi e minidesvalorizações, pacotes econômicos, choques heterodoxos, queda de ministros, rombo nas estatais etc.. Afinal estamos no Brasil, não é mesmo?

A pesquisa temática – esta é uma das mais importantes fases da criação de um jogo e não obstante a que nunca é executada quando o autor tem uma atitude temperamental em relação ao seu trabalho. Isso decorre de uma imaturidade 'projectual' onde o jogo não é visto como um produto sério ou que deva cumprir um determinado objetivo.

A necessidade de se criar uma prática de pesquisa que anteceda a todas as outras etapas do jogo é de extrema importância. Esta pesquisa é fundamental na principal avaliação a ser feita sobre um jogo comercial: se ele tem condições de despertar o interesse das pessoas para a compra.

A motivação para o consumo de jogos é bastante simples e não guarda nenhum componente inovador. Compra-se um jogo porque há um interesse naquela forma de entretenimento. O consumidor procura um tema que corresponda às suas expectativas e anseios e não por causa da capa, ou da fita, ou do disco, ou por ele ser desta ou daquela marca, ou deste ou daquele autor. Compra-se um jogo porque quer-se jogar e neste caso a decisão de compra está fundamentada no tema do jogo.

Todos os elementos citados acima são componentes que reforçam, sobremaneira, o poder de atração de um determinado produto, porém é a temática do jogo quem define se ele é interessante ou não.

A pesquisa propriamente dita deve ser estruturada para encampar os diversos aspectos da elaboração de um projeto. Tomemos como exemplo um adventure onde pretenderíamos dar vazão a toda sorte de seres fantasiosos e situações emocionantes a la *Spielberg;* um *Indiana Jones* tropical.

A planificação da pesquisa deverá resultar algo parecido com:

- Analisar programas semelhantes (fazer uma lista de programas que corresponderiam mais ou menos ao que se propõe). Isto é muito importante para que se tenha uma idéia de como certos problemas conceituais foram resolvidos por outros autores. Não se trata obviamente de plagiar algum trabalho, mas puramente ter algum tipo de contato com os problemas;
- Coletar informações e desenhos de armas, cenários, paisagens, veículos etc.. Mesmo que o projeto se reporte a um adventure de texto, a ilustração é importante para situar o autor quando for criar as frases descritoras;
- Ler alguns livros de aventuras que tenham parentesco com o tema (geralmente uma tarde de visitas a uma livraria é suficiente para coletar uma série de exemplares; e
- Coletar e avaliar todo material que porventura possa ser útil de alguma forma, tais como: coletânea de rotinas, seqüências de movimentos, vocabulário específico, gírias, coisas ou fatos reais etc..

A perfeita planificação de uma pesquisa temática pode resultar numa fatídica conclusão: não vale a pena fazer um determinado jogo. Mas não devemos desanimar, pois um bom adventure é composto principalmente por obstinação.

Os leitores podem participar desta coluna enviando idéias, opiniões, críticas etc.. Para tanto, basta destacar no envelope "coluna ADVENTURES".

Renato Degiovani

Reconhecendo Palavras

Nesta altura do campeonato o leitor interessado em adventures pode estar impaciente para chegar ao estágio da criação de um jogo. É preciso ter um pouco mais de paciência pois chegaremos lá. Quanto mais se conhece a estrutura operacional destes sistemas, mais requisitos se tem para a compreensão dos jogos propriamente ditos.

Existem certas coisas na programação de adventures que são complexas e, aparentemente, desprovidas de sentido. Porém elas são muito importantes dentro do processo de costura de um jogo (afinal todos nós estamos interessados em adventures e não em idiotaventures ou panacajogos de palavras).

Fiz esta observação pois nesta edição vamos tratar de um tema um tanto quanto chato: o reconhecimento de palavras. Creio que não preciso alertar o leitor quanto ao que sucede com um adventure que mal consegue entender o que o jogador está tentando fazer.

Inicialmente temos que definir o tipo de comparação que o nosso jogo adotará. O sistema mais primitivo é aquele que define uma matriz, por exemplo **PALAVRA$(MÁXIMO)**, e que executa a busca direta, como estamos acostumados a ver em Basic:

```
FOR I=1 TO MAXIMO
IF FRASE$=PALAVRA$(I) THEN...
NEXT I
```

saindo deste looping poderíamos chegar numa instrução do tipo

```
ON I GOTO...
```

ou algo semelhante.

O que importa aqui são duas coisas distintas, ou melhor, dois conceitos. O primeiro é que o sistema "esqueceu" a palavra original mas passou a manipular um pequeno código (I) que identificará esta mesma palavra em outra circunstância.

O segundo conceito é que o sistema exigiu, para o reconhecimento, não só a palavra completa mas também a sua grafia correta. Por exemplo: se eu desejasse que o jogo reconhecesse MACA e MAÇÃ como uma determinada fruta, o sistema geraria dois códigos diferentes. Mais à frente veremos como sair desta embrulhada.

Voltando à forma de reconhecimento, poderíamos adotar uma estrutura de abreviações, onde as definições se restringissem, por exemplo, às três primeiras letras de cada palavra. Por exemplo:

EXA para **EXAMINE**
NOR para **NORTE**
DEI para **DEIXE**

Desta forma, o programa isola cada palavra da frase; obtém a forma abreviada e... craul Este processo é uma simplificação do anterior e o ganho de desempenho obtido com ele não é lá essas coisas. Fica muito fácil, neste caso, para o jogador confundir o sistema do jogo e se confundir também.

Podemos criar sistemas alternativos das mais variadas formas. Adotei, para as primeiras versões do jogo AMAZÔNIA, um sistema que gerava um código por palavra, obtido com a primeira letra e mais a somatória dos valores das outras letras. Este sistema se mostrou extremamente rápido porém não permitia um conjunto muito extenso de palavras.

Outro problema apresentado foi o de que certas palavras produziam o mesmo código e isto confundia o jogador.

Uma das formas mais interessantes, usadas atualmente, é aquela que identifica a palavra independentemente do número de letras fornecidas pelo jogador. O processo é o seguinte: dada as palavras

FRASE$="EXAM"** e **PALAVRA$(5)="EXAMINE"

a comparação é feita letra por letra e enquanto houver paridade de **FRASE$** para **PALAVRA$** o processo se repete. Quando o final de **FRASE$** for encontrado e havendo paridade para todas as letras, então o sistema assume que **PALAVRA$(5)** era a palavra com a qual o jogador estava tentando se expressar.

A beleza deste processo é que ele simula uma situação real. Imagine o que acontecerá com o sistema se forem fornecidas apenas as letras **EX** e o banco de palavras possuir **EXAMINE, EXECUTE, EXTERMINE**, etc. Poderíamos criar um jogo só de adivinhações.

Este exemplo nos leva a uma conclusão interessante: em certos casos, para sermos compreendidos pelo adventure somos obrigados a nos expressar coerentemente. Pense o que um amigo seu compreenderia se você chegasse até ele e dissesse: "EX"...

Não se deve esperar que o computador entenda certas coisas que nós humanos temos dificuldades em assimilar. Jogar adventures por iniciais ou abreviações pode ser muito rápido, mas certamente foge da essência deste tipo de jogo, que é simular o real.

Com exceção dos movimentos NORTE/SUL/LESTE/OESTE, é preferível que o sistema manipule a palavra toda ou pelo menos uma parte dela que faça sentido.

Quanto ao código gerado a partir do reconhecimento, temos alguns problemas pela frente. Já deu para perceber que o indexador de dados não é muito útil quando tratamos com sinônimos. A saída, nestes casos, é relacionar cada palavra com um código independente. Por exemplo: **EXAMINE, OLHE, VEJA, ESPIE**, etc, receberiam um código qualquer como primeiro caracter da palavra.

```
PALAVRA$(5)=CHR$(7)+PALAVRA$(5)
```

Desta forma, quando a palavra for identificada obtém-se código correspondente. O resultado é que uma frase extensa pode ser reduzida a dois ou três códigos. Mas o que identificar? E como?

A ORGANIZAÇÃO DOS ELEMENTOS

Em primeiro lugar vamos obedecer a fórmulas atuais de construção de frases dada por verbo + objeto. Faremos isto não tanto por tradição, mas porque o adventure é um jogo onde o jogador irá determinar o que será feito e portanto é esta a forma ocidental de se expressar: a ação seguida do objeto.

Isto introduz um fato novo ao nosso sistema: uma vez que vamos trabalhar com dois tipos de elementos e que eles não se confundirão em termos de definição, então devemos trabalhar com duas listas separadas de palavras. É mais inteligente, mais objetivo, mais simples e mais higiênico.

Eu sei que o primeiro elemento de uma frase será sempre o verbo e portanto a tabela de verbos será a primeira a ser consultada (incluiremos aqui as direções). Os outros elementos da frase são objetos e estarão definidos numa outra tabela: a tabela de objetos.

Peço desculpas aos imortais da academia por ser obrigado a considerar a palavra NORTE como um verbo, mas compreendam, é para o bem dos adventures. Ademais, vamos tratar NORTE não como um verbo propriamente dito, mas como uma abreviação de "VÁ PARA O NORTE".

Vamos partir de um exemplo clássico: PEGUE A CANETA. Nesta frase o que interessa realmente é o verbo PEGUE e o objeto CANETA. Podemos desprezar certos elementos, tais como artigos, que o sentido não se perderá.

Supondo que tenham códigos correspondentes 7 e 9 e após ter passado pelo nosso *turbo-analisador-gramatical-sintático*, o que sobrará da frase são apenas esses dois números, sendo que o primeiro corresponde ao verbo e o segundo ao objeto. Para evitarmos armadilhas futuras, proponho a adoção de uma diferenciação para os códigos dos verbos, por exemplo o sinal "-". Desta forma teríamos -7 e 9.

Para quem está programando em Assembler, dou uma dica: use o bit 7, do byte do código, pois 127 verbos e igual quantidade de objetos é um volume respeitável de elementos e poucos adventures chegam a isso.

Esta estrutura verbo + objeto está presente na esmagadora maioria dos adventures e data dos primórdios da criação desses jogos. O resultado de tal sistema é a criação de adventures muito pobres em termos de possibilidades de interações com o jogador. Quem pretende levar a sério a criação de adventures não pode ficar preso a estruturas tão simples quanto essa.

Há que se direcionar esforços de programação no sentido de buscar avanços nesta área. O que vou propor, daqui para frente, são conceitos que ainda não vi integralmente aplicados em adventures comerciais, até porque a programação necessária não é tarefa para tardes chuvosas de domingo.

Voltando ao nosso exemplo, vamos acrescentar mais um objeto à frase: **PEGUE A TAMPA E A CANETA.** Codificando temos: -7, 5 e 9 (**TAMPA**=5). Novamente, o que interessa aqui são as palavras PEGUE, TAMPA e CANETA não é mesmo? ERRADOOO....

O que acontece se eu escrever **PEGUE A TAMPA DA CANETA** (códigos -7, 5 e 9)? As frases codificadas são exatamente idênticas, mas a intenção é completamente diferente. Acontece que na primeira frase há mais um verbo, que ficou oculto pela partícula aditiva "e". O certo seria desmembrar a ação em duas frases: **PEGUE A TAMPA** e **PEGUE A CANETA.**

Para visualizar melhor, veja o diagrama abaixo:

PEGUE A TAMPA (E) A CANETA = -7 5 -7 9

PEGUE A TAMPA (DA) CANETA = -7 5 9

PEGUE A TAMPA DA CANETA = -7 5

Para complicar mais um pouco, podemos considerar **TAMPA DA CANETA** como um objeto para diferenciar de **TAMPA DO BULE, TAMPA DA CHALEIRA** etc. Isto, porém, é uma outra estória. Vamos nos concentrar apenas nos objetos formados por uma única palavra.

Temos que reconhecer a partícula "e" e assumir que a sua ocorrência na frase implica numa duplicação do verbo, a menos que:

PEGUE A TAMPA (E) COLOQUE (A) NA CANETA = -7 5 -8 5 9

-8

Neste caso temos realmente duas ações distintas: **PEGUE A TAMPA** e **COLOQUE A TAMPA NA CANETA.**

Condensamos tudo numa única frase e ainda usamos um pronome para referenciar o objeto **TAMPA**, mas a partícula "e" precisou ser ignorada. Isto ocorrerá sempre que o verbo não estiver oculto.

Complicando um pouco mais podemos ter:

PEGUE A TAMPA (E) A CANETA (E) COLOQUE (AS) NA MALA = -7 5 -7 9 -8 5 3 -8 9 3

3

Decompondo temos:

PEGUE A TAMPA
PEGUE A CANETA
COLOQUE A TAMPA NA MALA
COLOQUE A CANETA NA MALA

Aqui vai um brinde especial para o leitor esquematizar e codificar:

PEGUE A CANETA, COLOQUE-LHE A TAMPA E GUARDE-A NA MALA

CONCLUSÃO

Como vimos nesta edição, a análise de uma frase pode se tornar algo perigoso e cheio de armadilhas, mas com um pouco de bom senso podemos equacionar o problema e resolvê-lo. Apesar desta complexidade intrínseca, a coisa vai piorar muito mais quando formos distinguir também os advérbios, adjetivos e principalmente os personagens que podem fazer parte do jogo.

Variáveis e Registradores

Muito bem, já estamos aptos a reconhecer as palavras numa frase e transformá-las em códigos. Agora então vem a pergunta fatal: o que fazer com esses códigos?

Deste ponto em diante, a programação do sistema funcional do jogo irá variar muito de autor para autor, pois não existe uma estrutura que seja considerada a melhor. Vou ilustrar portanto alguns pontos que podem, ou não, estar implementados nos editores de adventures mais conhecidos.

Uma vez que estabelecemos a codificação para os verbos e palavras, podemos dizer que os códigos —7 e 5 correspondem à frase PEGUE A TAMPA, por exemplo. A primeira providência é de que para se pegar a TAMPA, no jogo, será necessário obedecer a uma lista de verificações, como se segue:

1 — Existe o objeto TAMPA no jogo?
2 — A TAMPA está no mesmo local que o jogador?
3 — A TAMPA pode ser pega pelo jogador?
4 — O jogador pode carregar mais esse objeto?
5 — Existe alguma implicação especial no ato de PEGAR a TAMPA?
6 — Ok: pega a TAMPA.

Tudo isto é feito através do uso de variáveis ou melhor, de registradores. Supondo que o registrador POS guarda a posição do jogador, então POS=11 significa que o jogador está na posição 11. Supondo também que dispomos de uma série de registradores chamados R1, R2, R3 etc, e que o registrador R3 está associado ao objeto TAMPA, então se R3=0 o objeto TAMPA não existe; se R3=11 a TAMPA também está na posição 11 e se R3=255 então ela está sendo carregada pelo jogador. Fácil não?

Veja agora como fica a lista de verificações:

1 — IF R3=0 THEN...
2 — IF R3<>POS THEN...
3 — IF R4<>0 THEN...
4 — IF QTO<5 THEN...
5 — IF R5<>0 THEN...
6 — R3=255:QTO=QOT+1

QTO — quantidade de objetos carregados pelo jogador.
R4 — se igual a zero indica que a TAMPA é um objeto normal. Uma CASA, por exemplo, também seria definida como objeto porém não poderia ser pega pelo jogador.
R5 — se igual a zero indica que não há nada especial com este objeto. Se fosse uma tampa de garrafa, por exemplo, poderíamos programar o jogo para que o líquido contido na mesma fosse derramado assim que o jogador pegasse a TAMPA..

Note como tudo não passa de convenções para variáveis. O único segredo aqui é estabelecer claramente o que cada variável, ou registrador, representa.

Num jogo existirão diversos objetos que se assemelham à TAMPA, no que se refere à sua existência. Por exemplo: CAPA, MAPA, DISCO, JARRO, etc. Podemos automatizar a lista de verificações e transformá-la em sub-rotina. Para tanto basta substituir os registradores Rx por variáveis Xz. Antes da chamada é só fazer a conversão X1=R1, X2=R2, X3=R3 e assim por diante.

Uma vez que já reconhecemos o código do verbo e do objeto (supondo VRB=—7 e OBJ=5) podemos criar um looping onde:

ON VRB GOSUB...

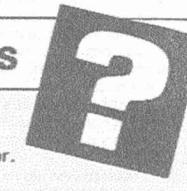
A ordem de processamento pode ser esta mesmo, ou seja, localizamos a rotina que executa as ações de PEGAR e nela processamos o objeto em questão. Para cada verbo será necessário definir um procedimento específico. Faça uma lista das verificações necessárias para diversos tipos de verbos, tais como: QUEBRAR, EXAMINAR, FAZER, SOLTAR, TROCAR, etc.

Para estabelecer procedimentos especiais, será necessário o domínio absoluto do que cada registrador representa para o enredo do jogo e então criar desvios condicionais na programação. Digamos que uma determinada ação exija a presença da TAMPA no mesmo local que o jogador. Resolvemos a parada com IF R3=POS THEN...

O volume de objetos e situações manipulados por um adventure pode chegar a tal ordem que a capacidade de memória do micro seja insuficiente. É comum avaliarmos a riqueza e o desempenho de um jogo pelo seu tamanho físico. Apesar disto, este não é um parâmetro cem por cento confiável mas nos ajuda bastante a formar uma idéia do que nos espera durante o desenrolar de uma partida.

Por outro lado, a tendência mais comum é ir criando apenas as situações que levam à saída do jogo e mais uma ou duas "armadilhas". Isto é muito pouco para qualquer adventure, mes-

Problema do mês



Supondo que:

255 = objeto sendo carregado pelo jogador.
POS = posição atual do jogador.
R3 = registrador da CANETA
R4 = registrador da BOLSA
R5 = registrador da SACOLA

como podemos resolver e como ficarão os registradores na seguinte situação:

PEGUE A CANETA
COLOQUE-A NA BOLSA
FECHE A BOLSA
COLOQUE-A NA SACOLA

Note que serão três objetos envolvidos numa mesma situação e que no final somente um ficará aparente.

mo escrito em Basic e para micros com 16 Kbytes de RAM. A beleza do adventure está justamente em tratarmos cada objeto e cada situação de forma autônoma.

Vou ilustrar um exemplo: será difícil termos uma situação onde a TAMPA de uma caneta tenha maior importância para o jogo. Isto é o que normalmente pensamos, então se um comando simples do tipo COLOQUE A TAMPA NA CANETA resulta em uma resposta do tipo ISTO NÃO É POSSÍVEL já obtemos aí a confirmação de que a TAMPA é um objeto dispensável.

Podemos ter, apesar de tudo, situações absurdas onde seja necessário o uso da TAMPA. Por exemplo, colocando a TAMPA da caneta na orelha o jogador assustará um morcego que o está atacando dentro da caverna. Esta é uma forma equivocada de fazer adventures pois uma TAMPA de caneta na orelha não assusta nem mesmo as moscas do cérebro de certos autores. Pelo menos é isto o que penso. Se quisermos fazer algo realmente interessante temos que criar situações curiosas e não estúpidas.

O mais correto, no que diz respeito ao comportamento dos objetos num adventure, é estabelecer e definir todas as situações normais que são passíveis de ocorrer. Por exemplo, sempre que um adventure tiver objetos como TAMPA e CINZEIRO deverá ser possível colocar a TAMPA dentro do CINZEIRO. Isto é o normal, a menos que seja uma TAMPA de caixa d'água. É isto que é fundamentalmente um adventure.

Proponho um tema para reflexão: os adventures são normalmente estruturas fechadas dentro de uma construção lógica estabelecida pelo autor do jogo. O imponderável fica conseqüentemente de fora deste tipo de jogo, a menos que...

Podemos dividir os verbos de um adventure em duas classes: primários e secundários. Os verbos primários são aqueles que necessitam ser definidos pelo programador, como por exemplo, PEGAR, CORTAR, AMARRAR, EXAMINAR, SOLTAR, etc. Os verbos secundários são aqueles que utilizam os conceitos embutidos nos verbos primários. Temos como exemplo o verbo FAZER.

Suponha que num dado adventure precisamos construir uma BALSA e o computador não sabe como fazê-la. Ao comando FAÇA UMA BALSA o jogo responde sempre ISTO EU NÃO SEI FAZER. Pense, só por um momento, como seria se pudéssemos executar a seguinte seqüência de comandos:

APRENDA A FAZER UMA BALSA
PEGUE O MACHADO
CORTE *ALGUNS* TRONCOS
PEGUE O FACÃO E A CORDA
CORTE-A EM PEDAÇOS
AMARRE OS PEDAÇOS NO TRONCO (isto não dará certo)
AMARRE OS TRONCOS *COM* OS PEDAÇOS
OK.

Num sistema desses podemos ver duas coisas importantes. Em primeiro lugar, teríamos uma maior integração do adventure com a inteligência artificial e a capacidade de aprendizado do jogo poderia abrir fronteiras inimagináveis.

Outro ponto é que seria necessário uma capacidade de processamento razoável para se testar o mínimo de consistência da ação que será ensinada ao jogo, caso contrário poderíamos criar verdadeiros monstros. Por exemplo:

APRENDA A FAZER UMA BALSA
PISQUE O OLHO
OK.

Isto seria desonesto, a menos que o jogador atingisse o estágio GRAN MESTRE MAGO DOS MAGOS e então ele pudesse fazer uma BALSA com um "piscar de olhos".

Cartas

A coluna adventures, a partir desta edição, ganha mais uma página e um espaço dedicado às cartas dos leitores. Nesta edição publico uma dúvida técnica e duas soluções para o problema proposto na edição 76 de Micro Sistemas. Com a palavra os autores:

• Ao invés de se usar uma matriz numérica do tipo (X,4), onde X é o número de locais do jogo, para se controlar os movimentos do jogador nas quatro direções possíveis (N,S,L,O), pode-se usar uma matriz do tipo (X,8), onde passariam a existir então oito direções possíveis: N, S, L, O, Suba, Desça, Entre, Saia.

Obviamente serão necessárias alterações na sub-rotina de movimentação do jogador, publicada na revista, para que a mesma passe a operar com as quatro novas direções do jogo.

Fernando Paiola — São José dos Campos

• Movimentos de subir e descer podem aparecer em muitos locais de um adventure. Neste caso, seria mais prático tratá-los como direções, do mesmo modo que Norte, Sul, Leste e Oeste. Assim, a matriz de limites teria seis elementos na segunda dimensão. Para seguir o exemplo do Renato, ficaria DIM LIM (50,6). O programa também deverá reconhecer tanto VÁ PARA CIMA como SUBA.

Já ENTRAR e SAIR prendem-se à descrição do local. Por exemplo, estamos num longo corredor. Há uma porta aberta a oeste. Seria lógico entrar pela porta tanto com Oeste como com ENTRE. Isto pode ser feito com pouco gasto de memória se o programa do jogo zerasse as variáveis EN e SA (para ENTRA e SAI) antes de receber a frase do jogador, enquanto que os trechos responsáveis pela descrição dos locais atribuiriam novos valores a essas variáveis, caso fosse necessário. É claro que os valores atribuídos às variáveis apontariam para onde o jogador entraria ou sairia. Caso o seu adventure tenha poucas subidas e descidas, estes movimentos poderiam ser feitos de forma semelhante à sugerida para ENTRE e SAIA.

Outra forma de codificar esses movimentos seria por ações compostas (verbo + objeto) como ENTRE NA CAVERNA ou SUBA A ESCADA.

Fernando Silva Lozano — Rio de Janeiro

• ... Já criei alguns adventures usando o seu EDITOR porém não consigo ajustar o relógio para uma cadência mais adequada. Como devo proceder para obter resultados melhores?

Paulo C. Guimarães — Porto Alegre

O ciclo do relógio varia muito de computador para computador e ainda é preciso considerar o que já foi programado principalmente na FUNÇÃO 4, que é o looping de tempo real do jogo.

A melhor forma é ajustar visualmente, construindo um pequeno relógio digital na tela. Siga as seguintes etapas:

1 — edite a mensagem 3 (ou qualquer outra) e coloque o sinal ">" no local onde você deseja ter o mostrador. É importante mudar este sinal do seu lugar padrão.

2 — edite a FUNÇÃO 4 e insira as seguintes instruções:

```
MSG    5
REGN   12
CHRS   58
REGN   11
CHRS   32
CHRS   32
REF
```

3 — inicialize o registrador 14 com o valor que você deseja testar e execute o teste do jogo desde o início (comando TJ do EDITOR).

Altere o valor do registrador 14 até que o relógio fique com a cadência desejada.

Os leitores podem participar desta coluna enviando idéias, opiniões, críticas, dúvidas, etc. Destaque na sua carta Coluna ADVENTURES e remeta para a ATI Editora.

GAC: o mestre dos adventures do ZX-Spectrum

Luiz Fernandes de Moraes

Uma das coisas que eu nunca consegui entender bem foi o descaso dos programadores do ZX-Spectrum com relação a alguns programas incríveis que essa máquina e os seus compatíveis nacipnais possuem. Não estou falando de jogos. Estou falando de alguns utilitários realmente surpreendentes, largamente difundidos no mercado, e que possibilitam a criação de programas de excelente nível.

Dentre esses utilitários, um dos melhores é o GAC, The Graphic Adventure Creator, da empresa inglesa Incentive Software Ltd. Como explicar que após dois anos no mercado, somente um adventure nacional foi criado com o auxílio deste poderoso editor?

Lembro-me que quando escrevi a **Lenda da Gávea**, eu acreditava que seria apenas o primeiro autor nacional a me utilizar do GAC. No meu modo de ver as coisas, achava que em seguida viriam dezenas de adventures feitos por outros autores.

Infelizmente eu estava enganado e, a despeito da ótima vendagem da "Lenda", comecei a pensar que os nossos programadores não estavam interessados em adventures. Mas eu estava novamente enganado (será que não dou uma dentro?). O número de pessoas que escreve para MS comprova o grande interesse e mostra que a principal lacuna é a falta de informações sobre o GAC.

Mas não há de ser nada! Talvez ainda haja tempo para contornar o problema.

A FILOSOFIA DE TRABALHO DO GAC

Uma das melhores coisas do GAC é a filosofia adotada na sua concepção, que permite criar adventures ilustrados com um mínimo de metodologia. Não existe uma ordem preestabelecida para trabalhar. Você pode fazer uma ilustração, criar um novo local, criar novos objetos, testar o jogo, criar um novo verbo, testar novamente, etc.

A parte do programa que edita o jogo e a parte que o executa convivem perfeitamente na memória. O único inconveniente é que isso acarreta uma certa perda de espaço para o desenvolvimento de jogos muito extensos. Mas esse inconveniente resulta num fato primordial: em qualquer momento podemos testar o nosso adventure. Com o teste on line, podemos ir aperfeiçoando o jogo e eliminando os bugs que possam ter permanecido.

Isso é muito importante para testarmos se não houve erro de lógica em alguma das instruções do jogo. Esse tipo de erro é mais freqüente quando o programador ainda não entendeu perfeitamente a forma como o GAC lida com cada tipo de evento do jogo.

Outro dado importante é que o editor de telas do GAC é ao mesmo tempo simples e eficiente. Ele possui a função UNDO e o recurso de ELASTIC, que permite traçarmos retas ou círculos acompanhando a formação do traço que está sendo feito na tela.

Cada tela do GAC é armazenada como seqüência de comandos em um buffer próprio. Esse buffer não é fixo e permite até 400 bytes por tela, isto é, uma média de 150 comandos gráficos do tipo DRAW, CIRCLE, INK, PAPER, BORDER, FILL, etc. Isso é suficiente para criarmos uma tela bastante detalhada, mas pode resultar em um jogo com poucas telas, já que quanto mais detalhes, maior o espaço ocupado.

Fazer desenhos cheios de pormenores requer um planejamento bem estruturado, para que possamos montar uma tela através de várias outras telas (ou cenas). Isso é relativamente fácil de conseguir, pois o GAC permite o "merge" de telas. Você pode definir a tela 10 como sendo a tela 1 + tela 3 + tela 5 + tela 200.

Como exemplo, o adventure **A Lenda da Gávea** possui 56 telas diferentes, sendo todas bastante detalhadas. Isso foi possível criando-se trinta pequenas telas que continham cenários, detalhes de árvores, detalhes de trilhas na floresta e várias máscaras de montagem. Cada tela era formada por, no mínimo, duas outras telas básicas com partes do cenário, acrescida dos detalhes individuais de cada cenário. Em suma, só foi gasto o espaço de 38 telas, mas o jogador pode visualizar 56.

O TRATAMENTO DE EVENTOS DO GAC

O GAC trabalha com três níveis de condições: as condições primárias (**high priority conditions**), as condições secundárias (**low priority conditions**) e as condições locais. É fundamental entender o que isto significa a nível de processamento para que possamos construir um adventure.

As condições primárias são testadas ANTES de ser liberado o cursor para um input do jogador. As condições secundárias e locais são testadas DEPOIS do jogador digitar um novo comando. Para entender melhor, imaginemos que você está jogando a "Lenda" e se encontra no primeiro local. Você digita SUL e se movimenta para outro local. Antes de se formar a nova tela e ANTES de surgir o cursor com a frase O QUE VOCÊ PRETENDE FAZER, são testadas as condições primárias.

Se você estivesse carregando uma cobra, seria picado por ela e, ao invés da tela do local, surgiria a tela da "morte". A sua pontuação seria mostrada e, em seguida o jogo recomeçaria.

É para isso que serve a condição primária: ela é testada no momento em que entramos em uma nova área e não interage com o jogador. Todas as funções automáticas, transparentes ao jogador (atualização de ponteiros, inicialização da partida, etc.) devem ser tratadas como condições primárias.

Mas digamos que não exista uma cobra. Você chega ao novo local e digita BEBA ÁGUA. Como não existe uma condição local (façamos de conta que não), a tabela de condições secundárias é testada. Se você tiver um cantil com água você poderá beber. Caso contrário você receberá a resposta AQUI NÃO TEM NADA DISSO.

Beber água é uma condição secundária porque o ato de beber resulta de uma ordem expressa do jogador e a ação deve ser executável em QUALQUER LOCAL, desde que existam condições para isso (um cantil cheio, por exemplo).

Mas existem ações que só podem ser realizadas em um determinado local. Suponhamos que você esteja na frente do casebre e diga ENTRE NO CASEBRE. Esta ação não pode ser feita em nenhum outro lugar. Sendo assim ela deve ser considerada como condição local.

O processamento do GAC ocorre da seguinte forma: ao chegar a um novo local ele testa as condições primárias, libera o cursor para uma ordem do jogador e, após a ordem, testa se existe uma condição local para aquela área do jogo. Caso não haja, testa a tabela de condições secundárias até encontrar uma condição válida ou até chegar ao fim da tabela.

É possível escrever um adventure usando apenas as tabelas de condições primárias e condições secundárias. Só que isso resultaria numa perda de tempo de processamento após a ordem do jogador, pois a tabela de condições secundárias ficaria muito extensa. Ficou claro?

VERBOS, ADVÉRBIOS, SUBSTANTIVOS E OBJETOS

As tabelas de verbos (**verbs**), advérbios (**adverbs**), substantivos (**nouns**) e objetos (**objects**) são construídas a partir de um índice numérico.

O verbo deve ser interpretado como item principal de uma frase-comando do jogador. Nesse caso as palavras NORTE, SUL, LESTE e OESTE, devem ser consideradas como verbos (se meu professor de português visse isso... benza Deus!).

Os verbos podem ter sinônimos. O sistema reconhece como sinônimos os verbos que possuem o mesmo índice numérico, exemplo: 1—ANDE, 1—CAMINHE, 1—PERCORRA.

Como todos possuem o mesmo índice (1), não existe diferença entre digitar ANDE e CAMINHE. Além disso você pode digitar palavras abreviadas. Nesse caso é bom tomar cuidado com a ordem das palavras na tabela. Digitar PE pode significar PEGUE ou PERCORRA, dependendo apenas de qual verbo foi definido primeiro.

Os advérbios atuam da mesma maneira, sendo que normalmente são pouco utilizados em um adventure. Exemplo: se você digitar PEGUE A NITROGLICERINA, você poderia explodir. Se você digitasse PEGUE GENTILMENTE A NITROGLICERINA, você pegaria e continuaria vivo para contar a história. O GAC permite isso.

Já substantivos e objetos são coisas intimamente relacionadas. Para cada objeto deve existir um substantivo de mesmo índice, exemplo: objeto 1 — uma faca; substantivo 1 — faca. Substantivos são palavras que o jogo pode reconhecer e geralmente são em maior número que os objetos (um objeto precisa de um substantivo, mas um substantivo não precisa de um objeto).

Para usarmos no jogo a frase ABRA A PORTA, não devemos criar o objeto PORTA. Basta criar o substantivo PORTA e testar a frase na tabela de condições locais (só se pode abrir portas onde existirem portas para serem abertas). Se for criado o objeto PORTA você terá nas mãos um grave problema: alguém digitará PEGUE A PORTA e o adventure não só aceitará, como sairá carregando a porta por aí.

Não esqueça de que objetos são coisas passíveis de serem pegas, largadas, comidas, bebidas, cheiradas, etc. Fica a seu critério considerar como objeto coisas como ÁGUA ou ÁRVORE (particularmente, acho isso arriscado).

Cada objeto deve ter definido o seu peso e o local onde ele se encontra no início do jogo. A questão do peso é importante pois é aqui que podemos limitar o jogador, impedindo que ele possa pegar tudo o que vê. Já a questão do local pode ser burlada para fazer aparecer os objetos durante o jogo. Basta iniciar todos eles no local 0 (zero) e fazer um SWAP (troca) de locais mediante a satisfação de uma ou mais condições.

Como exemplo, você poderia iniciar o jogo com uma garrafa no local 1 e cacos de garrafa no local 0. Se o jogador pegar a garrafa e quebrá-la, um SWAP colocará a garrafa no local 0 (sumirá com ela) e colocará os cacos de garrafa no local 1. O local 0 existe para arquivar objetos que serão futuramente transportados para o jogo.

AS MENSAGENS

O GAC permite até 255 mensagens de 255 caracteres cada uma (o número de verbos, advérbios, substantivos e objetos também é 255, sendo que o máximo de caracteres permitido é 35). Isso faz com que o adventure tenha uma incrível capacidade de conversação com o jogador.

O sistema utiliza as mensagens de 240 a 255 em seu processamento. Quando fizer um adventure não esqueça de incluí-las, mesmo que você só tenha mensagens de 1 a 50. Esses índices (240 em diante) são respostas automáticas do processamento. Como exemplo, sempre que o jogador digita uma frase não prevista (não existe verbo ou substantivo), o sistema imprime a mensagem 242. A mensagem 242 deve ser definida então como: DESCULPE, EU NÃO ENTENDI.

Se o jogador digitar BEBA ÁGUA e não houver água, a nível de processamento o sistema reconhece o comando mas, com ele não satisfaz nenhuma condição (secundária ou local), será impressa a mensagem 241. Sua definição correta seria algo como: LAMENTO... ISSO NÃO É POSSÍVEL.

Essas mensagens devem ser criadas juntamente com o jogo e a correlação com os índices é a seguinte:

240 — O que você pretende fazer?
241 — Isso não é possível!
242 — Não entendi o que você disse!
243 — Digite qualquer tecla para reiniciar.
244 — Tem certeza? (S/N).
245 — Nós já temos isso.
246 — Nós não temos isso.
247 — Eu não vejo nada disso.
248 — Não dá para pegar mais nada.
249 — Você marcou
250 — pontos em
251 — Está escuro. Não consigo ver nada.
252 — Não encontrei o que você quer.
253 — Eu também posso ver.
254 — OK.
255 — Jogadas.

Estas frases são apenas sugestões e podem ser modificadas desde que mantenham o mesmo sentido. É bom tomar um cuidado especial com as mensagens 249, 250 e 255. Essas três mensagens resultam em uma quarta que dá a pontuação do jogador. A mensagem seria algo como: VOCÊ MARCOU + (contador) + PONTOS EM.+ (contador) + JOGADAS.

Com relação às outras mensagens, vale a pena reservar as primeiras para a descrição de objetos. Isso é bom pois exige um pequeno truque de programação que permite subordinar todas as descrições a uma única condição. Esse mesmo truque é utilizado para pegar ou largar qualquer objeto, testando apenas uma condição para PEGAR e uma para LARGAR. Nós veremos isso adiante quando estivermos falando da programação.

CONTADORES E MARCADORES

O GAC possui 128 contadores (**counters**) e 255 marcadores (**markers**). Os contadores servem para armazenar valores numéricos (1 ou 2 bytes), e podem ser incrementados, decrementados, somados, etc. Já os marcadores podem ter apenas dois estados: 0 ou 1. Um marcador setado representa o estado de uma determinada coisa (objeto ou condição).

Exemplificando, você poderia definir o contador 10 como o contador da sede. A cada movimento do jogador ele seria incrementado em uma unidade (condição secundária) e caso estivéssemos no deserto ele seria duplicado (condição local). Quando chegasse a um número predeterminado (condição primária), o marcador 10 seria testado. Se estivesse ressetado, significaria que o jogador estaria sendo avisado pela primeira vez. O marcador 10 seria então setado, a mensagem ESTOU COM SEDE seria impressa e o contador 10 seria zerado.

Se o jogador pudesse beber água, o marcador 10 seria ressetado para reiniciar o ciclo (condição secundária). Quando o contador 10 chegasse novamente ao limite, estando o marcador 10 setado, saberíamos que o jogador já havia sido avisado da sede e ainda não tinha bebido água. Este seria o momento de matá-lo (condição primária).

Como você pode ver, contadores e marcadores são coisas muito importantes para um adventure. Com eles podemos saber o número de pontos do jogador, a quantidade de movimentos no jogo, se uma porta foi aberta ou fechada, se o Drácula está dentro ou fora do caixão, ou qualquer outra loucura que lhe der na veneta.

Mas é bom saber que o sistema utiliza três contadores e quatro marcadores. O contador 0 acumula os pontos do jogador e os contadores 126 e 127 atuam como um contador de 2 bytes, que serve para ir guardando o número de jogadas. O marcador 0 é setado após um comando de descrição de local. Se você não mudar de local, pode imprimir uma descrição diferente mediante o teste deste marcador.

Digamos que você chegou num local e recebeu a seguinte descrição: ESTAMOS NO BATEAU MOUCHE. Se você pedir uma nova descrição e testar o marcador 0, a condição de "marcador setado" poderia indicar uma nova descrição que, como o adventure é nacional, infelizmente seria: GLUB! GLUB! GLUB...

O marcador 1 indica a iluminação do local. Se estiver setado o local é iluminado. O marcador 2 indica que o jogador possui uma fonte de luz (lâmpada, vela, etc.). Basta estar setado. Se ambos os marcadores estiverem ressetados, o sistema não descreverá os locais e imprimirá sempre a mensagem 251 (a da escuridão).

O último marcador usado pelo sistema é o de número 3. Se esse marcador for setado, não será impressa a pontuação quando se encerrar a partida. Isso permite criar um novo sistema de contagem de pontos.

AS INSTRUÇÕES DO GAC

As condições são construídas usando-se as instruções de programação. Sendo assim, aqui vai a descrição de cada uma delas:

VERB x	— O verbo "x" foi digitado?
NOUN n	— O substantivo "n" foi digitado?
ADVE z	— O advérbio "z" foi digitado?
HERE x	— O objeto "x" está no mesmo local que o jogador?
CARR n	— O jogador está com o objeto "n"?
AVAI n	— O objeto "n" está disponível para uso (isto é, está aqui ou está com o jogador?)
n IN z	— O objeto "n" está no local "z"?
WEIG n	— Informa o peso do objeto "n".
SET? x	— O marcador "x" está setado?
RES? x	— O marcador "x" está ressetado?
CTR c	— Informa o valor do contador "c".
x EQU? c	— O valor do contador "c" é igual a "x"?
TURN	— Informa o número de jogadas.
ROOM	— Informa o número do local corrente.
AT z	— Estamos no local "z"?
AND	— Verdadeiro se todas as palavras foram digitadas. Ex: VERB 1 AND NOUN 1 A condição é satisfeita se ambas as palavras forem digitadas pelo jogador.
OR	— Verdadeiro se uma das palavras foi digitada.
XOR	— Verdadeiro se uma palavra foi digitada e a outra não.
NOT	— Verdadeiro se nenhuma das palavras foi digitada.
x<y	— X é menor que Y?
x>y	— X é maior que Y?
x=y	— X é igual a Y?
RAND z	— Gera um número randômico entre 0 e Z−1.
VBNO	— Informa o número do verbo em um comando.
NO1	— Informa o número do primeiro substantivo do comando.
NO2	— Informa o número do segundo substantivo do comando.
LOOK	— Descreve o local corrente.
DESC z	— Descreve o local de número "z".
PICT	— Ativa a tela de ilustração (modo normal).
TEXT	— Desativa a tela (para adventures só de texto).
GET y	— Pega o objeto número "y" (se ele estiver no local).
DROP y	— Solta o objeto número "y" (se ele estiver com o jogador).
x SWAP y	— Troca o objeto "x" (local 0) pelo "y".
OBJ z	— Descreve o objeto número "z".
LIST r	— Lista todos os objetos do local "r".
LIST WITH	— Lista todos os objetos em poder do jogador.
c TO r	— Move o objeto "c" para o local "r".
SET m	— Seta o marcador "m".
RESE m	— Resseta o marcador "m".
x CSET y	— Armazena o valor "x" no contador número "y".

INCR y	— Incrementa o contador "y" em uma unidade.
DECR y	— Decrementa o contador "y" em uma unidade.
GOTO z	— Vai para o local "z" e faz sua descrição.
STRE x	— Determina "x" como o peso máximo para carregar.
FIND y	— Procura o objeto número "y" e move o jogador para o local onde o objeto está. Cria o efeito de mágica no adventure.
SAVE	— Salva o jogo em fita ou disco.
LOAD	— Carrega o jogo em fita ou disco (uma partida pode ser interrompida, gravada e continuada em outro dia).
WAIT	— Espera um novo comando do jogador.
OKAY	— Imprime a mensagem 254 e aguarda um novo comando.
EXIT	— Encerra o jogo sem perguntar nada ao jogador.
QUIT	— Imprime a mensagem 244 e se a resposta do jogador for afirmativa, imprime os pontos e encerra o jogo.
MESS z	— Imprime a mensagem número "z".
PRIN x	— Imprime o número "x".
LF	— Imprime um line feed.
HOLD c	— Delay. Para interromper por 10 segundos, use HOLD 500.
x+y	— Retorna o valor da soma de dois números.
x−y	— Retorna o valor da subtração de x−y.

Cada comando inicia sempre com um IF e a sintaxe é bastante simples. Os testes ou comparações devem ficar sempre entre parênteses e a linha deve sempre encerrar com um END. Vejamos os seguintes exemplos de condições secundárias:

IF (VERB 7 AND NOUN 5) MESS 14 HOLD 200 EXIT END

Se o jogador digitar uma frase com o verbo 7 e o substantivo 5, o sistema imprime a mensagem 14, dá um tempo para o jogador ler a mensagem e encerra o jogo. Digamos que o verbo 7 seja BEIJE e o substantivo 5 seja OGRO. Se digitarmos BEIJE O OGRO, seria impressa uma mensagem do tipo: OGRO NÃO É CHEGADO! ELE TE DEU UMA PORRETADA E VOCÊ MORREU. HOLD 200 daria tempo suficiente para a leitura e o jogo encerraria.

IF (VERB 8 AND NOUN 7 AND CARR 7) DROP 7 7 TO 0 MESS 17 WAIT END

Se o jogador digitar o verbo 8 e o substantivo 7, e estiver com o objeto 7, o sistema solta o objeto, transfere-o para o local 0 (destrói o objeto), imprime a mensagem 17 e aguarda um novo comando. Digamos que o verbo 8 seja COMA, o substantivo 7 seja RATO e o objeto 7 seja UM RATO. Se digitarmos COMA O RATO, o rato sumiria e imprimiríamos uma mensagem específica (CHOMP! HUUM... QUE RATINHO ZZZENZAZIONAL!). Em seguida tornaria a aparecer o cursor, aguardando uma nova frase-comando do jogador.

Os dois exemplos são bastante simples, mas ilustram perfeitamente uma linha de comando. A estrutura é a mesma para qualquer uma das três tabelas de condições (primária, secundária e local), com a vantagem do GAC fazer um ajuste automático para a sintaxe correta no caso de erro do programador.

Falta falarmos do truque que permite pegar, soltar ou descrever qualquer objeto.

Para isso usamos a função NO1 que fornece o número do primeiro substantivo da frase. Supondo que o verbo 7 fosse PEGAR, o comando seria: IF (NO1 < 10 AND VERB 7) GET NO1 OKAY END

Com essa estrutura poderíamos pegar qualquer objeto de número menor que 10. Esse limite é dado pelo programador. Se o número de objetos for 20, ao invés de 10 coloque 21. Ou então coloque 16 e considere os 5 últimos objetos como "impossíveis" de pegar sem algum tipo de auxílio (objeto pesado demais ou grande demais).

Para soltar ou descrever objetos, suponhamos que os respectivos verbos possuam os números 8 e 9. As instruções seriam:

IF (NO1 < 10 AND VERB 8) DROP NO1 OKAY END
IF (NO1 < 10 AND VERB 9) MESS NO1 OKAY END

Fácil, não? Tenha apenas o cuidado de escrever suas instruções com um espaço entre cada palavra. Isso é axiomático.

EDITANDO AS TELAS

O editor de telas é bastante simples de operar. Anote as teclas de comando e faça várias experiências antes de desenhar para valer. As teclas são as seguintes:

SETAS	— Movem o cursor.
SETAS + CAPS	— Movem o cursor de 8 em 8 pixels.
L	— Posiciona a caneta para traçar uma linha. Uma segunda pressão em "L" levanta a caneta e encerra o traço (todas as funções de "riscar" sao desativadas com uma segunda pressão).
E	— Elipse ou círculo.
R	— Retângulo.
D	— Desenha um ponto (acende um pixel).
I	— Define o INK.
P	— Define o PAPER.
V	— FLASH.
B	— BRIGHT.
T	— Cor do BORDER e da área de texto.
G	— GRID (quadriculado).
F	— FILL de INK.
A	— FILL de PAPER.
S	— FILL sombreado (SHADOW).
M	— Merge de telas.
C	— Pisca o ponto onde se situa o cursor.
SETAS + SYMB	— Avança ou retrocede entre os passos de confecção da tela (buffer de comandos).
Z	— Salta para o início da tela.
DELETE	— Deleta o último comando.
CAPS + 9	— Deleta da posição corrente até o final do buffer de comandos de formação da tela.

As teclas CAPS + SPACE (BREAK) funcionam como uma espécie de "chave-geral" para o editor. Teclar BREAK interrompe o que quer que esteja sendo feito em qualquer etapa da confecção do adventure. Este será o seu PANIC-BUTTON. Insista no BREAK até recuperar o controle (seu e do jogo).

Está tudo aí, não faltou nada. Treine bastante! Se o seu entusiasmo coroar o meu esforço, e você resolver escrever um adventure com o GAC, aqui vão alguns conselhos: faça o mapa do jogo defina a solução da aventura e escreva tudo no papel. Defina todas as condições necessárias (também no papel) e comece o trabalho de criação das ilustrações.

Procure fazer com que após a confecção de todas as telas você ainda tenha uns 8 Kb para a programação. As rotinas de compactação do GAC permitem criar adventures gigantescos em apenas 8 Kb. Ao fim de tudo, selecione o BEGIN HERE através do menu principal, e informe em que local o adventure se inicia. Tire uma cópia do jogo (runnable) e envie de forma singela para este seu criado: afinal eu também gosto de adventures, certo?

PERIFÉRICOS

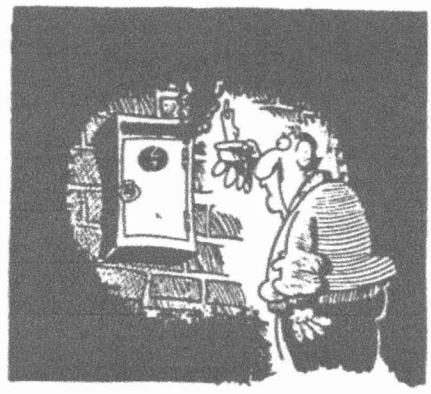
MICROHOBBY

Ano II - Nº 21 - 1985 Cr$ 6.000.

A REVOLUÇÃO DOS MICROS PESSOAIS COMEÇA AGORA

TK 90X
color computer

A ALTA TECNOLOGIA AO ALCANCE DOS USUÁRIOS BRASILEIROS



O Instan-TK e a amnésia em micros.

Tanios Hamzo

Não há nada mais medonho para um digitador do que a idéia de ver um programa quilométrico (aqueles com 15,9 k Bytes) evaporar, antes de tê-lo gravado em fita.

Alguns computadores mais recentes possuem um tipo de memória que jamais esquece, a não ser propositadamente. Os TKs e tantos outros computadores pessoais têm o que se chama de "memória volátil", ou seja: interrompendo-se a alimentação elétrica, mesmo por muito pouco tempo, a memória "esquece" para sempre seu conteúdo.

Para espantar este "fantasma da amnésia" de uma vez por todas, costuma-se usar um sistema conhecido por NO BREAK, que não interrompe a alimentação elétrica ao micro, mesmo que haja falta de energia elétrica na rede, ou, num caso comum (pelo menos para mim) onde o usuário, muito excitado durante um jogo, chute sem querer o fio da tomada.

Em sistemas de responsabilidade, a instalação de um NO BREAK é crucial. O TK, que pode ter aplicações diversas, pode ter diferentes motivos para conservar sua memória por prazos igualmente variados.

Segundo minha concepção, são três as categorias de NO BREAK:

1. INSTANTÂNEA: Quando se quer que a memória não se perca por alguns instantes, enquanto se troca de tomada; quando houver instalação elétrica intermitente (com bruscas variações de tensão ou em tomadas, extensões ou "benjamins" com mau contato, evidenciado quando se mexe neles para ligar um gravador, por exemplo) ou ainda quando aquele seu querido priminho vem te visitar e quer descobrir o que acontece se o computador for desligado quando você está usando. A memória será mantida por alguns minutos.

2. PORTÁTIL: Quando se quer que a memória seja mantida, enquanto o computador é transportado de sua casa até a casa de um colega; da escola até o clube ou do escritório até um cliente. A memória poderá ser mantida por algumas horas.

3. PERPÉTUA: Em sistemas realmente NO BREAK, que não podem parar: alarmes residenciais, comerciais ou industriais; controle de processos; em softhouses e em ocasiões em que o micro deva funcionar indefinidamente sem interrupções, independente da disponibilidade da rede elétrica. A memória poderá ser mantida por dias.

Abordarei o primeiro e o segundo caso, que me parecem mais interessantes para o leitor, porém ficarei atento à correspondência pedindo complementação do artigo.

Chamei o projeto de INSTAN-TK e seu esquema elétrico pode ser visto na figura 1.

Todos estes componentes podem ser facilmente encontrados em casas do ramo. Atenção para a polaridade de diodos e do L.E.D., bem como a sua sensibilidade ao calor excessivo. Verifique as ligações antes de ligar o aparelho.

FUNCIONAMENTO DO CIRCUITO: Apesar de ser um circuito aparentemente simples em sua configuração física, há diversos detalhes dinâmicos acumulados em cada componente. Farei uma explanação do funcionamento conjunto, descrevendo a(s) função(ões) de cada parte.

Ao se ligar o plugue da fonte no circuito, através de J1 e este ao computador através de J2, observa-se:

Com a fonte ligada, haverá passagem de corrente para o computador através de D1, que está diretamente polarizado, sempre que a fonte estiver ligada. No entanto, quando ela não for mais capaz de suprir energia, ficará o diodo D1 inversamente polarizado, não conduzindo energia das pilhas para outro lugar senão para o computador. Com isso, ficarão com carga destinada exclusivamente para o computador, garantindo seu funcionamento. Além disto, este diodo melhora a atuação da fonte, aumentando sua retificação, sua impedância e ainda reduzindo em cerca de 0,6 volts sua tensão de saída, o que pode contribuir para minimizar o aquecimento do regulador interno de tensão do computador.

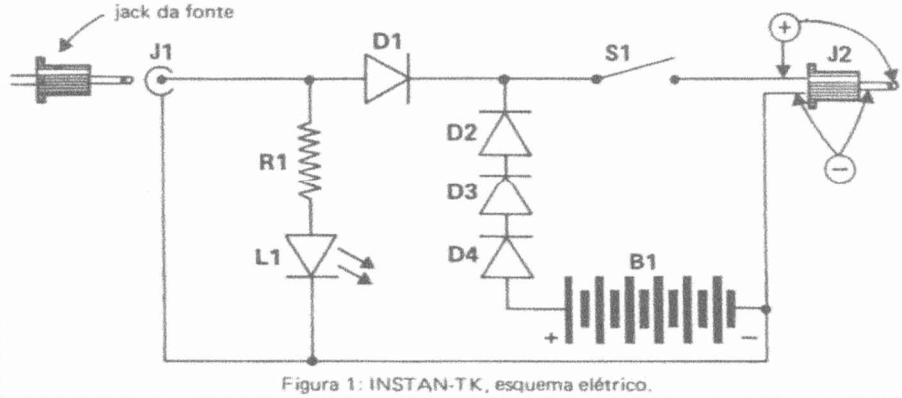


Figura 1: INSTAN-TK, esquema elétrico.

LISTA DE MATERIAL:

Quantidade	Componente/Tipo	Legenda
4	Diodo 1N4002	D1, D2, D3 e D4
1	L.E.D. de qualquer côr	L1
1	Interruptor de 1 pólo	S1
1	Jack fêmea	J1
1	Jack macho	J2
1	Resistor de 820 Ω, 1/4W	R1
6	Pilhas alcalinas de 1,5V	B1
6/1	Suporte(s) para pilhas	—
1	Caixa plástica	—

Diversos: fio, solda, placa de circuito impresso ou ponte de terminais, etc.

O L.E.D. (diodo emissor de luz) L1 serve para se monitorar a atividade da fonte, apagando, quando ela não estiver operando e atestando seu funcionamento, quando estiver ligada. O resistor R1 limita o consumo do L.E.D. em cerca de 0,01 ampères.

O interruptor S1, além de proporcionar a interrupção da corrente, desligando o aparelho, também pode ser útil como um RESET, pois não tendo a inércia dos eletrolíticos da fonte, proporciona um rápido POWER-UP e POWER-DOWN.

Os diodos D2, D3 e D4 formam, em série uma diferença de potencial tal que os 9 Volts provenientes da associação das pilhas acabam se equiparando aos vales da flutuação da tensão da fonte. Estes vales podem chegar próximos a 7 volts, equilibrados com os 9 menos os 2 volts resultantes dos três diodos. No protótipo, a d.d.p. foi de 0,015 volts, insuficientes para provocar uma descarga nas pilhas (que aliás andam bem carinhas). Estes diodos também impedem que os picos de tensão da fonte polarizem reversamente as mesmas; fazem com que sejam mais lentamente consumidas, aumentando sua vida útil. A impedância adicional que estes diodos criam, também contribui para reduzir a corrente de descarga.

As pilhas substituem a energia da fonte automática e instantaneamente após sua falta, o que torna imperceptível ao usuário e ao computador saber se está consumindo energia da rede, via fonte ou das pilhas.

Quando em teste, o INSTAN-TK foi capaz de suprir a ausência da fonte por 13 horas e 56 minutos ininterruptamente, a um consumo constante de 600 mA no modo SLOW, pois verifiquei ligeiro declínio no consumo quando o computador, em modo FAST, não mostrava a tela. Conclui-se então que o consumo varia se a tela é mostrada ou não e que o computador consome menos energia em modo FAST e rodando um programa (sem mostrar a tela) do que em SLOW ou durante uma instrução PAUSE ou ainda parado (ocasiões em que a tela é mostrada).

Condições de teste:
Computador usado: TK 82-C sem expansão de memória.
Características das pilhas: 6 pilhas novas tipo alcalina, tamanho grande, 1,5 V cada.
Circuito: INSTAN-TK original, aqui descrito.
Efeitos colaterais: nenhum efeito colateral foi observado sob condições normais de temperatura e pressão.
Melhoras adicionais: melhor imagem, livre de interferências da rede e menor aquecimento do computador.

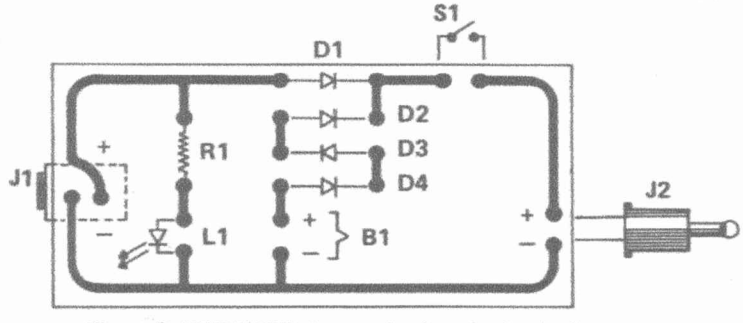


Figura 2: INSTAN-TK, *lay-out* da placa de circuito impresso.

Dependendo da aplicação específica do computador, o tempo que o aparelho é capaz de substituir a fonte pode diminuir com o uso de periféricos, tais como expansão de memória, impressora, etc. Se este for o seu caso ou se assim preferir, uma bateria de maior porte (e com consequente maior potência) de 9 Volts poderá ser usada (como aquelas usadas por fotógrafos para flashes eletrônicos, por exemplo), que ainda estará o aparelho na categoria de "temporariamente portátil", segundo os itens 1 e 2. Pode-se ainda usar pilhas recarregáveis, com a mesma vantagem da bateria de flash; longa durabilidade e possibilidade de recarga.

Por experiência própria, me sinto em posição de aconselhar a montagem e uso (ou só o uso) do INSTAN-TK pela sua grande utilidade, segurança (esqueça a tomada) e economia (de tempo e paciência). E por falar em economia (o que agrada a gregos e troianos e principalmente a árabes e judeus) o aparelho tem a característica de não inutilizar completamente as (caras) pilhas, que ainda ficam com carga suficiente para fazer um rádio portátil funcionar (principalmente as alcalinas).

A figura 2 mostra um exemplo de lay-out da placa de circuito impresso do INSTAN-TK (que, aliás, poderia muito bem ser batizado de JUMBO. Não é verdade que os elefantes jamais esquecem?) com as devidas indicações de onde e como montar os componentes. Aos pontos da placa marcados com +B1 e –B1, solde os terminais do porta-pilhas, devidamente polarizados.

Até o próximo número! O

Sinclair Place

O lugar compatível com os mini-micros.

(Z X 81, TK 83, TK 85, CP 200, APPLY 300, TS 1000, RINGO, AS 1000, ETC...)

MICROS
ACESSÓRIOS
SOFTWARE
LIVROS
REVISTAS

Sinclair Place do Brasil
Com. de Microcomputadores Ltda.
Rua Dias da Cruz, 215 – Sala 804
Ed. Meyer Golden Center
CEP 20.720 – Meyer – Rio de Janeiro – RJ.
Tel.: (021) 594-2699

GRÁTIS: UM MICROCOMPUTADOR
CURSOS
BASIC
COBOL
TURMAS
MANHÃ – TARDE – NOITE
INCLUSIVE AOS SÁBADOS
Turmas especiais para
TK 82 – TK 83 – TK 85 – CP 200
Cursos para alunos a partir de 10 anos de idade
MATRÍCULAS ABERTAS
PROMOÇÃO 40% DE DESCONTO
COMPUTATION CENTER
R. Teodoro Sampaio, 2534 – Loja 40
Pinheiros S.P.

Apertem os cintos, o piloto sumiu!

Álvaro A. L. Domingues

Você é um piloto que, com auxílio de seu TK 90X, deve pousar um avião em circunstâncias de emergência.

Você está num vôo com destino à Europa e tudo parece correr bem. Entretanto, no almoço são servidos dois pratos diferentes: peixe e frango. Você come frango. Parte dos passageiros e da tripulação prefere peixe. Só que o peixe está estragado e provoca intoxicação em todos, inclusive no piloto. O único com condições físicas e com algum conhecimento de pilotagem é você.

Nestas condições, você terá que pousar o avião sem problemas, salvando os passageiros e evitando um grave acidente.

Este foi o tema de "Voando para o perigo", *best-seller* que deu origem a vários outros livros sobre desastres aéreos e ao cinema-catástrofe dos anos 50 e posteriormente nos anos 70 (vide a série Aeroporto). Mais tarde surgiu "Apertem os cintos, o piloto sumiu", que ridicularizava a situação, mostrando todos os estereótipos do genêro (de personagens e situações). A partir daí ninguém mais teve coragem de fazer um filme sobre desastres aéreos (a não ser os russos com "A Tripulação ").

Porém, isso não nos impede de explorarmos o tema, mais uma vez, num programa destinado ao TK 90X. Digite-o e divirta-se.

O Jogo

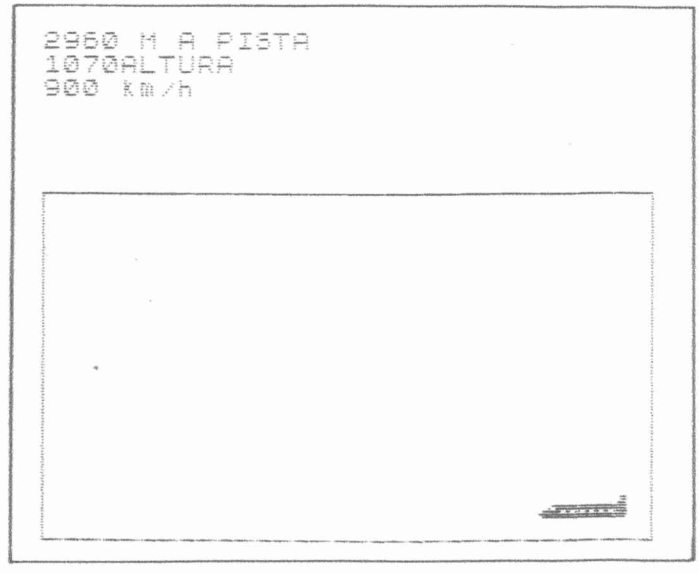
Este programa pode ser considerado como o embrião de um Simulador de Vôo. Indicamos o funcionamento de cada uma das sub-rotinas para que você mesmo possa fazer as modificações.

O programa, inicialmente, apresenta uma tela com a descrição do jogo, bem como dos comandos empregados na sua utilização:

p - para subir
o - para descer
w - aumenta a velocidade
q - diminui a velocidade
t - abaixa o trem de pouso

A seguir, é mostrada uma tela como a da figura 1. Nela você recebe informações sobre a sua altura atual, a sua velocidade e a distância que o separa da pista. Posteriormente lhe indicará o tamanho da pista disponível.

FIG. 1 — Tela inicial



O avião aparecerá no canto inferior direito do vídeo, com uma altura inicial de 1070 metros. Entretanto, não é esta a altura mínima. Se você digitar *O* ele aparecerá no canto superior direito, com a indicação de que a altura diminuiu. É como se a janela desenhada na tela descesse para mostrar a nova posição do avião. Continue digitando *O*. O avião se deslocará no sentido vertical, de baixo para cima.

Use agora o comando *P*. O avião se deslocará no sentido vertical de cima para baixo. Quando atingir o canto superior direito, recomeçará o movimento do canto inferior direito, como se a janela tivesse se movimentando para cima.

Você começa com uma velocidade de 900 Km/h. Esta velocidade não pode ser aumentada, já que é a máxima que pode ser atingida por seu avião. Para diminuí-la, use a tecla *Q*. Cuidado para não baixar muito a velocidade, causando a queda do avião.

Se a velocidade for inferior a 900 Km por hora você poderá aumentá-la até este limite, por meio da tecla *W*.

Você não verá o movimento do avião na horizontal, mas terá a indicação dele no canto superior esquerdo, onde aparecerá um número correpondente à distância até a pista. Quando a distância for inferior a 1000 metros, você deverá baixar o trem de pouso. Quando a distância for inferior a 500 metros, você verá a torre de comando. Como o avião fica "parado", (referencial) a torre de comando se deslocara', ao longo da tela. Quando a distância for 0, você estará sobrevoando o aeroporto. A partir daí, você terá dois quilômetros de pista para pousar. Você deverá então, tocar a pista (altura = 0) e parar o avião (velocidade=0). Se a pista acabar, você sofrerá um acidente.

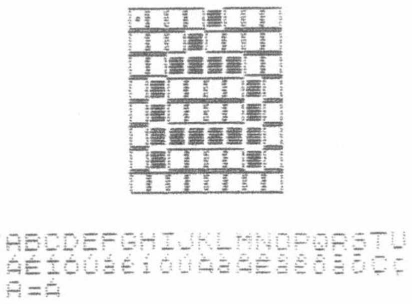
Terminada a missão com êxito, você receberá os parabéns em meio a um festival de sons e uma mensagem piscante na tela.

Caso ocorra uma falha, o avião cairá incendiando-se.

O programa

Antes de digitar o programa, digite a função UDG O (SYMBOL SHIFT + X) para obter os caracteres acentuados. Assim você poderá usar o sinal til (˜) que aparece em algumas mensagens. Para ver os sinais, digite UDG 2 (FIG.2).

Caracteres de UDG Ø



A primeira sub-rotina que o programa usa é a que forma os caracteres gráficos especiais, armazenando-os na região UDG (uma área da RAM reservada para esta finalidade). Ela está nas linhas de 9.900 a 9.999 e tem a incumbência de criar os caracteres gráficos utilizados para construir as figuras.

Para obtê-los, digite primeiro o CAPS SHIFT 9, seguido pela tecla que contém o caractere definido.

Consultando a figura 3, você poderá visualizar quais as teclas que contém o caractere em questão. Na tabela I você verá quais as letras que devem ser digitadas no modo gráfico para obter cada desenho. *Importante*: Elas só tomarão a forma definitiva quando você rodar a sub-rotina 9.900.

FIG. 3

Caracteres após ter sido rodada a rotina 9900

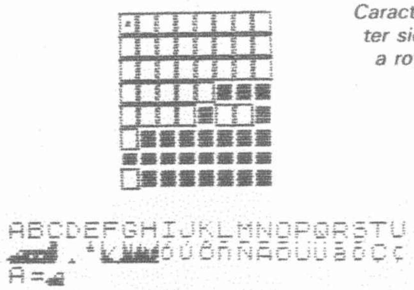


TABELA I - Caracteres gráficos

Linha	Caractere no modo gráfico
210	ABBBC, GHGHG
215	E,E
8520	GHGHG
8810	F

Se você quiser visualizar o que está fazendo, digite esta rotina primeiro, e as linhas 1 e 2.

A seguir rode o programa. Para verificar se você digitou tudo certo, acione o modo UDG 2 e compare com a figura 3.

Agora você pode digitar o restante do programa. Mas, antes de fazer isso consulte a tabela 2, que contém as informações necessárias para se compreender o programa e o seu funcionamento.

Tabela II - Rotinas e Sub-rotinas

Linhas	Função dentro do programa
1 - 10	Estabelece o estado inicial da tela
15 - 40	Estabelece as condições iniciais
150-250	"Gerencia" o jogo
1000 - 1090	Verifica as condições de acidentes
3000	Fecha em um Loop as rotinas de gerenciamento e de verificação de acidentes
4000-4900	Mostra a história do jogo e fornece os comandos
8000-8020	Verifica se o trem de pouso foi baixado
8500-8550	Rotina de acidente
8800-8827	Rotina de aterrissagem
8830-8888	Aterrissagem foi um sucesso(finalização)
8900-8990	Controles
9900-9999	Caracteres gráficos

```
   1 REM aperte os cintos,o pilo
to sumiu
   2 GOSUB 9900
   5 BORDER 5: PAPER 5: CLS
   7 GOSUB 4000
  10 CLS : PLOT 0,0: DRAW 0,119:
 DRAW 255,0: DRAW 0,-119: DRAW -
255,0
  15 LET A=1070
  20 LET S=1
  25 LET C=5000
  30 LET X=8: LET Y=26
  35 LET K=0
  40 LET T=0
 150 LET C=C-10
 160 LET D=C-2000
 170 IF D<=0 THEN PRINT AT 0,0;
INK 6; PAPER 1;C;" M DE PISTA ":
 GOTO 190
 180 PRINT AT 0,0; INK 6; PAPER
1;D;" M A PISTA "
 200 GOSUB 8000
 210 PRINT AT X,Y;"     "
 215 IF T=1 THEN PRINT AT X+1,Y+
1;" ";AT X+1,Y+4;" "
 220 PRINT AT 1,0; INK 7; PAPER
0;A;"ALTURA     "
 235 GOSUB 8960
```

→

```
 240 LET B=1000-S*100
 250 PRINT AT 2,0; INK 1; PAPER
6;B;" km/h
1000 REM ▓▓▓▓▓▓▓▓CONDIÇÕES▓▓▓▓▓▓▓▓
1010 IF C<=0 THEN PRINT AT 15,1;
"VOCE PASSOU DA PISTA";AT 17,3;"
ACABOU O COMBUSTIVEL";AT 19,5;"A
CIDENTE": GOSUB 8500
1020 IF C<2000 AND A<120 THEN GO
SUB 8820
1030 IF A<=100 AND D>0 AND D<500
 THEN GOSUB 8800
1040 IF A<=0 AND D>0 THEN GOTO 8
500
1050 IF B<=0 AND A>0 THEN GOTO 8
500
1060 IF B=0 AND A=0 AND C<2000 A
ND T=1 THEN GOTO 8830
1070 IF D<=1000 AND A<=500 AND T
=0 THEN PRINT AT 5,0; INK 2; FLA
SH 1;"BAIXE O TREM DE POUSO": GO
SUB 8000
1080 IF C<2000 AND A<=0 AND T=0
THEN GOTO 8500
1090 IF (A<0 AND D<0) OR (D>500
AND A<100) THEN GOTO 8500
3000 GOTO 150
4000 PRINT AT 0,5; FLASH 1;"Voan
do para o perigo"; FLASH 0
4010 PRINT AT 2,0;"VOCE VIAJA TR
ANQUILAMENTE PELAS"
4020 PRINT AT 4,7;"ILHAS CANARIA
S"
4030 PAUSE 100
4040 PRINT AT 7,1;"O PILOTO ADOE
CE REPENTINAMENTE!": PAUSE 100
4050 PRINT AT 9,3;"E VOCE E' O U
NICO CAPAZ"
4055 PRINT AT 11,5;"DE`FAZER O P
OUSO...": PAUSE 150
4060 PRINT AT 13,7;"OS COMANDOS
São:";AT 15,0;"(Q,P) PARA CONTRO
LAR A ALTURA": PAUSE 50
4070 PRINT AT 17,3;"(Q,W) PARA A
 VELOCIDADE"
4075 PAUSE 200
4080 PRINT AT 19,3;"(T) PARA ABA
IXAR O TREM "
4085 PAUSE 350
4090 CLS
4100 PRINT AT 10,3;"VOCE SE SENT
E CAPACITADO ?"
4110 PRINT AT 15,5;"PRESSIONE QU
ALQUER TECLA"
4115 PAUSE 0
4200 CLS
4900 RETURN
8010 IF INKEY$="t" THEN LET T=1:
 PRINT AT 5,0; FLASH 0; INK 5;"
                "
8020 RETURN
8500 REM ROTINA DE ERROS
8505 FOR H=X TO 21
8510 PRINT AT H,Y;"▓▓▓▓▓";AT H,Y
; INK 2; PAPER 6;"▓▓▓▓▓";AT H-1,
Y; PAPER 5; INK 5;"  "
8512 PAUSE 5
8520 IF H=21 THEN PRINT AT 21,0;
 INK 4;"▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓
▓▓";AT H,Y; INK 2; PAPER 6;"▓▓▓▓▓
▓": PAUSE 150
8525 NEXT H
8527 PRINT AT 15,1;"VOCE MATOU O
S PASSAGEIROS"
8528 FOR q=1 TO 300: NEXT q
8530 CLS : PRINT AT 10,0;"OUTRA
VEZ ?"
8535 PAUSE 0
8540 IF INKEY$="s" THEN GOTO 10
8550 CLS : STOP
8800 REM ATERRISAGEM
8805 LET K=K+1
8810 PRINT AT 20,K;"▙";AT 21,K;"
▜": PAUSE 10: PRINT AT 20,K;" ";
AT 21,K;" ": IF K=30 THEN LET K=
0: PRINT AT 21,30;" ";AT 20,30;"
"
8815 RETURN
8820 LET K=K+1: IF K=31 THEN LET
 K=0
8825 PRINT AT 21,K; INK 0; PAPER
 5;"▓▓";AT 20,K;"▓▓"
8827 RETURN
8830 REM sucesso
8840 FOR X=0 TO 50
8850 SOUND ,1,X: NEXT X
8855 PRINT AT 12,7; INK 1; PAPER
 6; FLASH 1;"VOCE CONSEGUIU !";
FLASH 0
8860 PAUSE 300: PRINT AT 15,5;"O
UTRA VEZ?"
8870 PAUSE 0
8880 IF INKEY$="s" THEN GOTO 10
8888 CLS : STOP
8900 REM INKEY$
8905 LET N$=INKEY$
8910 IF N$="o" AND X<21 THEN LET
 X=X+1: PRINT AT X-1,Y;"      ":
LET A=A-10
8920 IF X>20 THEN LET X=9: PRINT
 AT 20,Y;"      "
8930 IF N$="p" AND X>8 THEN LET
X=X-1: PRINT AT X+1,Y;"      ": L
ET A=A+10
8940 IF X<=8 THEN LET X=20: PRIN
T AT 9,Y;"      "
8950 RETURN
8960 LET N$=INKEY$
8970 IF N$="q" AND S<10 THEN LET
 S=S+1
8980 IF N$="w" AND S>1 THEN LET
S=S-1
8990 RETURN
9900 FOR z=0 TO 7: READ y: READ
x: READ w: READ v: READ u: READ
t: READ s: READ r
9910 POKE USR "a"+z,y
9920 POKE USR "b"+z,x
9930 POKE USR "c"+z,w
9940 POKE USR "d"+z,v
9950 POKE USR "e"+z,u
9960 POKE USR "f"+z,t
9970 POKE USR "g"+z,s
9980 POKE USR "h"+z,r
9990 DATA 0,0,56,0,16,128,64,1
9991 DATA 0,0,56,0,16,128,72,129
9992 DATA 0,0,120,0,56,132,105,1
63
9993 DATA 7,255,248,0,56,200,105
,183
9994 DATA 9,255,248,0,0,208,123,
247
9995 DATA 127,153,248,0,0,224,12
3,247
9996 DATA 255,255,248,16,0,248,2
55,254
9997 DATA 127,255,240,56,0,255,2
55,255
9998 NEXT z
9999 RETURN
```

```
9967 IF lineno=Inicio THEN LET s
tlin=lineno: LET stlineadd=linea
dd: LET stlinlen=linelen: GOTO 9
971
9968 LET lineadd=lineadd+4+linel
en
9969 GOTO 9964
9970 REM Qual o endereco da linh
a final
9971 LET lineadd=lineadd+4+linel
en
9972 IF lineadd=vars THEN CLS :
PRINT "Linha";Fim;"Nao existe":
STOP
9973 LET lineno=FN b(lineadd)
9974 LET linelen=FN a(lineadd+2)
9975 IF lineno=Fim THEN LET endl
en=lineno: LET endlineadd=linead
d: LET endlinlen=linelen: GOTO 9
980
9976 LET lineadd=lineadd+4+linel
en
9977 GOTO 9972
9978 PRINT
9979 REM Imprimir Linhas
```

```
9980 INPUT "Imprimir listagem na
 Tela ou na Impressora(t/i)? "
;q$
9981 CLS
9982 LET dev=2: IF q$="I" OR q$=
"i" THEN LET dev=3
9983 REM Escolha Caracteres para
 Impressao
9984 INPUT "Tipo de Espacejament
o? (1=simples,2=duplo etc.)";sp
9985 PRINT #dev; INK 2;FN b(stli
neadd);TAB 5;
9986 FOR n=stlineadd+4 TO endlin
eadd+2+endlinlen
9987 PRINT #dev; INK 1;CHR$ PEEK
 n AND (PEEK n=13 OR PEEK n>31);
9988 IF PEEK n=14 THEN LET n=n+5
9989 IF PEEK n=13 THEN FOR z=2 T
O sp: PRINT #dev: NEXT z: PRINT
#dev; INK 2;FN b(n+1);TAB 5;: LE
T n=n+4
9990 NEXT n
9991 PRINT #dev
9992 STOP
9999 SAVE "llist" LINE 9951
```

Controle de Estoque no TK 90X

Este aplicativo foi desenvolvido para ser usado por pequenas empresas que possuam um TK 90X e até 100 produtos diferentes em estoque. E fornece um resumo, com dados sobre estoque mínimo necessário, estoque atual e quantidade a ser encomendada, permitindo também a gravação e impressão do arquivo gerado. Para edição do mesmo, cada registro é acessado pelo seu número e não por seu código. O registro pode ser alterado ou deletado. Toda vez que este for alterado grave-o novamente. Quando for iniciá-lo utilize o comando GOTO 1000, nunca o comando RUN. Isso porque as matrizes dos valores, de cada registro, estão dimensionadas no início do programa (não poderia ser de outra forma!). Se ele é reiniciado pelo comando RUN, todas as matrizes, contendo os dados introduzidos, serão redimencionadas, fazendo com que todos os valores sejam zerados.

A lógica empregada neste programa é do tipo modular, ou seja, várias sub-rotinas gerenciadas por GOSUB. As linhas de decisão (Opções do MENU) estão também no início, após a rotina de formatação do MENU (linhas 1000 a 1090).

Vale ressaltar um detalhe interessante de seu algorítimo. O número do código do produto (b$) é tratado como STRING e armazenado numa matriz alfabética r$(100,12). Cada linha da matriz armazenará, um código de produto. Sendo assim, cada elemento será um caractere de cada um dos códigos.

Este processo é feito com o comando no formato: LET r$ (a,1 TO12)=b$ que significa: "atribua cada caractere de b$ (de 1 a 12 no máximo) a cada elemento da linha a".

Por exemplo, digamos que você introduziu os códigos: 101010, 202020, 303030.

A matriz ficara na forma:

$$r\$ = [a_{11}=1\ a_{12}=0\ a_{13}=1\ a_{14}=0\ a_{15}=1\ a_{16}]$$

$$r\$ = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & a_{1,5} & a_{1,6} & a_{1,7} & a_{1,8} & a_{1,9} & a_{1,10} & a_{1,11} & a_{1,12} \\ " & " & " & " & " & " & " & " & " & " & " & " \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} & a_{2,6} & a_{2,7} & a_{2,8} & a_{2,9} & a_{2,10} & a_{2,11} & a_{2,12} \\ 2 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 & 0 \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & a_{3,6} & a_{3,7} & a_{3,8} & a_{3,9} & a_{3,10} & a_{3,11} & a_{3,12} \\ 3 & 0 & 3 & 0 & 3 & 0 & 3 & 0 & 3 & 0 & 3 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

Estaremos dessa forma, limitando o número de código a 12 caracteres. É lógico que não é essa a vantagem.

Para cada caractere numérico introduzido na memória, o computador reservará dois bytes. Mas, para cada caractere alfanumérico ele reservará apenas 1. Então, ao tratarmos um número como STRING, economizaremos inúmeros Bytes, pois da forma descrita, o campo código do produto ocupará 12 Bytes no máximo.

Se esse procedimento não fosse tomado ocuparia sempre 24 Bytes!

Ao digitar o programa, para obtenção dos caracteres em português, utilizaremos o recurso de redifinição de caracteres UDG, residente na ROM do TK 90X. Na utilização de todos os caracteres em Português, maísculos e minúsculos devemos acessar UDG0 e logo após UDG1, como linha direta.

Esta operação deve ser feita antes de iniciarmos a digitação. Quando gravado o programa é automaticamente verificado.

```
    1 REM Controle de Estoque
    2 REM *******TK90X*******
   10 DIM r$(100,12): DIM w(100):
 DIM x(100): DIM y$(100,9)
   20 LET a=0: LET i=2
   30 BORDER 0: PAPER 0: INK 4: C
LS
   40 GOSUB 9000
1000 BORDER 0: PAPER 0: INK 4: C
LS
1010 PRINT AT 4,4;"**********MEN
U**********"
1020 PRINT AT 6,4;"TECLA A-ALTER
AR ARQUIVO"
1030 PRINT AT 8,4;"TECLA D-DELET
AR REGISTRO"
1040 PRINT AT 10,4;"TECLA E-ENTR
AR REGISTRO"
1050 PRINT AT 12,4;"TECLA N-NIVE
L DE ESTOQUE"
1060 PRINT AT 14,4;"TECLA C-CARR
EGAR ARQUIVO"
1070 PRINT AT 16,4;"TECLA S-SALV
AR ARQUIVO"
1080 PRINT AT 18,4;"**********#
**********"
1090 PRINT AT 21,8;"QUAL SUA OPÇ
AO ?": INPUT a$
1100 CLS
1110 IF a$="e" THEN GOTO 3000
1120 IF a$="n" THEN GOTO 4000
1130 IF a$="c" THEN GOTO 5000
1140 IF a$="s" THEN GOTO 6000
1150 IF a$="d" THEN GOTO 7000
1160 IF a$="a" THEN GOTO 2000
1170 GOTO 1090
2000 LET o=1: CLS
2010 PRINT AT 21,0;"ENTRE O CODI
GO A SER ALTERADO": INPUT b
2020 PRINT AT 21,0;"
                    "
2030 LET o=o+1: PRINT AT 0,3;"PR
ODUTO";AT 0,20;"NIVEL": PRINT AT
 1,0;"No.";AT 1,6;"Código";AT 1,
14;"Mín.*Atual*Encargo"
2040 FOR z=1 TO a: IF z=b THEN I
NK 6: PRINT AT 21,0;z;TAB 3;r$(z
,1 TO 12);TAB 16;w(z);TAB 20;x(z
);TAB 23;y$(z)
2050 NEXT z
2060 INPUT g: LET w(b)=w(b)+g
2070 PRINT AT o+1,0;b;TAB 3;r$(b
,1 TO 12);TAB 16;w(b);TAB 20;x(b
);TAB 23;y$(b)
2080 PRINT INK 5;AT 21,0;"Tecla
BREAK-proximo registro"
2090 IF INKEY$="m" THEN GOTO 100
0
2100 IF INKEY$<>" " THEN SOUND .
1,3: PRINT AT 21,0; INK 9;"Tecla
 BREAK-Proximo registro": SOUND
.1,2: GOTO 2080
2110 GOTO 2010
3000 PRINT AT 0,3;"PRODUTO";AT 0
,20;"NIVEL": PRINT AT 1,0;"No.";
AT 1,6;"Código";AT 1,14;"Mín.*At
ual*Encargo"
3010 INPUT b$: SOUND .1,2: LET a
=a+1: LET r$(a,1 TO 12)=b$
3020 INPUT b: SOUND .1,2: LET w(
a)=b: INPUT c: SOUND .1,1: LET x
(a)=c: INPUT d$: SOUND .1,2: LET
 y$(a)=d$
3030 PRINT AT i,0;a;TAB 3;b$;TAB
 14;b;TAB 19;c;TAB 25;d$: PRINT
AT 21,0; INK 4;"Tecla ENTER-Prox
imo registro": SOUND .5,0: PRINT
 INK 4;AT 21,0;"Tecla ENTER-Prox
ima entrada ": SOUND .5,1: INPUT
 a$: IF a$="m" THEN GOTO 1000
3035 IF a$<>"" THEN GOTO 3030
3040 PRINT INK 7;AT 21,0;"
                    ": LET
 i=i+1: GOTO 3010
4000 PRINT AT 0,3;"PRODUTO";AT 0
,20;"NIVEL": PRINT AT 1,0;"No.";
AT 1,6;"Código";AT 1,14;"Mín.*At
ual*Encargo"
4010 FOR z=1 TO a: IF r$(z,1 TO
12)="            " THEN GO
TO 4040
4020 PRINT z;TAB 3;r$(z,1 TO 12)
;TAB 16;w(z);TAB 20;x(z);TAB 24;
y$(z): NEXT z
4040 NEXT z
4050 PRINT INK 5;AT 21,0;"
    TECLA M - MENU          ": I
F INKEY$="m" THEN GOTO 1000
4060 IF INKEY$="s" THEN COPY
4070 GOTO 4050
5000 PRINT AT 0,3;"PRODUTO";AT 0
,20;"NIVEL": PRINT AT 1,0;"No.";
AT 1,6;"Código";AT 1,14;"Mín.*At
ual*Encargo"
5010 FOR z=1 TO a
5020 IF w(z)<x(z) THEN PRINT z;T
AB 3;r$(z,1 TO 12);TAB 16;w(z);T
AB 20;x(z);TAB 23;y$(z): NEXT z
5030 NEXT z
5035 INPUT u$
5040 GOTO 1000
6000 CLS : PRINT AT 21,0;"Nome d
o Arquivo";: INPUT a$: SAVE a$:
PRINT AT 21,0;"Término de gravaç
ão.Pressione uma tecla para veri
ficação": INPUT t$: VERIFY a$: P
RINT AT 21,0;"Verificado
                ": INPUT t$: GOTO 1
000
7000 CLS   INPUT "Entre o numero
 do registro a seralterado";p
7010 FOR q=1 TO a: IF q=p THEN G
OTO 7030
7020 NEXT q: PRINT AT 21,0; INK
5;"Codigo ";p;"Não Arquivado
              ": INPUT a$: GOTO 1000
7030 LET r$(q)=r$(a+1): LET w(q)
=w(a+1): LET x(q)=x(a+1): LET y$
(q)=y$(a+1): PRINT AT 21,0; INK
6;"Registro deletado.Pressione E
NTER                ": INPUT a$: GOTO
 1000
9000 PRINT INK 4;TAB 8; FLASH 1;
"CONTROLE DE ESTOQUE"; FLASH 0;T
AB 8;"********************"
9010 PRINT : PRINT INK 7;TAB 8;
9020 PRINT : PRINT TAB 2;"Toda v
ez que alterar os regis-tros,gra
ve-os novamente(opçao 5 do MENU)
. Os dados são gravados juntamen
te com o programa."
9030 PRINT TAB 2;"Quando usar um
programa com da-dos gravados não
 digite RUN, mas GOTO 1000."
9040 PRINT AT 21,0;"Pressione qu
alquer tecla": IF INKEY$="" THEN
 GOTO 9040
9050 RETURN
9998 SAVE "CTRL.EST0.": GOTO 999
8
9999 BORDER 7: PAPER 7: INK 0
```

Faça Música com seu Computador

Alexander Gromow

A evolução dos computadores pessoais ampliou consideravelmente seu campo de aplicação. O TK 90X e seus compatíveis, como o ZX-Spectrum, representam uma grande evolução quando comparados com o seu "irmão mais velho", o TK 85 (ou o ZX-81, CP 200, Ringo, etc.). Isto porque, além da cor e da alta-resolução gráfica, ele abre as fronteiras do som sem a necessidade de acessórios adicionais. Essa abertura é feita de modo limitado, mas bastante útil, e pode até surpreender quando corretamente explorada.

Neste artigo vamos nos ater à programação de som em Basic, se bem que existe um amplo espectro de possibilidades de gerar sons, ruídos, efeitos sonoros em geral, a partir de programação em Linguagem de Máquina.

Comentaremos as características tanto do TK 90X como do ZX - Spectrum, pois existem diferenças entre ambos, como também das aplicações.

Como ponto central deste artigo descreveremos as chaves de conversão entre a pauta de uma determinada música e a programação que permitirá ao computador executar a música em questão. Apesar do trabalho envolvido na decodificação de uma dada melodia e na posterior programação, o resultado pode ser gratificante. É possível utilizar trechos musicais ou até músicas completas em seus programas. Com isto o computador assume a tarefa de um executante exímio e tecnicamente correto, das músicas que forem programadas.

Finalizando, o artigo apresenta uma música completa: o tema do filme Golpe de Mestre, um "ragtime" composto por Scott Joplin.

Características do som do seu computador

Em Basic o som é gerado a partir de um comando específico, e é aqui que aparece a primeira diferença entre o TK 90X e o ZX-Spectrum. Esta diferença é, na verdade, uma mera questão de forma, a saber:

	TK 90X	ZX-Spectrum
comando	SOUND d,t	BEEP d,t

Ambos os comandos, apesar dos nomes diferentes, têm função exatamente iguais. Cada um dos computadores aceita o comando do outro e o converte para o seu próprio, isto é, um programa escrito num ZX-Spectrum com o emprego de BEEP's roda no TK 90X. Ao ser dado o comando LIST observa-se que os BEEP's foram automaticamente transformados em SOUND'S. As duas variáveis que acompanham estes comandos são:

d = duração da nota; t = tonalidade da nota.

Outra diferença, desta vez real, é que o ZX-Spectrum possui um pequeno alto-falante embutido, cujo "volume" fixo é um constante teste audiométrico. O TK 90X, por sua vez, gera o som através do amplificador do televisor. Como alternativa para os possuidores do ZX-Spectrum basta ligar os conectores EAR ou MIC a um amplificador. Em ambos os computadores, o comando de som em Basic interrompe a execução de um programa, agindo como o comando PAUSE.

O que podemos definir é a duração e a tonalidade de uma nota; assim sendo, é necessário programar nota por nota, indicando ambas as variáveis "d" e "t". A duração pode ser expressa por números fracionários. A tonalidade da nota é expressa por um número definido em função de uma escala e pode ser positivo ou negativo, incluindo o zero. Aliás, conforme estabelecido na Conferência Internacional de Londres, o Dó central tem a freqüência de referência de 261,6 Hz e no seu computador este valor corresponde exatamente a t=0.

Neste artigo partiremos da programação mais direta e simples, nota por nota, compasso por compasso, deixando aos leitores o caminho aberto para sofisticações; com a prática podem ser empregados comandos READ, DATA e RESTORE para, através de sub-rotinas, programar trechos de músicas que se repetem.

Tipos de aplicação do som

Vamos agora diferenciar efeitos sonoros e reprodução de música propriamente dita. Como efeitos sonoros ou "vinhetas" aplicam-se muito bem as chamadas músicas aleatórias ou randômicas. Veja o exemplo da Listagem 1.

```
  5 REM por A.Gromow em 1985
 10 REM vinheta aleatoria
 20 FOR x=30 TO 1 STEP -1
 30 SOUND .1,RND*60
 40 SOUND .001,RND*60
 50 SOUND .01,RND*60
 70 NEXT x
 80 REM experimente variar o va
lor de 'x' bem como as duracoes
dos BEEP's!
```

Listagem 1 ■

Conforme mencionamos anteriormente, para a programação de efeitos sonoros é mais adequado o emprego da Linguagem de Máquina. Com ela é possível aumentar o suave "click" que ocorre quando você aciona uma tecla. Para que isto aconteça basta entrar o seguinte:

POKE 23609, 255

Ao ligar o computador o valor após a vírgula é 0, mas pode variar até 255. Este procedimento pode ser incluído em programas para propiciar a confirmação sonora da entrada de dados.

Chaves de conversão entre a pauta musical e a programação de seu computador

As chaves de conversão que passamos a apresentar permitem a programação de uma partitura musical, sem que seja necessário possuir profundos conhecimentos de música. Iremos identificar os aspectos ligados à pauta em si, ao compasso e à sua fórmula, às notas e às pausas. Parece complicado, mas simplificaremos as coisas ao máximo.

Aspectos ligados às pautas

Cinco linhas e quatro espaços formam a **pauta** ou o **pentagrama**. Nela são dispostas as notas e demais sinais que formam a melodia. Além das cinco linhas, quando necessário, são empregadas linhas suplementares. As linhas e espaços são numerados de baixo para cima, ao passo que as linhas suplementares inferiores são numeradas de cima para baixo e as superiores, de baixo para cima (veja Figura 1). Juntamente com as linhas suplementares existem os espaços suplementares.

Os **sinais convencionais** são empregados na pauta, para indicar a seqüência de execução de uma música. Convém observar que nem sempre os editores de música observam os sinais convencionais à risca, o que poderá implicar num cuidado maior ao se definir a seqüência de execução de uma dada música. A letra de uma música, quando disposta na pauta, dá uma orientação adicional de sua seqüência de execução. (observar símbolo no original)

- "S" ou "segno" - sinal empregado para balizar a música;
- D.S. - "Del Segno" - volta até o "segno" e toca até o final;
- D.C. - "Da Capo" - volta até o início e toca até o final;
- "Coda" - geralmente indica o trecho final da música.

Na figura 4 apresenta-se um exemplo da aplicação de alguns sinais convencionais.

Compasso e fórmula de compasso

O compasso é a porção de música contida entre duas barras de divisão adjacentes. Estas barras dividem a música em trechos de duração igual. A fórmula de compasso, geralmente representada sob forma de fração, indica o número de tempos (batidas) e o inferior indica o tipo de nota que corresponde a um tempo unitário. A Figura 5 mostra um exemplo de compasso e fórmula de compasso.

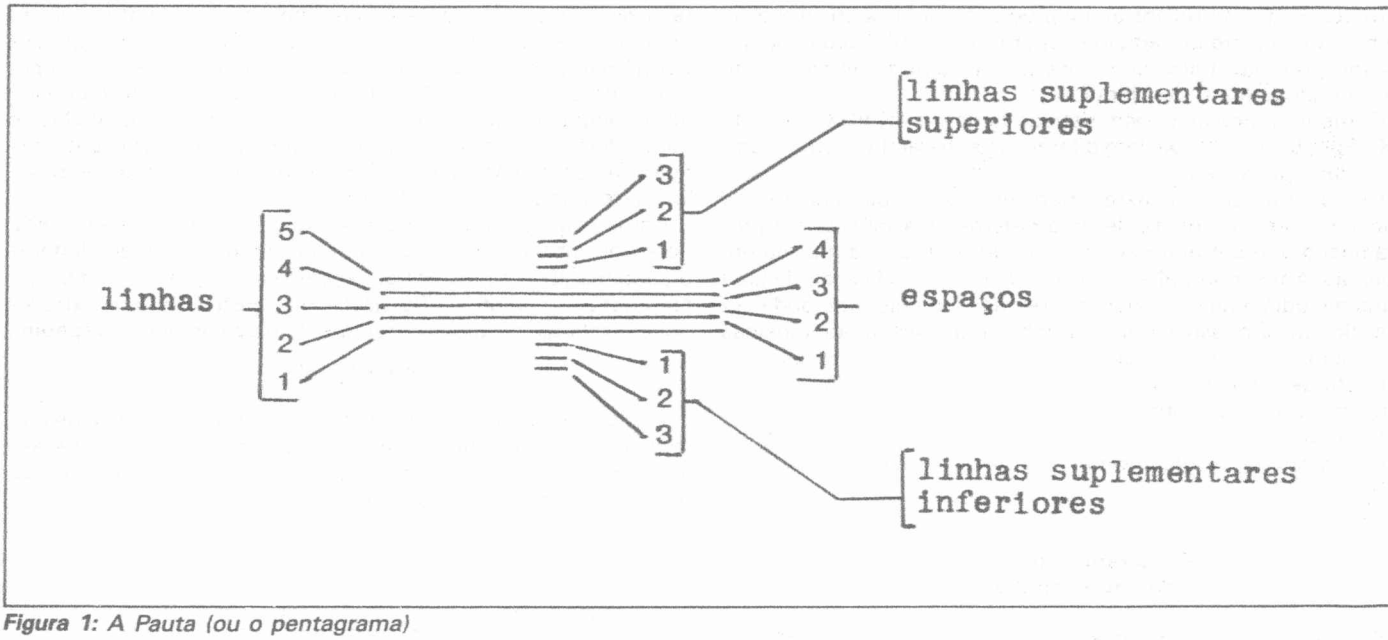


Figura 1: A Pauta (ou o pentagrama)

As **chaves** são colocadas no início de uma partitura e, com isto, identificam as notas dispostas na pauta. Músicas para piano, violino, flauta, guitarra e violão são freqüentemente escritas na clave de Sol. Para sons mais graves, como o contrabaixo, é empregada a clave de Fá. A figura 2 apresenta ambas as claves, bem como as notas dispostas na pauta e os correspondentes valores de "t".

As **barras** são linhas perpendiculares à pauta e são de quatro tipos:

simples: dividem os compassos entre si;
duplas: dividem os períodos;
repetição: duplas com dois pontos;
finalização: colocadas no fim da melodia.

A figura 3 mostra os vários tipos de barras.

A duração de cada tempo depende do andamento da música. Exatamente por este motivo é interessante programar uma dada música usando a variável "d", definindo-a antes que o computador a execute. Desta forma, podemos selecionar o andamento adequado.

Notas

A figura 6 mostra os tipos de notas (ou sons) em função do seu valor no tempo, e indica a fórmula a ser empregada na programação, sob a forma de um submúltiplo do valor "d". Veja exemplo na figura 7.

As notas, quando dispostas na pauta, assumem valores musicais em função da clave empregada. O seu computador pode

Figura 2: Acima clave de Sol e abaixo clave de Fá. Os números são os valores de "t" para o seu computador.

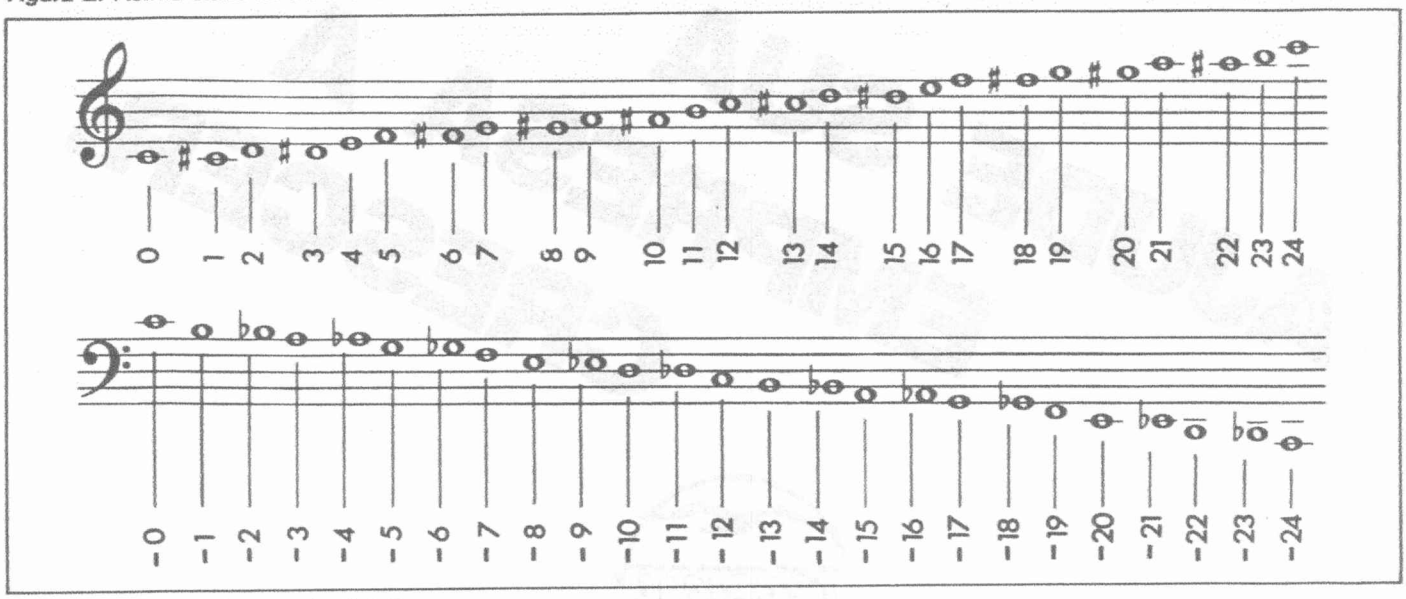


Figura 3: Tipos de barras de divisão.

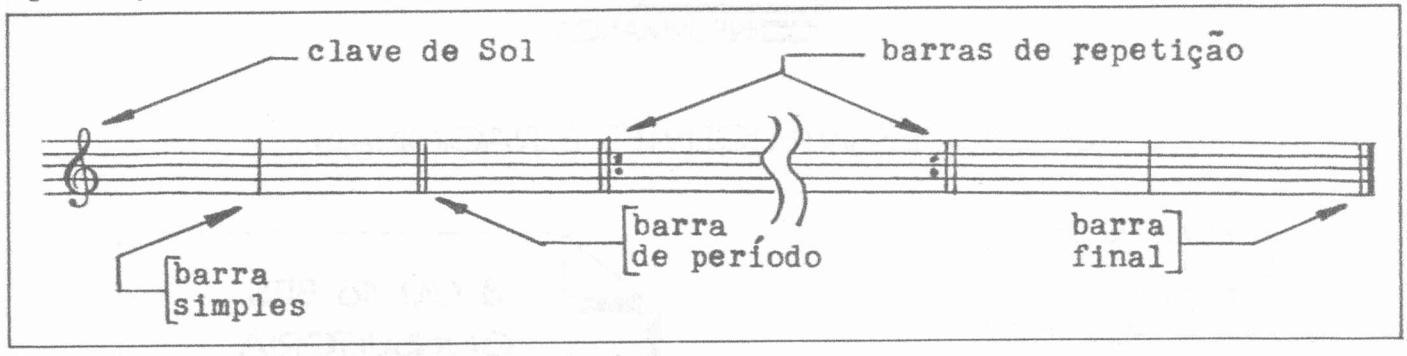


Figura 4: Exemplo de aplicação de sinais convencionais.

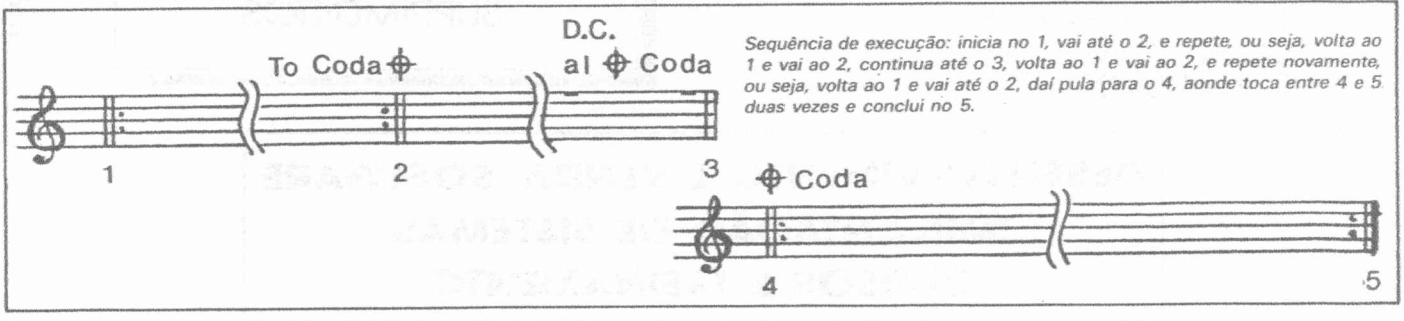
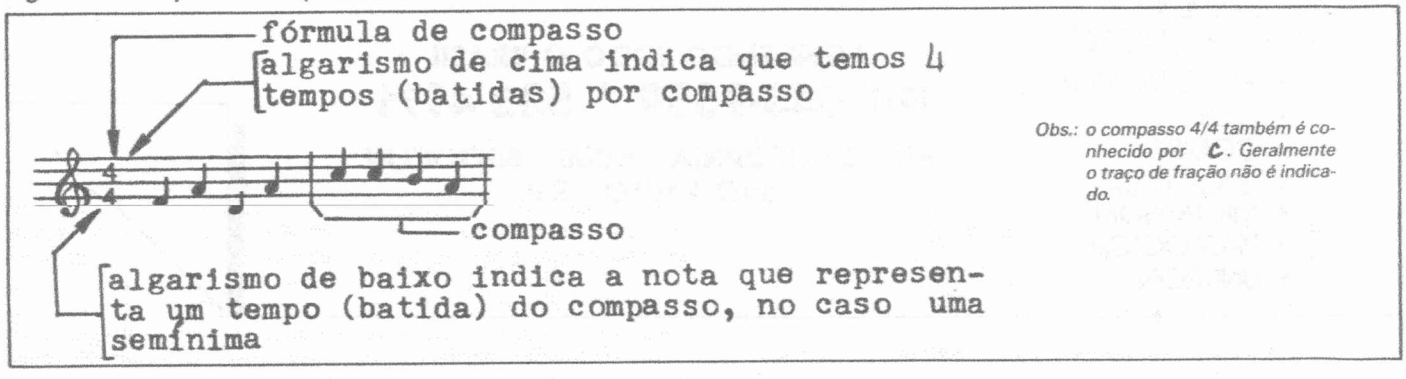


Figura 5: Exemplo de Compasso e Fórmula de compasso.



AJUDE SUA EMPRESA A CRESCER.

TROPICAL®
INFORMÁTICA

SOLUÇÕES INTELIGENTES E RACIONAIS

A TROPICAL DISPÕE
DE LARGA
EXPERIÊNCIA E
MODERNA TECNOLOGIA
PARA OFERECER A
EMPRESAS, ESCRITÓRIOS,
ESCOLAS, LOJAS
E CONSULTÓRIOS

8 OU 16 BITS
CONSULTORIA
SUPRIMENTOS

DESENVOLVIMENTO E VENDA SOFTWARE
IMPLANTAÇÃO DE SISTEMAS
CURSOS E TREINAMENTO

REVENDEDOR
AUTORIZADO:
- APPLETRONICA
- CCE
- COMPO
- LOGODATA
- MICROSOFT
- PROLÓGICA
- UNITRON

ATENDENDO TODO O BRASIL
(011) 543-9859 * 533-4971
RUA CALIFÓRNIA - 1.000 - BROOKLIN
SÃO PAULO - S.P.

EDIMAQ

gerar dez oitavas com tons e semitons, sendo que o valor de "t" pode variar entre -60 e 69. Para indicar a variação de um ou mais semitons, em relação a uma dada nota, são empregados os seguintes sinais:

♯ - sustenido - sobe um semitom;
b - bemol - desce um semitom;
♮ - bequadro - cancela um sustenido ou um bemol anteriores;
♯♯ - dobrado sustenido - sobe um tom inteiro;
bb - dobrado bemol - desce um tom inteiro.

Subir ou descer um semitom em seu computador representa somar ou subtrair,respectivamente, uma unidade no valor de "t".

Os sinais ♯ e b quando dispostos no início de uma pauta são chamados de acidentes fixos e têm efeito durante todo o transcorrer da peça musical. A sua presença indica o tom de uma dada escala. As escalas usuais e seus acidentes são mostrados na Figura 8.

NOME	NOTAS (sons) sim-bolo	NOTAS (sons) formula p/comp	PAUSAS (silencios) sim-bolo	PAUSAS (silencios) formula p/comp
Semibreve	O	d	𝄻	PAUSE d*60
Minima	𝅗𝅥	d/2	𝄼	PAUSE (d/2)*60
Seminima	♩	d/4	𝄽	PAUSE (d/4)*60
Colcheia	♪	d/8	𝄾	PAUSE (d/8)*60
Semicol-cheia	𝅘𝅥𝅯	d/16	𝄿	PAUSE (d/16)*60
Fusa	𝅘𝅥𝅰	d/32	𝅀	PAUSE (d/32)*60
Semifusa	𝅘𝅥𝅱	d/64	𝅁	PAUSE (d/64)*60

Figura 6

Vamos usar como exemplo a escala de Lá maior. Observamos que ela possui 3 notas sustenidas (elevadas de meio tom), a saber: Fá, Sol e Dó. Isto só não ocorrerá quando houver uma notação específica antes destas notas.

As notas musicais têm várias denominações. As usuais são a latina e a grega. A equivalência entre as duas é a seguinte:

notas musicais	Dó	Ré	Mi	Fá	Sol	Lá	Si	Latina
	C	D	E	F	G	A	B	Grega

Sinais e notações que

alteram a duração das notas

O **ponto** aumenta a duração da nota em metade do seu valor original:

𝅗𝅥. = d+ d/2 (fórmula para o computador)

A **ligadura** prolonga o valor de uma nota resultando em um tempo final que é a soma dos tempos das notas ligadas (veja exemplo na Figura 9). A ligadura pode se processar entre figuras de mesma duração ou de durações diferentes; pode se processar dentro de um compasso ou se expandir para outros compassos, ligando notas iguais na mesma altura.

A **Fermata** 𝄐 quando escrita sobre uma nota,indica que esta deve ser prolongada o máximo possível, a critério do intérprete.

A **Tercina** é um sinal que faz com que três notas unidas por ele passem a ter o valor de duas da mesma espécie (veja Figura 10).

Escalas de conversão para a programação de notas em seu computador

Primeiramente, vamos ver na Figura 11 uma parte da escala básica de notas que o seu computador possui, fazendo a comparação com o teclado de um piano. Notamos que a partir do Dó central, que tem o valor de t= 0, as demais oitavas distam de 12 da central, isto é, o Dó uma escala acima e uma escala abaixo do central tem o valor +12 e -12, respectivamente. Isto

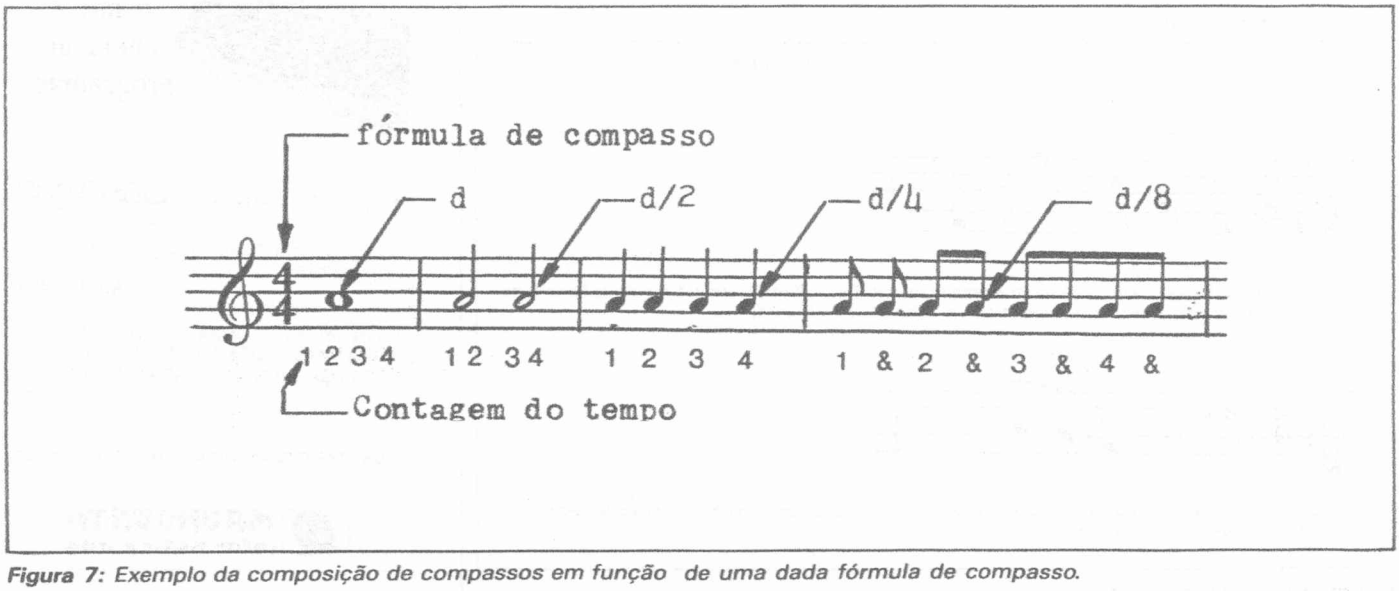


Figura 7: Exemplo da composição de compassos em função de uma dada fórmula de compasso.

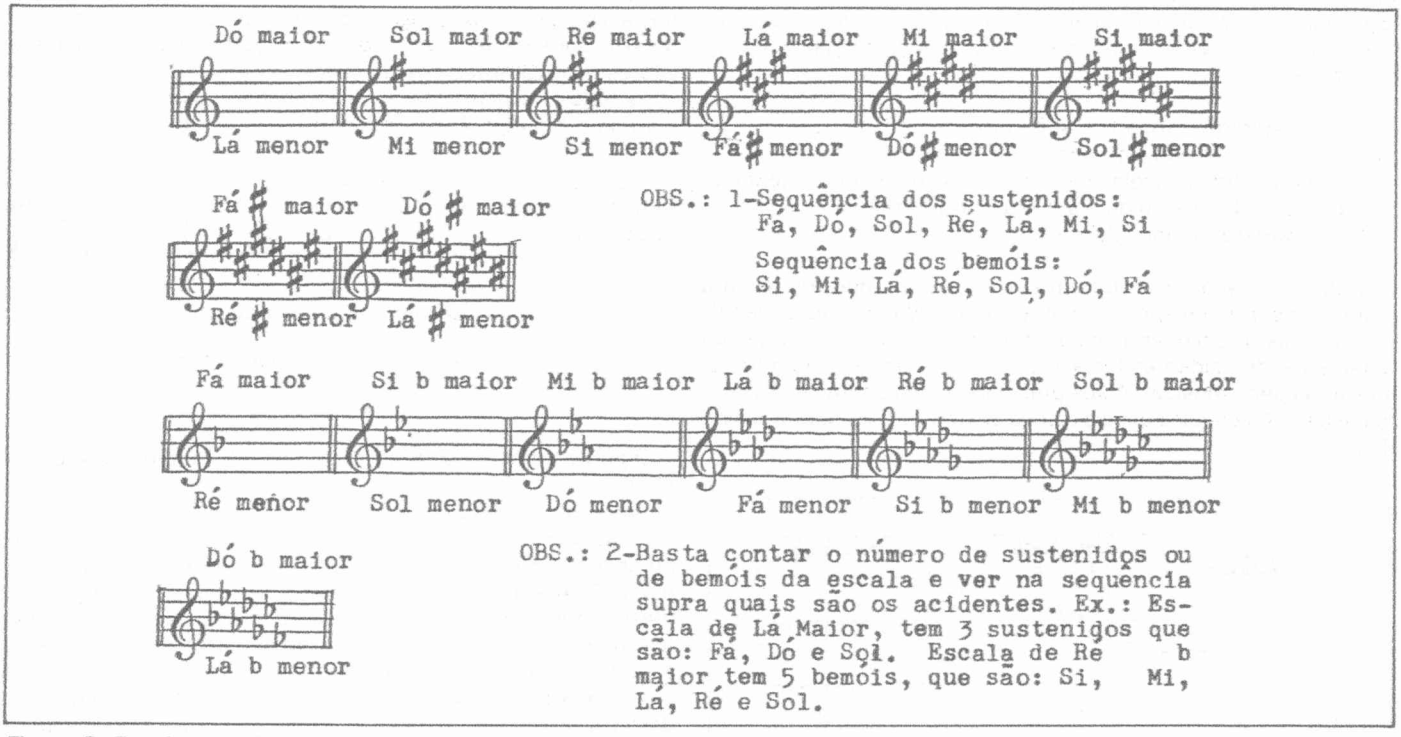
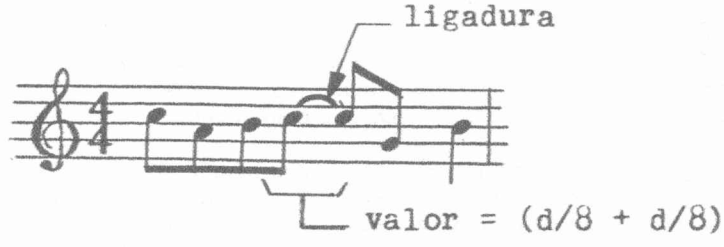
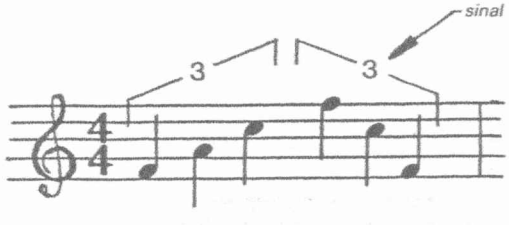


Figura 8: Escalas usuais e seus acidentes.

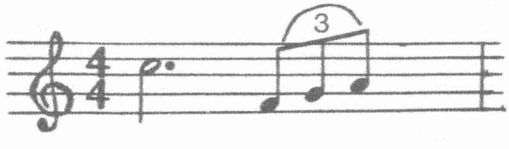


Observe que a segunda nota não deve ser tocada. A primeira nota é mantida pelo tempo igual à soma dos tempos das notas unidas pela ligadura.
Figura 9:

Figura 9: Exemplo de ligadura.



Sem a tercina cada nota valeria d/4. Com a tercina temos 3 X = 2 (d/4), onde X é a duração de uma das notas componentes destas tercinas e tem o valor de X = (2 (d/4)) / 3



Sinal de tercina entre colcheias, um arco que envolve o n.º 3. Chamando de Y a duração de cada nota desta tercina temos Y = (2 (d/8)) /3.

Figura 10: Exemplo de tercinas.

GUARDE ESTE NÚMERO:
(011) 255.7653
É onde você encontra tudo para os micros da linha TK.
• SOFTWARE (Jogos e Aplicativos)
• ACESSÓRIOS
• PERIFÉRICOS
• LITERATURA
• E O NOVO TK 90X - MICRODIGITAL
o micro cheio de programas.
• JOGOS APLICATIVOS CIBERTRON SOFTWARE
• APLICATIVOS TK-85 / TK-2000II EM FITA E DISKETE Microidéia
PASSE SEU PROGRAMA EM FITA SEM PROBLEMAS UTILIZANDO A INTERFACE LH-3.
CONHEÇA TAMBÉM O MONITOR LH-12
R.C.C.
FAÇA O CURSO DE BASIC NA MELHOR ESCOLA DE S.PAULO
MAGNODATA informática ltda.

se aplica a todas as notas. Por exemplo, o Mi imediatamente após o Dó central tem t=4, e o Mi uma escala acima tem t=4 + 12 = 16.

Como vimos, existem várias escalas. Tomando as da clave de Sol, vamos estabelecer uma escala de conversão para o seu computador. Na figura 12 temos o "formulário" para todas as **escalas da clave de Sol**. Para cada uma delas temos que acrescentar os acidentes correspondentes e completar os valores de "t".

Na figura 2 temos a indicação dos valores de "t" para completar as escalas de conversão. Como exemplo, tomemos a escala de Lá maior. O que se pretende é ter um instrumento de leitura de uma partitura, de modo a definir o parâmetro "t" que complementará, juntamente com "d", o comando SOUND d,t ou BEEP d, t. A escala de Lá maior tem todas as notas Fá, Sol e Dó sustenidas. Na figura 13 está a escala de conversão completa.

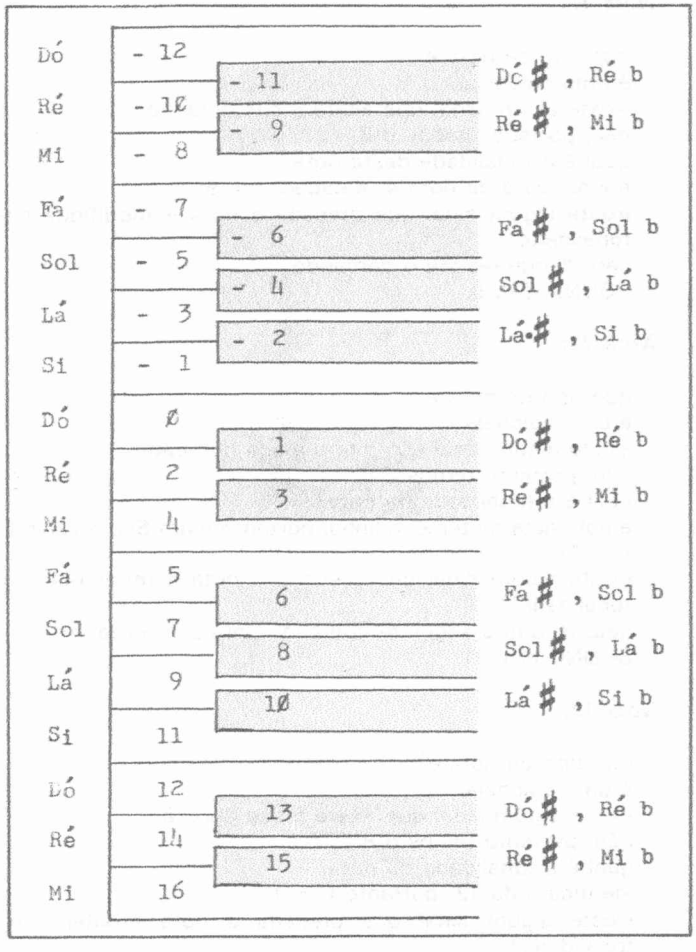


***Figura 11:** Comparação entre o teclado de um piano e o valor de "t".*

Pausas

Para completar os elementos que compõem uma pauta passemos a descrever as pausas ou silêncios. No seu computador elas serão efetuadas através do comando PAUSE seguido da duração correspondente, expressa sob a forma de uma fração do valor "d". As pausas correspondem às notas em duração; têm, portanto, os mesmos nomes, como pode ser visto na Figura 6.

Empregamos o "d" também para programar as pausas. Deste modo, é mantida a homogeneidade quanto ao andamento da música.

Também para as pausas se aplica o ponto que, como no caso das notas, aumenta a duração das pausas em metade do seu valor original.

O valor de "60" empregado na fórmula para o computador deve-se ao fato de que o comando PAUSE, para um valor unitário, tem a duração de 1/60 segundos (para países com a freqüência de 60 Hz). Com isto, o valor da pausa é mantido diretamente em função do valor atribuído a "d".

Exemplo detalhado de programação

Para programar uma música devemos seguir um roteiro, que se inicia com a análise da escala e da fórmula do compasso. A seguir, deve ser verificada a seqüência da música. A decodificação de cada nota se inicia pela verificação do seu tipo, a fim de ser determinada sua duração. Deve-se sempre observar a eventual existência de um sinal que altere o valor da nota. A programação das pausas segue esta seqüência. Voltando às notas, falta identificar sua tonalidade e o valor de "t". Para tanto usamos a escala de conversão, previamente preparada, observamos se existe algum sinal que altere a tonalidade da nota.

Passemos ao exemplo propriamente dito, e vamos programar o trecho da pauta da figura 14, seguindo o roteiro de programação.

a) Qual é a clave?
   Neste caso é a de Sol.
b) Qual é a escala?
   Como o Fá, o Dó e o Sol são sustenidos, temos a escala de Lá maior; aplica-se, portanto, o esquema da Figura 13.
c) Qual é a fórmula do compasso?
   4/4.
d) Decodificação das notas (e das pausas): Numeramos as notas deste exemplo para facilitar a identificação das mesmas.

Nota 1

- que tipo de nota é?
  uma colcheia, segundo a Figura 6.
- existe algum sinal que altere sua duração?
  não; portanto, também conforme a Figura 6, a fórmula de sua duração é d/8.
- qual é a tonalidade da nota?
  ela está imediatamente abaixo da primeira linha suplementar inferior, portanto, consultando a Figura 13, temos t = -1 e a nota é um Si.
- existe algum sinal que preceda a nota e modifique sua tonalidade?
  não; portanto o comando completo para programar esta nota é: SOUND d/8, -1 ou BEEP d/8, -1.

Nota 2

- que tipo de nota é?
  também é uma colcheia.
- existe algum sinal que altere sua duração?
  não; portanto a fórmula de sua duração é d/8.
- qual é a tonalidade da nota?
  é idêntica à anterior; portanto, até aqui t = -1.
- existe algum sinal que preceda a nota e modifique sua tonalidade?
  sim, um sinal de sustenido que sobe a nota em meio tom (ou seja, soma-se ao valor original de t), donde t = -1 + 1 = 0; como resultado o comando para esta nota é o seguinte:

  SOUND d/8, 0.

A fórmula de compasso de 4/4 indica que estas duas notas não completam um compasso inteiro (faltam 3 tempos ou batidas); isto porque duas colcheias formam um tempo. Trocando em miúdos: com a fórmula de 4/4, as quatro notas de valor igual

que comporiam um compasso são as semínimas, de duração d/4 (cada uma que têm a duração d/8, se somadas equivalem a uma semínima, ou seja d/8 + d/8 = d/4.
Aproveitando este raciocínio, podemos concluir que duas mínimas equivalem a uma semibreve, duas semínimas equivalem a uma mínima, e assim por diante. Assim sendo, uma semibreve equivale a oito colcheias. Resulta, portanto, a seguinte linha para programar a introdução composta pelas notas 1 e 2:
10 SOUND d/8, -1: SOUND d/8,0.

Nota 3:

- que tipo de nota é?
  uma colcheia.
- existe algum sinal que altere sua duração?
  não; portanto, resulta d/8.
- qual é a tonalidade da nota?
  ela está na primeira linha suplementar inferior. É, portanto, um Dó, que na escala de Lá maior é sustenido (este fato já foi considerado no esquema da **Figura 13**, logo, temos t = 1).
- existe algum sinal que preceda a nota e modifique sua tonalidade?
  não; portanto, para esta nota temos:
  SOUND d/8,1.

Nota 4:

- que tipo de nota é?
  trata-se de uma semínima.
- existe algum sinal que altere a sua duração?
  não; logo, a fórmula de sua duração é d/4.
- qual é a tonalidade da nota?
  ela está no segundo espaço do pentagrama, portanto é um Lá e, de acordo com o esquema da *Figura 13*, temos t = 9.
- existe algum sinal que preceda a nota e modifique a sua tonalidade?
  não; donde o comando para esta nota é:
  SOUND d/4, 9.

Nota 5:

- é idêntica à nota 3.

Nota 6:

- é idêntica à nota 4.

Nota 7:

- é idêntica à nota 3.

Nota 8 (9 e 10):
- que tipo de nota é?
  uma colcheia.
- existe algum sinal que altere sua duração?
  sim, uma ligadura, que une esta nota a outra no compasso seguinte.
- qual é o tipo de nota na extremidade desta ligadura?
  uma mínima.
- existe algum sinal que altere sua duração?
  sim, outra ligadura.
- qual é o tipo de nota na extremidade desta ligadura?
  uma colcheia.
- existe algum sinal que altere sua duração?
  não; resulta que a mesma nota é mantida por um tempo igual à soma dos tempos das notas unidas por estas duas ligaduras, a saber:
  d/8 + d/2 + d/8.

Podemos racionalizar e simplificar esta expressão. Mas, se não houver limitação da capacidade de memória em seu computador, é mais fácil deixar o problema para que ele mesmo resolva.

- qual é a tonalidade das notas unidas pelas ligaduras?
  idêntica à nota 4.
- existe algum sinal que preceda a nota e altere a sua tonalidade?
  não; portanto resulta o seguinte comando:
  SOUND (d/8 + d/2 + d/8), 9.

Como uma ligadura passou de um compasso para o outro, vamos tratar das notas 11, 12 e 13, que fecham o segundo compasso, para depois escrever a linha de programação para os primeiro e segundo compassos.

Nota 11:

- que tipo de nota é?
  é uma colcheia.
- existe algum sinal que altere a sua duração?
  não; portanto temos d/8.
- qual é a tonalidade desta nota?
  é idêntica à da nota 4, a saber: t = 9.
- existe algum sinal que preceda a nota e modifique sua tonalidade?
  não; donde resulta o comando:
  SOUND d/8, 9.

Nota 12:

- que tipo de nota é?
  é uma colcheia.
- existe algum sinal que altere a sua duração?
  não; portanto temos d/8.
- qual é a tonalidade da nota?
  a nota está na terceira linha, portanto é um Si e o valor de t = 11.
- existe algum sinal que preceda a nota e modifique sua tonalidade?
  não; resulta o seguinte comando para esta nota:
  SOUND d/8, 11.

Nota 13:

- que tipo de nota é?
  é uma colcheia.
- existe algum sinal que altere a sua duração?
  não; portanto temos d/8.
- qual é a tonalidade da nota?
  idêntica à da 12, portanto t = 11.
- existe algum sinal que preceda a nota e altere sua tonalidade?
  sim, um sustenido; portanto a tonalidade final é t = 11 + 1 = 12. O comando para esta nota é:
  SOUND d/8, 12.

Para os primeiro e segundo compassos que englobam as notas de 3 até 13 temos a seguinte linha de programação:

20 SOUND d/8, 1; SOUND d/4, 9; SOUND d/8, 1; SOUND d/4, 9; SOUND d/8, 1; SOUND (d/8 + d/2 + d/8), 9; SOUND d/8, 9; SOUND d/8, 11; SOUND d/8, 12.

Ainda como parte deste exemplo a Figura 15 mostra mais dois compassos. Porém, sem o detalhamento feito acima. Tente fazer a decodificação sem consultar o resultado apresentado a seguir.

Para verificar se você acertou vamos ao resultado destes dois compassos:

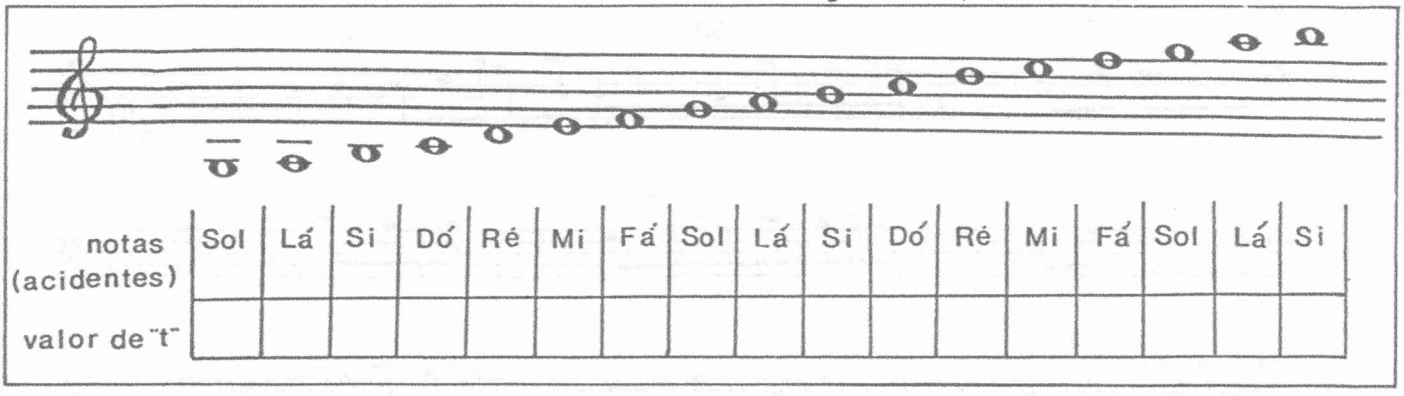
```
40 SOUND d/8, 13; SOUND d/8, 9; SOUND d/8, 11; SOUND
d/4, 13; SOUND d/8,8; SOUND d/4,ll.
50 SOUND d/2, 9; PAUSE (d/4) * 60;
   SOUND d/8, -1; SOUND d/8, 0.
```

Exemplo de uma música completa:

Finalizando temos na Figura 16 a partitura completa da música "The Entertainer". Os compassos foram numerados. Observe que nos compassos números 16 e 25 existem pausas de colcheia.

Além das constatações feitas no exemplo anterior, quando verificarmos a clave, a escala e a fórmula de compasso, devemos estabelecer a seqüência de execução da música. Para tanto, vamos identificar os sinais de repetição existentes, utilizando as letras A até F. A execução começa em A e vai até C, pula o décimo sétimo compasso e vai do E até o F, volta ao B e conclui em D. Neste caso é necessário observar as expressões "To Interlude", "Interlude", "D.S. al Fine", "Fine" e o sinal (Segno). Esta seqüência de execução pode ser estabelecida com auxílio dos comandos FOR e NEXT. A partir destas considerações iniciais você pode programar esta música.

Figura 12: Esquema básico para escala de conversão.



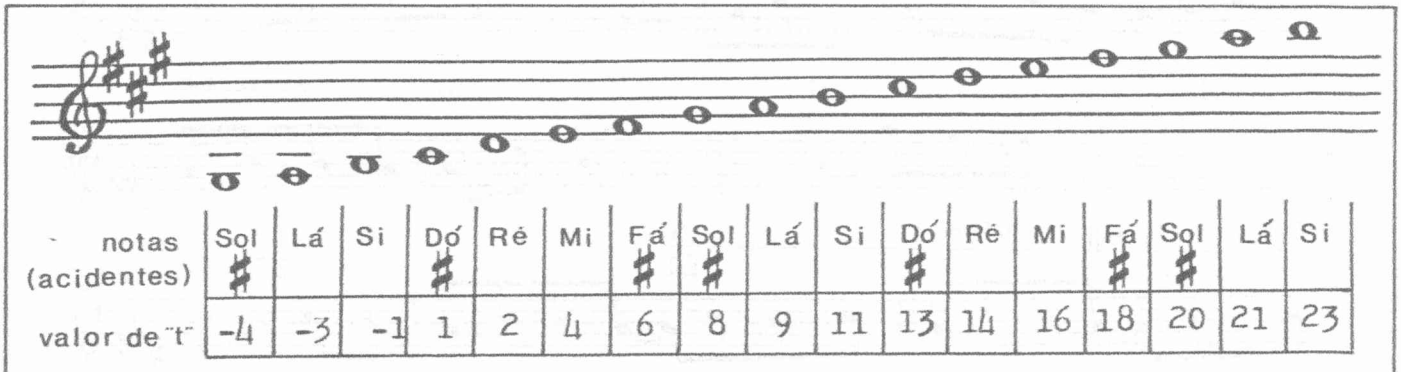


Figura 13: Escala de Lá maior (Fá menor).

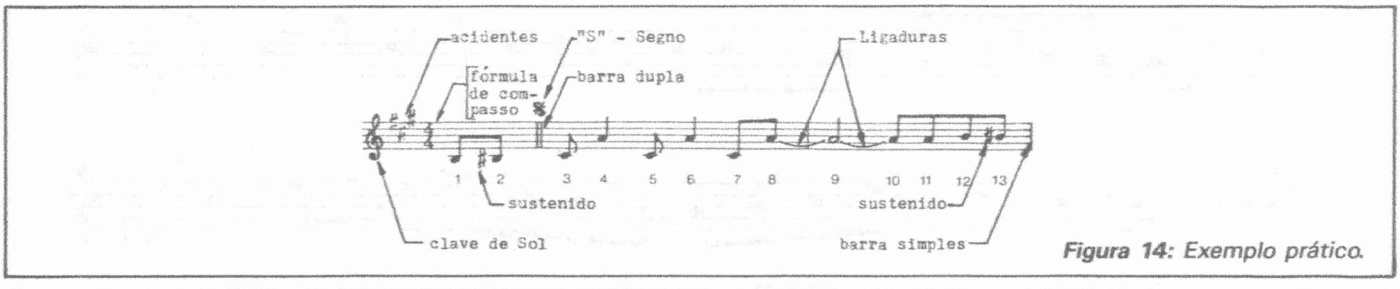


Figura 14: Exemplo prático.

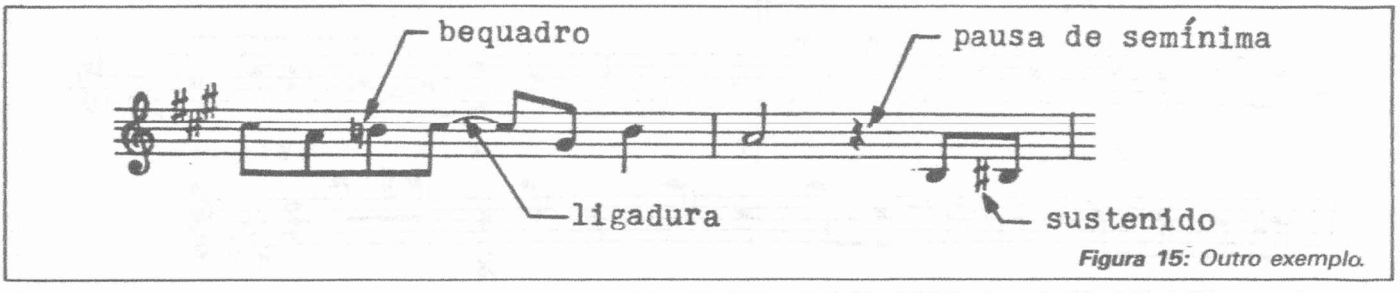
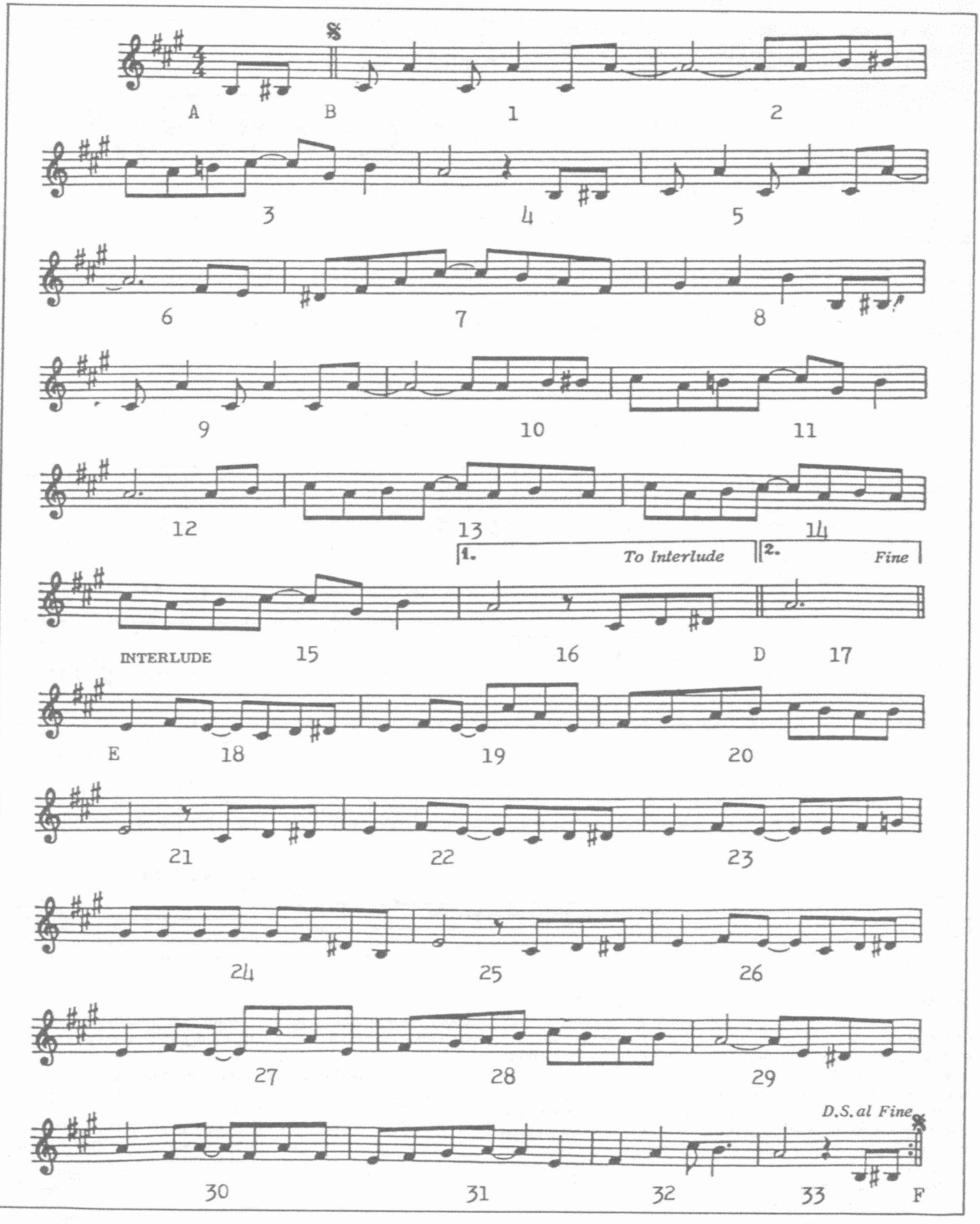


Figura 15: Outro exemplo.

Figura 16: Pauta completa do tema do filme "Golpe de Mestre".



A listagem 2 é o programa completo da música "The Entertainer". Observe que nesta lista incluímos a impressão do título da música e de seu autor. Além disso, foi acrescentada a seleção do valor "d". (Os vários "REM's" com esclarecimentos podem ser omitidos.) Adicionamos, também, uma pequena rotina para gravar este programa; para tanto, basta digitar GO TO 900.

Conclusão

Embora este artigo esteja direcionado aos possuidores de TK 90X e ZX-Spectrum, os rudimentos de música nele apresentados são universais e válidos para qualquer tipo de computador. Basta adaptá-los às ordens de programação específicas de cada um dos equipamentos.

Consultoria e Bibliografia: Consultoria Musical - Professora Maria Eugenia Pacheco de Brito; - Manual do ZX-Spectrum - Steven Vickers; ZX-Spectrum User's Handbock - R.J. Simpson & T.J. Terrell.

Alexander Gromow é engenheiro da Siemens e trabalha com coordenação de projetos em hidrogeração.

***Listagem 2:** Programa completo da música "The Entertainer".*

```
   1 REM por A.Gromow em 1985
   2 CLS : PRINT AT 0,7;"THE EN
TERTAINER"
   3 PRINT AT 5,3;"LA+";AT 5,13;
"(Scott Joplin)"
   4 PRINT AT 7,7;"(ragtime nove
lty)"
   6 INPUT "Tempo desejado (sugi
ro d=1.6)     d=";d: PRINT AT 10,
11;"d=";d
   8 PRINT AT 18,0;"para parar
'CAPS SHIFT' & 'BREAK'"
  10 SOUND d/8,-1: SOUND d/8,0:
REM inicio
  15 FOR a=1 TO 2
  20 SOUND d/8,1: SOUND d/4,9: S
OUND d/8,1: SOUND d/4,9: SOUND d
/8,1: SOUND (d/8+d/2+d/8),9: SOU
ND d/8,9: SOUND d/8,11: SOUND d/
8,12: REM compassos ns.1 & 2
  40 SOUND d/8,13: SOUND d/8,9:
SOUND d/8,11: SOUND (d/8+d/8),13
: SOUND d/8,8: SOUND d/4,11: REM
 com-passo no.3
  50 SOUND d/2,9: PAUSE (d/4)*60
: SOUND d/8,-1: SOUND d/8,0: REM
  compasso no.4
  60 SOUND d/8,1: SOUND d/4,9: S
OUND d/8,1: SOUND d/4,9: SOUND d
/8,1: SOUND (d/8+d/2+d/4),0: S
OUND d/8,6: SOUND d/8,4: REM com
passos  ns.5 & 6
  80 SOUND d/8,3: SOUND d/8,6: S
OUND d/8,9: SOUND (d/8+d/8),13:
SOUND d/8,11: SOUND d/8,9: SOUND
 d/8,6: REM compasso no.7
  90 SOUND d/4,8: SOUND d/4,9: S
OUND d/4,11: SOUND d/8,-1: SOUND
 d/8,0: REM compasso no.8
 100 SOUND d/8,1: SOUND d/4,9: S
OUND d/8,1: SOUND d/4,9: SOUND d
/8,1: SOUND (d/8+d/2+d/8),9: SOU
ND d/8,9: SOUND d/8,11: SOUND d/
8,12: REM compassos ns.9 & 10, i
dem linha 20
 120 SOUND d/8,13: SOUND d/8,9:
SOUND d/8,11: SOUND (d/8+d/8),13
: SOUND d/8,8: SOUND d/4,11: REM
 com- passo no.11,idem linha 40
 130 SOUND (d/2+d/4),9: SOUND d/
8,9: SOUND d/8,9: SOUND d/8,11:
REM compasso no.12
 140 SOUND d/8,13: SOUND d/8,9:
SOUND d/8,11: SOUND (d/8+d/8),13
: SOUND d/8,9: SOUND d/8,11: SOU
ND d/8,9: REM compasso no.13
 150 SOUND d/8,13: SOUND d/8,9:
SOUND d/8,11: SOUND (d/8+d/8),13
: SOUND d/8,9: SOUND d/8,11: SOU
ND d/8,9: REM compasso no.14, id
em li- nha 140
 160 SOUND d/8,13: SOUND d/8,9:
SOUND d/8,11: SOUND (d/8+d/8),13
: SOUND d/8,8: SOUND d/4,11: REM
 com- passo no.15
 165 IF a=2 THEN SOUND (d/2+d/4)
,9: PAUSE 4*(d/4)*60: GOTO 350:
REM compasso no.17, marcado com
'2. FINE',com o comando FOR TO
estabelecemos a ordem dos tre-
chos da pauta.
 170 SOUND (d/2),9: PAUSE (d/8)*
60: SOUND d/8,1: SOUND d/8,2: SO
UND d/8,3: REM compasso no.16, m
arcado com '1. TO INTERLUDE', co
m isto na primeira passada o com
pas-so no.17 e' pulado.
 190 SOUND d/4,4: SOUND d/8,6: S
OUND (d/8+d/8),4: SOUND d/8,1: S
OUND d/8,2: SOUND d/8,3: REM com
passo no.18
 200 SOUND d/4,4: SOUND d/8,6: S
OUND d/4,4: SOUND d/8,13: SOUND
d/8,9: SOUND d/8,4: REM compasso
 no.19
 210 SOUND d/8,6: SOUND d/8,8: S
OUND d/8,9: SOUND d/8,11: SOUND
d/8,13: SOUND d/8,11: SOUND d/8,
9: SOUND d/8,11: REM compasso no
.20
 220 SOUND (d/2),4: PAUSE (d/8)*
60: SOUND d/8,1: SOUND d/8,2: SO
UND d/8,3: REM compasso no.21, i
dem linha 170
 230 SOUND d/4,4: SOUND d/8,6: S
OUND (d/8+d/8),4: SOUND d/8,1: S
OUND d/8,2: SOUND d/8,3: REM com
passo no.22, idem linha 190
 240 SOUND d/4,4: SOUND d/8,6: S
OUND (d/8+d/8),4: SOUND d/8,4: S
OUND d/8,6: SOUND d/8,7: REM com
passo no.23
 250 SOUND d/8,8: SOUND d/8,8: S
OUND d/8,8: SOUND d/8,8: SOUND d
/8,8: SOUND d/8,6: SOUND d/8,3:
SOUND d/8,-1: REM compasso no.24
 260 SOUND (d/2),4: PAUSE (d/8)*
60: SOUND d/8,1: SOUND d/8,2: SO
```

```
UND d/8,3: REM compasso no.25, i
dem linha 170
 270 SOUND d/4,4: SOUND d/8,6: S
OUND (d/8+d/8),4: SOUND d/8,1: S
OUND d/8,2: SOUND d/8,3: REM com
passo no.26, idem linha 190
 280 SOUND d/4,4: SOUND d/8,6: S
OUND d/4,4: SOUND d/8,13: SOUND
d/8,9: SOUND d/8,4: REM compasso
 no.27-idem linha 200
 290 SOUND d/8,6: SOUND d/8,8: S
OUND d/8,9: SOUND d/8,11: SOUND
d/8,13: SOUND d/8,11: SOUND d/8,
9: SOUND d/8,11: REM compasso no
.28,     idem linha 210
 300 SOUND (d/2+d/8),9: SOUND d/
8,4: SOUND d/8,3: SOUND d/8,4: R
EM  compasso no.29
 310 SOUND d/4,9: SOUND d/8,6: S
OUND (d/8+d/8),9: SOUND d/8,6: S
OUND d/8,9: SOUND d/8,6: REM com
passo no.30
 320 SOUND d/8,4: SOUND d/8,6: S
OUND d/8,8: SOUND ((d/8)+(d/4)),
9: SOUND d/4,4: REM c0mpasso no.
31
 330 SOUND d/4,6: SOUND d/4,9: S
OUND d/8,13: SOUND ((d/4)+(d/8))
,11: REM compasso no.32
 340 SOUND (d/2),9: PAUSE (d/4)*
60: SOUND d/8,-1: SOUND d/8,0: R
EM compasso no.33, o 'D.S. al Fi
ne'indica que devemos voltar ao
'S'e ir até' o compasso 17,  ten
```

```
do pulado o 16
 345 NEXT a
 350 INPUT "Quer ouvir novamente
? S(ENTER)/N "; LINE z$: IF z$="
n" OR z$="N" THEN STOP
 360 GOTO 10
 900 REM rotina p/gravar
 910 CLS : PRINT AT 3,3;"Coloque
 uma fita virgem no"
 920 PRINT AT 4,10;"seu gravador
"
 930 PRINT AT 6,3;"Conecte o cab
o para gravar"
 940 PRINT AT 0,0;#1;"  Aperte u
ma tecla para gravar": PAUSE 0
 950 CLS : PRINT AT 10,3; FLASH
1;"PROGRAMA"; INVERSE 1;" 'RAGTI
ME' "; INVERSE 0;"GRAVANDO"
 960 SAVE "RAGTIME" LINE 1: REM
com o uso de 'LINE 1' damos ini-
cio automatico ao programa
 965 CLS : PRINT AT 3,8;"Rebobin
e a fita"
 970 PRINT AT 6,1;"Conecte o cab
o para verificar!"
 975 PRINT AT 9,5; FLASH 1;">lig
ue o gravador e...."
 980 PRINT AT 0,0;#1;"Aperte uma
 tecla para verificar!": PAUSE 0
1000 CLS : PRINT AT 10,1; FLASH
1;"PROGRAMA"; INVERSE 1;" 'RAGTIM
E' "; INVERSE 0;"VERIFICANDO"
1010 VERIFY "RAGTIME"
1020 CLS : STOP
```


SEU MICRO EXPLORADO AO MÁXIMO!

O **SOFTWARE CIBERTRON**, já tradicional no TK 85 e consagrado pelos peritos como a melhor opção para o TK 2000, está agora disponível também para o modernissimo TK 90X com as mesmas características de qualidade: Sistema Multigravação, poderosas rotinas em linguagem de máquina e instruções detalhadas. Qualquer que seja o seu TK, explore-o ao máximo com **CIBERTRON SOFTWARE.**

TK 2000

Falcons	Simulador de Vôo
Xadrez	Ceiling Zero
Bolo	Gamma Goblins
Eliminator	**Os Utilitários:**
Space Eggs	Graphs 2000
Grand Prix	ROM 2000

Inquestionavelmente o melhor software produzido p/ TK 2000.

TK 90X

Simulador de Vôo (Manual em Português)
Túmulo do Drácula (Tridimensional)
Pac-man e Campo Minado
Alta resolução gráfica, som e cores.

TK 85

Pinball	Corrida Milionária
Tourada	TK Word (Aplicativo)

Outros 12 programas em nossa lista.

Se você preferir,envie cheque nominal cruzado à **Cibertron Eletrônica Ltda** - Caixa Postal 17.005 - CEP 02399 SP, para receber os programas em sua casa. Remessas em 3 dias úteis. Jogos a Cr$32.900 cada - Utilitários e Simulador de Vôo para TK 2000 e TK 90X Cr$37.900 cada - TK Word e Simulador de Vôo para TK 85 Cr$32.900 cada. Garantia integral. Encontre toda linha **Cibertron** na **Magnodata** Tel. (011) 255 7653.

Revendedores autorizados:

MAGNODATA jumbo ELETRO Mappin SANDIZ CIBERTRON SOFTWARE

JOGO

Pelicano

Mergulhe nesta aventura com seu TK-90X.

Elaine Kelly Lima

Como todos sabemos, o pelicano é uma ave aquática de pernas curtas que, para sobreviver, ocupa-se da pesca. Sendo um hábil pescador, ele espera o momento exato para mergulhar e apanhar, de improviso, seu alimento.

Neste joguinho desenvolvido para o TK-90X com RAM de 16 ou 48 Kbytes, você terá a oportunidade de simular um pelicano que, para sobreviver, ataca os habitantes do mundo submarino.

A manipulação do pelicano pode ser feita com joystick ou através do próprio teclado. No segundo caso, as teclas de controle são:

5 - muda o vôo para a esquerda;
6 - faz com que a ave abaixe o vôo;
7 - a ave voa mais alto;
8 - o sentido do vôo é para a direita;
9 - provoca mergulho do pelicano.

Para passar da primeira fase a ave, controlada por você, terá que comer 7 peixes, sendo que, para isso, não poderá ultrapassar o tempo especificado pelo programa. Caso não consiga, seu pelicano perderá o vigor e por conseqüência, morrerá.

A cada nível o tempo especificado para completar a pescaria diminui e deste modo, somente a sua habilidade como pescador poderá mantê-lo vivo.

Sub-rotinas do programa

A linha 10 inicializa a tela e chama duas sub-rotinas: linha 1050 que forma caracteres gráficos e linha 510, que exibe a tela inicial. A linha 20 inicializa as variáveis que auxiliam a formatação da tela de jogo. A linha 30 exibe fundo da tela do jogo. A rotina das linhas 40 a 82 controla o jogo. Da 90 a 120 fazem a formatação da tela do jogo. A rotina entre as linhas 200 e 240 exibe tela final.

```
  10 PAPER 0: INK 7: BORDER 0: B
RIGHT 1: CLS : LET hi=0: GOSUB 1
050: GOSUB 510
  20 LET t=300: LET p=1: LET sc=
0: LET l=0: LET d=0: LET a$="
": LET b$=a$: LET c$="
": LET d$="
": LET e$="
                ";
LET a=5: LET b=15: LET c=1
  30 BORDER 0: BRIGHT 1: PRINT A
T 0,0; INK 0; PAPER 6;"  PLACAR
      RECORD                ";AT 0,10;
sc;AT 0,25;hi;AT 21,0;"   LAGO 1
      TEMPO 300             ": FOR f
=1 TO 15: PRINT AT f,0; PAPER 5;
"
 ": NEXT f: PRINT PAPER 1;,,,,,,
,,,,
  40 LET e$=e$(32)+e$(1 TO 31)+e
$(34 TO 64)+e$(33): PRINT AT 16,
0; PAPER 1; INK 6; BRIGHT 1;e$:
IF d=1 THEN GOTO 90
  45 PRINT AT a,b; PAPER 5; BRIG
HT 1;"  ";AT a+1,b;"  ": IF INKE
Y$="8" THEN LET b$=b$(5 TO )+b$(
1 TO 4): LET a$=b$: LET c=1: LET
 d$="
": LET b=b+1-(30 AND b=
30): GOTO 70
  50 IF INKEY$="5" THEN LET c$=c
$(5 TO )+c$(1 TO 4): LET a$=c$:
LET b=b-1+(30 AND b=0): LET c=-1
: LET d$="
": GOTO 70
  60 LET a=a+(INKEY$="6" AND a<9
)-(INKEY$="7" AND a>1): IF INKEY
$="9" THEN LET d=1
  70 LET a$=a$(5 TO )+a$(1 TO 4)
: PRINT AT a,b; INK 0; PAPER 5;a
$(1 TO 2);AT a+1,b;a$(3 TO 4)
  80 PRINT AT 18,0; PAPER 1;"
                          ";AT
 0,10; PAPER 6; INK 0;sc: IF sc>
```

```
0 AND sc/7=INT (sc/7) AND l=0 TH
EN LET e$="
    
          ": LET p=p+1: LET t=3
00-(((p-1)*20) AND p<10)-(160 AN
D p>9): PRINT AT 21,9; PAPER 6;
INK 0;p;AT 21,21;t: LET l=1
  81 LET t=t-1: PRINT AT 21,21;
PAPER 6; INK 0;t;" ": IF t>0 THE
N GOTO 40
  82 GOTO 200
  90 SOUND .001,60-a: LET p1=5:
LET p2=5: PRINT AT a,b; PAPER p1
; BRIGHT 1;"  ";AT a+1,b; PAPER
p2;"  ": LET a=a+1: LET b=b+c+(2
6 AND b<=2)-(26 AND b>=29): PRIN
T AT a,b; INK 0; PAPER p1; BRIGH
T 1;d$(1 TO 2);AT a+1,b; PAPER p
2;d$(3 TO 4): IF a=16 THEN LET p
1=5: LET p2=1: IF e$(b TO b+1)<>
"  " THEN LET e$(b-1 TO b+2)="
": PRINT AT a,b; PAPER 1; BRIG
HT 1;"  ";AT a+1,b;" ": LET d=0:
 LET a=8: LET sc=sc+1: GOTO 81
 95 IF a=15 THEN SOUND .01,10:
SOUND .01,12
 100 IF a=17 THEN LET p1=1: LET
p2=1: IF e$(b+32)<>" " THEN LET
e$(b+31 TO b+33)="   ": PRINT AT
 a,b-1; PAPER 1; BRIGHT 1;"   "
;AT a+1,b;"  ": LET d=0: LET a=8
: LET sc=sc+1: LET l=0: GOTO 81
 110 IF a>17 THEN PRINT AT a,b-1
; PAPER 1; BRIGHT 1;"   ";AT a+
1,b-1;"  ": LET a=8: LET d=0
 120 GOTO 81
 200 FOR g=0 TO 10: FOR f=0 TO 7
: SOUND .001,f+40: OUT 254,f*8:
OUT 254,200: PRINT AT a,b; INK f
; PAPER 5; OVER 1;"  ";AT a+1,b;
"  ": NEXT f: NEXT g: CLS
 210 PAPER 0: INK 7: BORDER 0: C
```

PROGRAMAS

```
LS : PRINT AT 10,0; INK 5;"
                          "
                          "
 215 PRINT AT 14,0;"
                          "
 217 PRINT AT 15,0;"
                          "
 220 IF sc>hi THEN SOUND .5,10:
SOUND .5,0: SOUND .2,12: SOUND .
2,12: SOUND 1,20: PRINT AT 15,0;
 INK 0; PAPER 6; BRIGHT 1; FLASH
 1;"          UM NOVO RECORD
  ": LET hi=sc
 230 PRINT AT 13,0; INK 7; BRIGH
T 1;" PRECIONE TECLA P/ CONTINUA
R  "
 240 PAUSE 1: PAUSE 0: GOTO 20
 510 PRINT AT 5,4;"Voce e um pel
icano  pescadore adora comer pei
xes. Porem vocetera um determina
do  tempo  paraisso. No primeiro
 lago voce teraque comer todos o
s  peixes  parapoder ir ao proxi
mo.Porem a cadanovo lago voce te
ra menos tempo."
 520 PRINT AT 16,4; INK 6;"USE A
S SEGUINTES TECLAS:          5-ESQ
UERDA      8-DIREITA       6-DESCE
     7-SOBE      9-PESCA"
 530 PRINT AT 20,0; INK 0; PAPER
 6; FLASH 1;"   PRECIONE TECLA P
ARA COMECAR  "
 540 LET a$="          ": LET b$="
          "
 550 PRINT AT 1,5; INK 6;a$(1 TO
 2);AT 1,25;b$(1 TO 2);AT 2,5;a$
(3 TO 4);AT 2,25;b$(3 TO 4): OUT
 254,24: IF INKEY$="" THEN LET a
$=a$(5 TO )+a$(1 TO 4): LET b$=b
$(5 TO )+b$(1 TO 4): OUT 254,200
: GOTO 550
 560 RETURN
 600 STOP
1000 DATA 0,0,1,3,187,255,235,85
,43,54,44,52,40,88,112,224,224,1
76,89,41,181,215,235,127,0,192,2
24,56,191,245,224,192
1010 DATA 0,0,128,192,221,255,21
5,170,212,108,52,44,20,24,14,7,7
,13,154,148,173,235,215,254,0,3,
7,28,253,31,7,3
1020 DATA 0,32,65,115,127,115,65
,128,0,96,248,204,232,252,248,96
,0,6,31,51,23,63,31,6,0,4,130,20
6,254,206,130,1
1030 DATA 152,148,218,213,250,12
5,122,61,0,0,0,0,128,128,128,192
,30,7,0,0,0,0,0,0,240,216,232,24
8,124,12,2,1
1040 DATA 0,0,0,0,1,1,1,3,25,41,
91,171,95,190,94,188,15,27,23,31
,62,48,64,128,120,224,0,0,0,0,0,
0
1050 FOR f=USR "a" TO USR "t"+7:
 READ a: POKE f,a: NEXT f: RETUR
N
```

```
ABCDEFGHIJKLMNOPQRSTU
                     U
A=
```

Voce e um pelicano pescador e adora comer peixes. Porem voce tera um determinado tempo para isso. No primeiro lago voce tera que comer todos os peixes para poder ir ao proximo. Porem a cada novo lago voce tera menos tempo.

USE AS SEGUINTES TECLAS:
5-ESQUERDA 8-DIREITA
6-DESCE 7-SOBE 9-PESCA

PRECIONE TECLA PARA COMECAR

TELA INICIAL

Tente Estas!

TK-90X

POKE 23609,N
onde N é a tonalidade do som do teclado

TK-90X

POKE 23613,0
protege programa contra listagem

TK-90X

POKE 23562,N
onde N é a pausa na repetição

TK-90X

POKE 23658,8
letras maiúsculas

TK-90X

POKE 23692,255
roda o programa sem SCROLL

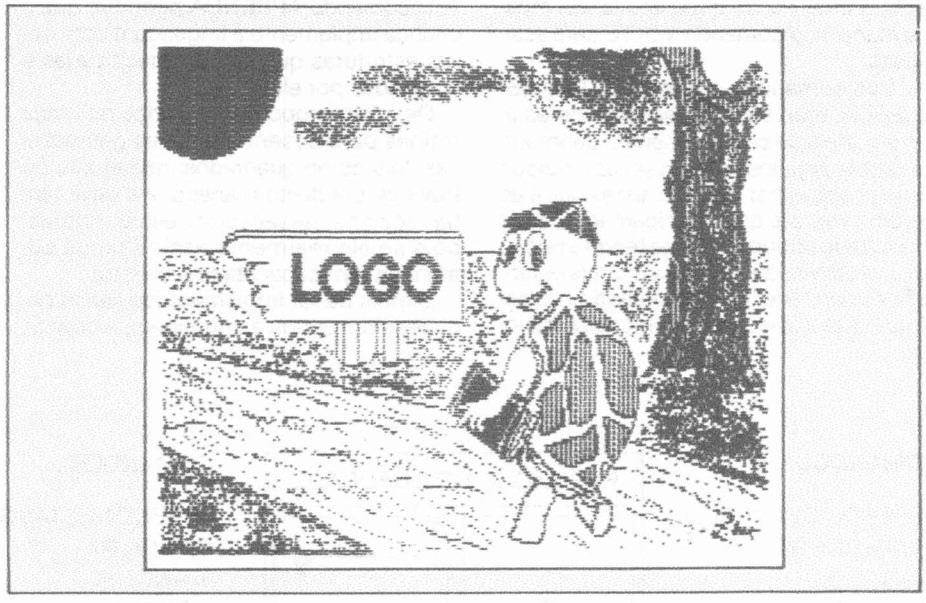
TK-90X

POKE 23756,0
numera primeira linha igual a 0

Uma introdução, através do TK-90X

Além de desenvolver a criatividade da criança, o LOGO aumenta o poder de concentração e agiliza o raciocínio.

Elaine Kelly Lima



Ao final da década de 60 um grupo liderado pelo matemático Seygmour Papert desenvolveu, no Instituto de Tecnologia de Massachusetts, a primeira versão do LOGO. O objetivo deles era criar uma linguagem que pudesse ser usada no ensino de crianças e para isso basearam-se em estudos realizados por Jean Piaget que comprovavam não ser aconselhável ensinar crianças a programar em determinada linguagem, pois elas sentem dificuldades para compreender conceitos abstratos.

Segundo o grupo, a linguagem deveria ajudar a criança a desvendar o mundo sem exigir que ela decorasse comandos. Pelo contrário, o aprendizado deveria ser espontâneo e à medida que houvesse necessidade de ampliar seus conhecimentos, a própria criança se interessaria por descobrir as novas instruções e recursos oferecidos, com os quais seria capaz de inovar e melhorar as rotinas que quisesse desenvolver.

Como resultado final da pesquisa do grupo nasceu o LOGO, que conseguiu o sucesso imediato devido às suas características. E, atualmente é inúmera a quantidade de versões que a linguagem possui para diferentes equipamentos.

O TK-90X e o LOGO

Por ser um micro acessível e oferecer recursos tais como cor e som, os micros da linha Spectrum como o TK-90X sempre tiveram um número muito grande de usuários.

Entre as usuários que adquirem o TK-90X muitas preferem deixar os filhos afastados do equipamento, ou seja, o menor tempo possível em contato com ele, por temerem que o televisor cause tensão ocular à criança. Porém, os pais não devem se preocupar com isso, pois raramente a criança consegue ficar muito tempo concentrada em alguma coisa e desta maneira, elas não estão sujeitas a este tipo de problema.

A única coisa pela qual os pais devem ficar alertas é que algumas linhas de televisores a cores fabricados antes de 1970 emitem uma pequena carga de radiação, que é prejudicial. Por este motivo é bom certificar-se de que o seu televisor a cores utilizado tenha sido fabricado posteriormente à esta data.

Agora, com o desenvolvimento de uma versão do LOGO para esse equipamento, ficou provado que no processo de educação infantil este pode ser muito favorável às crianças, pois mistura seus recursos com tecnologia desenvolvida especialmente para o ensino. Além disso introduz a criança no universo da computação de maneira gostosa e descontraída, o que a entusiasma, criando um novo interesse quanto à utilização do micro.

Utilizando o LOGO

Tendo como objetivo primordial ambientar a criança com processamento de dados sem forçar sua cabecinha, os comandos da linguagem foram nomeados de acordo com o conceito de que a criança não seja obrigada a decorá-los.

E dessa forma, os comandos foram transformados em palavras do seu vocabulário habitual, que geram respostas cujo sentido tenha a mesma idéia do significado da palavra.

Para que a criança tenha algo visível à sua frente que lhe forneça a idéia de execução das suas "ordens" (comandos), o LOGO possui uma tartaruga que á a responsável pelo traçamento dos desenhos.

O método de aprendizagem da linguagem é simples, sendo que a criança assimila os comandos através da prática própria, ou seja, ela testa várias vezes os comandos e instruções e verifica, à cada nova tentativa, a resposta dada pela tartaruga.

É bom lembrar que desta maneira a assimilação dos conhecimentos será muito mais rápida e duradoura, visto que a criança está aprendendo por si mesma.

De acordo com o dito acima ao iniciar o contato da criança com o micro você deve mostrar a ela um comando, deixando-a testá-lo. Desta maneira, ela irá errar várias vezes, e assim se ambientará com os limites da "tartaruga".

Alguns dos principais comandos da versão LOGO para o TK-90X são:

Comando	Utilidade
VENHA	Faz com que a tartaruga apareça na tela
LIMPE TELA	Apaga tela
AVANCE N	Faz a tartaruga avançar N passos
VOLTE N	Faz a tartaruga voltar N passos
DIREITA G	Faz a tartaruga girar G graus à direita
ESQUERDA G	Faz a tartaruga girar G graus à esquerda

Por exemplo, no início a primeira instrução que será apresentada à criança será VENHA e desta maneira, ela perceberá que este comando provocará o aparecimento de uma figura no centro da tela, figura que não é nada mais nada menos, do que a famosa tartaruga que acompanhará todas os traçados de cada desenho.

Vendo a tartaruga, a criança sentirá necessidade de aprender como fazê-la andar e assim, sentirá a necessidade de usar e desvendar os segredos do comando AVANCE que a faz avançar *n* passos na di-

reção à qual ela foi direcionada. Da mesma maneira, o comando VOLTE será assimilado.

Já os comandos como DIREITA e ESQUERDA, (que são utilizados pelos adultos por meio de conhecimentos geométricos sobre ângulos), devem ser aprendidos pelas crianças por meio de testes que elas devem fazer até que consigam identificar o resultado obtido. Uma vez tendo a noção sobre a dimensão do giro, a criança estará apta a coonhecer novos comandos ou ensinar a tartaruga a construir coisas maiores.

O comando APRENDA permitirá que a criança implemente a linguagem com novas estruturas que serão especificadas e nomeadas por elas.

De início proponha à criança que faça rotinas para desenhar formas geométricas, tais como quadrados, retângulos ou triângulos e desta maneira, ela deve tentar, sózinha, desenvolver essas rotinas, pois se ela realmente assimilou os comandos tem capacidade para isto.

Depois de ter implementado vários pequenos desenhos à linguagem, a criança começa a juntá-los e assim idealiza projetos maiores e passa a perceber que o LOGO tem uma amplitude muito maior do que ela imaginava.

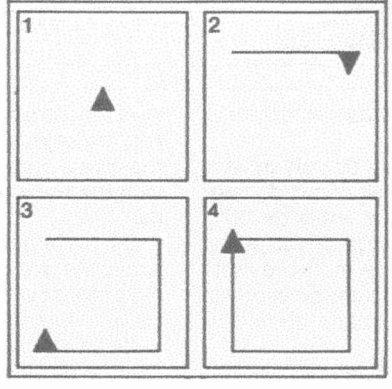
Rotinas para formar uma casa no vídeo

Através da junção de três rotininhas que são criadas, separadamente, a criança pode fazer o formato de uma casa na tela. Essas rotinas foram desenvolvidas somente com a utilização dos comandos básicos da linguagem.

COMANDOS

```
APRENDA QUADRADO
ESQUERDA 90
AVANCE 30
ESQUERDA 90
AVANCE 30
ESQUERDA 90
AVANCE 30
ESQUERDA 90
AVANCE 30
FIM
```

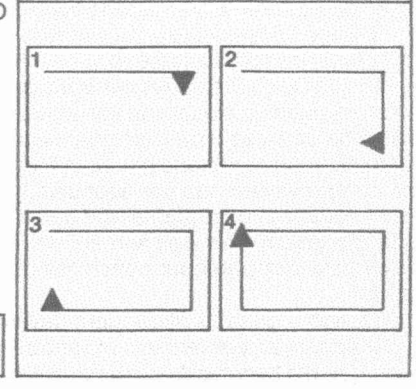
FORMAÇÃO DO QUADRADO



COMANDOS

```
APRENDA RETÂNGULO
DIREITA 90
AVANCE 50
DIREITA 90
AVANCE 30
DIREITA 90
AVANCE 50
DIREITA 90
AVANCE 30
FIM
```

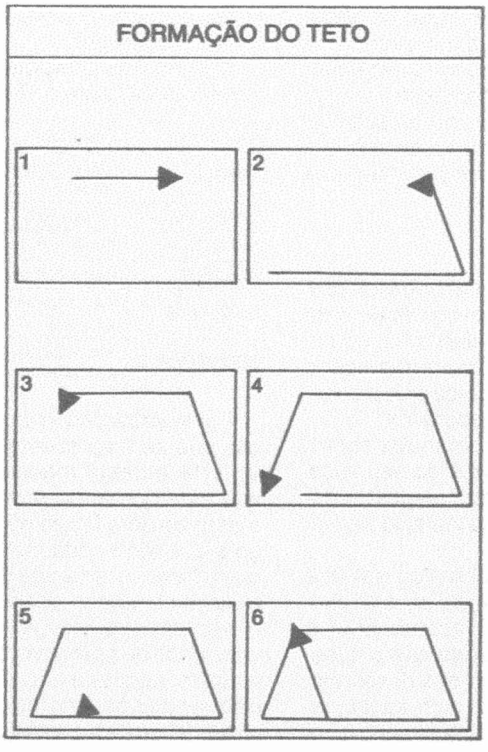
FORMAÇÃO DE RETÂNGULO



COMANDOS

```
APRENDA TETO
DIREITA 90
AVANCE 50
ESQUERDA 135
AVANCE 30
ESQUERDA 45
AVANCE 45
ESQUERDA 60
AVANCE 25
ESQUERDA 120
AVANCE 30
ESQUERDA 127.5
AVANCE 27
FIM
```

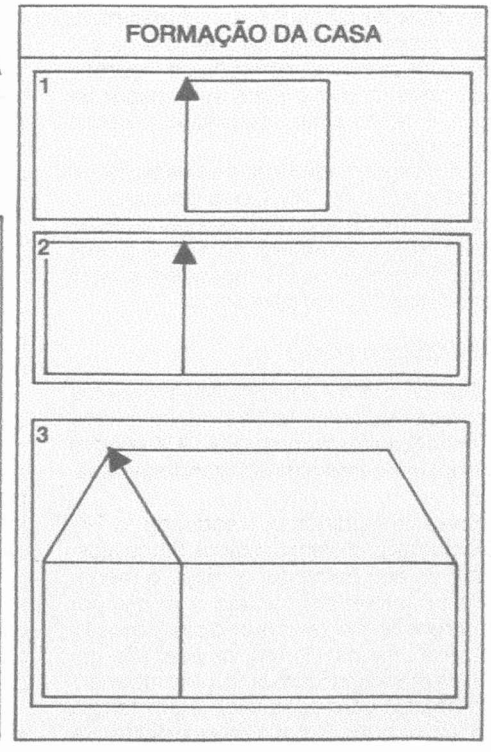
FORMAÇÃO DO TETO



COMANDOS

```
APRENDA CASA
QUADRADO
RETÂNGULO
TETO
FIM
```

FORMAÇÃO DA CASA



A ROM DO TK-90X

```
LD BC, 16
LD DE, MESS 6   } IMPRIME A MESS 6
CALL 8252
LD HL, (30177)  ; HL - INÍCIO OU LINHA INICIAL } IMPRIME O
CALL TI                                          } INÍCIO PROG.
LD BC, 17
LD DE, MESS 7   } IMPRIME A MESS 7
CALL 8252
LD HL, (30179)  ; HL CONTÉM O TAMANHO DO PROG. BASIC } IMPRIME
CALL TI                                              } TAMANHO
RET                                                  } BASIC
```

Tabela B

Nota		*Freqüência (Hz)*
DÓ	C 0	16.35
	C#0	17.32
RÉ	D 0	18.35
	D#0	19.44
MI	E 0	20.60
FÁ	F 0	21.83
	F#0	23.12
SOL	G 0	24.50
	G#0	25.96
LÁ	A 0	27.50
	A#0	29.14
SI	B 0	30.87
	C 1	32.70
	C#1	34.64
	D 1	36.70
	D#1	38.88
	E 1	41.20
	F 1	43.66
	F#1	46.24
	G 1	49.00
	G#1	51.92

Som

Existem duas formas principais de executarmos sons com o auxílio da ROM do TK-90X.

A primeira usa a rotina BEEPER, e a segunda a rotina BEEP (ou SOUND) de modo muito semelhante ao BASIC. Veremos como usar cada uma:

A rotina BEEPER (0385h) necessita, como dados de entrada, a freqüência da nota e a que duração. A freqüência fica nos registradores HL e a duração nos DE, por exemplo:

O comando SOUND 1,0 pode ser simulado por:

```
30000 11 05 01    ld de,duração
30003 21 6A 06    ld h1,nota
30006 CD B5 03    call BEEPER
30009 C9          ret
```

Para calcularmos a duração e a nota, usam-se os seguintes algorítmos:
duração = freqüência * tempo
nota = 437500/freqüência - 30.125

Verifique que a freqüência de um DO ou C4 é 261.60 Hz (dada na tabela B). Se substituirmos esse valor nas equações acima, para tempo igual a um segundo, teremos os valores de HL e DE. Verifique isso através do programa abaixo:

```
10 CLS
20 INPUT"FREQUENCIA (Hz) = ";F
30 INPUT"DURAÇÃO(s) = ";D
40 CLS
50 PRINT"HL = ";437500/F-30.125
60 PRINT"DE = ";F*D
70 PAUSE 9999
80 GOTO 10
```

Bom divertimento a todos. ◇

CNTK®

CLUBE NACIONAL DO TK

— Fitoteca com **1000** programas em todas as áreas.
— Ganhe gratuitamente uma fita gravada por mês com **10 programas de sua escolha.**
— Periféricos e livros com descontos especiais.
— Intercâmbio de programas.
— Sorteios mensais de periféricos.
— Programas sob encomenda.
— Mensalidade: **apenas Cz$ 106,00.**
Promoção especial: fique sócio e ganhe uma fita brinde com 5 jogos sensacionais para TK-90X, TK-2000 e TK-85.

Informações: Caixa Postal 6605
CEP 01051 — São Paulo — SP
Tel.: (011) 222-5977

Tente Estas! TK 90X

```
  1 INPUT a$: LET c=LEN
(a$)-1: PRINT a$
  8 LET a=16384
  9 FOR m=0 TO 7
 10 FOR i=a TO a+c
 20 POKE i,0
 30 NEXT i
 40 LET a=a+256
 50 NEXT m
```

```
 10 INPUT "Linha:";l;"
Coluna:";c;" Raio:";r
 20 FOR i=1 TO 102
 30 LET a=c+r*SIN (i/8)
 LET b=l+r*COS (i/8)
 40 PLOT a,b
 50 NEXT i
```

```
30213 - TI:  LD BC - 30129
             LD IX - 30154          ; BUF
F2:          LD D, (IX + 1)         ; BASE
             LD E, (IX + 0)
             LD A, 47
             AND A
F1:          INC A
             SBC HL,DE
             JR NC, F1
             ADD HL,DE
             LD (BC),A
             INC IX
             INC IX
             INC BC
             DEC E
             JR NZ,F2
             LD DE, 30129
             LD BC,5
             CALL 8252 ; PRINT STRING
             RET
```

Essa rotina transforma o conteúdo dos registradres HL em string e depois o imprime.

```
30254 - LEHEAD: LD DE, 17
                LD IX, 30164        ; HEADER
                XOR A
                SCF
                CALL 1366           ; LOAD DATA
                RET
```

LE HEADER

```
30267 - INÍCIO: LD A,2
                CALL 5633           ; ABRE O CANAL 5
                CALL CLS            ; 30190
                LD DE, MESS 1
                LD BC, 65           ; 65 = COMPRIMENTO DA MESS 1
                CALL PRINT STRING, 8252
                CALL LEHEAD         ; LEO HEADER
                CALL CLS
                LD A,18
                RST 16
                LDA,0
                RST 16

                LD DE, MESS 2
                LD BC, 12
                CALL 8252           ; PRINT STRING
                LD DE, 30165        ; INÍCIO DO NOME DO PRG
                LD BC, 10
                CALL 8252
                LD DE, MESS 3
                LD BC, 8
                CALL 8252
                LDA, (30164)        ; A CONTÉM O TIPO DE PROGRAMA
                LD e,A
                SLA A
                SLA A
                ADD A,e
                LD HL, MESS 8
                LD e,A
                LD D,0
                ADD HL,DE
                EX DE, HL
                LD BC,5
                CALL 8252
                LD DE, MESS 5
                LD BC, 11
                CALL 8252
                LD HL, (10175)      ; HL CONTÉM O TAMANHO DO PROG.
                CALL TI
```

IMPRIME A MESS 1

FLASH 0

IMPRIME A MESS 2

IMPRIME O NOME DO PROG.

IMPRIME A MESS 3

MULTIPLICA A*5

HL APONTA P/ O TIPO DO PROG.

IMPRIME O TIPO DE PROG.

IMPRIME A MESS 5

IMPRIME O TAMANHO

A ROM DO TK-90X

O byte 0 indica o tipo de programa:
0 - implica em programa BASIC;
1 - matriz numérica;
2 - matriz alfanumérica;
3 - bloco de dados.

os bytes de 1 a 10 contêm o nome do programa;

os bytes 11 e 12 contêm o tamanho do bloco se o byte 0 for 3;

os bytes 13 e 14 contêm o endereço inicial do bloco ou linha inicial do programa BASIC.

os bytes 15 e 16 contêm o tamanho da área ocupada pelo programa BASIC.

O programa referente às listagens A, B e C lê e exibe os dados do ***header*** de qualquer programa. Procure entendê-lo. Vale a pena!

```
30000 16 00 05 4C 45 49 54 4F
30008 52 20 44 45 20 48 45 41
30016 44 45 52 16 01 05 70 6F
30024 72 20 44 61 6E 69 65 6C
30032 20 4A 6F 73 65 20 42 75
30040 72 64 2E 16 0A 08 12 01
30048 45 53 50 45 52 41 4E 44
30056 4F 20 48 45 41 44 45 52
30064 2E 16 05 00 50 52 4F 47
30072 52 41 4D 41 3A 16 07 00
30080 54 49 50 4F 3A 16 09 00
30088 54 41 4D 41 4E 48 4F 3A
30096 16 0B 00 49 4E 49 43 49
30104 4F 2F 4C 49 4E 48 41 3A
30112 16 0D 00 54 41 4D 41 4E
30120 48 4F 20 42 41 53 49 43
30128 3A 30 35 37 36 38 62 61
30136 73 69 63 6D 2E 6E 75 6D
30144 6D 2E 61 6C 66 62 6C 6F
30152 63 6F 10 27 E8 03 64 00
30160 0A 00 01 00 00 41 53 53
30168 45 4D 42 4C 45 52 20 88
30176 16 00 00 88 16 00 00 00
30184 00 00 00 00 00 00 21 00
30192 40 11 01 40 01 FF 17 36
```

```
30200 00 ED B0 23 13 01 FF 02
30208 36 20 ED B0 C9 01 B1 75
30216 DD 21 CA 75 DD 56 01 DD
30224 5E 00 3E 2F A7 3C ED 52
30232 30 FB 19 02 DD 23 DD 23
30240 03 1D 20 E8 11 B1 75 01
30248 05 00 CD 3C 20 C9 11 11
30256 00 DD 21 D4 75 AF 37 CD
30264 56 05 C9 3E 02 CD 01 16
30272 CD EE 75 11 30 75 01 41
30280 00 CD 3C 20 CD 2E 76 CD
30288 EE 75 3E 12 D7 3E 00 D7
30296 11 71 75 01 0C 00 CD 3C
30304 20 11 D5 75 01 0A 00 CD
30312 3C 20 11 7D 75 01 08 00
30320 CD 3C 20 3A D4 75 5F CB
30328 27 CB 27 83 21 B6 75 5F
30336 16 00 19 EB 01 05 00 CD
30344 3C 20 11 85 75 01 0B 00
30352 CD 3C 20 2A DF 75 CD 05
30360 76 01 10 00 11 90 75 CD
30368 3C 20 2A E1 75 CD 05 76
30376 01 11 00 11 A0 75 CD 3C
30384 20 2A E3 75 CD 05 76 C9
00
```

LISTAGEM B

```
30000  MESS 1:   DEFM  22,0,5, "LEITOR DE HEADER"
30019            DEFM  22,1,5, "POR DANIEL JOSÉ BURD."
300H3            DEFM  22,10,8,18,1, "ESPERANDO HEADER."
30065  MESS 2:   DEFM  22,5,0, "PROGRAMA:"
30077  MESS 3:   DEFM  22,7,0, "TIPO:"
30085  MESS 5:   DEFM  22,9,0, "TAMANHO:"
30096  MESS 6:   DEFM  22,11,0, "INÍCIO/LINHA:"
30112  MESS 7:   DEFM  22,13,0, "TAMANHO BASIC:"
30129  BUF:      DEFb  0,0,0,0,0, ←—— BUFFER P/ IMPRESSÃO.
30134  MESS 8:   DEFM  "BASIC"; "M.NUM", "M,ALF", "BLOCO"
30154  BASE:     DEFW  10000,1000,100,10,1 ←—— BASE DECIMAL
30164  HEADER:   DEFb  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, ←——
                       └→ armazena o header
```

LISTAGEM C

```
30190 - CLS LD HL - 16384
            LD DE - 16385
            LD BC - 6143      } LIMPA A TELA
            LD (HL),0
            LDIR
            INC HL
            INC DE
            LD BC - 767       } COLORE A TELA DE VERDE
            LD (HL),32
            LDIR
            RET
```

Aula 5

Daniel José Burd

Nesta lição aprenderemos a enviar dados para o gravador e a recebê-los. Veremos também, as rotinas da ROM que geram sons.

Manipulando o gravador cassete

Quando mandamos o computador enviar dados para o gravador, através do comando SAVE, dois blocos de bytes são deslocados. O primeiro bloco, conhecido como ***header,*** contém as seguintes informações:

- tipo de programa (exemplo: BASIC);
- nome do programa;
- tamanho;
- endereço inicial.

O segundo bloco é o programa em sí.

Dessa forma, ao executarmos um LOAD, o primeiro bloco a ser lido será o ***header,*** o qual informará ao micro onde o programa deve ser armazenado, o nome do programa etc.

A rotina LOAD DATA

Quando desejamos ler dados do gravador através de instruções Assembler, devemos dar várias informações ao computador. O computador necessita saber se o que ele vai receber é um ***header*** ou não, onde colocar o bloco de bytes, qual o tamanho do bloco etc. Para informamos todos esses dados ao micro devemos usar a seguinte convenção:

acumulador - se for 0, implica em leitura ou escrita de ***header;***
se for 255 implica em leitura ou escrita de bloco:

flag de carry - se for 0 implica em verificação de dados (VERIFY);
se for 1 implica na leitura de dados (LOAD);

registradores IX - contêm o endereço a partir do qual devem ser colocados (ou enviados) os bytes;

registradores DE - contêm o tamanho do bloco.

Após especificarmos todos os parâmetros, basta dar um CALL LOAD DATA (0556h) e pronto. Aqui estão as possíveis combinações dos parâmetros anteriores:

```
LOAD BYTES: 1d de,nbytes     ;nbytes = tamanho do bloco
            1d ix,início
            1d a,255         ;não header
            scf              ;LOAD
            call LOAD DATA
            ret

LOAD HEADER: 1d de,17        ;17 bytes e o tamanho do
                             header
             1d ix, ínico do header
             xor a
             scf
             call LOAD DATA
             ret

VERIFYBYTES: 1d de,nbytes
             1d ix,ínicio
             1d a,255
             and a
             call LOAD DATA
             ret

VERIFYHEADER: 1d de,17
              1d ix,ínicio do header
              xor a
              call LOAD DATA
              ret
```

A rotina SAVE DATA (04C2h)

Essa rotina usa os mesmos parâmetros da LOAD DATA, exceto a flag de carry. Aqui estão os possíveis usos dessa rotina:

```
SAVEBYTES:  1d de,nbytes
            1d ix, ínicio     ;ínicio do bloco a ser salvo
            1d a,255
            call SAVE DATA
            ret

SAVEHEADER: 1d de,17
            1d ix,ínicio do header
            xor a             ;a = 0 implica em header
            call SAVE DATA
            ret
```

Vamos, a título de exemplo, salvar os bytes que compõem a imagem desenhada na tela. Como sabemos, de lições anteriores, a região ocupada pela tela inicia-se no endereço 16384 e tern comprimento 6912 bytes, incluindo a região de atributos.

Dessa forma, um programa que salva essa região é mostrado abaixo:

```
30000 11 00 18    1d de,6912     ;salva 6912 bytes;
30003 DD 21 00 40 1d ix,16384    ;aponta para o arquivo da
                                  tela;
30007 3E FF       1d a,255       ;indica que não é header;
30009 CD C2 04    call SAVE DATA
30011 C9          ret
```

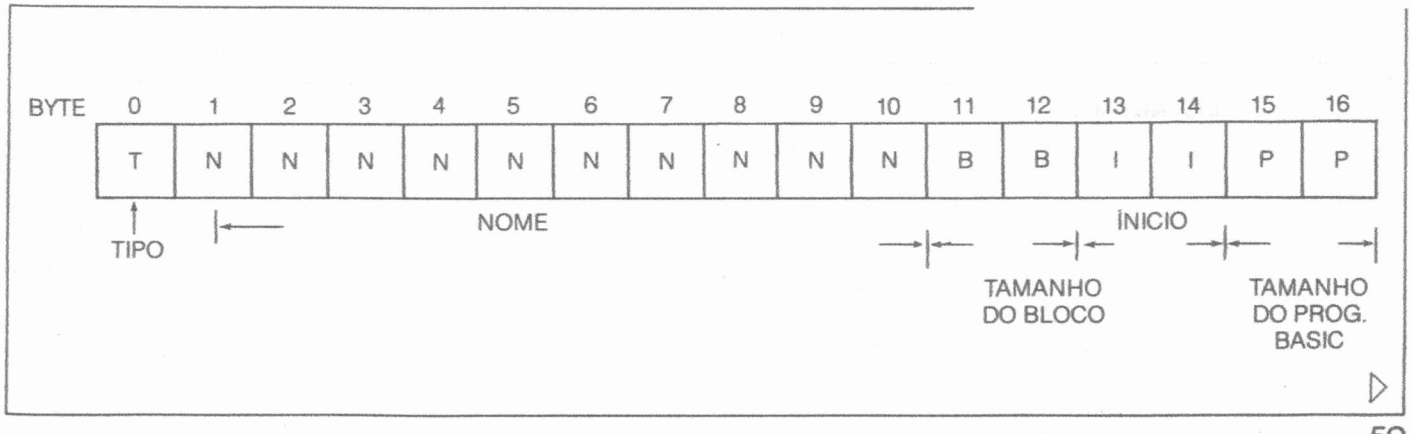
Simples não! Para usá-lo, "suje" a tela com qualquer coisa (exemplo: PRINT AT 10,10;"TELA 1"). Digite RAND USR 30000, não "dê" ENTER ainda, primeiro ligue o gravador (RECORD), e só depois dê ENTER.

Vamos agora "LOADar" a nossa tela:

```
30500 11 00 18    1d de,6912
30503 DD 21 00 40 1d ix,16384
30507 3E FF       1d a,255
30509 37          scf
30510 CD 56 05    call LOAD DATA
30513 C9          ret
```

Agora ligue o gravador (PLAY), logicamente, após rebobinar a fita) e só então "dê" RAND USR 30500. Aí está a sua tela!

Você deve estar curioso quanto aos 17 bytes do ***header.*** Veremos agora como criá-lo e lê-lo:

O HEADER

O ***header*** possui o seguinte formato: