

# MICROCOMPUTADOR CURSO PRÁTICO

TIRE O MÁXIMO DO MICRO EM 30 SEMANAS

PUBLICAÇÃO SEMANAL ILUSTRADA

Cr\$ 10.250

24



**MSX: padrão internacional • O jogo da centopéia**  
**Máquinas sob comando da mente**

*rioGráfica*

Depois de ler com atenção todo o fascículo, responda a estas perguntas. Veja a solução no próximo número, junto com um novo teste.

- 1) Em que fenômeno orgânico se baseiam o detector de mentiras e o "acoplamento mental" do homem ao computador?
- 2) Como se chamam os motores que produzem movimentos extremamente precisos, utilizados nos robôs e nas tartarugas mecânicas?
- 3) Qual o microprocessador adotado pelos micros padrão MSX?
- 4) Quais os dois primeiros micros padrão MSX introduzidos no Brasil?

## RESPOSTAS DO TESTE ANTERIOR

- 1) Numa comparação da visão humana com a robótica, os olhos são os sensores e o cérebro é o processador.
- 2) O dispositivo mais usado como "olho" robótico é a câmara de vídeo.
- 3) O processador onde se executam as operações aritméticas e lógicas é a ALU (Arithmetic and Logic Unit, "unidade lógica e aritmética").
- 4) Um sistema eficiente de mala direta deve ser utilizado em conjunto com um processador de texto e um gerenciador de banco de dados.



## APLICAÇÕES

Comando mental	477
O passo do robô	490



## PROJETOS DE PROGRAMAS

Centopéia faminta	480
-------------------	-----



## CHIPS & BYTES

Solução para compatibilidade	482
Força bruta	488



## RAIO X

Padrão MSX	485
------------	-----



## LINGUAGENS

Mudança de endereço	493
---------------------	-----



## PERFIL

Labo	496
------	-----

## PLANO DA OBRA

MICROCOMPUTADOR - CURSO PRÁTICO é uma coleção de 30 fascículos de periodicidade semanal. Os 29 primeiros terão, cada um, vinte páginas internas (miolo) e quatro capas, além de uma introdução de dezesseis páginas acompanhando o fascículo 1. O fascículo 30 incluirá o índice geral da obra. Os miolos, encadernados, constituem dois volumes de um curso prático de microcomputação, com seções indicadas por tarjas de diferentes cores. As duas primeiras capas são descartáveis; as 3.<sup>as</sup> e 4.<sup>as</sup> trazem programas de jogos. Encadernadas, formam um volume de cem páginas.

## COMO ENCADERNAR

As capas duras, incluindo as guardas, estarão à venda simultaneamente com os fascículos 10 (**Volume 1**) e 30 (**Volume 2 e Jogos**).

**Volume 1** - Deve ser encadernado com os elementos dispostos nesta ordem: guardas; frontispício e introdução (publicados com o fascículo 1); miolos dos fascículos de 1 a 15; e guardas.

**Volume 2** - Guardas; frontispício (que virá com o fascículo 30); miolos dos fascículos de 16 a 30; e guardas.

**Jogos** - Guardas; frontispício e sumário (que constituem as 3.<sup>as</sup> e 4.<sup>as</sup> capas do fascículo 30); 3.<sup>as</sup> e 4.<sup>as</sup> capas dos fascículos de 1 a 29; e guardas.

**rioGráfica**

**CONSELHO DE ADMINISTRAÇÃO:** Roberto Irineu Marinho; João Roberto Marinho; José Roberto Marinho; Oscar Neves; **Diretoria:** Oscar Neves; Danilo Esteves Costa; João J. Noro; Marcos Marchi; Nilo Sérgio de Almeida; **GRUPO EDUCAÇÃO E CULTURA - Divisão Comercial - Diretor:** Jaime Rodrigues; **Gerente de Planejamento:** João de Azeredo Coutinho; **Gerente de Vendas:** Rubens Barbosa; **Coordenadores de Promoção:** Paulo Cesar M. Seixas; Edgard Mello M. Neto; **Gerente de Circulação:** Norberto Martin; **Supervisor de Operações:** Abel Pereira Pinto; **Editorial - Diretor:** João J. Noro; **Editor-chefe:** Maurício

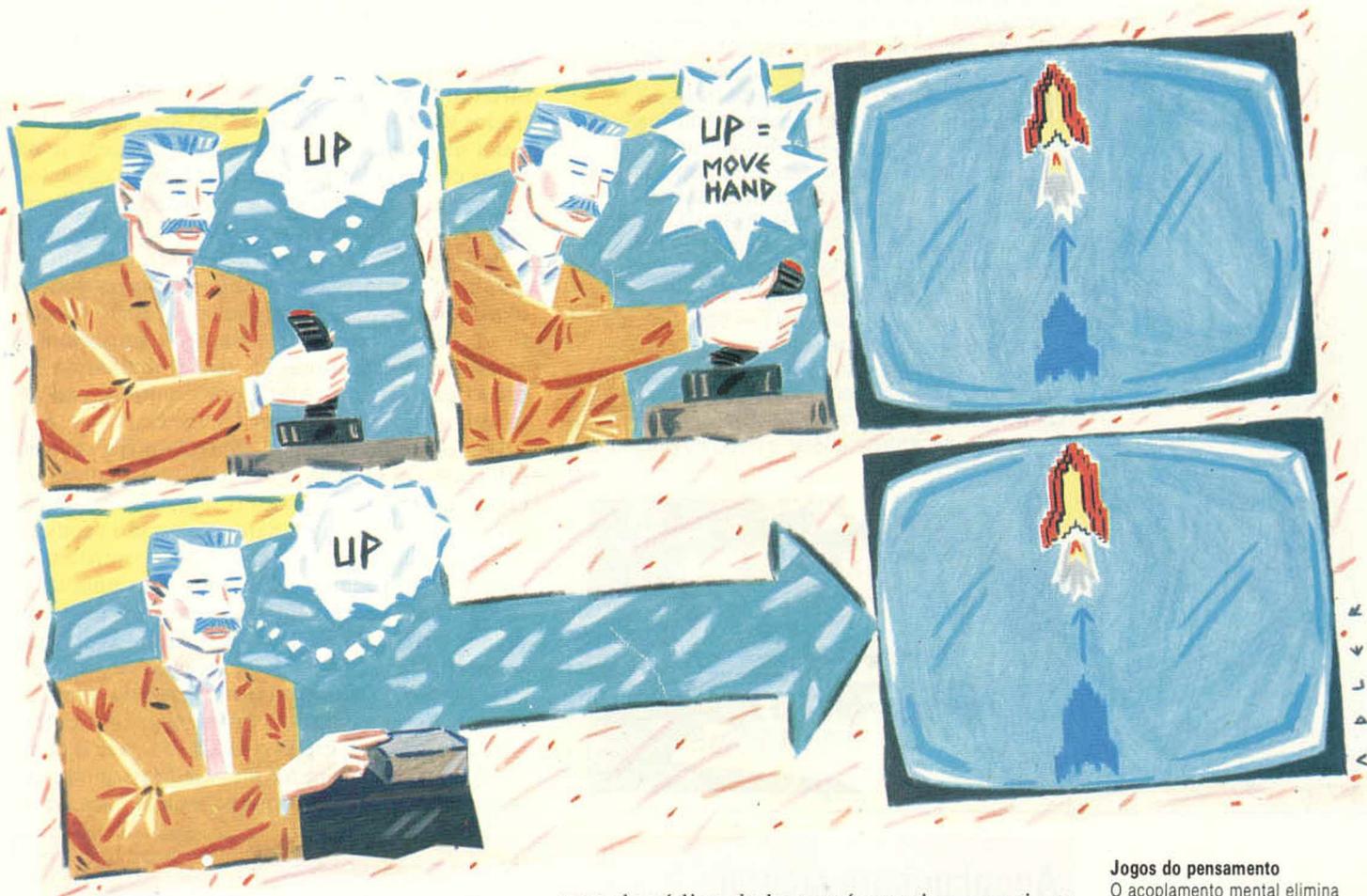
Rittner; **Diretor de Arte:** Adalberto Cornavaca; **Administração Editorial:** Heitor de Souza Paixão; **Editor:** Jesse Navarro; **Editor-assistente:** Claudio Marcondes; **Coordenadora de Revisão:** Isis Augusta Loyolla; **Revisores:** Cacilda Guerra; Luiz Vicente Vieira Filho; Maria da Graça Mendonça Couto; Sérgio Roberto Torres; **Chefe de Arte:** Bonifácio Duardes Miranda; **Chefe de Estúdio:** José Yuji Kuribayashi; **Assistentes de Arte:** Miguel Luis Escámez Simón; Edemilson Ribeiro da Silva; Wagner Celestino; **Colaboradores - Editores-assistentes:** Isa Mara Lando; Luiz Carlos Pizarro Marin; **Texto:** Carlos Eduardo Silveira Matos; José Antônio Arantes; Jussara Maturu; Paulo Brito; Solange A. Menezes; **Preparação de Texto:** Luiz Carlos Cardoso; **Tradução:** Luiz Roberto de Godoy Vidal; Maria Clara Cascato; Maria Ester Menezes Marinho; Fernando Dorfman Knijnik; Newton Roberval Eichenberg; **Pesquisa:** Stella Maria Quentel; **Assessoria Técnica:** Geraldo Coen; Gustavo José Ferreira Grubba; José Rubens Salles Toledo; Lauro de Lauro; Pierluigi Piazzì. As fotos não creditadas pertencem à obra original. Segue-se o crédito das demais: Labo; © APSIF Copenhagen, 1984; © Orbis Publishing Ltd., 1984; © Editora Rio Gráfica Ltda., 1985, para a língua portuguesa. **Composição:** Editora Rio Gráfica Ltda. e AM Produções Gráficas Ltda. **Impressão:** JBIG.

**NÚMEROS ATRASADOS** - Você poderá comprar os exemplares que faltam em sua coleção, pelo preço do último fascículo posto à venda, em sua banca de jornal preferida. Caso não consiga obtê-los, faça sua solicitação por carta endereçada diretamente à Editora Rio Gráfica Ltda., rua Itapiru, 1209, Rio Comprido, Rio de Janeiro, CEP 20251. O atendimento será feito por via postal. Nas cidades do Rio de Janeiro e de São Paulo, as compras poderão ser efetuadas pessoalmente nos seguintes endereços: Rio de Janeiro - Rua Itapiru, 1209, Rio Comprido; São Paulo - Rua Frei Caneca, 1152, Consolação.

Distribuidor exclusivo para todo o Brasil: Fernando Chinaglia - Rua Teodoro da Silva, 907, Rio de Janeiro. Distribuidor para Portugal: Electrolibr Lda. - Rua Prof. Reinaldo dos Santos, 1488, Lisboa.



# COMANDO MENTAL



**O controle de um computador pelo pensamento deixou de ser ficção científica. Conheça o fenômeno que fundamenta essa tecnologia e o dispositivo que acopla o sistema nervoso ao computador.**

Desde que surgiram os primeiros computadores, o teclado tem sido o principal dispositivo de entrada de informações. Talvez seja o método mais eficaz de se introduzir textos e, em menor grau, números — pelo menos até que se desenvolvam sistemas confiáveis de reconhecimento de voz e destinados a objetivos gerais. Entretanto, por mais adequado que seja para se fornecer textos à máquina, o teclado não é, necessariamente, o melhor dispositivo para se introduzirem outros tipos de informação. Se precisamos inserir dados direcionais com rapidez, como em vários jogos, damos preferência ao joystick ou ao paddle; no caso de dados posicionais, como numa seleção de opções num menu, o mouse, a caneta óptica ou a tela sensível ao toque mostram-se em geral mais apropriados; e a lei-

tora de código de barras é o mais conveniente para longos números ou códigos seriais.

Mas todos esses dispositivos apresentam um inconveniente: são indiretos. Nos jogos, o joystick é mais adequado que o teclado (para fazer a nave subir, por exemplo, empurramos a empunhadura para a frente), mas ele ainda representa um hiato entre o que pretendemos e o que de fato acontece. Por exemplo, pensamos “para cima”, traduzimos isso mentalmente como “empurrar para a frente” e só então empurramos. Para uma interação mais rápida e direta entre nós e o computador, seria preciso eliminar essa mediação: por que não apenas pensar “para cima” e obter do aparelho pronta resposta ao nosso pensamento?

Assim, jogaríamos Invasores do Espaço apenas ordenando mentalmente “Para cima”, “Para baixo”, “Vire”, “Fogo”, e assim por diante; ou, então, escreveríamos uma carta apenas pensando nas palavras. Claro que o processador de textos telepático ainda não existe, mas jogar Invasores do Espaço por controle mental já é uma realidade, graças à noção de “acoplamento mental”. Segundo esta, empregamos o pensamento — mais precisamente, as alterações fisiológicas resultantes das mudanças nos padrões de pensa-

## Jogos do pensamento

O acoplamento mental elimina o estágio mecânico intermediário da conversão dos impulsos do jogador em sinais significativos para o computador. Colocando usuário e máquina em relação direta, o acoplamento mental possibilita programas mais interativos e entradas mais rápidas.

## Notas mentais

A foto ilustra uma demonstração de um protótipo do sistema RGP funcionando com um computador Apple. Sistemas semelhantes já foram desenvolvidos para uso no IBM PC e no Commodore, mas é bastante provável que o software efetivo ainda leve mais tempo. Roger Dilts, presidente da Behavioral Engineering e autor de software para RGP, tem especial interesse na possibilidade de combinar o acoplamento mental com a NLP (Neuro-Linguistic Programming, "programação neurolíngüística"), um ramo da psicologia dedicado ao estudo do aprendizado. Com a utilização da técnica de RGP para monitorar o estado mental do usuário, o software pode medir o grau de tensão do mesmo e ajustar o andamento do programa. Roger Dilts acredita que os programas educacionais — para citar um exemplo — serão capazes de avaliar as reações emocionais dos estudantes ao material apresentado. Se este for demasiado complexo, o computador captará a tensão emocional provocada, permitindo alterar o ritmo do programa ou simplificar o tema.



mento — para controlar um dispositivo eletrônico, que funciona como uma interface para um computador, tal como o joystick ou outro dispositivo de entrada.

O acoplamento mental baseia-se no conhecido fenômeno RGP (Reação Galvânica da Pele), pelo qual mudanças no estado emocional se manifestam na condutividade elétrica da pele. Eletrodos são colocados na pele do usuário e ligados a um medidor de resistência. Introduzem-se os sinais do medidor no micro através do conector de expansões, e um software especial os interpreta e processa. A empresa Behavioral Engineering (Engenharia Comportamental), da Califórnia, baseou tanto jogos como softwares aplicativos nessa técnica, inclusive uma versão simplificada do Invasores do Espaço. Importantes empresas de computação vêm trabalhando há anos nesse setor, a Atari, por exemplo, mas só recentemente alcançaram algum êxito.

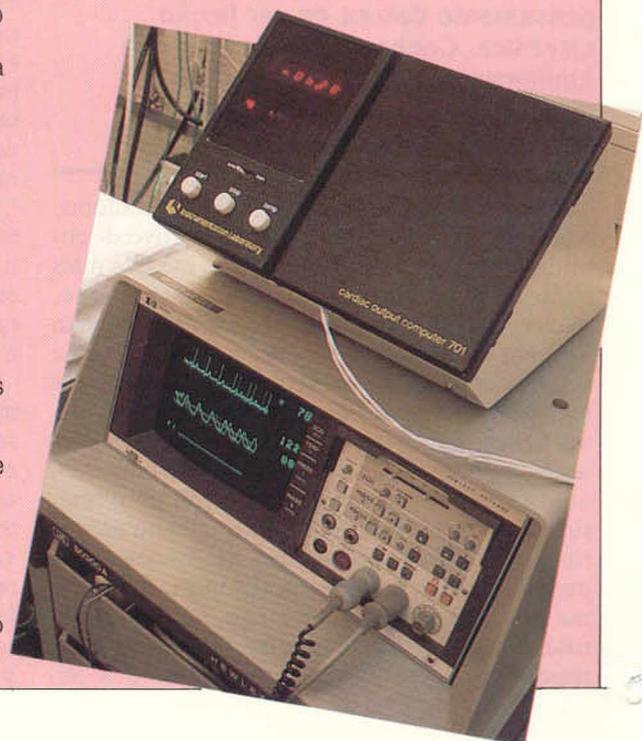
No jogo convencional, controlamos a nave na órbita de um planeta. O objetivo é atingir naves inimigas sem que a nossa o seja, e sem que ela se espatife contra o planeta. Na versão produzida pela Behavioral Engineering, controlamos apenas a altitude da nave, mas com uma diferença: pelo pensamento! A empresa projetou uma interface de RGP para o Apple II em que o usuário tem apenas de colocar os dedos indicador e médio num aparelho parecido com um mouse. Este, medindo a resistência elétrica entre os dois dedos, envia o resultado para o computador. O software foi planejado para que um aumento na tensão comande a nave para cima,

## Acoplamento mental

Esse princípio baseia-se num fenômeno conhecido por três nomes alternativos: RGP (Reação Galvânica da Pele); RPG (Reflexo Psicogalvânico); e RED (Reflexo Eletrodérmico). Preferimos adotar o primeiro termo. A RGP refere-se a mudanças na condutividade elétrica da pele que correspondem a alterações no estado emocional de uma pessoa. Experimentos mostram que, quanto mais tensa uma pessoa, mais baixa se torna a resistência elétrica de sua pele. A aplicação mais conhecida da RGP está nos polígrafos, os chamados detectores de mentiras.

Embora se observe o fenômeno da RGP em animais e seres humanos desde o século XIX, pouco se conhece sobre sua causa. Segundo a teoria inicial, o suor produzido pela excitação ou ansiedade age como condutor eletrolítico, baixando assim a resistência da pele. Experiências mais recentes, porém, lançaram dúvidas sobre essa teoria tão simplista. Seja qual for a razão do efeito, sabe-se que a RGP relaciona-se diretamente ao grau de tensão no sistema nervoso simpático. Este depende do sistema nervoso central e, portanto, do cérebro. Daí ser possível o controle de um computador pelo pensamento. Alterações na atividade cerebral acarretam mudanças no estado do sistema nervoso central; isso modifica o estado do sistema nervoso simpático, resultando

em variação na resistência elétrica da pele. E a alteração numa corrente elétrica é a base do funcionamento de qualquer dispositivo de entrada.





e uma redução a leve para baixo. O objetivo é controlar esses movimentos e alinhar a nave com os alvos que se aproximam.

O termo “acoplamento mental” talvez seja enganoso, pois essa ligação se dá não pela mente, mas pelo sistema nervoso. Eis a questão: se ambos são interdependentes, faz sentido separá-los? Muitas pessoas mostram-se capazes, após cerca de 20 minutos, de exercer um razoável grau de controle, quase sempre sem saber como o conseguem. Uma, de forma consciente, tensionam e relaxam levemente o corpo; outras simplesmente pensam “Para cima” ou “Para baixo” e deixam o sistema nervoso fazer o resto.

O acoplamento mental tem outras aplicações mais práticas. Uma pessoa totalmente parálitica ainda conta com o recurso de gerar, de modo consciente, efeitos de RGP, embora não controle os músculos. Um dispositivo de RGP ligado a um paralítico foi acoplado com sucesso a um robô Topo, através de um computador Apple, dando a essa pessoa a possibilidade de controlar o robô. Uma outra aplicação prática estaria no espaço sideral, sob gravidade zero. É extremamente difícil operar controles mecânicos sem a gravidade para contrabalançar a energia aplicada. Um dispositivo de RGP poderia ser o substituto perfeito para muitos controles mecânicos.

## Um software que capta o ânimo

Entre as aplicações exóticas está a captação da disposição de ânimo. Os softwares educacionais, em particular, poderiam se beneficiar com a retroalimentação do estado interior do usuário por meio da RGP. O indivíduo usaria em torno do pulso uma faixa com eletrodos, que o software calibraria no estado de relaxamento normal. Se, num dado momento, o aparelho registrasse acentuado aumento de tensão, significando estar o usuário em dificuldades, ele ajustaria o andamento do programa. Softwares comerciais seriam capazes de detectar a tensão mental de usuários — e de sugerir um descanso.

A rapidez do desenvolvimento dessa tecnologia depende não só da criação de dispositivos mais precisos e de respostas mais rápidas, como também de nossa habilidade em interpretar os efeitos. O primeiro problema está em que a RGP ocorre 2 segundos após o evento e leva de 2 a 10 para desaparecer. Os dispositivos atuais superam em parte o problema, medindo o ritmo de mudança da reação e não sua intensidade. O segundo problema está na imprecisão das nossas deduções a partir dos dados fornecidos pela RGP. Uma queda brusca na resistência elétrica da pele indica tensão ou estresse, mas não se pode precisar se a causa é agradável ou dolorosa.

Como se vê, apesar de todos os problemas, a RGP descortina possibilidades entusiasmantes. E a Atari não é a única a levá-la a sério: a Commodore ofereceu a respeitável soma de 2 milhões de dólares à Behavioral Engineering pelos direitos do dispositivo e do software de RGP — devidamente recusados!



## O detector de mentiras

Numa investigação ou julgamento, um dos maiores problemas enfrentados pela polícia e sistema judicial é saber se um suspeito ou uma testemunha está mentindo. Embora um perito possa detectar pistas mínimas (como alterações na coloração da pele e na respiração, por exemplo), a medição desses fatores não está definida nem catalogada — e muito menos é admissível como prova. Por isso se pesquisa no sentido de desenvolver meios objetivos de distinguir entre declarações falsas e verdadeiras. Um dos resultados nesse sentido foi o polígrafo, ou detector de mentiras.

O polígrafo, na verdade, consiste apenas num medidor de resistência elétrica. Supõe-se que o indivíduo sofrerá aumentos significativos de tensão ao dizer uma mentira ou ouvir palavras associadas ao crime, e que isso se revelará por uma queda na resistência elétrica da pele.

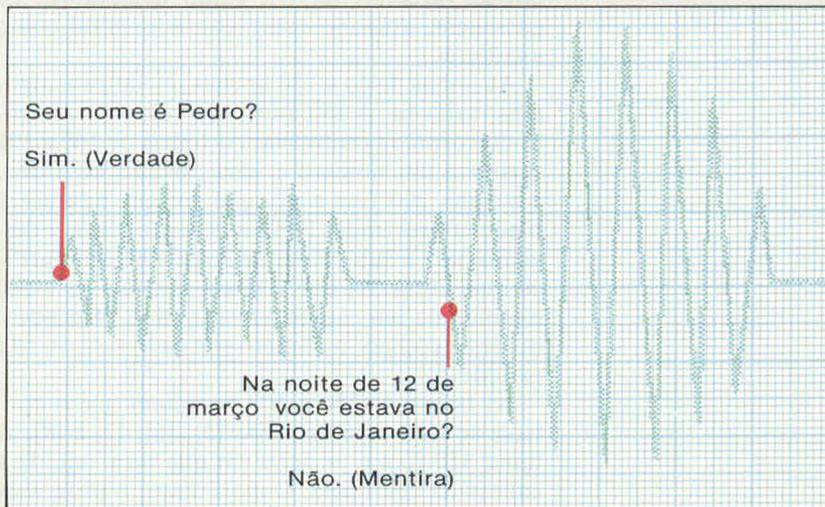
O polígrafo vem sendo objeto de muita controvérsia. É defendido pela polícia, nos EUA em especial, mas seus críticos alegam que o nosso conhecimento sobre o fenômeno RGP é insuficiente para uma interpretação segura e coerente. Além do fato de indivíduos diferentes reagirem de forma também diferente, sejam eles culpados ou inocentes.

### Na conquista de um emprego

Às vezes se usa a RGP na avaliação de candidatos a empregos, alimentando um detector de mentiras na análise das respostas do entrevistado. Na foto acima, o interessado recebe os resultados do teste que acabou de fazer.

### Mentiras inofensivas

O detector de mentiras mede mudanças de RGP causadas pelas reações emocionais do indivíduo. Mas é pouco provável que esse aparelho forneça respostas tão precisas quanto as que vemos abaixo. Reações emocionais variam conforme a sensibilidade do indivíduo a certos temas (se os acha constrangedores, por exemplo) e não dependem apenas da falsidade ou veracidade das respostas.



# CENTOPÉIA FAMINTA

**Diversos programas criados para o TK 85 são adaptáveis ao TK 90X. Analisamos aqui duas versões de um mesmo jogo para esses micros. Aproveite para fazer suas próprias modificações.**

Nosso programa é um jogo de ação onde você dirige uma centopéia esfomeada, em busca de comida, dentro de um jardim. O controle se faz com as teclas [5], [6], [7] e [8] (as setas de controle do cursor), e você deve evitar que a centopéia “morda” a si mesma ou os tijolos que delimitam o jardim. Se isso acontecer, ela morrerá.

Os alimentos são pequenas larvas (números) que surgem ao acaso na superfície do jardim. Podem ser mais ou menos gordas: quanto mais gorda, maior o número por ela representado, e cada número que a centopéia devora a faz crescer proporcionalmente em segmentos.

Mas o jardim permite que a centopéia tenha no máximo quatrocentos segmentos. Como ela já “nasce” com um, significa que poderá crescer até 399. Seu objetivo final, então, consiste em fazer com que todo o jardim seja ocupado pela centopéia.

## A versão para o TK 85

As linhas 20, 30 e 40 desenham na tela os “tijolos”. As variáveis L e C armazenam a posição do último segmento da centopéia, e as variáveis Y e X armazenam a posição da cabeça. O string A\$ armazena os movimentos que a cauda deve realizar para chegar à posição da cabeça. As outras variáveis servem como indicadores de passagem, atuando como contadores ou como auxiliares.

As linhas de 40 a 130 apenas inicializam algumas variáveis. A 150 mostra na tela um segmento do corpo da centopéia. As linhas de 160 a 190 sorteiam um número e uma posição no vídeo, e a linha 200 fixa a posição em que aparecerá o número sorteado. Feito isso, o programa é desviado para a sub-rotina da linha 490, que verifica qual caractere existe nessa posição do vídeo e armazena seu código na variável E. Dependendo desse código, o número sorteado será ou não exibido.

As linhas de 250 a 300 permitem que o controle se faça com as teclas [5], [6], [7] e [8]. A linha 310 fixa a posição em que será exibida a cabeça da centopéia. Depois disso, o programa se desvia para a sub-rotina da linha 490.

As linhas 330 e 340 verificam se a centopéia simplesmente andou, se ela mordeu um tijolo, se devorou a si mesma ou a uma larva. Em cada caso, algumas das linhas entre 350 e 480 são executadas.

As linhas 350, 355 e 360 são executadas se a centopéia morre. Já as 370, 380 e 390 são executadas no caso de ela devorar alguma larva. A linha 400 exhibe a cabeça no vídeo e a 420 apaga a cauda. O movimento da cauda é produzido nas linhas 430 e 440.

A variável B indica se vai ou não haver crescimento, Z conta quantos segmentos a centopéia aumentou, e W armazena o total de alimentos na superfície do canteiro.

As linhas 410 e 470 estão relacionadas com o conteúdo de B, e as 460 e 480 desviam incondicionalmente o programa para a linha 150, onde um outro segmento do corpo da centopéia é exibido.

O programa deve ser executado no modo slow, extremamente lento quando o micro tem mais de 2 Kbytes de RAM. Isso ocorre porque, quando o aparelho possui menos de 3,25 Kbytes de RAM disponíveis, gerencia o armazenamento do vídeo de uma forma mais econômica.

Caso seu micro tenha 16 Kbytes, para aumentar-lhe um pouco mais a velocidade, você poderá “enganá-lo” alterando o valor da variável do sistema que armazena o endereço do último byte existente na RAM: a RAMTOP ou RTP. Para isso, basta comandar diretamente (antes de digitar o programa):

```
POKE 16388,200
```

e

```
POKE 16389,11
```

Isso fará o micro “pensar” que está apenas com cerca de 3 Kbytes de RAM, tornando-se um pouco mais rápido. Quanto mais à esquerda estiver a imagem no vídeo, menos memória será necessária para o armazenamento. Devido a isso, o programa para o TK 85 (e similares) constrói a moldura descentralizada, mais à esquerda.

Devido à lentidão, impõe-se que os números sorteados na linha 170 (larvas) sejam sempre superiores a 3, para que o crescimento se acelere mais.

## A versão para o TK 90X

O TK 90X possui muitos recursos visuais e sonoros que podem ser usados em quase todos os tipos de programas. Em nosso caso específico, esses recursos são essenciais para transformar o programa Centopéia (lento e um pouco pobre no



TK 85) num programa ao nível de qualquer bom videogame.

Uma grande vantagem do TK 90X sobre o TK 85 é patente: a velocidade de execução do programa. O TK 85 possui duas velocidades de processamento. No modo slow, o vídeo se apresenta permanentemente ao usuário, o que torna a execução lenta. No modo fast, a execução acontece mais rápida, porém o vídeo não é mostrado.

Num jogo de ação, precisa-se utilizar o modo slow. O TK 90X funciona sempre mostrando o vídeo, porém com uma velocidade de processamento próxima à do fast.

Outro fator que também torna o programa lento no TK 85 é a utilização da sub-rotina da linha 490, necessária para checar a posição do vídeo em que ocorrerá a próxima impressão. No TK 90X existe uma instrução extremamente simples, a SCREEN\$, utilizada nas linhas 200 e 310, que substitui com vantagem a sub-rotina usada no TK 85.

O uso das cores também é muito simples. Os comandos BORDER, PAPER e INK são usados na linha 10 de modo a gerar uma tela branca com caracteres pretos e uma moldura vermelha.

Acrescentamos som ao programa com o comando SOUND. A linha 250 gera um sinal cada vez que o teclado é verificado, e a 350 gera um sinal de "morte" da centopéia (sua marcha fúnebre!).

Outro grande melhoramento possibilitado pelo TK 90X é o uso de caracteres determinados pelo usuário. Foram definidos três deles: um para os "tijolos" do jardim (linha 30), um para o corpo da centopéia (linha 150) e outro para sua cabeça (linha 400). Os três estão ilustrados em detalhes na figura para que possam ser reproduzidos.

Por fim, como fica evidente, muitas vezes utilizamos linhas com multiinstrução, isto é, várias instruções numa mesma linha do programa.

Como no TK 90X não temos problemas com a velocidade de processamento, as larvas são representadas por qualquer número entre 1 e 9. Além de muito mais rápido, mais estético e mais interessante, o programa no TK 90X também fica mais compacto.

## Outras opções

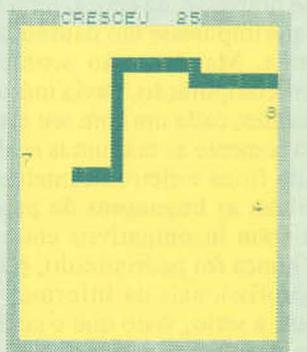
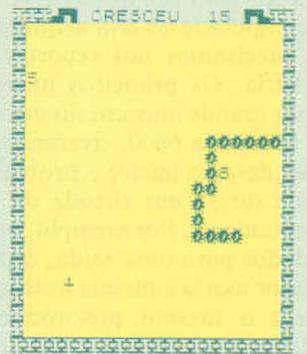
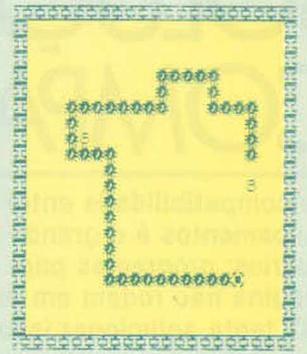
Existem várias maneiras de se produzir um programa como este; utilizamos apenas uma delas. No caso específico do TK 85, certamente não é a melhor (seria muito mais rápido imprimir no vídeo, colocando, por meio do comando POKE, caracteres diretamente na memória de vídeo).

No TK 90X, entretanto, a lógica usada é bastante viável (nesse caso, acabaria sendo bem mais complicado utilizar o POKE para colocar caracteres diretamente no vídeo).

Constituiria um exercício bastante instrutivo tentar tornar o programa para o TK 85 mais rápido usando POKE ao invés de PRINT. Ainda mais instrutivo seria adaptar esse programa para o TK 90X, dada a sua complexidade. Fica como sugestão para um fim de semana chuvoso.

```

5 REM PARA O TK90X
6 BORDER 2: PAPER 7: INK 0
7 FOR A=0 TO 21
8 PRINT AT A,0;" ";TAB 21;" ";
9 AT 0,A;" ";AT 21,A;" ";
10 NEXT A
11 LET L=11: LET C=L: LET Y=L
12 LET X=C: LET A$="" : LET C$=""
13 LET Z=1: LET B=0: LET W=B
14 PRINT AT Y,X;" "
15 IF W>15 THEN GOTO 250
16 LET S=INT (1+9*RAND)
17 LET T=1+20*AND
18 LET U=1+20*AND
19 GOTO 200
20 IF SCREEN$ (T,U)<>" " THEN
21 GOTO 130
22 PRINT AT T,U;S
23 LET W=W+S
24 LET D$=INKEY$: SOUND .01,25
25 IF D$<>" " THEN LET C$=D$
26 IF C$<>"5" AND C$<>"8" AND
27 C$<>"7" AND C$<>"8" THEN GOTO 25
28
29 LET A$=A$+C$
30 LET Y=Y+(C$="6")-(C$="7")
31 LET X=X+(C$="8")-(C$="5")
32 PRINT AT Y,X;
33 GOSUB 490
34 IF E=0 THEN GOTO 400
35 IF E<>"8" AND E<>"128" THEN GOT
36 PRINT AT 0,4;"CRESCEU ";Z-
37
38 PAUSE 200
39 RUN
40 LET B=B+E-20
41 LET Z=Z+E-20
42 LET U=W-E+20
43 PRINT " "
44 IF B>0 THEN GOTO 470
45 PRINT AT L,C;" "
46 LET L=L+(A$(1)="6")-(A$(1)="7")
47 LET C=C+(A$(1)="6")-(A$(1)="5")
48 IF LEN A$>0 THEN LET A$=A$(
49 TO
50 GOTO 150
51 LET B=B-1
52 GOTO 150
53
54 REM PARA O TK85
55
56 CLS
57 FOR A=0 TO 21
58 PRINT AT A,0;" ";TAB 21;" "
59 AT 0,A;" ";AT 21,A;" "
60 NEXT A
61 LET L=11
62 LET C=L
63 LET Y=L
64 LET X=C
65 LET A$=""
66 LET Z=1
67 LET B=0
68 LET W=B
69 PRINT AT Y,X;" "
70 IF W>15 THEN GOTO 250
71 LET S=INT (4+6*AND)
72 LET T=1+20*AND
73 LET U=1+20*AND
74 PRINT AT T,U;S
75 GOSUB 490
76 IF E<>0 THEN GOTO 180
77 PRINT " "
78 LET W=W+S
79 IF D$<>" " THEN LET C$=D$
80 IF C$<>"5" AND C$<>"8" AND
81 C$<>"7" AND C$<>"8" THEN GOTO 25
82
83 LET A$=A$+C$
84 LET Y=Y+(C$="6")-(C$="7")
85 LET X=X+(C$="8")-(C$="5")
86 PRINT AT Y,X;
87 GOSUB 490
88 IF E=0 THEN GOTO 400
89 IF E<>"8" AND E<>"128" THEN GOT
90 PRINT AT 0,4;"CRESCEU ";Z-
91
92 PAUSE 200
93 RUN
94 LET B=B+E-20
95 LET Z=Z+E-20
96 LET U=W-E+20
97 PRINT " "
98 IF B>0 THEN GOTO 470
99 PRINT AT L,C;" "
100 LET L=L+(A$(1)="6")-(A$(1)="7")
101 LET C=C+(A$(1)="6")-(A$(1)="5")
102 IF LEN A$>0 THEN LET A$=A$(
103 TO
104 GOTO 150
105 LET B=B-1
106 GOTO 150
107
108 LET E=PEEK (PEEK 16398+256+
109 PEEK 16399)
110 RETURN
    
```





# SOLUÇÃO PARA COMPATIBILIDADE

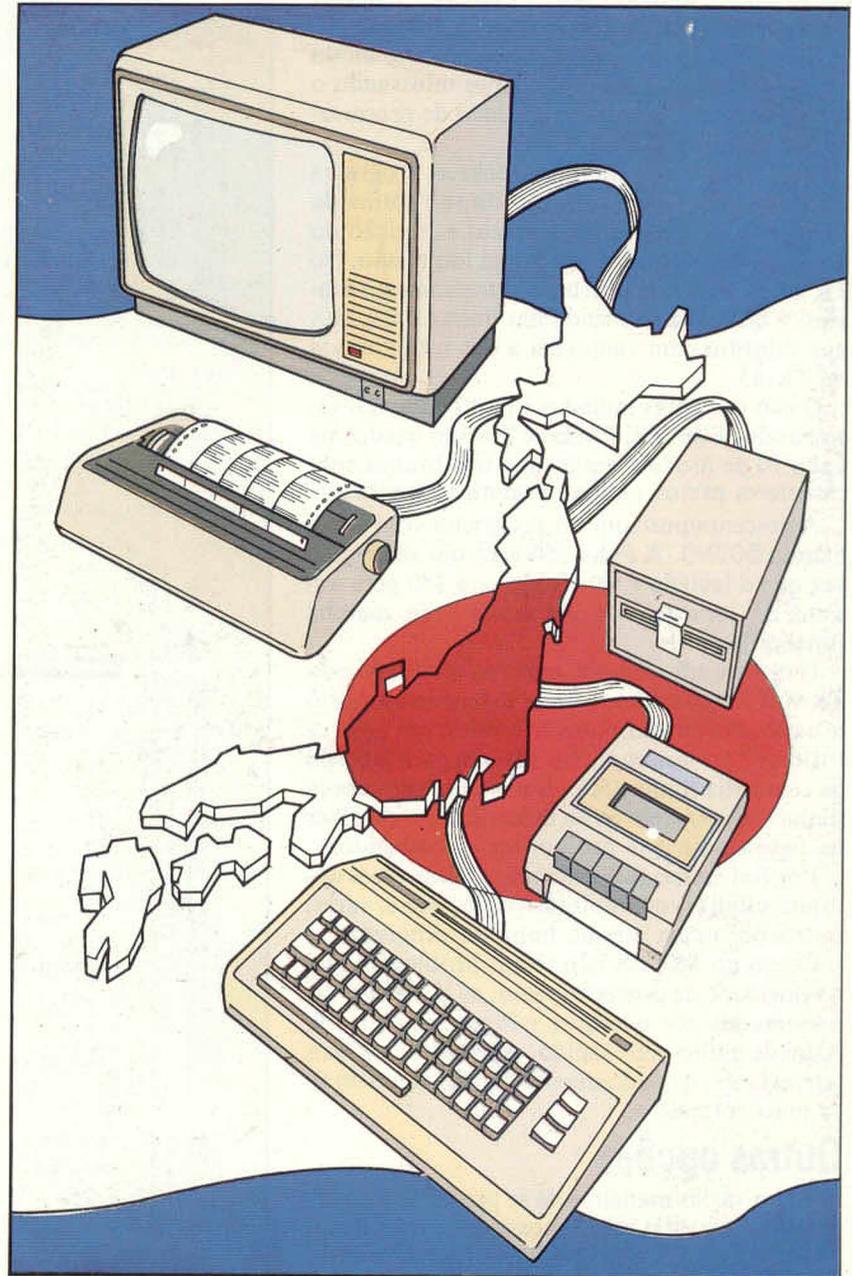
**A incompatibilidade entre os diversos equipamentos é o grande problema dos usuários: programas para determinada máquina não rodam em outra. O padrão MSX tenta solucionar isso.**

Para entender por que as diversas marcas de microcomputadores têm arquiteturas tão diferentes, precisamos nos reportar à história dessa indústria. Os primeiros microprocessadores a causar grande impacto no mercado, o Intel 8080 e o Motorola 6800, tiveram suas instruções fixadas desde o início; o problema da compatibilidade surgiu em virtude da versatilidade dos processadores. Por exemplo, para enviar um byte de dados para uma saída, qualquer microcomputador usaria a mesma instrução, desde que utilizasse o mesmo processador. Mas a saída poderia ter qualquer endereço, entre centenas ou milhares.

Os primeiros micros foram produzidos por diversos fabricantes "de fundo de quintal"; dessa forma, não houve uniformidade sobre a localização das portas de entrada e saída no mapa da memória. Um fabricante poderia escolher o endereço 255 para a saída de impressora paralela, enquanto outro escolheria o endereço 254 para o mesmo fim. Essa situação tornou-se ainda mais caótica com o advento dos chips controladores de telas gráficas e de efeitos sonoros. Qualquer programa que utilizasse os recursos especiais de um determinado computador certamente não rodaria em outro, a menos que sofresse consideráveis modificações.

Se houvesse, na época, uma indústria mais forte, que impusesse um padrão, a situação seria diferente. Mas isso não aconteceu: no início da microcomputação, havia inúmeros pequenos fabricantes, cada um com seu estilo e seus padrões. Não somente as máquinas acabavam se apresentando física e eletronicamente diferentes, como também as linguagens de programação utilizadas eram incompatíveis entre si. Mesmo o BASIC nunca foi padronizado, pois na década de 70 os profissionais da informática não o levaram muito a sério, visto que o consideravam linguagem para principiantes.

No final da década de 70 e início da de 80, os microcomputadores se desenvolveram a enorme velocidade. Os projetos pioneiros — o Apple, por exemplo — incorporaram refinamentos como gráficos e cores; mas para pôr em prática essas inovações, os fabricantes desenvolveram suas próprias versões do BASIC, que assim proliferaram. Se por um lado essa expansão possibilitou



ampla liberdade de escolha ao consumidor, por outro gerou muitas frustrações aos proprietários, aos fabricantes de micros e aos projetistas de software. O orgulhoso proprietário de um SORD ficaria desesperado para obter o último jogo da moda; mas, se o programa fosse escrito para o Sinclair, ele teria de esperar muito pela versão compatível com sua máquina.

O fabricante que tem um novo micro a lançar sabe que existe pouquíssimo software para seu equipamento até que haja um número substancial de usuários. Mas, com pouco ou nenhum

## Rumo ao sucesso

O padrão MSX é a rota do Japão para entrar nos mercados mundiais de computação. Se obtiver sucesso, poderá dominar esse mercado, da mesma maneira que dominou as áreas de aparelhos de som e das câmaras fotográficas.



software disponível, o potencial de vendas de sua máquina é muito limitado, pondo em risco todo o investimento feito no desenvolvimento do produto.

Se você cria software comercial, suas vendas limitam-se (na melhor das hipóteses) às pessoas que possuem o modelo de computador para o qual se destina o programa. Suponhamos que você crie um jogo de aventuras para o TK 90X chamado Masmorras do Inferno, com personagens como o Fantasma Invisível, o Carcereiro Sádico e incontáveis Antros do Terror para apanhar os incautos. A equipe de pesquisa de mercado afirma ser imenso o potencial de vendas, desde que se lance o produto a um certo preço e se vendam, no mínimo, 65.000 cópias. Entretanto, o número de proprietários do TK 90X não basta para garantir a vendagem; é necessário pelo menos mais uma versão. O custo de produção de novas versões, para, digamos, o CP 400 e o Apple, aumentaria o valor unitário em 20%; a esse preço, as vendas ficariam abaixo de um nível economicamente viável. É esse o dilema que frustra muitos projetistas de software em potencial.

O problema da compatibilidade de software não foi ignorado pela indústria de computadores. A óptica individualista do Ocidente não favorece a padronização; mas, no Extremo Oriente, os fabricantes preferem que as coisas sejam mais sistemáticas, organizadas e uniformes, especialmente quando isso se transforma em lucro.

A ASCII/Microsoft está tentando organizar o caos. A companhia resulta de uma fusão da Microsoft Corporation com a ASCII, uma bem-sucedida editora japonesa de revistas populares. Como linha secundária, a ASCII também publica software comercial e, quando a Microsoft se impõe o desafio de entrar no "impenetrável" mercado japonês, a ASCII foi a primeira opção de parceria num empreendimento conjunto: a Microsoft possuía a tecnologia e a ASCII a experiência de marketing.

A Microsoft Corporation construiu sua reputação sobre a versão do BASIC mais aceita como padrão, o MBASIC, adotado por fabricantes de micros do mundo todo. Mesmo assim, não havia garantias de que um programa em MBASIC rodasse em todos os computadores que usassem essa linguagem, pois sempre que houvesse recursos especiais haveria incompatibilidade de hardware.

O BASIC da Microsoft foi vendido para numerosos fabricantes japoneses através da ASCII/Microsoft. Mas isso também não resolveu o problema de compatibilidade de hardware e software. Como solução, a ASCII/Microsoft criou, em cooperação com os principais fabricantes japoneses, um padrão de aceitação internacional. O resultado desses esforços foi o chamado padrão MSX, que inclui requisitos básicos de hardware (centrados no microprocessador Z80 e alguns outros chips) e também uma linguagem padronizada. No final de 1985, os mi-

cro MSX já faziam bastante sucesso no mercado internacional; no Brasil, os pioneiros foram o Gradiente Expert e o Sharp Hot-Bit.

## Especificações do MSX

O BASIC MSX assemelha-se muito ao MBASIC (da Microsoft), mas com várias melhorias que aproveitam as modernas possibilidades gráficas e sonoras. Seguem-se os novos comandos: SCREEN, para especificar o modo de tela, o tamanho dos sprites, o bip das teclas, a velocidade de transmissão de cassete e as opções da impressora; LOCATE, que posiciona os caracteres na tela; COLOUR seleciona uma das dezesseis cores de linha e de fundo; PUT SPRITE estabelece atributos dos sprites; CIRCLE faz círculos e elipses; DRAW executa desenhos; LINE traça retas entre coordenadas especificadas; e PAINT preenche figuras com uma determinada cor. Há também o comando KEY, para alocar strings a teclas de função. Outros comandos permitem introduzir valores na RAM de vídeo (VPOKE), inscrever valores nos registradores do chip de efeitos sonoros (SOUND) e controlar o motor do gravador cassete (MOTOR).

O MSX, entretanto, envolve não apenas a padronização do software, mas também especificações quanto ao hardware. A CPU é um processador Z80 funcionando a 3,58 MHz. Deve haver no mínimo 32 Kbytes de ROM para armazenar o software MSX, e no mínimo 8 Kbytes de RAM. Não há limite máximo para a capacidade de ROM e de RAM. Um computador MSX deve incorporar um chip controlador de vídeo TMS9918A, da Texas Instruments (ou equivalente), e um AY38910—um chip gerador de som com três vozes. O vídeo deve apresentar 24 linhas, com 32 ou 40 colunas. No momento, não há recursos para apresentação em 80 colunas. Quanto à resolução, exige-se 256 x 192 pixels.

Escolheu-se a fita cassete como meio básico de armazenamento de programas e dados; ela deve usar o sistema de codificação FSK, com velocidade de transmissão de 1.200 ou 2.400 bits por segundo. O teclado também é padronizado, e inclui teclas de função e caracteres gráficos. No Brasil, a Gradiente optou por dotar o Expert com o gerador internacional de caracteres, que permite grafar todas as letras e acentos da língua portuguesa da mesma forma que nas máquinas de escrever.

Uma entrada para cartuchos de ROM permite a utilização de softwares aplicativos. Há um conector de cinquenta pinos para entrada e saída, e interface para dois joysticks.

Os formatos dos discos também são padronizados, assim como o sistema operacional em disco, o MSX-DOS, que equivale funcionalmente ao MS-DOS e permite a leitura de arquivos de dados gravados com esse sistema. Parece ser compatível, ainda, com o popular sistema operacional CP/M 2.2. Também se estabeleceram formatos para os disquetes de 3 1/2 (Sony e Gradiente), 5 1/4 e 8 polegadas.



# Arquitetura do MSX

**CPU**  
O popular microprocessador Z80 é a base de todas as máquinas MSX.

**Dispositivos adicionais**  
As máquinas MSX podem incorporar qualquer dispositivo extra que o fabricante queira incluir.

**Chip de som**  
Um 8910 proporciona som com efeitos especiais.

**Gravador cassette**  
As máquinas MSX operam com gravadores comuns em alta velocidade: 1.200 ou 2.400 bauds.

**Teclado**  
Todas as máquinas MSX têm as mesmas teclas, embora o posicionamento possa variar.

**Entrada para cartucho**  
Padronizada, permite a inserção de qualquer cartucho MSX.

**Unidade de disco**  
O MSX inclui especificações para discos que utilizam um formato padrão.

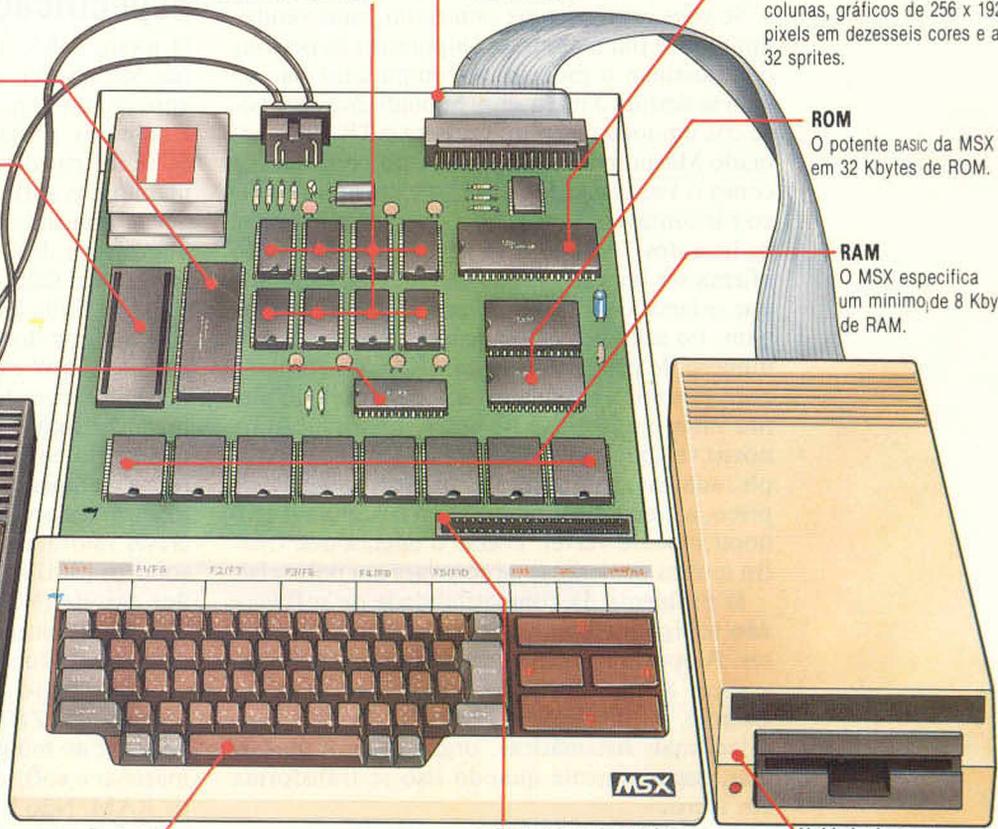
**RAM de vídeo**  
Separada em alguns sistemas, libera áreas de memória para os programas em BASIC.

**Conector de expansão**  
Todas as máquinas têm conector de expansão padronizado.

**Chip de vídeo**  
Um Texas 9918 ou equivalente proporciona a apresentação na tela, com 24 linhas de 32 colunas, gráficos de 256 x 192 pixels em dezesseis cores e até 32 sprites.

**ROM**  
O potente BASIC da MSX reside em 32 Kbytes de ROM.

**RAM**  
O MSX especifica um mínimo de 8 Kbytes de RAM.



Isso tudo significa que programas escritos para uma máquina MSX e armazenados em qualquer disco certamente rodarão sem problemas em qualquer outro computador da linha MSX, com plena utilização de seus recursos gráficos e sonoros. As vantagens são óbvias, tanto para o fabricante como para o consumidor.

O padrão MSX corre apenas alguns riscos. O primeiro é que qualquer "padrão", uma vez adotado, não conseguirá incorporar as inovações da área. Se, por exemplo, surgir no mercado um novo chip controlador de vídeo com recursos muito superiores, os programas MSX não poderão aproveitá-lo e deixarão o campo aberto aos concorrentes capazes de utilizá-lo.

O segundo risco está nos microprocessadores de 8 bits, dos quais o mais popular é o Z80, pois por princípio são incapazes de endereçar diretamente mais de 64 Kbytes de memória principal; também não podem processar de uma só vez dados de valor superior a 256.

À época do lançamento do MSX, parecia provável que nos próximos dez anos o mercado de micros passasse a ser dominado pelos microprocessadores de 16 bits. Se o padrão MSX conseguisse o sucesso esperado, sua consequência

mais positiva seria a de alertar os fabricantes que a padronização deve vir cedo — e não tarde — no projeto de qualquer inovação.

O MSX poderia facilitar a venda imediata dos produtos com base no Z80, mas dificilmente isso seria vantajoso a longo prazo. Seu real impacto estava em convencer o resto do mundo de que a padronização é importante.

No campo dos micros de 16 bits, a IBM provou que "contra a força não há argumentos", ao impor, com seu computador pessoal, na prática, um padrão. O MSX seria capaz de fazer o mesmo para os micros de 8 bits? Até fins de 1985, ele foi apoiado por muitos fabricantes, como a Yamaha, JVC, Hitachi, Sony, Sanyo, National, Pioneer, Canon, Fujitsu e Mitsubishi do Japão, a americana Spectravideo, a coreana Daewoo e Gradiente e Sharp no Brasil. Contudo, nenhum fabricante europeu entrara para o clube. Somente o tempo e a reação do mercado diriam, nos próximos anos, se havia necessidade de mais padronização, ou do tipo de inovação individualista que se espera de empresários como os pioneiros da computação, que, com seu engenho, criaram em precárias condições os primeiros microcomputadores.

**Convenções do projeto**  
Para que os programas e dispositivos adicionais sejam compatíveis com todos os sistemas MSX, o hardware de um micro MSX segue regras estritas. Uma vez que a máquina se conforma à configuração básica mostrada aqui, os projetistas podem acrescentar dispositivos extras.



# PADRÃO MSX

**O MSX é o padrão para micros adotado por mais de doze fabricantes japoneses, alguns bastante conhecidos pelos seus produtos eletrônicos. Vamos examinar dois desses equipamentos: o Hit-Bit e o HX-10.**

O padrão MSX prescreve: o microprocessador a ser usado (Z80), a quantidade mínima de ROM (32 Kbytes) e de RAM (8 Kbytes), o tipo de chip de gráficos e som, as teclas que devem estar presentes (embora o layout do teclado possa variar), o número mínimo de interfaces (e seus modelos), as telas gráficas e de texto, e a linguagem BASIC contida na ROM.

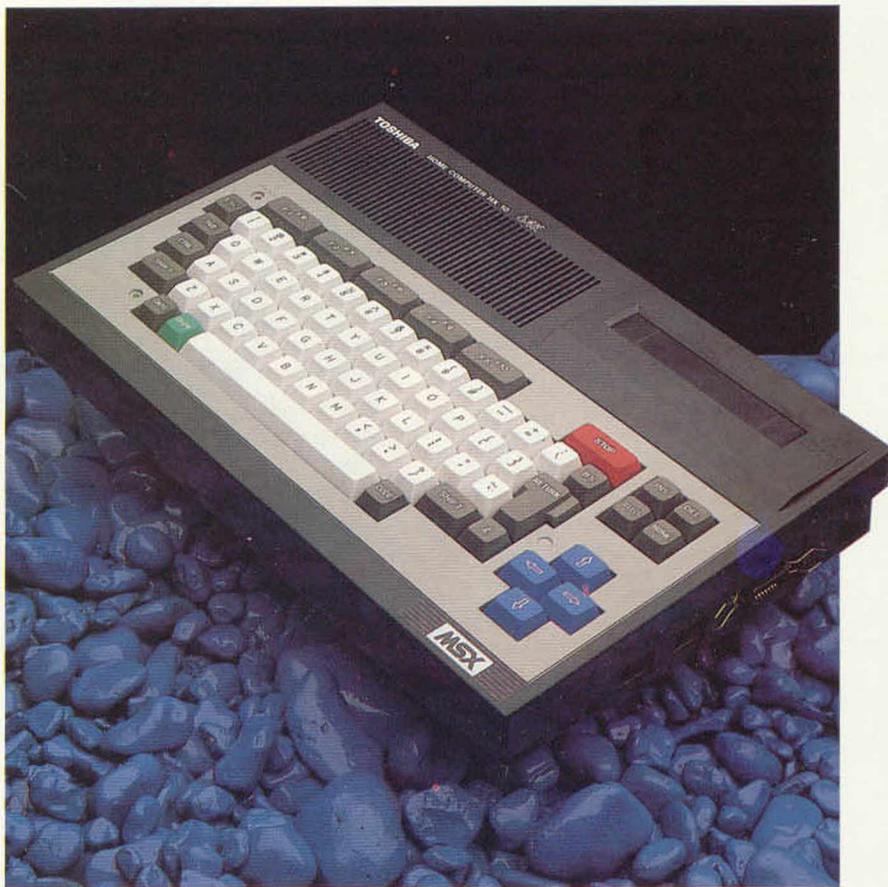
Sendo o MSX um projeto padronizado, é de se esperar que todos os micros MSX se assemelhem. Os diversos modelos, no entanto, admitem flexibilidade: a quantidade de memória que pode ir além daquele mínimo, o tipo de teclado e o número extra de interfaces. Assim, a maioria dos equipamentos MSX atingiu especificações acima dos requisitos mínimos.

Entre os modelos estrangeiros, o Hit-Bit da Sony e o HX-10 da Toshiba possuem um teclado de boa qualidade, mas há quem ache as teclas demasiadamente sensíveis ao toque. Os dois micros vêm com 64 Kbytes de memória principal e 16 Kbytes adicionais de RAM, exclusivos para o funcionamento da tela. Esse total de 80 Kbytes é superior ao fornecido pela maioria dos micros. Ambos os modelos possuem interfaces para impressora com saída paralela padrão Centronics e duas saídas para joysticks — itens frequentemente opcionais.

Esperava-se que os micros MSX se tornassem máquinas baratas, mas as flutuações no câmbio e o aumento dos custos de fabricação empurraram os preços para cima. Outra razão para o encarecimento foi a pressa com que se procurou colocar essas máquinas no mercado europeu.

Uma das primeiras coisas que nos chama a atenção quando ligamos um micro MSX é uma linha com palavras na parte inferior da tela. São comandos da linguagem BASIC, tais como RUN, CLOAD, LIST etc. Cinco teclas de função apresentam as palavras mais usadas, que, na tela, servem de rótulo para as teclas de função, de modo que o usuário não precisa memorizar a função de cada tecla.

Essas teclas são automaticamente definidas quando se liga a máquina, mas é fácil mudar a definição usando-se o comando KEY. Embora só existam cinco teclas, podem ser acessadas até dez funções. Para tanto, pressionam-se, ao mesmo



## Sistema padronizado

O Toshiba HX-10 possui uma interface para impressora com saída paralela padrão Centronics e saídas para joystick (duas) e para cartucho de ROM. O BASIC do MSX lida com os joysticks da mesma maneira como comanda os movimentos do cursor, de modo que jogos desenvolvidos para um tipo de controle podem automaticamente utilizar outro. À esquerda, o Sony Hit-Bit e seus periféricos.

tempo, a função desejada e a tecla [Shift]. Esta, quando acionada, muda as palavras na parte inferior da tela, apresentando as novas funções designadas às teclas. Cada atribuição de função pode conter até quinze caracteres, mas apenas os sete primeiros aparecem na tela.

O teclado e o editor da tela funcionam concatenados para facilitar a edição. Quatro teclas movimentam o cursor pela tela, podendo-se fazer



Saída para tevê

Saída para cartucho de ROM  
Conforme o projeto padrão MSX

Saída para impressora

Chip adaptador de interface periférica  
Este chip, chamado PIA (Peripheral Interface Adaptor), controla a entrada e a saída dos periféricos.

Saídas para joystick  
Estas saídas usam o padrão Atari para joysticks.

Controlador de som AY-389 10  
Este chip fornece som em três canais.

Chip de vídeo TI TMM 9929  
O padrão MSX requer um TI 9918 ou equivalente. O chip 9929 é o equivalente para as telas do sistema PAL.

ROM do BASIC  
O BASIC Extended, da Microsoft, está gravado neste chip de 32 Kbytes.

RAM de vídeo  
Estes chips, com capacidade de 16 Kbytes, contêm os requisitos de memória para a informação de vídeo, liberando a memória principal para o usuário.

Modulador de PAL

RAM do usuário com 64 Kbytes

CPU Z80A

alterações em qualquer lugar da mesma, bastando escrever em cima dos caracteres existentes. A inserção e supressão de caracteres requerem um único toque. No HX-10, as teclas do cursor têm o mesmo tamanho das outras; já no Hit-Bit, esse conjunto é bem maior e bastante diferenciado. Como tais teclas são intensamente usadas, esse tipo de layout revela-se muito conveniente.

Assim como o hardware e o software do MSX estão cheios de recursos extras, seu BASIC inclui comandos como AUTO e RENUM — que facilitam a codificação — e contém vários comandos para manipulação de som, gráficos e interrupções. Três deles são básicos para a geração de gráficos. O comando LINE traça uma linha ligando dois pontos, mas pode também ser usado para desenhar um quadrado — basta acrescentar a letra B (box) logo após as coordenadas. Adicionando-se as letras BF (box fill), obtém-se

um quadrado preenchido com uma cor. O comando CIRCLE é usado para desenhar elipses e arcos, assim como círculos básicos. E PAINT preencherá totalmente qualquer contorno, mesmo os mais complicados, com uma cor.

O BASIC do MSX inclui muitas outras características úteis, mas o conjunto mais sofisticado de comandos, para “manipulação de interrupções”, pode não ser apreciado de imediato. São comandos muito úteis para programar gráficos em alta velocidade. Há várias situações em que um programa deve realizar uma tarefa, ao mesmo tempo que verifica se outra coisa está acontecendo.

Exemplo típico disso são jogos do tipo Invasores do Espaço. O programa deve manter os alienígenas em constante movimento pela tela e, simultaneamente, verificar se o botão de “fogo” foi acionado. Precisa, portanto, fazer as duas



## Desfraldando a bandeira

### Padrão MSX

<b>CPU</b>	Z80A, de 3,58 MHz
<b>RAM</b>	8 Kbytes, no mínimo
<b>ROM</b>	32 Kbytes, incluindo o BASIC
<b>TELA</b>	Dezesseis cores, gráficos de 256 x 132 pixels, 32 sprites, texto de 24 linhas x 40 colunas (ou 24 x 32); chip de vídeo TI 9918 ou equivalente
<b>SOM</b>	Três canais, podendo ser acessado diretamente pelos comandos do BASIC; chip controlador de som AY 38910
<b>INTERFACES</b>	Para cartucho MSX, monitor de tevê, impressora e para cassete
<b>TECLADO</b>	Tipo QWERTY, com teclas especiais de função, quatro de movimentação do cursor e dez de funções programáveis

### Variações MSX

<b>SONY HIT-BIT</b>	Software de banco de dados gravado em ROM, saída RGB, expansões opcionais de 4 Kbytes de RAM
<b>TOSHIBA HX-10</b>	Bus de expansão, duas saídas para joystick
<b>YAMAHA CX-5</b>	Miniteclado musical e software para a interface MIDI
<b>PIONEER</b>	Interface controladora de videodisco a laser
<b>SANYO MPC100</b>	Caneta óptica opcional e software
<b>JVC HC7GB</b>	Saída RGB
<b>SVI 728</b>	Teclado numérico completo
<b>SPECTRAVIDEO</b>	

Apesar de o padrão MSX requerer um mínimo de 8 Kbytes de memória, todos os fabricantes acima fornecem 64 Kbytes de RAM para o usuário, mais 16 Kbytes para a RAM de tela.

coisas ao mesmo tempo, trocando rápido de tarefa.

A solução do MSX é designar certas coisas como "eventos". São fornecidas instruções ao computador para que esteja atento a um evento. Se algum ocorre, o computador chama automaticamente uma sub-rotina para ocupar-se dele.

O modo gráfico do MSX pode mostrar dezesseis cores com uma resolução de 256 x 192 pixels. Definem-se até 32 sprites de 8 x 8 pixels (ou 16 sprites de 16 x 16 pixels; ou 8 sprites de 32 x 32 pixels). Para o aproveitamento máximo dos sprites, o BASIC do MSX inclui um conjunto completo de comandos dedicados a eles; por exemplo, SPRITE para definir um sprite, e PUT SPRITE para posicioná-lo em qualquer lugar da tela.

Já se encontra disponível um grande número de software em cartuchos. E a promessa de compatibilidade parece estar sendo cumprida — os aplicativos para o HX-10 rodam perfeitamente no Hit-Bit e vice-versa. Isso vale tanto para aplicativos em cartucho como para programas em cassete. Depois de anos de sistemas não compatíveis, parece quase mágico poder tirar um cartucho de um micro e usá-lo em outro. Os fabricantes do padrão MSX, confiantes no sucesso do equipamento graças a essa característica, procuram colocar bem rápido no mercado uma ampla gama de aplicativos.

No Brasil, foi com o padrão MSX que a Gradient e a Sharp ingressaram, em 1985, na área da informática. Seus micros, o Expert e o Hit-Bit, respectivamente, visam sobretudo o mercado educacional. É a contribuição brasileira a esse padrão de computadores bem equipados, divertidos de usar e de preço bem razoável. E que vêm cumprindo à altura as promessas dos fabricantes.

## MSX TOSHIBA HX-10

### MICROPROCESSADOR

Z80A.

### CLOCK

3,58 MHz.

### MEMÓRIA

64 Kbytes de RAM (28 Kbytes disponíveis para o BASIC), 16 Kbytes de RAM de tela, 32 Kbytes de ROM, incluindo o BASIC.

### VIDEO

Modo texto: 4 linhas x 40 colunas; modo gráfico: 256 x 192 pixels, dezesseis cores e até 32 sprites.

### TECLADO

Estilo máquina de escrever, com 68 teclas, sendo quatro para o cursor e cinco de funções programáveis.

### LINGUAGEM

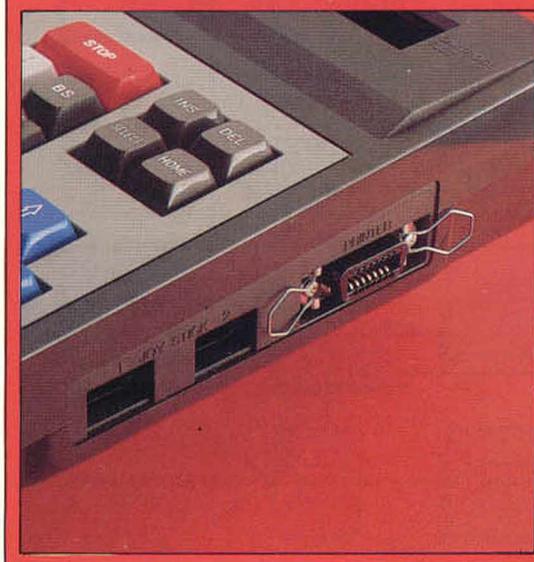
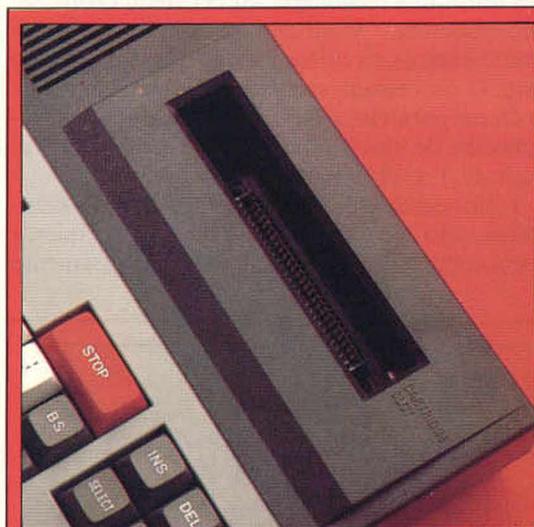
BASIC expandido da Microsoft.

### INTERFACES

Saídas para impressora paralela (padrão Centronics), monitor, tevê, áudio, cassete, joystick, duas aberturas para cartucho de ROM, bus de expansão.

### DOCUMENTAÇÃO

Guia para instalação e guia de referência de programação em BASIC. Ambos são bem feitos, mas não suficientemente completos.





# FORÇA BRUTA



## O favorito

O Cray-1 (foto acima) oferece uma enorme capacidade de processamento, sob o controle de um computador frontal como um IBM, DEC, ou similar. A arquitetura do Cray-1 (diagrama abaixo) apresenta 32 Mbytes de memória principal e 4.888 bytes de espaço de registradores. Todo esse sistema aloja-se em nada menos que 3.400 painéis de circuitos impressos, com mais de 90 km de fios para conectá-los.

Muitas áreas da ciência exigem processamento rápido de imensas quantidades de dados. Para esse fim, são usados "supercomputadores", como o Cray-1, que estudamos neste artigo.

A potência de um computador, em termos simples, é função de seu tamanho de palavra (quantos bits são processados por vez), velocidade de transferência de dados, tamanho da memória principal e velocidade de ciclos da unidade central de processamento. Nos micros, as funções principais da CPU residem num único chip microprocessador. Este pode ser o conhecido Z80, os Intel 8088 e 8086, e o Motorola M68000. Todos utilizam a tecnologia MOS (Metal Oxide Semiconductor, "semicondutor de óxido metálico") para os circuitos lógicos e a memória do chip. O processamento e transferência de dados se dá em paralelo, a 8 ou 16 bits por vez. As frequências de clock dos microprocessadores variam de 1 a 12 MHz.

Todos esses recursos, mesmo parecendo avançados, não são suficientes para que os micros deem conta do colossal volume de processamento

de dados necessário para aplicações como animação de imagens, dinâmica dos fluidos e previsão de tempo.

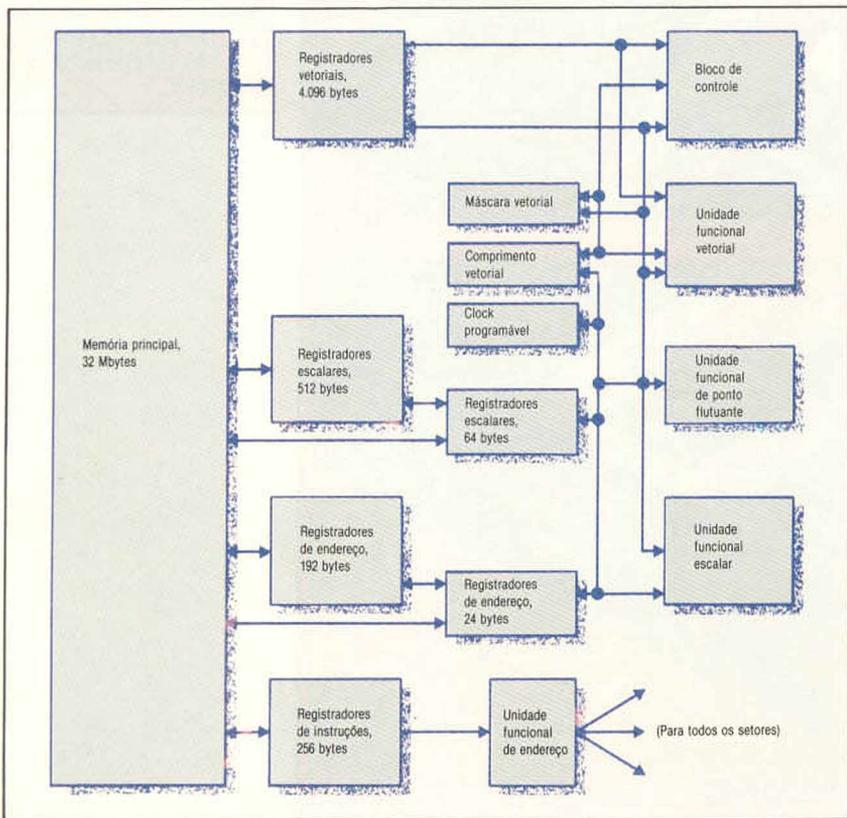
Vejamos um exemplo concreto. Imagine os requisitos necessários para fazer um filme com gráficos animados gerados por computador, com resolução de 6.000 x 6.000 pixels e 24 fotogramas, ou quadros, por segundo. Como os gráficos são animados, os pontos na imagem deslocam-se de quadro em quadro, de modo que a posição de cada ponto terá de ser calculada a cada novo fotograma. Isso significa 864 milhões de cálculos por segundo, cada um bastante complexo e envolvendo dezenas ou centenas de instruções em código de máquina. Isso resulta em bilhões de instruções por segundo.

Os supercomputadores, nome dado às máquinas de maior porte, funcionam de modo muito semelhante aos micros. As instruções e os dados vêm da memória; o processador manipula os dados de acordo com as instruções e armazena os resultados na memória. A diferença principal está na escala e na velocidade de execução dessas operações, bem como na arquitetura dos componentes do hardware.

Vejamos um exemplo típico de supercomputador: o Cray-1 S/4400, fabricado pela Cray Research. O primeiro Cray-1 foi instalado em 1976 e vem se firmando como o mais famoso e o mais popular supercomputador de grande porte. A CPU do Cray-1 ocupa um gabinete semicircular de quase 2 m de altura, parecido com um sofá curvo (o sistema de resfriamento e a fonte de alimentação ficam sob os "assentos"). Sua velocidade resulta da lógica de semicondutores bipolares, que são transistores "comuns", ao contrário do MOS, CMOS, NMOS, FET e tantos outros tipos. A lógica bipolar e a memória provêm de mais de 200.000 circuitos integrados, dispostos em 3.400 painéis; usam-se mais de 90 km de fios para interconectá-los.

O tamanho de palavra usado nas computações do Cray-1 é de 64 bits, ou seja, oito vezes mais que os 8 bits processados por um Z80. O clock do sistema tem uma velocidade de 80 MHz (80 milhões de ciclos por segundo); como resultado, uma adição em 64 bits leva apenas 37,5 nanossegundos, ao passo que uma adição em 8 bits, num Z80 operando a 4 MHz, leva 1,75 microssegundo.

A memória principal no Cray é parte integrante da CPU. Ela contém 32 Mbytes, dispostos sob a forma de 4.194.304 palavras, e transfere até 2,56 bilhões de bytes por segundo. Como se pode ver, o Cray possui um poderoso conjunto de registradores na CPU. Há 72 registradores de en-





dereço (cada um com 24 bits de comprimento), 72 registradores escalares (cada um com 64 bits de comprimento) e oito registradores vetoriais (cada qual com 64 palavras de comprimento). O espaço total dedicado aos registradores na CPU do Cray é, portanto, de 4.888 bytes; em comparação, o microprocessador Z80 possui apenas 26.

## O terminal frontal

Ao contrário da maioria dos computadores, que constituem unidades completas e independentes, os computadores Cray foram projetados para ser extensões de um computador de grande porte ou um minicomputador já existente. Este, chamado computador frontal, funciona como uma interface, aceitando entradas de terminais ou de leitoras de cartão e enviando as saídas para periféricos como impressoras ou fitas magnéticas.

Entre a CPU do Cray e o computador frontal fica o subsistema de entrada e saída do Cray, projetado para agilizar o fluxo de dados para a CPU. Há também uma interface, que adapta o sistema Cray às características específicas do IBM, DEC, Data General ou outro computador usado como terminal frontal. O subsistema de entrada e saída do Cray consiste em dois a quatro processadores de entrada e saída, sendo cada um deles, sozinho, um potente minicomputador.

Veja no boxe o diagrama de blocos da CPU do Cray-1, para uma idéia mais precisa de sua sofisticação. Poderíamos ainda citar o conjunto de instruções, as treze unidades funcionais que operam em paralelo e o processamento vetorial, que permite trabalhar com até 64 pares de operandos com uma única instrução. Em resumo, não exageramos ao dizer que o Cray é extremamente poderoso.

## Aplicações dos supercomputadores

### Meteorologia

Previsões de tempo resultam da coleta de dados em âmbito mundial, comunicações via satélite e modelagem computacional. Um "modelo" elaborado por computador requer centenas de milhões de cálculos para se obterem previsões em questão de horas. E apenas computadores extremamente potentes como o Cray processam tamanha quantidade de dados com a rapidez necessária.

### Dinâmica dos fluidos

Ciência fundamental no projeto de automóveis econômicos, sistemas de refrigeração de usinas nucleares, engenharia aeronáutica e muitas outras aplicações. Prever o movimento de fluidos exige a análise de imensas quantidades de dados, e os problemas se avolumam quando se necessita de resultados em tempo real. Cada partícula num fluido — e há incontáveis trilhões delas — atua sobre todas as outras do conjunto em movimento. Assim, o cálculo do comportamento de um sistema de fluidos exige uma capacidade maciça de processamento de dados.

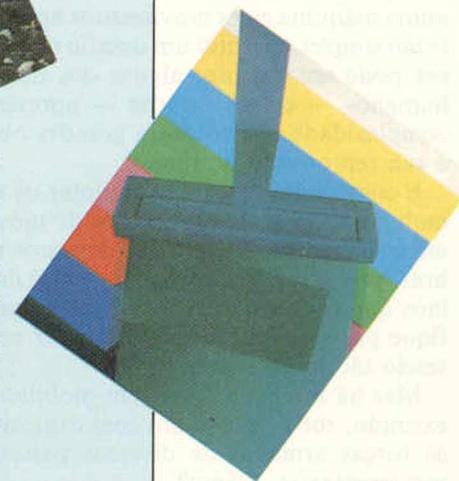
### Previsões econômicas

É difícil construir modelos econômicos, pois, como na dinâmica dos fluidos, pequenas mudanças num elemento repercutem sobre todo o sistema. Mesmo modelos simples são de enorme complexidade. Também aqui se recorre a computadores extremamente rápidos e potentes, para que os analistas não tenham de esperar meses pelos resultados.

## Alta velocidade

Para comparar a capacidade de processamento de um supercomputador Cray-1 à de um micro baseado no Z80, considere a produção de uma seqüência de 10 minutos de filme com gráficos animados em alta resolução. Vamos admitir uma resolução de 6.000x6.000 pixels, e 24 fotogramas por segundo; cada pixel precisa ser calculado para cada novo fotograma. Na versão para o Z80, necessita-se de cem instruções em código de máquina, com um tempo de execução médio de dezenove ciclos de clock (4,75 microssegundos). Para o Cray-1, com suas poderosas instruções de processamento vetorial, diremos que são necessárias 25 instruções em código de máquina (um cálculo pessimista), com um tempo médio de execução de quatro ciclos de clock (50 nanossegundos). Para a produção desse filme, um micro baseado no Z80 levaria

$$6.000^2 \times 24 \times 60 \times 10 \times 100 \times 4,75 \times 10^{-6} = 2,4624 \times 10^8 \text{ segundos, ou seja, } 7,8 \text{ anos.}$$

$$\text{Um Cray-1 levaria } 6.000^2 \times 24 \times 60 \times 10 \times 25 \times 50 \times 10^{-9} = 6,48 \times 10^5 \text{ segundos, ou seja, apenas } 7,5 \text{ dias.}$$


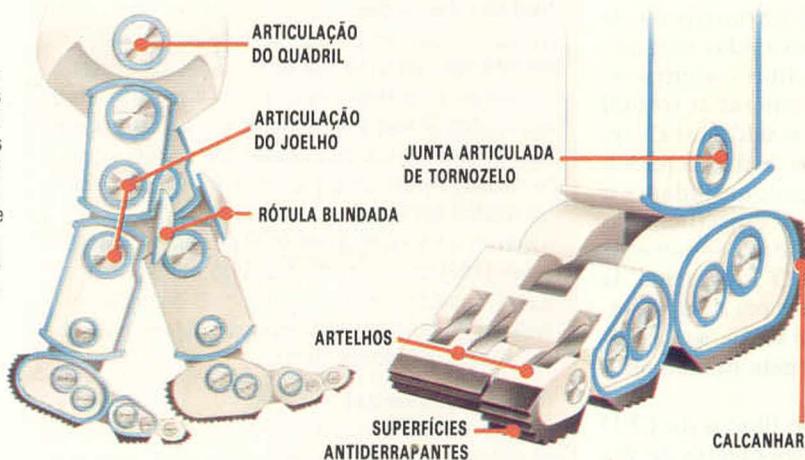
### Revolução visual

Os gráficos gerados por computador estão revolucionando a produção de vídeos e filmes. Criam-se imagens, como as mostradas, por meio da moderna tecnologia de micros e minicomputadores. Trabalhando em maior escala, usou-se o Cray-1 para gerar mais de 20 minutos de imagens no filme *The last starfighter (O último combatente do espaço)*, executando cálculos que ocuparam um micro de 8 bits por mais de quinze anos.

# O PASSO DO ROBÔ

## Caminhando

Andar é um movimento da maior complexidade. Para fazer isso, um robô deveria ter tantas articulações quanto o homem e estar programado para dar seus passos exatamente assim: deve mudar seu centro de gravidade, inclinando o corpo para a frente e levantando a perna que ficou atrás. O ombro gira e os braços balançam, equilibrando o corpo sobre o pé de apoio, com a ajuda dos artelhos, até que o outro pé vá à frente dar apoio.



**Para ser útil, um robô precisa de movimentos. Mas conseguir isso envolve a resolução de vários problemas, como a construção de pernas, pés, braços, mãos, articulações e “músculos”.**

Movimentar o corpo, para um ser humano, é mais do que trivial: é natural. Mas reproduzir numa máquina esses movimentos aparentemente tão simples constitui um desafio que nem sempre pode ser vencido: alguns dos movimentos humanos — como o andar — apresentam tal complexidade que colocam grandes obstáculos à sua reprodução mecânica.

E como fazer, então, para dotar os robôs de mobilidade, isto é, capacidade de mover-se de um lugar para outro? Bem, devemos nos lembrar, primeiro, de que a maior parte dos trabalhos que se pretende de um robô exige que ele fique parado. Assim, fazê-lo andar acaba não sendo tão importante.

Mas há tarefas que exigem mobilidade. Por exemplo, robôs que lidam com explosivos para as forças armadas de diversos países têm de movimentar-se. Como?

Há diversas maneiras de fazer um robô andar,

especialmente se ele tiver pernas. Nesse caso, estas podem se mover para a frente e para trás, simulando o andar humano, arrastando os pés pelo chão. Esses pés podem ter rodas com catracas, de modo que só se movam para diante. E aí o problema está resolvido — só que fica difícil manobrar o robô.

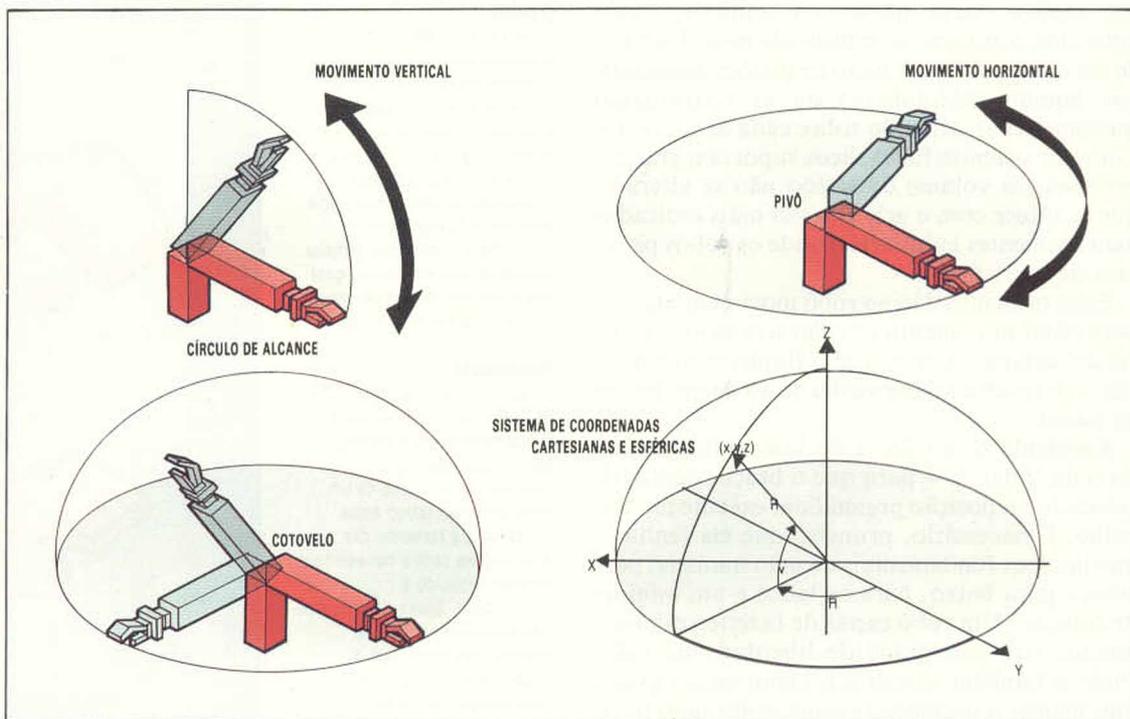
A solução seria fazê-lo erguer os pés do chão para caminhar. Mas aí surge outro problema: ele terá de equilibrar-se em uma das pernas enquanto anda. As soluções são complicadas, mas não impossíveis. Um robô assim até subiria escadas — só ficaria difícil montar o mecanismo para ele descobrir quando a escada acaba.

Muito mais simples é usar esteiras, como os tanques de guerra — com a desvantagem de que os robôs não são tão grandes, perdendo, portanto, para os tanques em estabilidade. Além disso, os movimentos carecem de precisão. Para manobrar, uma das esteiras tem de ficar parada, enquanto a outra continua a mover-se. O ideal seria que a esteira parada apenas girasse sobre o solo, em torno de seu centro. Mas isso não acontece, e ela acaba se deslocando, o que, mesmo sendo pouco, significa um problema. No caso do tanque, o deslocamento é corrigido pelo homem que o dirige.

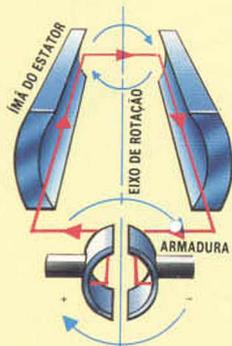


**rotações possíveis**

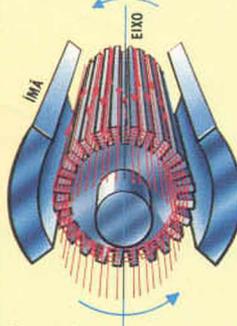
Mesmo o mais simples robô deve poder movimentar o braço como nestas ilustrações. Para isso, o cotovelo é articulado, permitindo um movimento vertical semicircular (de 90°) e horizontal circular (360°). Assim, o braço alcança qualquer ponto num hemisfério cujo raio tem seu tamanho. E a localização desse ponto é dada por coordenadas do seguinte modo (veja o desenho): o ângulo H corresponde ao do arco tangente x/y; o ângulo V é o do arco seno z/R (arc sen z/R). Assim, o programa que leva o braço a um determinado ponto deve conter as coordenadas x,y,z desse ponto. Esses valores são transformados em movimentos dos servomotores, que dirigem então o braço até o ponto exato de trabalho.



**Passo a passo**



Em qualquer motor elétrico, uma corrente na bobina do rotor cria um fluxo magnético contrário ao do estator. É justamente essa oposição de forças que produz movimento no rotor.



O rotor de um motor a passo pode ter centenas de bobinas. Passando-se a corrente de alimentação de uma para outra obtêm-se movimentos pequenos de grande precisão.

Por essas e outras razões, robôs usam rodas, que produzem movimento suave e são mais fáceis de controlar. Ficam faltando, então, um meio de o robô saber exatamente onde está e um mecanismo para movimentar as rodas com precisão.

Para o primeiro problema, a solução está no plano cartesiano, definido por dois eixos de coordenadas. Isso basta para localizar o robô e determinar os pontos para os quais ele tem de se deslocar.

Fazer isso com precisão, contudo, exige dispositivos especiais. Não podemos, nesse caso, usar motores elétricos comuns para movimentar as rodas, as quais, cortada a alimentação, ainda giram pelo menos meia volta, e isso constitui um grave inconveniente para aplicações que exigem precisão. Nesse caso, a solução está no uso de motores a passo, ou graduais, que produzem movimentos precisos, de qualquer amplitude. É esse o tipo de motor usado nas tartarugas, que têm rodas e um suporte para caneta, podendo, assim, executar desenhos.

Mas tudo se complica bem mais quando se trata de reproduzir os movimentos da mão e do braço do ser humano, necessários para que o robô trabalhe. Em primeiro lugar, deve-se estabelecer um sistema para determinar a posição do braço em qualquer instante; depois, esse braço precisa de uma estrutura e de "músculos".

A primeira questão tem solução também pelos eixos de coordenadas; mas nesse caso há um complicador, que é um terceiro eixo (já que o movimento do braço se faz em três dimensões). Com isso, pode-se descrever a posição do braço em qualquer ponto do espaço em que ele atua. É possível, então, contruir um robô cujo braço se desloque ao longo desses três eixos de coor-

denadas — para a frente e para trás, para cima e para baixo, para a direita e para a esquerda. Fica meio desajeitado, mas funciona bem para trabalhos numa área limitada.

Esse sistema funciona bem para robôs fixos a um suporte, mas não tem a mesma eficiência em outras situações. Para estas, há métodos de descrever o espaço e os pontos dentro dele. Um desses usa como modelo um cilindro e suas coordenadas. Imagine uma lata cilíndrica. Localiza-se qualquer ponto dentro da lata, associando: a distância a que está do centro do círculo que forma o fundo da lata (essa distância é um segmento de raio); o ângulo que ele forma em relação a determinado ponto de referência no círculo; e a altura em relação ao fundo da lata. Com essas três informações, há uma localização exata.

Um sistema parecido tem uma esfera como modelo de espaço. A diferença com o anterior está em que, ao invés de usar a altura em relação ao fundo, a informação necessária consiste no ângulo formado entre o ponto, o centro da esfera e seu diâmetro horizontal.

O processo mais comum para o posicionamento exato do braço de um robô, no entanto, usa somente ângulos: um para descrever a rotação na base, outro para a elevação do braço e um terceiro para uma segunda articulação.

**Força muscular**

O sistema de coordenadas adotado acabará de determinando o tipo de estrutura do braço, faltando, então, os "músculos" que vão produzir o movimento. Podem ser hidráulicos, elétricos ou pneumáticos.

Se a opção for pelos motores elétricos, estes deverão ser de passo, ligados direta ou indiretamente às partes que formam o braço. Um siste-



ma melhor seria imitar os animais, cujos músculos contraem-se e distendem-se. Isso pode ser conseguido por meio de pistões acionados por líquido (hidráulicos) ou ar comprimido (pneumáticos), atuando sobre cada articulação. Como os sistemas hidráulicos suportam grandes pressões e o volume do fluido não se altera (o que acontece com o ar), eles são mais indicados para ambientes industriais, onde os robôs precisam de mais força.

Esses músculos dão ao robô movimento linear; para obter movimento circular usa-se uma espécie de turbina sobre a qual o fluido exerce pressão — o resultado assemelha-se ao de um motor de passo.

Resolvido o problema do braço, fica faltando o da “mão” — para que o braço, depois de colocado na posição pretendida, execute um trabalho. É necessário, primeiro que ela tenha os movimentos fundamentais da mão humana: para cima e para baixo, para os lados e um mínimo de rotação. Um robô capaz de fazer esses movimentos tem seis graus de liberdade na mão. Pode-se também, construí-los com menos graus, mas manter a mobilidade implica dar mais liberdade às outras articulações do braço.

## A mão do robô

Tudo isso definido, fica faltando apenas estabelecer como será a mão do robô. O ideal, naturalmente, é fazê-la tão parecida com a do homem quanto possível (e até existem robôs assim). O formato mais comum consiste num “polegar” e dois “dedos”, habilitando o robô a apanhar objetos como o homem.

A energia para movimentar essa mão pode ser qualquer das três já citadas — elétrica, pneumática ou hidráulica —, dependendo da tarefa que o robô vai executar. Se ele tiver de trabalhar com objetos grandes e pesados, provavelmente precisará de musculatura hidráulica. Para outras aplicações, porém, motores elétricos ou a ar comprimido bastarão — especialmente se essas aplicações não exigirem precisão absoluta.

Haverá casos em que o robô não precisará de uma mão ou qualquer coisa parecida. Um robô de solda, por exemplo, só requer o equipamento de soldagem ligado diretamente ao punho. Existem robôs capazes até mesmo de escolher e trocar uma ferramenta do punho, de acordo com o tipo de trabalho a executar. Podem, digamos, descartar uma chave de fenda e inserir uma pistola de pintura — desde que o encaixe seja simples —, o que os torna extremamente versáteis.

A grande maioria, contudo, costuma trabalhar o tempo inteiro apenas com uma ferramenta — como os robôs de solda usados na indústria automobilística, que têm características totalmente distintas dos usados em pinturas. Seria inútil um robô de solda capaz de substituir ferramentas, já que seu trabalho é contínuo e exercido sempre no mesmo lugar. Mesmo porque, na fase de soldagem o carro ainda não está pronto para receber a pintura.

### O pulso

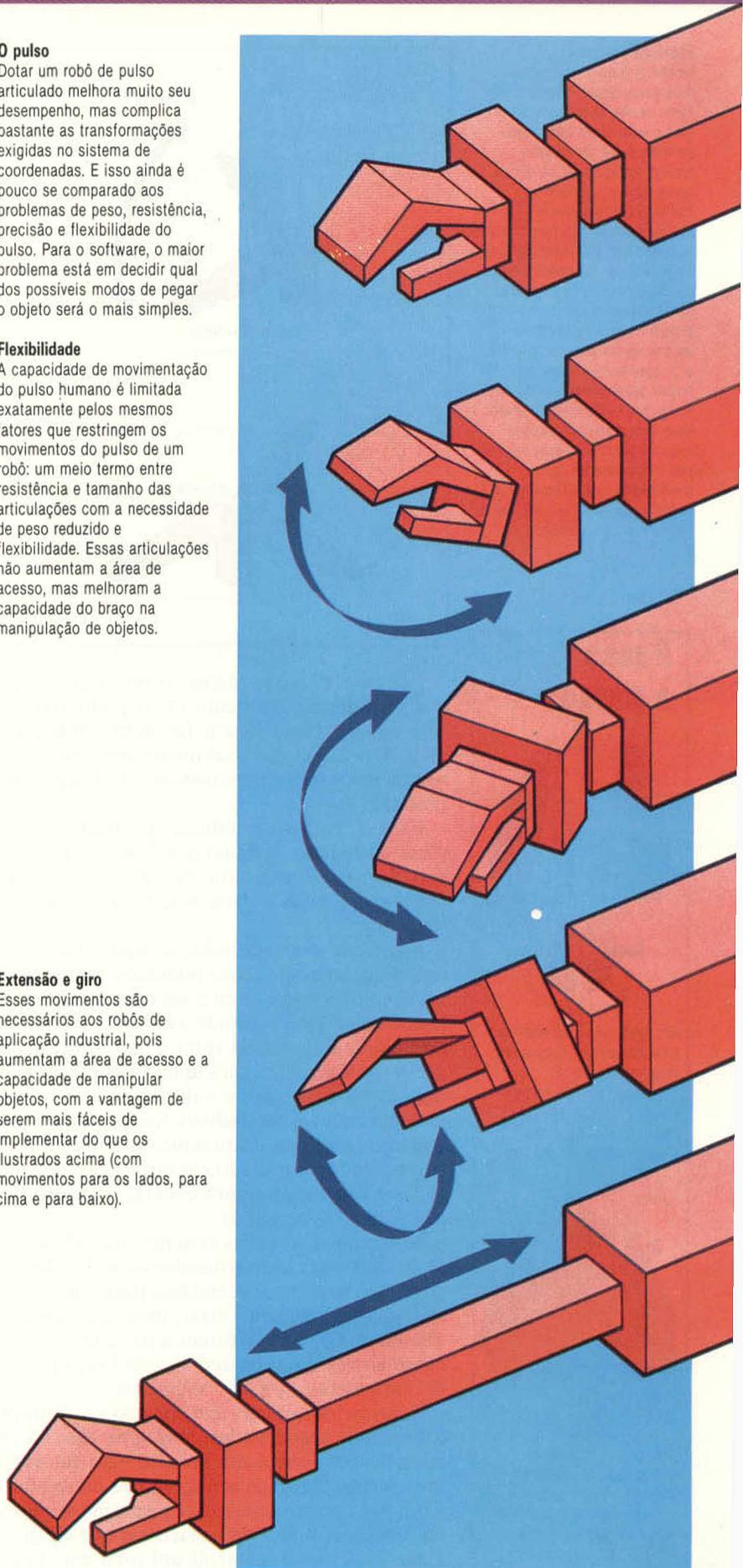
Dotar um robô de pulso articulado melhora muito seu desempenho, mas complica bastante as transformações exigidas no sistema de coordenadas. E isso ainda é pouco se comparado aos problemas de peso, resistência, precisão e flexibilidade do pulso. Para o software, o maior problema está em decidir qual dos possíveis modos de pegar o objeto será o mais simples.

### Flexibilidade

A capacidade de movimentação do pulso humano é limitada exatamente pelos mesmos fatores que restringem os movimentos do pulso de um robô: um meio termo entre resistência e tamanho das articulações com a necessidade de peso reduzido e flexibilidade. Essas articulações não aumentam a área de acesso, mas melhoram a capacidade do braço na manipulação de objetos.

### Extensão e giro

Esses movimentos são necessários aos robôs de aplicação industrial, pois aumentam a área de acesso e a capacidade de manipular objetos, com a vantagem de serem mais fáceis de implementar do que os ilustrados acima (com movimentos para os lados, para cima e para baixo).





# MUDANÇA DE ENDEREÇO

**A linguagem ASSEMBLY deve grande parte de sua versatilidade às diferentes maneiras de calcular endereços na memória. Examinamos aqui os endereçamentos direto, indireto e indexado, bem como suas combinações.**

Qualquer instrução em linguagem ASSEMBLY relaciona-se implícita ou explicitamente a um conteúdo da memória. Uma vez que um byte só se diferencia de outro por seu endereço, toda instrução em ASSEMBLY refere-se, portanto, a um ou mais endereços. Às vezes a referência é direta e óbvia, como em LDA \$E349, que significa “carregar o acumulador com o conteúdo do endereço \$E349”. Essa instrução menciona sem ambigüidade o endereço \$E349 e sua relação com o acumulador (cujo endereço é um nome, e não um número).

Outras instruções, contudo, referem-se a um endereço de maneira menos clara; por exemplo, RET, que significa “retornar de uma sub-rotina”. À primeira vista, aqui não há menção a nenhum endereço; mas RET, na realidade, significa “o endereço da próxima instrução a ser executada é o lugar de onde se chamou a última sub-rotina”. Aqui não se menciona o endereço cujo conteúdo será alterado (ou seja, o contador de programas — o registrador que contém o endereço da próxima instrução a ser executada). Também há referência ao endereço do novo conteúdo. Essas duas instruções representam modos de endereçamento totalmente diferentes.

Até agora, vimos instruções com dois diferentes modos de endereçamento: o imediato (como em LDA, \$45 ou ADC #31) e o direto absoluto (como em STA \$58A7 ou LD (\$696C),A). Esses modos, “naturais”, parecem abranger todos os casos possíveis, exceto os implícitos, como RST ou RET; há, porém, outras possibilidades, que examinaremos agora.

## Endereçamento da página zero

Usamos esse endereçamento, ou endereçamento curto, sempre que uma instrução se refere a um endereço entre \$0000 e \$00FF. Todos os endereços nessa faixa têm \$00 como byte mais alto e, portanto, ficam na página zero da memória. Tais instruções necessitam de apenas 2 bytes — um para o código de operação e outro para o byte mais baixo do endereço. Quando a CPU detecta um endereço de um só byte num ponto onde costuma haver dois, assume que o byte mais

alto é \$00 e, assim, refere-se diretamente à página zero. A vantagem desse processo está na velocidade de execução: o acesso aos dados da página zero é bem mais rápido que a qualquer outra página, exatamente porque requer um endereço de apenas 1 byte.

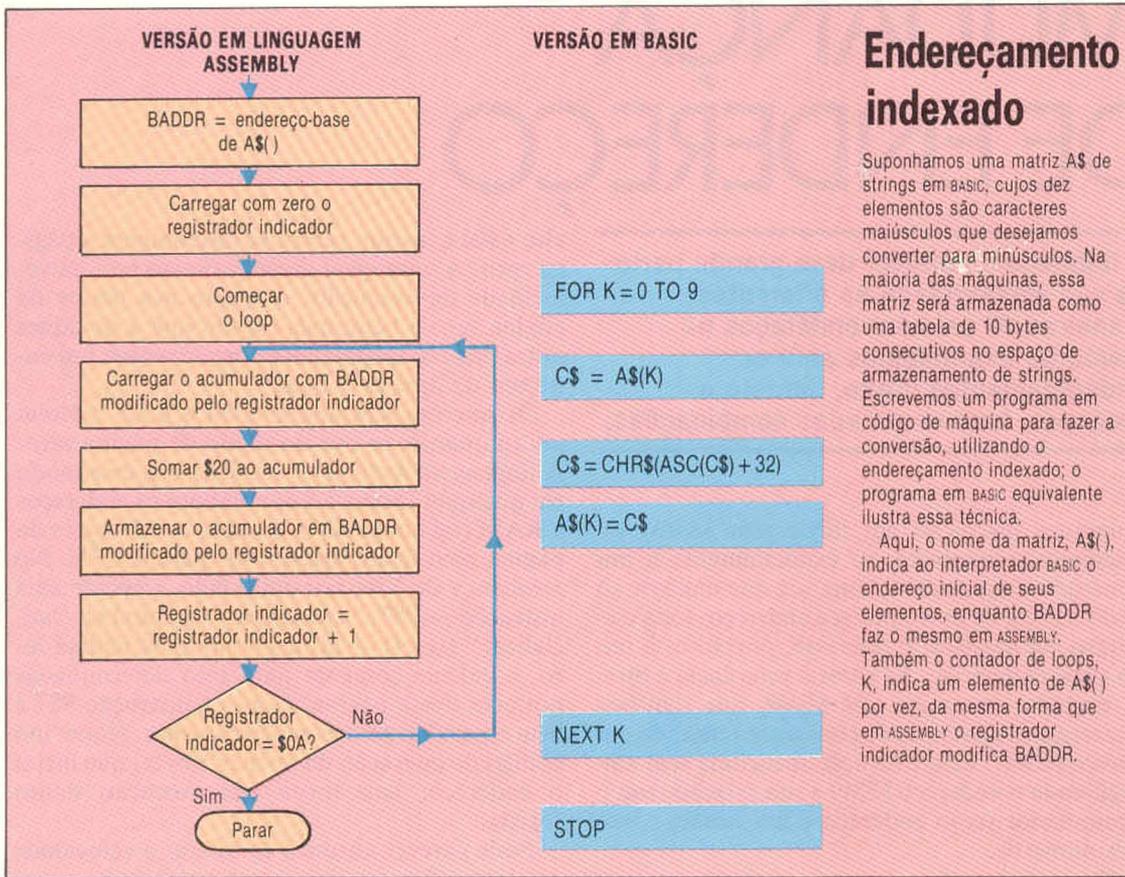
Os microprocessadores Z80 e 6502 diferem muito quanto ao uso do modo de página zero. A linguagem ASSEMBLY do 6502 usa esse modo em qualquer instrução que elabore os endereços à RAM (tal como LDA), e todos os 256 bytes da página zero ficam disponíveis para a CPU. No ASSEMBLY do Z80, por outro lado, somente uma instrução — RST, “recomeçar” (restart) ou “inicializar” (reset) — utiliza o modo de página zero, e calcula somente para oito determinadas posições dessa página. Como a instrução RST é bem específica em seu endereçamento, requer um código de operação de apenas 1 byte, que inclui o endereço; isso torna sua execução muito rápida.

Pode parecer absurdo comparar a velocidade das instruções em linguagem ASSEMBLY, quando sabemos que o tempo de execução da mais lenta é medido em microssegundos. Entretanto, às vezes a economia de 1 microssegundo por instrução representa a diferença entre o sucesso e o fracasso de um programa. Por exemplo, os jogos que mostram gráficos animados em alta resolução, coloridos e tridimensionais, envolvem milhões de instruções por operação em tela, e devem executar esses comandos o mais rápido possível para obter movimentos suaves e contínuos. Cortar 1 microssegundo de uma operação é muito importante quando ela faz parte de um loop a ser executado 64.000 vezes seguidas.

## Endereçamento indexado

O modo indexado é vital para a programação em ASSEMBLY, pois permite a elaboração de estruturas matriciais de dados. Sem tais estruturas, os programas se restringem ao endereçamento de posições de memória uma a uma; foi o que fizemos até agora em nossos programas. Por meio da indexação, entretanto, manipulam-se muito mais dados, e o programador conta com quantidade bem maior de recursos.

Os elementos essenciais do modo indexado são o endereço-base e o *índice*. Vamos supor que queremos montar uma tabela de dados — os códigos dos caracteres ASCII, por exemplo — em bytes consecutivos. O endereço-base da tabela é o do primeiro byte. A partir disso, podemos nos referir a qualquer byte subsequente da tabela por sua posição em relação ao endereço-base; o primeiro byte fica na posição zero, o segundo na



### Endereçamento indexado

Suponhamos uma matriz A\$ de strings em BASIC, cujos dez elementos são caracteres maiúsculos que desejamos converter para minúsculos. Na maioria das máquinas, essa matriz será armazenada como uma tabela de 10 bytes consecutivos no espaço de armazenamento de strings. Escrevemos um programa em código de máquina para fazer a conversão, utilizando o endereçamento indexado; o programa em BASIC equivalente ilustra essa técnica.

Aqui, o nome da matriz, A\$( ), indica ao interpretador BASIC o endereço inicial de seus elementos, enquanto BADDR faz o mesmo em ASSEMBLY. Também o contador de loops, K, indica um elemento de A\$( ) por vez, da mesma forma que em ASSEMBLY o registrador indicador modifica BADDR.

posição um, o terceiro na posição dois, e assim por diante. A posição de um byte com relação ao endereço-base da tabela chama-se índice, e calcula-se o endereço absoluto de qualquer byte da tabela pela soma do endereço-base com o índice do byte. Se construirmos um loop num programa em ASSEMBLY, e usarmos o contador do loop como índice, poderemos endereçar cada byte da tabela seqüencialmente, da mesma forma como acessamos os elementos de uma matriz em BASIC usando um loop FOR-NEXT.

Uma vez mais, as versões do ASSEMBLY do Z80 e do 6502 diferem quanto ao endereçamento indexado. O chip do 6502 contém dois registradores de um só byte, chamados X e Y; cada um dispõe de um índice que modifica um endereço-base. Isso limita o comprimento da tabela a 256 bytes, que é o maior número representado por um byte. Já o chip do Z80 contém dois registradores de 2 bytes, IX e IY, que contêm o próprio endereço-base e podem ser incrementados ou decrementados para indicar os sucessivos bytes da tabela. Sendo registradores de dois bytes, IX e IY podem conter o endereço de qualquer byte a que a própria CPU tem acesso. Seu conteúdo também se modifica por um índice de um só byte.

### Endereçamento indireto

Envolve o uso de endereços indicadores. Para compreender melhor esse conceito, imaginemos um grupo de pessoas que formou um cineclube

e se encontra todas as semanas para assistir a um filme. Este pode estar passando em um dos vários cinemas da cidade. Então, escolhido o filme da semana, o presidente escreve a hora e o lugar da projeção num cartaz e afixa-o na vitrine de uma loja no centro da cidade. Os membros do cineclube não sabem qual será o cinema escolhido, mas sabem onde fica a loja, e esta lhes "indica" o local correto. O endereço da loja é, indiretamente, o endereço do cinema.

No modo de endereçamento indireto escrevem-se instruções que contêm o endereço de um indicador e que atuam sobre o conteúdo da po-



#### Indicador

Exemplo de endereçamento indireto encontrado no cotidiano. Na foto, o quadro indicador dos trens, com destinos e plataformas, contém dados necessários a quem viaja. A seta "Train indicator" (indicador de trens) mostra onde encontrar o quadro; assim, ela indiretamente endereça os dados. Em ASSEMBLY, o endereçamento indireto significa que o endereço no operando é o do byte que armazena os dados; o do operando, um indicador que aponta para o endereço real.



sição para onde o indicador aponta (e não sobre o conteúdo do próprio indicador). As vantagens desse modo são consideráveis, especialmente quando combinado com o tipo indexado. Suponhamos, por exemplo, que você escreva uma rotina em ASSEMBLY que pesquise, numa tabela de dados, um certo caractere, e retorne com sua posição indexada. Suponhamos ainda que você queira manter várias tabelas em diferentes locais da memória e usar a mesma rotina para pesquisar qualquer uma delas. Se sua rotina encontrar o endereço-base da tabela indiretamente, por meio de um indicador, poderá ser usada com qualquer tabela, desde que se modifique o conteúdo do indicador antes de se chamar a rotina.

Em geral, os programas requerem combinações desses modos de endereçamento. A instrução LDA do 6502, por exemplo, pode ser usada nos seguintes modos:

Código de operação	Operando	Modo
LDA	#\$34	Imediato
LDA	\$A2	Direto, página zero
LDA	\$967F	Direto, absoluto
LDA	\$A2,X	Página zero, indexado com X
LDA	\$967F,X	Absoluto, indexado com X
LDA	\$967F,Y	Absoluto, indexado com Y
LDA	(\$A2,X)	Indireto, pré-indexado com X
LDA	(\$A2),Y	Indireto, pós-indexado com Y

Exemplificamos com uma instrução do 6502 porque mostra com clareza as combinações de diversos modos de endereçamento. Essa tabela pode, de início, parecer um pouco confusa, mas, na verdade, o uso de vários modos torna claro seu significado; até agora usamos LDA e ADC em dois modos — imediato e absoluto — sem confusão.

A tabela responde a uma pergunta natural: como saber qual o modo de endereçamento de uma instrução quando o código mnemônico é o mesmo em todos os casos? Vemos que o formato do operando difere em cada modo, e a única ambigüidade possível é quanto a uma instrução como LDA SYMB1 — ela está no modo absoluto ou página zero? Um programa assembler resolve esse problema. Mas, se você quiser compilar o programa manualmente, terá de determinar se SYMB1 foi definida como um valor de 1 ou de 2 bytes.

De modo geral, uma vez que se comece a usar um programa assembler, pode-se esquecer dos códigos de operação e do número de bytes por instrução e concentrar-se em aprender as técnicas de programação do ASSEMBLY. É importante entender como funciona o código de máquina, mas a linguagem ASSEMBLY, usada em conjun-

to com um bom programa assembler, é um método de programação muito melhor, que combina a potência do código de máquina com muitos recursos das linguagens de alto nível.

Eis dois programinhas úteis em ASSEMBLY prontos para execução.

```
; CONVHA — CONVERTE BYTES HEXA EM CARACTERES ASCII
; SEQUENCIA DA CHAMADA

; LXI    D,<OP1>   CARREGA O ENDEREÇO DE DESTINO
; LXI    H,<OP2>   CARREGA O ENDEREÇO-FONTE
; MVI    C,<CONT>  CONTADOR DE BYTES HEXA (1-255)
; CALL   CONVHA

CONVHA:  MOV    A,M      ; CARREGA DOIS DIGITOS HEXA DE 4 BITS
        RAR          ; EXAMINA O PRIMEIRO DA ESQUERDA
        RAR          ;
        RAR          ;
        RAR          ;
        CALL  COHAD   ;
        MOV    A,M      ; CARREGA-OS NOVAMENTE E
        CALL  COHAD   ; EXAMINA O DIGITO DA DIREITA
        INX    H        ; INDICA O BYTE HEXA SEGUINTE
        DCR    C        ; CONT = CONT-1
        JNZ   CONVHA  ; CONTINUA O LOOP
        RET          ; RETORNA AO USUARIO

COHAD:   XCHG         ; TROCA O CONTEUDO DOS INDICADORES
        ANI    OFH     ; MASCARA ZERO NOS 4 BYTES DA ESQUERDA
        OPI    10      ; EXAMINA A FAIXA
        JNO   COHA1    ;
        ADI    D30H    ; FAIXA 0-9
        JMP   COHA2    ;
COHA1:   ADI    037H    ; NA FAIXA A-F
COHA2:   MOV    M,A     ; ARMAZENA O RESULTADO
        INX    H        ; INDICA O PROXIMO BYTE
        XCHG         ; DESTROCA OS INDICADORES
        RET          ; RETORNA

; TRANS: TRANSFERE DADOS DE UMA AREA (OP1) PARA OUTRA AREA (OP2)
; SEQUENCIA DA CHAMADA:

; LXI    D,<OP1>   ; CARREGA ENDEREÇO DE ORIGEM
; LXI    H,<OP2>   ; CARREGA ENDEREÇO DE DESTINO
; MVI    C,<CONT>  ; CARREGA TAMANHO DO CAMPO OP1
; CALL   TRANS    ; TRANSFERE OS DADOS DE OP1

TRANS:   PUSH   H      ; SALVA ENDEREÇO DE OP2
TRANS1:  PUSH   D      ; SALVA ENDEREÇO DE OP1
        LDAX  D      ; ACESSA UM DADO DE OP1
        MVI  D,0     ; ZERA O REGISTRADOR D
        MOV  E,A     ; CARREGA O ACUMULADOR NO REGISTRADOR
        E      ;
        DAD  D      ; ENDEREÇO DE ENTRADA DE OP2
        MOV  A,M     ; ACESSA UM DADO DE OP1
        POP  D      ; RESTAURA ENDEREÇO DE OP1
        POP  H      ; RESTAURA ENDEREÇO DE OP2
        PUSH H      ;
        STAX D      ;
        INX  D      ; INCREMENTA O INDICADOR DE OP1
        DCR  C      ; INCREMENTA A CONTAGEM
        JNZ  TRANS1  ; LOOP ATE O FINAL
        POP  H      ;
        RET          ; RETORNA
```



# LABO

**A Labo, indústria brasileira de minis, produz também micros, terminais bancários e sistemas aplicativos. Modulares e compatíveis entre si, seus equipamentos vêm conquistando crescente fatia do mercado.**

Criada em 1961, a Labo Eletrônica S.A. iniciou suas atividades fabricando instrumentos eletrônicos de alta precisão. Em 1979 entrou na área da informática, firmando um acordo de transferência integral de tecnologia com a empresa alemã Nixdorf Computer A.G. Esta, a maior fábrica européia de computadores e detentora de tecnologia avançada, já instalou mais de 110.000 sistemas em todo o mundo. Em 1983, a Labo participou com 27,7% das unidades comercializadas no Brasil, alcançando em 1984 a marca dos 1.500 equipamentos instalados.

A linha de produção da empresa inclui os minicomputadores da série Labo 8000, com vários modelos totalmente compatíveis em hardware e software; os superminis, de arquitetura modular e atualizada; e os microcomputadores, que constituem a série Labo 8200. Fornece, ainda, sistemas bancários on line e off line e terminais de venda para lojas e supermercados.

Modulares e compatíveis entre si, os equipamentos Labo expandem-se e integram-se facilmente. Seus micros e minis comunicam-se em rede. Mantendo os investimentos já realizados em hardware, software e treinamento, converte-se o míni 8034 num 8038, com um processador

central duas vezes mais rápido e um poderoso software básico.

O micro Labo 8221, concebido para operar em configuração multiterminal, dispõe de uma capacidade de memória superior a qualquer outro equipamento de seu porte. Nele, o processador central, a memória e o controlador da unidade de discos estão montados numa única placa, o que simplifica sua manutenção. Trabalhando com três modos de operação hierarquizados — usuário, sistema e núcleo (kernel) —, proporciona uma proteção aos dados só encontrada nos sistemas de maior porte. Seu software básico é o SOL (Sistema Operacional Labo), que permite, operando com diferentes terminais, executar dois programas simultaneamente.

Também o minicomputador Labo 8043, o maior fabricado pela empresa, oferece total proteção a programas e dados: qualquer acesso às informações é protegido por um conjunto de procedimentos conhecidos apenas pelo pessoal autorizado. Outro valioso recurso desse míni está no módulo ASA (Aceleração de Software Aplicativo). Ele reduz o número de acessos aos discos, mantendo na memória, de forma dinâmica, os blocos mais lidos e prevenindo por estatística os próximos blocos a serem acessados. Totalmente desenvolvido no Brasil, o ASA agiliza o trabalho do míni, equiparando-se, nos recursos, aos equipamentos de grande porte.

Os componentes utilizados pela Labo são rigorosamente testados em seu laboratório de controle de qualidade, onde uma câmara de simulação de envelhecimento descarta todos os elementos que apresentarem falhas.

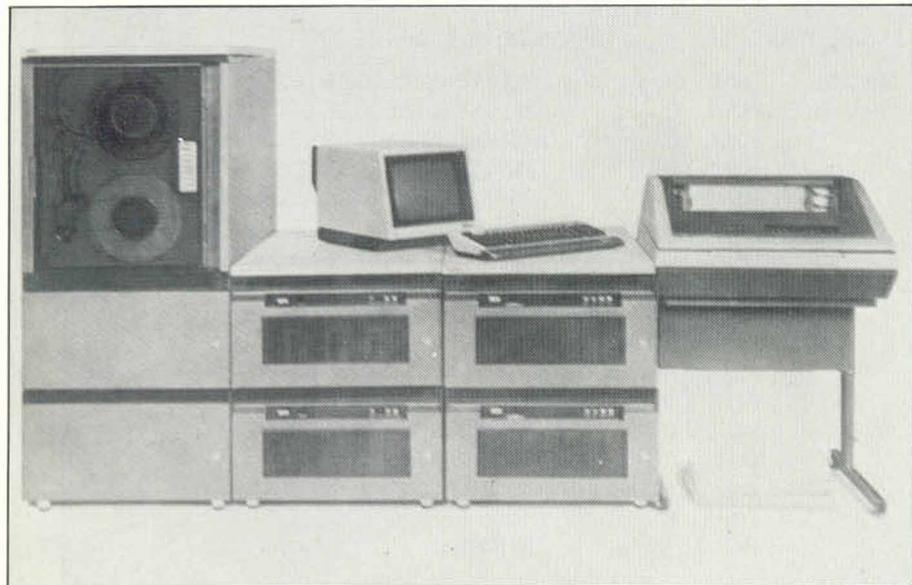
Visando oferecer soluções imediatas aos problemas de seu cliente, a Labo dispõe de uma série de programas prontos, destinados a diversos segmentos do mercado: consórcios, hospitais, transportadoras, frotistas, administradoras de imóveis, operadoras de open market, agroindústrias etc. Todos os programas são protegidos por códigos e procedimentos que impedem o acesso de pessoas não autorizadas às informações.

Com filiais nas principais cidades do Brasil, a Labo dedica especial atenção à manutenção de seus equipamentos, incluindo suporte remoto por telediagnóstico — serviço inédito que, através das linhas telefônicas, detecta irregularidades nos sistemas e programas.

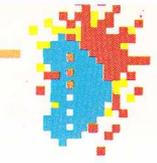
Entre esses programas destacam-se o SACIL — Sistemas Aplicativos Comerciais Integrados Labo e o PLANCOI — Planejamento e Controle Industrial, destinado, principalmente, a empresas de médio e grande porte.

#### Programas protegidos

O 8043, o maior dos minis fabricados pela Labo, conta com recursos de proteção a programas e dados confidenciais. Dispõe também de um módulo ASA, que reduz os acessos aos discos, agilizando a execução.





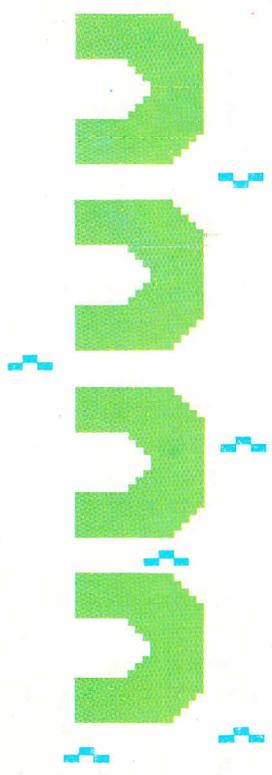


do para alvejar grupos de alienígenas invasores. Os inimigos se movem ameaçadoramente na direção da superfície da Terra. Perde-se uma "vida" quando o canhão de raios laser é atingido pelo fogo inimigo, ou quando os invasores conseguem chegar ao solo (alcançam a parte inferior da tela).

Há várias diferenças entre o fliperama original e as versões disponíveis para micros domésticos. Em vez de surgirem do nada, as hordas invasoras agora saem de um grande foguete, situado à esquerda da tela de tevê. Os próprios alienígenas são mais vivamente coloridos, formados por sprites mais complexos. As barreiras defensivas, atrás das quais o canhão de raios laser podia esconder-se na versão original, agora desaparecem, e os invasores percorrem uma distância menor antes de alcançar a parte inferior da tela. Mas um elemento permanece o mesmo: o som intimidante de batidas de coração, que acompanha a descida dos alienígenas, intensificando-se à medida que os monstros se aproximam. Isso acaba provocando uma estonteante descarga de adrenalina — provável razão do enorme sucesso do jogo.

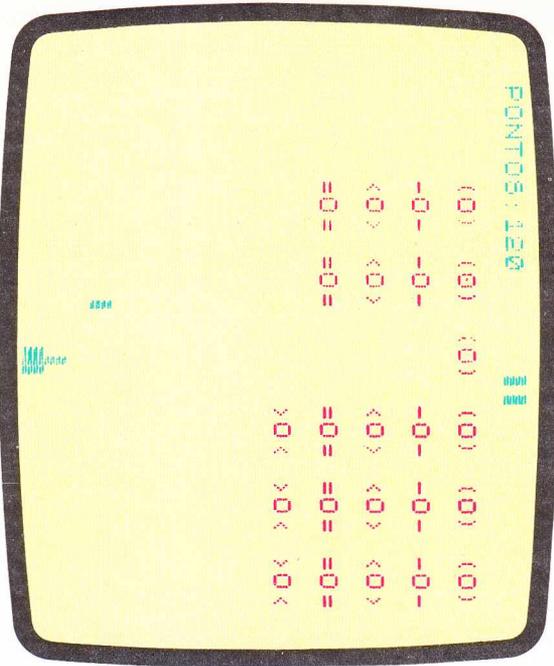
Outra característica tanto do jogo de fliperama como das versões para microcomputadores é o bônus misterioso, um prêmio oferecido ao jogador quando ele consegue atingir um dos discos voadores que ocasionalmente cruzam a tela da esquerda para a direita. Apesar do tempo e da disponibilidade de componentes de software mais sofisticados, o Space Invaders vem mantendo seu apelo, mostrando-se um jogo extremamente excitante e agradável — um verdadeiro "clássico do software".

**Space Invaders:** Para os computadores Apple II, Atari, Commodore 64, Sinclair Spectrum e TRS-80.  
**Editor:** Atari Corporation, Inglaterra.  
**Autor:** Atari.  
**Joystick:** Necessário.  
**Formato:** Cartucho.



# INVASORES PARA A LINHA SINCLAIR

Este é um jogo inspirado num clássico do fliperama, o Space Invaders. Você precisa defender a Terra dos invasores que se aproximam para destruí-la com suas bombas. No final, o planeta acabará sendo invadido, mas seu objetivo é conquistar o maior número de pontos. Quando atingir 1.500 pontos, você receberá um canhão extra e iniciará nova fase, mais difícil que a anterior. Seus controles são: [5] para mover para a esquerda, [8] para a direita e [0] para atirar. Após digitar o programa, inicie-o com o RUN.



**Prova de resistência**  
 Aproximando-se da Terra, os inimigos jogam suas bombas para eliminar toda e qualquer resistência. O jogo termina quando os alienígenas invadem o planeta ou destroem todos os canhões.

```

10 REM
11 SCREEN=0
12 GOTO 100
130 REM *****
140 REM *****
150 REM *****
160 REM *****
170 REM *****
180 REM *****
190 REM *****
200 REM *****
210 REM *****
220 REM *****
230 REM *****
240 REM *****
250 REM *****
260 REM *****
270 REM *****
280 REM *****
290 REM *****
300 REM *****
310 REM *****
320 REM *****
330 REM *****
340 REM *****
350 REM *****
360 REM *****
370 REM *****
380 REM *****
390 REM *****
400 REM *****
410 REM *****
420 REM *****
430 REM *****
440 REM *****
450 REM *****
460 REM *****
470 REM *****
480 REM *****
490 REM *****
500 REM *****
510 REM *****
520 REM *****
530 REM *****
540 REM *****
550 REM *****
560 REM *****
570 REM *****
580 REM *****
590 REM *****
600 REM *****
610 REM *****
620 REM *****
630 REM *****
640 REM *****
650 REM *****
660 REM *****
670 REM *****
680 REM *****
690 REM *****
700 REM *****
710 REM *****
720 REM *****
730 REM *****
740 REM *****
750 REM *****
760 REM *****
770 REM *****
780 REM *****
790 REM *****
800 REM *****
810 REM *****
820 REM *****
830 REM *****
840 REM *****
850 REM *****
860 REM *****
870 REM *****
880 REM *****
890 REM *****
900 REM *****
910 REM *****
920 REM *****
930 REM *****
940 REM *****
950 REM *****
960 REM *****
970 REM *****
980 REM *****
990 REM *****
1000 REM *****

```