MICRO COMPUTADOR

CURSO BASICO

Sumário

VOLUME 1

VOLUME 1

Por dentro do hardware

Programação Basic

Software

Sumário

VOLUME 2

Chips & bytes

Conexões

Fundamentos

Hardware

Os precursores

Perspectivas

Por dentro do hardware

VOLUME 2

Programação BASIC

Software

Som e luz

Apresentando o som...

"Som e luz" é a nova seção que vai ajudá-lo a melhor utilizar os recursos sonoros e gráficos existentes em seu computador.

À medida que os microcomputadores foram se desenvolvendo, tornou-se maior a quantidade de recursos disponíveis. Os dispositivos para jogos concorreram para a aceitação comercial de cada novo modelo e também investiram-se muito tempo e esforço no aperfeiçoamento de capacidades gráficas. Embora sua importância não tenha sido reconhecida de imediato, os recursos sonoros e a produção de música igualmente se desenvolveram com rapidez.

Se você perguntar aos programadores de jogos de maior aceitação sobre a importância das rotinas de som em seus programas, eles as colocarão, provavelmente, em posição muito próxima ao plano conceitual do jogo e aos gráficos. O uso engenhoso de efeitos sonoros e de música amplia consideravelmente a capacidade de entretenimento e de atração de todos os jogos do tipo fliperama.

Além das aplicações em jogos, é possível estender conhecimentos de música com a utilização dos recursos sonoros de seu microcomputador. Em muitos casos, comandos especiais para música são proporcionados em BASIC, habilitando o usuário a desenvolver programas curtos para tocar melodias bastante complexas, que até mesmo incluem acordes. Alguns computadores também dispõem de meios de mudar a característica do som, para torná-lo mais agradável ao ouvido, ou para imitar os sons de instrumentos musicais conhecidos. Em todos esses casos, o teclado do computador pode ser "adaptado", por meio de um programa adequado, para atuar de modo semelhante ao teclado de um piano, o que capacita o usuário a tocar a música em "tempo real"

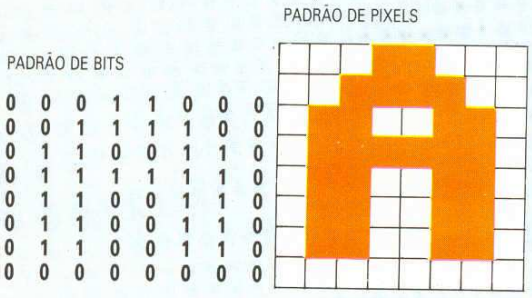
Mesmo que você tenha pouco conhecimento de programação, é possível escrever programas curtos e simples para a produção de sons musicais razoavelmente bem elaborados. Se quiser aproveitar melhor os recursos sonoros, a maioria das lojas de software tem programas de música completos, com os quais você poderá escrever e tocar melodias imediatamente. Qualquer que seja seu propósito, é interessante compreender como seu computador gera, modela e dirige a emissão de som.

...e a luz

Alta e baixa resolução

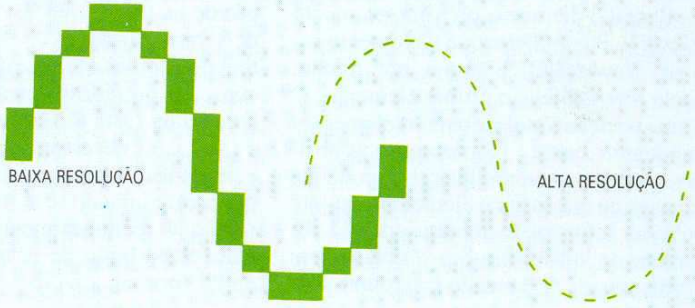
Os gráficos dos microcomputadores podem ser divididos em duas categorias: de baixa resolução e de alta resolução. A diferença entre as duas será melhor compreendida se examinarmos como é feito um caractere (letra, número ou forma).

Se você olhar detalhadamente um caractere standard impresso na tela da televisão, notará que sua forma é constituída por um conjunto de pequenos quadrados, chamados elementos da figura, ou pixels, e cada caractere ou forma que aparece na tela é um arranjo desses elementos de acordo com um padrão. Na maioria dos microcomputadores, os caracteres são formados por um quadriculado de 8 × 8 linhas, reunindo 64 pixels. A letra A pode ser constituída por um padrão de pixels como o seguinte:



Cada pixel iluminado no quadriculado pode ser representado na memória do computador pelo dígito 1 e cada pixel não iluminado pelo dígito 0. Oito bits constituem 1 byte e, assim, cada linha do quadriculado de caracteres pode ser armazenada em uma posição individual na memória do computador. Portanto, são necessárias oito posições de memória para comportar um único caractere.

As imagens gráficas às vezes são constituídas por blocos que têm o tamanho de um quarto, meio ou um quadriculado inteiro de caractere. Os gráficos projetados com esses blocos grandes e simples são denominados "de baixa resolução". Em muitos microcomputadores é agora possível desenhar imagens gráficas constituídas de pixels individuais; essas imagens são chamadas "de alta resolução". A diferença entre os dois tipos pode ser notada no gráfico de uma curva senoidal, como na ilustração abaixo.



Osciladores

São circuitos eletrônicos que produzem sinais (impulsos) repetitivos. Quando ampliados e transmitidos a um reprodutor de som, esses sinais geram emissões de uma determinada altura. A quantidade de osciladores nos microcomputadores varia de um a quatro — quanto maior o número de osciladores, maior o número de notas que você poderá tocar ao mesmo tempo.

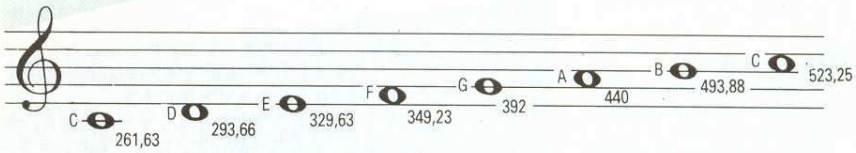
São três as características que descrevem o som criado: freqüência, envelope (que inclui o volume) e forma de onda. A freqüência é apresentada aqui; os geradores de envelope e a forma da onda, na próxima parte dessa seção.

Freqüência

É a mais importante das características necessárias ao controle, pois determina a altura do som. A freqüência é o número de vezes que um sinal (ou impulso) se repete a cada segundo, e é medida em hertz (Hz, ciclos por segundo). Os sons que podem ser captados pela audição humana estão em freqüências entre 20 Hz e 20.000 Hz. Embora não possam ser ouvidas, as freqüências abaixo de 20 Hz podem ser utilizadas para modificar as características dos sons audíveis. Esta técnica é chamada modulação.

Todavia, não é necessário aprofundar muito a descrição das freqüências. O que você realmente precisará saber é como tocar as notas musicais. A facilidade para fazê-lo varia enormemente de uma máquina para outra. Algumas têm comandos em BASIC que produzem as freqüências por você, de modo que seu trabalho consistirá apenas em especificar um número para a altura ou então o símbolo de uma nota musical — A, A#, B, e assim por diante. Em outras, esse procedimento é consideravelmente mais difícil, porque é preciso consultar uma tabela no manual do usuário, para ver a freqüência correspondente à nota, e armazenar, pelo comando POKE, o valor da freqüência em uma posição da memória. A ilustração abaixo apresenta as conversões exatas para a escala de dó central (C); é útil também para a programação de música em código de máquina, porque o BASIC, nesse caso, não poderá ajudá-lo a calcular as freqüências.

Das notas às freqüências
Você pode calcular a freqüência de cada nota da escala multiplicando um semitom abaixo dela por 1,0594631. Isto pode parecer um tanto estranho, mas, se a multiplicação for feita 12 vezes, a freqüência original será duplicada. Há 12 semitons em uma oitava (a diferença entre duas notas de mesmo nome); desse modo, a duplicação da freqüência eleva o tom para uma oitava acima. A ilustração abaixo fornece as conversões exatas dos símbolos das notas musicais (para a escala de dó central) nas freqüências correspondentes.



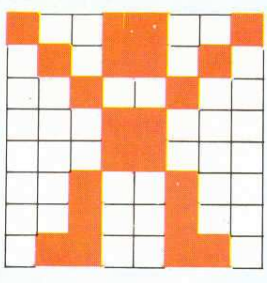
Definição pelo usuário

Para criar imagens atraentes e originais na tela será interessante recorrer a caracteres que não sejam do conjunto normal de caracteres alfanuméricos. O Vic-20 e o Commodore 64 possuem um conjunto especial de caracteres gráficos que pode ser utilizado diretamente a partir do teclado, porém, mesmo estes não abrangem todas as possibilidades. Na maioria dos microcomputadores é possível criar novos caracteres. Isso é conseguido pela redefinição dos padrões binários de oito posições de memória, nas quais o caractere é armazenado. No processo, o conjunto original de padrões binários geralmente se perde ou é "sobreposto", e o caractere "definido pelo usuário" assume algumas das propriedades daquele que está sendo substituído na memória. Desse modo, o novo caractere pode ser utilizado em instruções PRINT, pelo simples pressionar da tecla do caractere substituído. Eis aqui um exemplo de caractere definido pelo usuário, acompanhado por seus códigos binários correspondentes.

PADRÃO DE BITS

128	64	32	16	8	4	2	1
1	0	0	1	1	0	0	1
0	1	0	1	1	0	1	0
0	0	1	0	0	1	0	0
0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	0
0	0	1	0	0	1	0	0
0	0	1	0	0	1	0	0
0	1	1	0	0	1	1	0

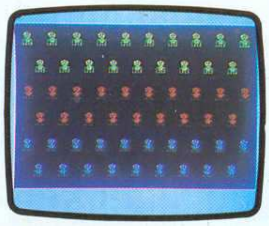
PADRÃO DE PIXELS



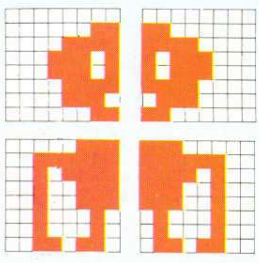
A facilidade com que os caracteres definidos pelo usuário podem ser constituídos varia enormemente, de acordo com o computador utilizado. Por exemplo, com o comando USR, dos computadores compatíveis com o Sinclair (TK's, CP 200, Ringo), é necessário apenas o fornecimento dos padrões binários adequados; no Commodore 64 o usuário tem de, primeiro, transpor o conjunto completo de caracteres, de ROM para RAM, antes de armazenar na memória, com o comando POKE, os oito equivalentes decimais dos padrões de bits que constituem a forma. Entretanto, vários programas para desenho de caracteres encontrados em fornecedores independentes facilitam o trabalho do usuário do Commodore 64.

Para criar figuras maiores é possível agrupar dois ou mais caracteres definidos pelo usuário. As figuras dos alienígenas (mostradas à direita) foram construídas a partir de quatro caracteres definidos pelo usuário. O programa, que pode ser processado no Commodore 64, imprime os grupos de caracteres na tela em três cores. Os caracteres foram criados usando-se uma pequena rotina para transportar o conjunto normal de caracteres de ROM para RAM e para substituir os caracteres gráficos ▞, ▌, ▚ e ' ' pela leitura de números decimais em instruções DATA, utilizando comandos POKE para colocá-los nas posições apropriadas. Você pode ter maiores detalhes de como fazer isso em outras partes do curso.

Mesmo quando o computador dispõe de sprites (ver p. 152), há geralmente um limite para a quantidade que pode ser apresentada ao mesmo tempo na tela; assim, os gráficos definidos pelo usuário mostram-se úteis quando muitas formas semelhantes têm de ser apresentadas na tela ao mesmo tempo.



Extraterrestre
Essas criaturas alienígenas foram criadas de quatro caracteres, definidos pelo programador. O método pode ser empregado em muitos micros que não têm sprites.



Dicas sobre o som

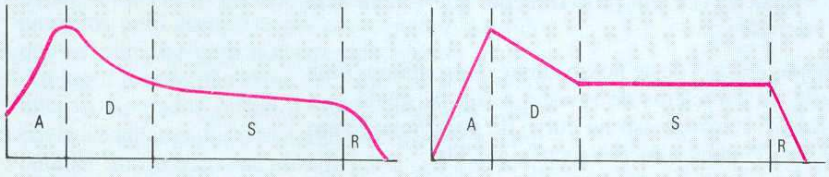
Continuamos a explicar a música por computação.

Como parte de nossa série sobre a produção de sons, vamos ver agora alguns dos aspectos mais desenvolvidos nos microcomputadores.

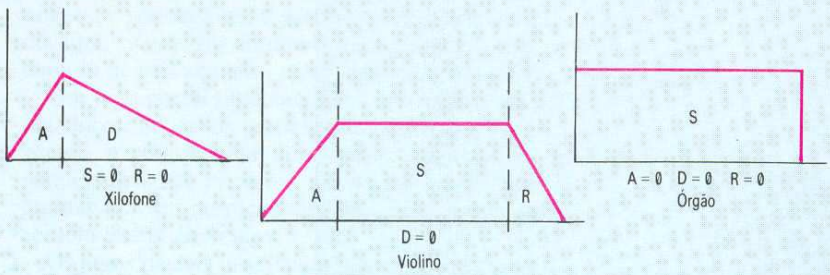
Geradores de envelope

O envelope de um som é o padrão das variações de volume, desde o instante da emissão do som até o momento em que se extingue. Envelopes semelhantes aos de uma nota de piano e de outros instrumentos estão ilustrados abaixo. Os envelopes geralmente se dividem em quatro partes, chamadas Attack (ataque), Decay (decaimento), Sustain (sustentação) e Release (liberação). A sigla ADSR é referente a essas quatro partes. No caso do piano, o volume se eleva rápido ao nível máximo, quando se pressiona a tecla (ataque); a seguir, decresce lentamente (decaimento) para um volume mais ou menos constante (nível de sustentação), enquanto se mantém a tecla pressionada; e, por fim, decai rapidamente para zero (liberação), quando a tecla deixa de ser pressionada. Observe que a sustentação é um nível de volume, enquanto ataque, decaimento e liberação representam intervalos de tempo. O dispositivo que controla os quatro aspectos do envelope é chamado gerador ADSR.

Aliás, qualquer dispositivo que ligue e desligue um som é uma espécie de gerador de envelope.



Na maioria dos computadores, para essa finalidade, não há nada que seja sofisticado: apenas um gerador de "bips", que consiste em um simples interruptor para ligar o som a um volume constante por determinado tempo. No exemplo acima, os elementos A, D e R equivalem a zero.



Como criar imagens

Animação simples pelos comandos em BASIC.

Tendo sido vistos os princípios básicos da produção de gráficos por computador, passamos agora a examinar os procedimentos para obtermos animação simples. Primeiro, é necessário detalhar os meios de fazer os caracteres aparecerem na tela e depois controlar seu posicionamento usando um programa em BASIC. Há dois métodos principais a que o programador pode recorrer: o comando POKE e o comando PRINT, com as funções a eles vinculadas.

O comando POKE coloca números em qualquer posição específica da memória. Há um conjunto especial de posições de memória no interior de cada computador, cada qual relacionada a uma determinada posição do caractere na tela. Em telas comuns, de 25 linhas por 40 colunas, há 1.000 posições reservadas para esta finalidade. Cada posição possui um número que corresponde a um caractere particular no conjunto de caracteres da máquina, que pode ser o código standard ASCII do caractere (ver p. 214) ou um código designado pelo fabricante da máquina. Além dessas posições de códigos de caracteres, há geralmente outro conjunto de posições que contém informação sobre a cor do caractere apresentado em qualquer posição da tela.

No Commodore 64, por exemplo, há pouquíssimos comandos para gráficos em BASIC que auxiliam o programador, e o comando POKE é com freqüência utilizado para criar imagens na tela. Os endereços das posições que contêm códigos de caracteres vão de 1024 a 2023, e as posições que contêm informação sobre as cores têm endereços que vão de 55296 a 56295. No Commodore, o caractere A tem o código de tela 1 e a cor preta é representada pelo código de cor 0; desse modo, os comandos exigidos para colocação de uma letra A na cor preta, no canto esquerdo superior da tela são:

```
10 POKE 1024,1
20 POKE 55296,0
30 END
```

Modificações simples podem ser feitas para apresentar uma linha de letras A na cor preta, na primeira linha superior da tela:

```
10 FOR X = 0 TO 39
20 POKE 1024+X,1
30 POKE 55296+X,0
40 NEXT X
50 END
```

As letras A são produzidas pelo loop FOR-NEXT, que aumenta os endereços das posições de caracteres e cores em uma unidade de cada vez. Se incorporarmos um comando que elimine uma letra A anterior, toda vez que uma nova letra A for criada, a letra

parecerá mover-se na tela: será uma forma elementar de animação.

O código de caractere do Commodore para um espaço é 32. É necessário apenas que o programador o coloque na posição adequada, isto é, atrás da nova letra A que será apresentada. Insira a seguinte linha no programa acima:

```
35 POKE 1024+X,32
```

Ou então:

```
15 POKE 1024+(X-1),32
```

O armazenamento de números em posições através do comando POKE, para a produção de gráficos, é um procedimento trabalhoso. Esse método, provavelmente, será mais bem empregado na criação de fundos estáticos, mediante as instruções READ e DATA, para dar entrada aos códigos de caracteres antes de armazená-los na posição correta pelo comando POKE.

A maioria dos microcomputadores tem muitos comandos para gráficos como parte do conjunto standard de instruções em BASIC, o que permite ao usuário criar imagens coloridas e atraentes, apenas com algumas instruções simples. Os comandos para construção de formas geométricas com alta resolução geralmente fazem parte do equipamento. Com essas instruções, o usuário pode formar uma trama de pontos na tela e uni-los com linhas retas; desenhar quadrados, arcos e círculos; e colorir o interior das figuras desenhadas.

É muitas vezes um procedimento simples retraçar as formas anteriores com a mesma cor do fundo, o que resulta no desaparecimento dessas formas. O traçamento, sua eliminação e o retraçamento em nova posição, realizados de modo rápido, são, de fato, a base da animação simples de gráficos. O realismo da ação depende em grande parte da rapidez com que o processo é realizado. Os sprites são muito mais eficazes porque não exigem a eliminação do traçado à medida que a forma muda de posição, e isto aumenta bastante a velocidade em que parecem se mover. Na verdade, o desenvolvimento dos sprites torna possível, pela primeira vez, escrever jogos tipo fliperama em BASIC, mas, anteriormente, o código de máquina era fundamental.

Os princípios de movimentação dos gráficos podem ser empregados em combinação com programas simples em BASIC. Muitos microcomputadores têm comandos que permitem ao usuário imprimir posições específicas na tela, tais como o comando PRINT AT e outros.

Traçando uma diagonal

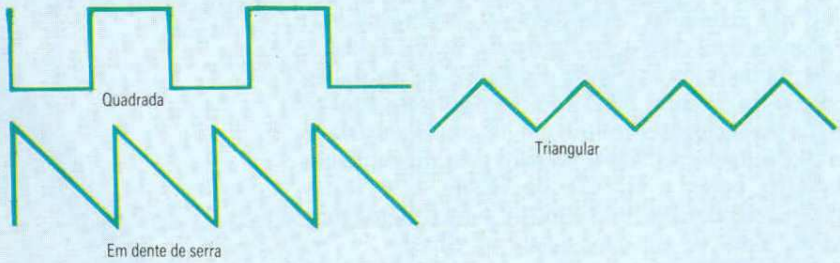
Eis um programa curto na linguagem BASIC compatível com os micros tipo Apple (Unitron, Maxxi, Exato, Micro Engenho e outros) que traça uma linha diagonal à tela, no modo de baixa resolução. As cores geradas em cada ponto são aleatórias:

```
10 REM LINHA DIAGONAL BAIXA RESOLUCAO
20 GR
30 COLOR = RND(16)*16
40 FOR X = 0 TO 39
50 PLOT X,X
60 NEXT X
70 GOTO 30
```

Forma da onda

A forma da onda é o "formato" ou contorno repetitivo do sinal produzido por um oscilador (ver p. 247) e que dá ao som sua característica própria. Dois instrumentos diferentes tocando notas na mesma altura não produzem sons iguais e isto, em parte, acontece porque as formas das ondas são diferentes. As formas de ondas mais comuns são quadradas (ou de pulso), triangulares e em dente de serra, como é mostrado na ilustração.

A maioria dos microcomputadores proporciona uma única forma de onda, geralmente do tipo pulso. Por isso, nota-se um áspero som sintetizado.



O Commodore 64 é atualmente o computador mais interessante em produção de música, porque permite a escolha entre as três formas básicas de ondas em cada um dos três osciladores. As formas de onda podem ser modificadas com o uso de filtros, que alteram a tonalidade, de modo muito semelhante aos controles grave/agudo dos equipamentos hi-fi e têm o efeito de tornar o som mais suave. Ainda mais útil é a capacidade de trocar os conjuntos de filtro durante a execução da nota. Isso permite a simulação de sons naturais de modo mais aproximado e a produção de sons não-naturais mais interessantes.

Ruído

O ruído é um tipo complexo de som resultante de vibrações não previsíveis. A audição humana não consegue distinguir entre padrões repetitivos e, assim, não pode discernir nenhuma tonalidade específica. Imagine alguns sons como os da chuva, do vento e do trovão. Esses ruídos não soam de forma igual porque são uma combinação de ruídos (imprevisíveis) em que há a predominância de alguns tons. A maioria dos microcomputadores com recursos para ruídos permite alguma forma de modulação do ruído ou a sua mistura com notas puras. Os efeitos possíveis variam do "vento zunindo" a explosões violentas.

Saída

A saída é feita, geralmente, através do alto-falante do televisor. Se for este o caso, você poderá ligar o televisor a seu equipamento de hi-fi, através de um videocassete. Alguns computadores, entretanto, só podem emitir sons por um pequeno alto-falante embutido. Nestes, é impossível obter sons de boa qualidade sem fazer uma adaptação do hardware ou adquirir equipamento acessório, externo. Seu computador poderá ter uma saída adequada para ligação direta com o aparelho de hi-fi, o que tornará compensador o esforço exigido na produção de configurações sonoras complexas.

O ressoar do Vic

Um exame detalhado da produção de som no Vic-20...

Um dos primeiros microcomputadores lançados na Inglaterra foi o Vic-20. Portanto, seus recursos podem parecer um pouco limitados em comparação com equipamentos que surgiram mais recentemente. Além disso, a Commodore não tornou muito fácil a elaboração de programas de som ou música, pois a linguagem BASIC do Vic-20, tanto como a do Commodore 64, não possui comandos especificamente vinculados à produção de som. Todo controle sonoro é obtido por uma série de comandos POKE armazenados nas posições de memória. Este princípio também se aplica ao Commodore 64, e as técnicas aqui descritas para o Vic-20 são úteis ainda ao usuário do Commodore 64. O grau de controle sonoro obtido está restrito ao volume (equivalente ao envelope com A = D = R = 0), à freqüência em três osciladores e a um gerador de ruído. A saída é feita exclusivamente através do alto-falante do televisor. Além disso, devido às imprecisões do método com que o Vic-20 seleciona freqüências, é impossível obter a altura exata para todas as notas da escala musical.

Dispondo só desses recursos, o Vic-20 é pouco útil na produção de música; mesmo assim seus recursos limitados podem ser empregados na criação de "melodias" com dois ou três acordes de notas.

Controle de som

O Vic-20 é equipado com três osciladores de ondas quadradas e um gerador de ruído. Cada oscilador alcança aproximadamente três oitavas de som, distribuídas nas seguintes freqüências:

Osc. 1	Osc. 2	Osc. 3	Amplitude da freqüência (Hz)	Oitava
●			(65,41-123,47)	1
●	●		(130,81-246,94)	2
●	●	●	(261,63-493,88)	3
	●	●	(523,25-987,77)	4
		●	(1046,5-1975,53)	5

Esta distribuição permite ao usuário o alcance de um total de cinco oitavas com a utilização de pelo menos um oscilador em cada uma delas. A oitava três, que se inicia em dó central e possui o padrão de referência A, a 440 Hz, pode ser utilizada com todos os três osciladores.

O controle dos osciladores é executado com a alteração dos conteúdos de cinco posições de memória, da seguinte forma:

Posição de memória	Oscilador
POKE 36874,X	1
POKE 36875,X	2
POKE 36876,X	3
POKE 36877,X	ruído

Em cada caso, X representa um número inteiro entre 135 e 241 (o valor 0 desliga o oscilador correspondente), que se refere à tabela de equivalentes de valores de notas, na página 73 do manual que acompanha o Vic-20. Antes que a freqüência escolhida possa ser ouvida, o nível do volume deve ser determinado, da seguinte forma:

```
POKE 36878,V
```

onde o valor de V pode ser determinado entre 0 (desligado) e 15 (alto), o que afeta a produção de ruídos e todos os osciladores. Por exemplo:

```
POKE 36874,219:POKE 36875,219:POKE
   36876,219:POKE 36878,7
```

Esse procedimento coloca a referência A (isto é, a nota lá) em 440 Hz, no oscilador 1, em uma oitava acima no oscilador 2 e em outra oitava acima no oscilador 3 — os três osciladores colocados a um volume médio de alcance 7. Lembre-se de armazenar, pelo comando POKE, o valor 0 em cada posição, quando for desligar os osciladores.

Notas e pausas

Sem uma determinada duração para cada nota e as pausas corretas entre elas, a seqüência torna-se embaralhada. Para simplificar os períodos de "espera", pode-se empregar dois métodos. O primeiro consiste nos loops FOR-NEXT, nos quais a pausa é cronometrada por um loop vazio, como o seguinte:

```
10 POKE 36878,7
20 POKE 36876,203
30 FOR P=1 TO 200
40 NEXT P
50 POKE 36878,0
60 POKE 36876,0
```

Essa seqüência de comandos produz a nota ré # durante a execução de 200 procedimentos FOR-NEXT. Entretanto, esse método depende de uma cuidadosa contagem externa do loop, para se obter precisão. Um modo mais fácil e elegante de determinar as durações e as pausas é usar o relógio embutido no Vic-20, que cronometra em sexagésimos de segundos (jiffys) e pode ser referenciado de dentro de um programa com o emprego da variável TI. Esse procedimento é útil, pois permite a elaboração de um comando para "esperar" durante um período de tempo cronometrado com precisão, como segue:

```
10 POKE 36878,7
20 POKE 36876,203:D=TI
30 IF TI-D<15 THEN 30
40 POKE 36878,0
50 POKE 36876,0
```

Esses comandos tocam as mesmas notas do programa anterior, mas por um período de 15 jiffys (um quarto de segundo). Quando o som é ligado, a variável D recebe o valor TI. A linha 30 conta 15 jiffys antes de prosseguir para a linha 40. As melodias podem ser elaboradas com o emprego do mesmo procedimento para realizar uma pausa antes de tocar uma nota diferente, e assim por diante.

Esclarecendo o Dragon

... e a capacidade gráfica do micro inglês Dragon 32.

O computador Dragon 32 dispõe de um dialeto próprio da linguagem BASIC, conhecido como Microsoft Extended Colour Basic — Basic Ampliado para Cores da Microsoft. Vários outros computadores também seguem essa versão do BASIC, especialmente os da linha Tandy (TRS) que operam em cores.

O BASIC da Microsoft é de uso fácil e possui uma boa variedade de comandos para desenhar linhas, círculos e outras figuras geométricas. Depois de desenhadas, as figuras podem ser coloridas, proporcionando imagens interessantes na tela e pouco empenho na programação.

O Dragon 32 tem sete níveis de resolução, o que dá ao usuário a possibilidade de trabalhar com a tela dividida em 512 pontos individuais, no nível mínimo, e até 49.152 pontos, no nível máximo. Há oito cores disponíveis, mas a escolha pode ser limitada a quatro ou mesmo a duas cores em operações com alta resolução.

Modos de resolução

A tela comum de dezesseis linhas por 32 colunas de caracteres forma o nível mais baixo de resolução e o comando PRINT@ permite a colocação do caractere em qualquer uma das 512 posições na tela. Além do conjunto normal de caracteres, há dezesseis caracteres gráficos de baixa resolução, disponíveis em oito cores.

O outro modo de resolução divide a tela em 32 linhas e 64 colunas. Assim, o tamanho de cada quadrado corresponde a um quarto do tamanho do caractere normal. Pontos desse tamanho podem ser traçados na tela pelo comando SET e eliminados pelo comando RESET.

Ambos os métodos podem ser utilizados ao mesmo tempo, quando são denominados telas de texto de baixa resolução. Há também cinco níveis de telas de alta resolução, mas não podem ser apresentados simultaneamente ou com telas de baixa resolução. Os cinco modos de alta resolução permitem escolhas de acordo com o padrão de resolução e o número disponível de cores, e são selecionados pelo comando PMODE.

PMODE	Resolução	Cores disponíveis
0	128*96	2
1	128*96	4
2	128*192	2
3	128*192	4
4	256*192	2

Naturalmente, há intercâmbio entre a resolução, a cor e a capacidade de memória necessária para armazenar dados da tela e isto deve ser levado em conta quando forem escritos programas extensos em BASIC que também empreguem imagens de alta resolução.

Embora permitindo apenas um número limitado de cores em alta resolução, o Dragon tem o recurso de, entre dois conjuntos de cores, selecionar um. Isso é feito pelo comando SCREEN. Por exemplo, a instrução SCREEN 1,0 escolhe uma tela de alta resolução e o conjunto de cores 0. A instrução SCREEN 1,1, por sua vez, escolhe uma tela de alta resolução, mas com um conjunto alternativo de cores.

PAINT
Este comando é muito útil como auxiliar na produção de figuras interessantes. Com o emprego do PAINT, o computador começa a colorir a tela a partir de determinado ponto, até uma linha limite. Isso significa que círculos, triângulos e outras figuras fechadas podem simplesmente ser preenchidas.

DRAW
O comando DRAW imita o movimento de um lápis na tela, permitindo ao usuário desenhar linhas em quatro direções. O comando DRAW também possibilita a rotação ou a ampliação da figura terminada pelo usuário.

GET e PUT
O comando GET instrui o computador a armazenar uma imagem da tela na memória e o PUT faz com que essa imagem seja reapresentada.

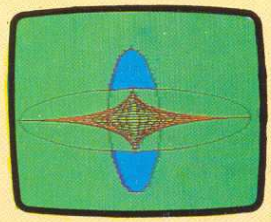
PSET e PRESET
Estes são os equivalentes em alta resolução dos comandos SET e RESET, que anteriormente examinamos. Eles permitem ligar e desligar pontos na tela. A cor do ponto também pode ser facilmente determinada.

LINE
Este comando une dois pontos específicos através de uma linha reta em alta resolução.

CIRCLE
Permite ao usuário desenhar círculos de alta resolução a partir de centro e raio determinados. As frações do círculo inteiro também são desenhadas de modo a formar arcos e a forma circular pode produzir elipses.

O micro Dragon 32 encontrado na Inglaterra é um computador de preço relativamente alto, que possui muitos comandos aperfeiçoados para auxílio da programação de gráficos. É mais adequado para apresentação de imagens estáticas do que para animação rápida. É apropriado para crianças que apreciam brincadeiras de aventuras.

O principal inconveniente do Dragon está em não apresentar na tela simultaneamente texto e gráficos de alta resolução. Isso significa que não pode ser usado para apresentar dados estatísticos na forma de gráficos em barras ou círculos.



Comando de cores
Imagem típica dos efeitos que podem ser obtidos no Dragon com apenas alguns de seus comandos de alto nível.

Alta resolução
Este pequeno programa para o Dragon 32 ilustra algumas de suas capacidades de alta resolução. O programa usa a instrução PMODE 3; este não é o nível mais alto, mas permite algum uso de cores.

```
10 PCLS:PMODE 3,1
20 SCREEN 1,0
30 COLOR 0,1
40 FOR X=0 TO 127 STEP 10
50 LINE(X,85)-(127,85-X/3),
   PSET
60 LINE(X,85)-(127,85+X/3),
   PSET
70 LINE(255-X,85)-(127,85-
   X/3), PSET
80 LINE(255-X,85)-(127,85+
   X/3), PSET
90 NEXT X
100 CIRCLE(127,85),128,4,0,3
110 CIRCLE(127,85),30,4,3
120 PAINT(130,30),3,4
130 PAINT(130,130),3,4
140 GOTO 140
150 END
```

Recursos modestos

A produção de som no Spectrum inglês, da Sinclair.

O Spectrum inglês está se tornando rapidamente o mais popular microcomputador de entretenimento. Dispõe de excelentes recursos gráficos coloridos e alternativas de memória disponíveis a baixo custo. Infelizmente, alguns dos recursos foram descartados em favor das vantagens de preço. Os maiores inconvenientes estão vinculados ao teclado e à linguagem BASIC fora do padrão, mas pode-se dizer que seu aspecto mais frágil são as limitações para produzir som. O Spectrum possibilita recursos mínimos de geração de som e sua produção de "música" a partir de um único oscilador, que opera por "impulsos", é insatisfatória. A duração e a altura das notas podem ser controladas, mas não é possível mudar o tom ou alterar seu envelope (ver p. 276). Outro inconveniente encontra-se no output padrão, feito através de um diminuto alto-falante piezelétrico, interno, que produz um "bip" estridente. Entretanto, a Sinclair oferece outputs alternativos de sinal, através das portas "mic" e "ear" do cassete, adequadas para amplificação externa por meio de sistema hi-fi. Esse, porém, é um benefício questionável, já que resulta em baixa qualidade sonora. Os recursos sonoros do Spectrum têm efetivamente a vantagem da simplicidade em BASIC dos comandos associados BEEP e PAUSE, que possibilitam ao usuário a compreensão mais fácil dos princípios do som produzido por meio do computador. Além disso, o equipamento tem a notável amplitude de freqüência de 10 oitavas.

Para produção de som no Spectrum é necessário ligá-lo a um equipamento hi-fi ou tornar o gerador interno de "bip" audível mediante o seguinte comando direto, antes do processamento dos programas de som:

```
POKE 23609,100
```

Esse procedimento também aumenta o volume do "clique" em feedback, que ocorre quando uma das teclas é pressionada.

Controle sonoro

Para instruir o computador quanto à saída de uma determinada nota, o comando BEEP deve ser empregado da seguinte forma:

```
BEEP d,n
```

onde "d" representa a duração da nota e "n" a sua altura. O valor da duração pode ser fixado entre 0,00125 e 10 segundos; a altura pode ser definida como o número de semitons contados a partir do dó central — o dó no meio da pauta; no piano, o dó menor central —, que recebe o valor 0, na amplitude de −60 a 69. Por exemplo, a instrução abaixo produz a nota lá, a 440 Hz, que corresponde a nove semitons acima da nota dó central, durante meio segundo:

```
BEEP 5,9
```

Para produção de uma série completa de notas com cronometragem precisa dos espaços entre elas, podemos empregar o comando PAUSE

```
PAUSE ms
```

Imagens primárias

Os recursos gráficos do Vic-20 da Commodore.

O Vic-20, como outros microcomputadores da Commodore — o PET e o Commodore 64 —, é um equipamento muito bem construído, mas isso não fica evidente em seu conjunto de instruções em BASIC. O Vic-20 não dispõe de comandos gráficos especiais para o usuário, e este precisa conhecer bem o funcionamento interno da máquina, ou comprar um dos acessórios projetados para tornar mais fácil a programação gráfica. No entanto, a Commodore proporciona um conjunto amplo de caracteres especiais que podem, com um pouco de engenhosidade, ser combinados para a obtenção de resultados interessantes.

Dispõe-se de dezesseis cores no Vic-20 e cada quadrado de caracteres pode apresentar quatro cores. A imagem na tela é constituída de 23 linhas e 22 colunas com células de caracteres de 8 por 8 pixels, que também podem ser apresentados em formato retangular de 16 por 8. O Vic-20 padronizado possibilita gráficos de alta resolução, mas essa programação é um tanto difícil para grande parte dos usuários.

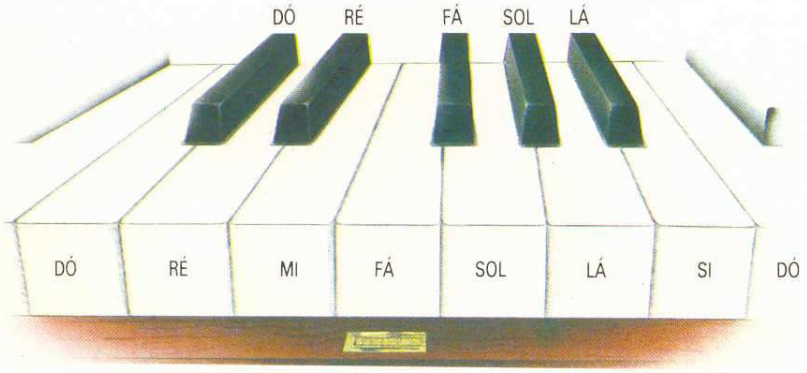
Baixa resolução

São disponíveis caracteres alfabéticos em maiúsculas ou minúsculas e mais de sessenta caracteres gráficos especiais PET. Muitas das teclas do Vic-20 são marcadas com dois pequenos quadrados, cada qual apresentando um padrão específico. Além dos quadrados de meio e de um quarto de caractere, há símbolos de cartas de baralho, desenhos de tabuleiros de xadrez, círculos e inúmeros outros símbolos que podem ser combinados na tela para desenhar gráficos, fazer tabelas, produzir inscrições em caracteres largos e criar outros efeitos. Cada carac-

onde "ms" representa o tempo em unidades de 0,001 de segundo (isto é, em milissegundos). Para produção de uma oitava na tonalidade de dó maior (dó, ré, mi, fá, sol, lá, si, dó) a partir do dó central, com a duração de meio segundo para cada nota e pausa de um quarto de segundo entre as notas, pode-se empregar o seguinte formato:

```
10 FOR I = 1 TO 8
20 READ N
30 BEEP .5,N
40 PAUSE 250
50 NEXT I
60 DATA 0,2,4,5
70 DATA 7,9,11,12
```

Este programa é um bom exemplo do formato de uma escala. A oitava abrange a nota base (nesta escala, dó central) até a nota seguinte com o mesmo nome (a nota dó seguinte, acima) e inclui 12 semitons. É chamada de uma oitava porque consiste em oito notas em uma escala maior.



Dó maior

Escalas e pianos

Mediante o comando INKEY$, o teclado do Spectrum pode ser usado de maneira a se assemelhar ao de um piano:

```
10 REM **********************
20 REM *OITAVA NO PIANO*
30 REM **********************
40 IF INKEY$ = "Q" THEN BEEP1,0
50 IF INKEY$ = "2" THEN BEEP1,1
60 IF INKEY$ = "W" THEN BEEP1,2
70 IF INKEY$ = "3" THEN BEEP1,3
80 IF INKEY$ = "E" THEN BEEP1,4
90 IF INKEY$ = "R" THEN BEEP1,5
100 IF INKEY$ = "5" THEN BEEP1,6
110 IF INKEY$ = "T" THEN BEEP1,7
120 IF INKEY$ = "6" THEN BEEP1,8
130 IF INKEY$ = "Y" THEN BEEP1,9
140 IF INKEY$ = "7" THEN BEEP1,10
150 IF INKEY$ = "U" THEN BEEP1,11
160 IF INKEY$ = "I" THEN BEEP1,12
170 GOTO 40
```

Podem ser feitos muitos aperfeiçoamentos em programas como o acima, a fim de criar um "instrumento" de teclado mais eficiente. Geralmente, o Spectrum é de pouca utilidade como fonte de som por falta de recursos para esse fim. Entretanto, o equipamento é muito útil como auxiliar no ensino de música. O único modo de obter som satisfatório será a compra de hardware para geração de som compatível com o equipamento.

tere pode ser apresentado de modo invertido (preto sobre branco, ao invés de branco sobre preto), ampliando ainda mais as potencialidades do equipamento. Assim, com paciência e imaginação, obtêm-se imagens de alta qualidade.

Os caracteres aparecem na tela pelo emprego da instrução PRINT ou pelo armazenamento, mediante o comando POKE, dos códigos correspondentes nas memórias de tela e de cores do Vic-20. A versão Commodore da instrução PRINT possui alta capacidade, permitindo ao usuário a definição da cor de cada caractere. O movimento do cursor também é controlado a partir do interior da instrução PRINT, para produzir facilmente imagens com movimento. Armazenar caracteres na tela por meio do comando POKE não é tão rápido quanto sua impressão mediante o comando PRINT, mas em certas circunstâncias este método pode ser mais adequado.

Alta resolução

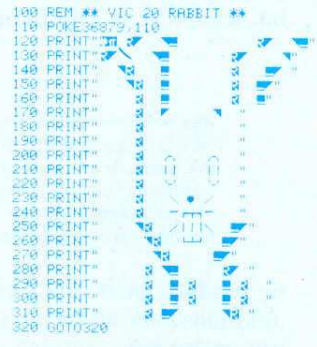
Obtêm-se gráficos de alta resolução em equipamentos Vic-20 sem acessórios complementares, mas há memória disponível suficiente para utilização de apenas aproximadamente metade da tela. O processo pelo qual a alta resolução é obtida denomina-se "mapeamento de bit", técnica que possibilita ao programador o controle de cada pixel no interior de uma determinada área da tela. Cada célula de caractere no quadriculado de 23 linhas por 22 colunas consiste em 64 pixels dispostos em oito linhas com oito pixels. Há uma relação matemática entre as coordenadas referentes a um ponto de pixel (x,y) e o bit correspondente na matriz de caracteres. Esta relação pode ser empregada em conjunto com uma combinação de comandos POKE, na produção de imagens constituídas por pixels individuais.

Cartucho Super Expander

A produção de gráficos de alta resolução por meio do mapeamento de bits pode tornar-se um processo longo e difícil. Uma abordagem alternativa consiste na aquisição do cartucho Super Expander, da Commodore, que permite ao usuário comandos de alta resolução, de música e de cores, em BASIC. Entre os comandos de alta resolução, estão o GRAPHIC, para determinar o modo da imagem, o POINT, para traçar pontos de pixel na tela, e o comando PAINT.

Há dois inconvenientes principais no emprego do cartucho: não existe comando UNPOINT (eliminação do ponto) para apagar os pontos de pixel e não é possível pintar ambos os lados de uma linha diagonal sem interferir na própria linha.

Os gráficos de baixa resolução do Vic-20 são bem elaborados e muito flexíveis, enquanto os de alta resolução implicam a aquisição de um cartucho para conexão, o que os encarece.



O coelho que corre
Esta listagem de programa para o Vic-20 exemplifica a versatilidade do conjunto especial de caracteres da Commodore. A figura de um coelho é inteiramente construída por esse conjunto de caracteres predefinidos.

O som ideal

O BASIC do Commodore 64 não combina com suas notáveis facilidades de som.

Entre os modelos existentes de microcomputadores, o Commodore 64 apresenta as mais sofisticadas facilidades para produção de som. Elas são atribuídas a um chip especial chamado Sound Interface Device — ou SID, como é mais conhecido.

A capacidade do SID é semelhante à do sintetizador monofônico comercial. Ele possui três osciladores com um alcance de oito oitavas (0-3.900 Hz em 65.536 etapas); um controle de volume principal que vai de 0 a 15; quatro formas de onda para cada oscilador (triângulo, serra, pulso variável e ruído); sincronização do oscilador; e geradores de envelope permitindo um controle ADSR para cada oscilador. Outras características são: modulação em anel; filtro programável com baixa passagem, passagem de sintonia, alta passagem, saída de entalhe (que bloqueia uma faixa estreita de freqüência) e ressonância variável; filtragem de envelope; duas interfaces para potenciômetro analógico-digital que podem ser usadas para controlar as facilidades SID; e uma entrada de áudio externo, permitindo que chips SID adicionais sejam ligados entre si. Outros sinais de áudio podem ser introduzidos, filtrados e misturados com a saída comum do SID.

Seria impossível detalhar aqui a operação de cada uma dessas funções (existem vários bons livros disponíveis), mas podemos explicar o significado de todas essas expressões. Antes de tudo, a sincronização do oscilador faz com que dois sinais separados (neste caso, duas vozes específicas) sejam reunidos harmonicamente, formando um único tom, mais complexo.

Modulação é a modificação de um sinal através de outro, afetando a freqüência ou a amplitude (volume) do som. Modulação em anel é a modulação da amplitude de uma voz por meio de uma outra. O resultado é um tom claro mas que tem um efeito desafinado, dissonante e pode ser usado para produzir um som de campainha, semelhante ao de um tambor de aço. Estes sons são considerados sobretons discordantes.

Através dos filtros, é possível eliminar tipos específicos de freqüência de um sinal. Os diferentes tipos de filtragem possíveis de se fazer no Commodore 64 têm os efeitos sugeridos pelos seus nomes: filtros de baixa freqüência cortam freqüências acima de uma freqüência específica; filtros de passagem de faixa eliminam freqüências acima e abaixo de uma determinada "faixa" de freqüência; filtros de entalhe são o inverso dos filtros de passagem de faixa — eles cortam uma determinada faixa; os filtros de alta freqüência cortam freqüências mais baixas do que uma freqüência específica; e a ressonância variável pode ser aplicada a todos os filtros acima citados

Luz-guia

Gráficos tipo Player-Missile constituem um dos pontos fortes dos Atari.

Gráficos Player-Missile

Gráficos do tipo Player-Missile ou "PM" fazem parte importante da capacidade gráfica do Atari. Eles são semelhantes em natureza aos gráficos sprite do Commodore 64, permitindo que o programador desenhe e controle até oito formas diferentes de alta resolução. Essas formas móveis operam à parte do cenário de fundo e podem ser programadas para se movimentar na frente ou atrás de quaisquer outras formas desenhadas na tela. Isto permite ao programador acrescentar uma terceira dimensão aos efeitos da tela. Os gráficos PM podem ser movimentados de modo uniforme na tela, em alta velocidade, e portanto são ideais para jogos rápidos do tipo fliperama. Eles também são usados para criar displays estáticos mais coloridos do que os elaborados com modos gráficos normais, uma vez que os objetos de PM podem ser coloridos independentemente uns dos outros e do cenário de fundo.

Assim como acontece com todos os gráficos sprite, o segredo das facilidades de gráficos PM está no hardware dedicado. Registros especiais são planejados para controlar movimento, cor e display de tela dos objetos de PM. Tudo que o programador deve fazer é colocar certos valores nesses registros para manipular os objetos. Em BASIC isso é feito usando-se o comando POKE. Assim que o número é colocado no respectivo registro através do POKE, o hardware do Atari assume o resto do trabalho. Isso ocorre em velocidade de código de máquina e, portanto, é muito mais rápido do que se o processo fosse controlado pelo BASIC.

Examinemos agora a criação de objetos de PM e os registros que os controlam. Os jogadores são desenhados a partir de uma faixa vertical, com uma largura de oito pixels e altura de 128 ou 256 pixels. Cada linha dessa faixa é representada por um único byte na memória do computador. Inserindo-se, por intermédio do comando POKE, os códigos binários adequados, é possível definir a forma de um jogador, empregando-se um método semelhante àquele usado para criar caracteres estabelecidos pelo usuário (ver p. 246). Até quatro jogadores podem ser de-

```
10 SID=54272
20 POKESID+23,0
30 POKESID+24,15
40 POKESID+5,40
50 POKESID+6,201
60 FOR N=1TO 5
70 READ FH,FL,D
80 POKESID+1,FH:
   POKESID,FL:
   REM * NOTA DE
   JOGO *
90 POKESID+4,33
100 FOR I=1 TO 300 *
   D: NEXT I
110 POKESID+4,32
120 FOR I = 1 TO 100:
   NEXT I
130 NEXT N
140 FOR I=1 TO 2000:
   NEXT I
150 POKESID+24,0
160 REM ** FH FL D **
170 DATA 57, 172, 1
180 DATA 64, 188, 1
190 DATA 51, 97, 1
200 DATA 25, 177, 1
210 DATA 38, 126, 2
220 END
```

para enfatizar as freqüências em torno dos pontos de corte. A filtragem do envelope é um caso especial: ela tem um efeito diferente das outras, pois os valores de ADSR digitalizados estabelecidos para a envolvente 3 podem ser lidos do chip SID e aplicados a um sinal, de tal forma que a estrutura harmônica muda através do curso de uma nota. Ela funciona como um filtro variável.

Estas características variadas permitem a formação de sons altamente complexos e, com eles, a produção de efeitos interessantes e emulações convincentes de instrumentos convencionais. O aspecto desfavorável do SID é que o CBM BASIC V2, dialeto fornecido juntamente com o Commodore 64, não apresenta qualquer espécie de comandos dedicados ao som. O controle é exercido através da utilização dos registros de controle 29 SID por meio dos comandos PEEK e POKE. Assim sendo, uma grande quantidade de código BASIC se faz necessária para gerar efeitos ainda que simples, e em alguns casos o BASIC não tem velocidade suficiente para fazer justiça a todas as possibilidades do SID.

Uma descrição completa dos registros de controle SID exigiria um espaço maior do que uma edição completa do MICROCOMPUTADOR — CURSO BÁSICO, mas ainda assim é possível apresentar a forma de obtenção de algumas notas harmoniosas, como se vê no programa à esquerda.

Embora o programa tenha 22 linhas, ele toca apenas cinco notas de uma melodia simples em um oscilador. A linha 20 desliga o filtro dos osciladores, a linha 30 eleva o volume principal ao máximo e as linhas 40 e 50 especificam um envelope do tipo piano. A linha 80 estabelece a freqüência da nota, 90 e 100 dão início e fim ao ciclo ADSR e selecionam uma onda serrilhada para a voz 1; e a cronometragem é realizada através de loops do tipo FOR-NEXT nas linhas 100, 120 e 140.

Para se programar som no Commodore 64 em BASIC, é necessário esforço em termos de aprender e escrever um código. Além do mais, isso pode se constituir em um exercício frustrante, uma vez que a única maneira de descobrir se um conjunto de instruções mais complexas em BASIC será executado em tempo hábil é através de tentativa e erro. Para simplificar, vale a pena pesquisar os diversos programas de edição de som. Eles são geralmente escritos em linguagem de máquina e tiram o máximo proveito das maravilhosas características do Commodore 64.

finidos dessa forma, cada um ocupando seus 256 ou 128 bytes de memória.

Cada um dos jogadores tem a figura de um míssil associada a ele, cuja largura é de 2 bits. Para criar jogadores e mísseis, é necessário inserir através do comando POKE os padrões de bits que irão definir sua forma numa determinada área da memória. A área de RAM usada pode ser escolhida pelo programador, mas o computador deve ser informado colocando-se um indicador no começo da área.

Se o programador decidir usar resolução vertical de pixel unitário, então será preciso utilizar o dobro da memória necessária para resolução vertical com dois pixels. O programa a seguir desenha o jogador 0 em resolução vertical com dois pixels, na forma de espaçonave:

```
10 REM *** DEFINIR UM JOGADOR ***
20 P = PEEK(106) - 8:REM ATRIBUI 2 K A P
   ABAIXO DO TOP DA RAM
30 POKE 54279,P: REM ATRIBUI O POINTER A
   AREA PM
40 BASE = 256*P:REM ATRIBUI O ENDERECO
   BASE DA AREA PM
50 FOR I = BASE+512 TO BASE+640
60 POKE I,0:REM LIMPA A AREA DO JOGADOR 0
70 NEXT I
80 FOR I = BASE+512+50 TO BASE+530+50
90 READ A:POKE I,A:REM DEFINE FIGURA
100 NEXT I
110 DATA 16, 16, 16, 56, 40, 56, 40, 56, 40
120 DATA 56, 56, 186, 186, 146, 186, 254, 186, 146
```

Cada jogador tem diversos registros associados a ele. Estes controlam cor, posição horizontal e tamanho. O último deles permite que o programador aumente a largura de um jogador com um coeficiente 2 ou 4. Outros registros controlam a prioridade da relação jogador-cenário de fundo. Os mísseis têm a mesma cor do seu jogador, mas seu tamanho pode ser mudado separadamente. Para aplicações de jogos, uma série de registros é separada para detectar colisões na tela entre jogadores, mísseis e cenário de fundo. Entretanto, não há registros para posição vertical de mísseis ou jogadores. Realiza-se o movimento vertical movendo o conteúdo de cada localização que mantém os padrões de bits para a figura através da área de memória destinada àquele jogador. Esta é uma tarefa executada diretamente quando se trata da linguagem Assembly, mas que seria um tanto lenta se processada em BASIC. É uma boa idéia fazer bem baixas e atarracadas as figuras que se movem no sentido vertical.

Os gráficos Player-Missile ampliam em muito o potencial gráfico do Atari, embora eles não sejam tão versáteis ou fáceis de usar como os sprites do Commodore 64. Eis aqui a continuação de um programa, para colorir uma espaçonave e movê-la da esquerda para a direita na tela.

```
130 POKE 559,46:REM PERMITE PM DISPLAY DE 2
    LINHAS
140 POKE 53277,3:REM DISPLAY DE PM
150 POKE 704,88:REM COR ROSA DO JOGADOR 0
160 GRAPHICS 0
170 SETCOLOUR 2,8,2:REM ATRIBUI
    AZUL-ESCURO AO FUNDO
180 FOR I = 0 TO 320
190 POKE 53248,I:REM ACERTA A POSICAO
    HORIZONTAL
200 NEXT I
210 END
```

Foguete PM
Antes que um objeto jogador possa ser definido, ele deve ser desenhado e os valores decimais para cada linha de pixels devem ser calculados.