MSX Article

A Arquitetura da Screen 2

Resumo

Este artigo propõe estudar a fundo a arquitetura da screen 2 do MSX, bem como apresentar algo que pode ser feito nesse modo de tela.

1- A Arquitetura da Screen 2

A tela da screen 2 do MSX 1 é composta por 768 caracteres de 8x8 pontos, possuindo 32 caracteres por linhas e 24 caracteres por coluna. A figura 1.1 mostra essa divisão da tela em caracteres.

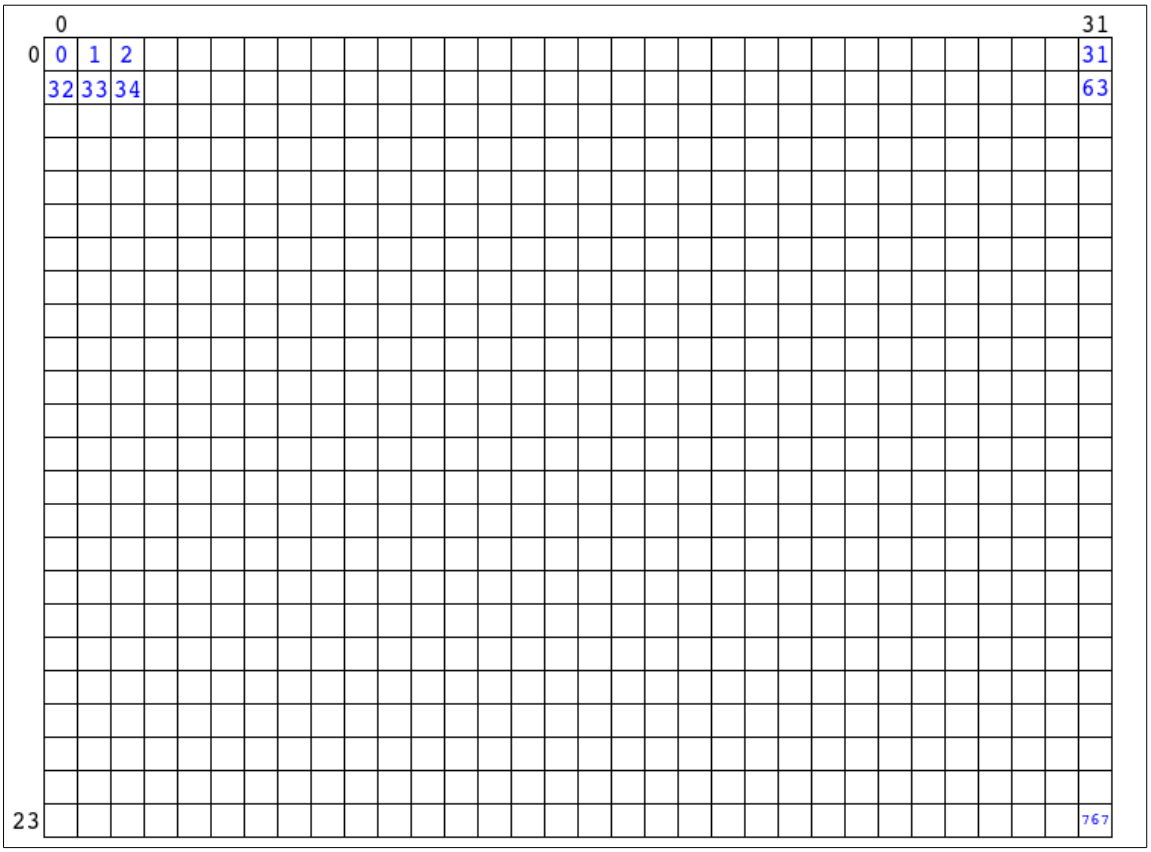

Figura 1.1. Divisão da screen 2 em caracteres de 8x8 pontos.

A ordem que os caracteres de 8x8 são armazenados na memória e são referenciados é apresentada pelos números em azul na figura 1.1.

Cada caractere é dividido por linhas, onde cada linha será controlada individualmente. Infelizmente a screen 2 não controla pontos individualmente, fazendo com que ela apresente “borrões” na tela em certas ocasiões. Controlar pontos individualmente só a partir da screen 5 do MSX 2.

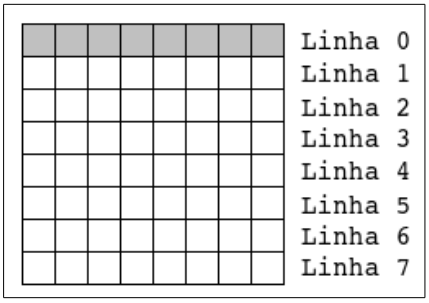

Figura 1.2. Divisão do caractere em linhas.

Os caracteres são formados por duas tabelas localizadas na memória de vídeo do MSX (VRAM): a tabela de padrões e a tabela de cores. A tabela de cores é responsável por definir a cor de frente e fundo de cada linha dos caracteres da tela, ou seja, um grupo de 1x8 pontos. Já a tabela de padrões é responsável por definir o padrão de acendimento de cada linha dos caracteres, ou seja, se cada ponto é frente ou fundo.

Por exemplo, vamos definir a linha 0 de um caractere com as cores 10 e 12 do MSX 1 com o padrão apresentado na figura 1.3.

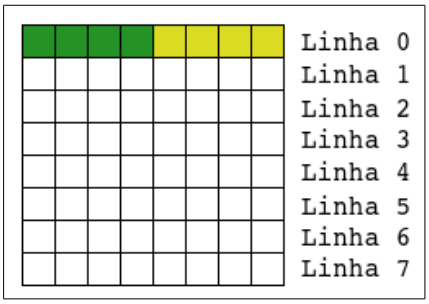

Figura 1.3. Configurando uma linha do caractere.

Cada linha do caractere é controlada pela mesma posição relativa nas tabelas de padrões e cores. Na tabela de cores, o byte controla com seus bits mais significativos a cor de frente, enquanto que os bits menos significativos controlam a cor de fundo.

	Cor de frente				Cor de fundo			
Bit	7	6	5	4	3	2	1	0

Assim, para formar a linha verde e amarela da figura 1.3, ajustaríamos o byte da tabela de cores correspondente a esta linha com o valor 10 (&B1010) para a cor de frente e 12 (&B1100) para a cor de fundo. A ordem não importa. Poderia ser 12 para a cor de frente e 10 para a de fundo. Entretanto, deverá haver a correspondência na tabela de padrões.

Cada bit da tabela de padrões controla um ponto na linha do caractere. O valor 0 indica que a cor do ponto é a cor de fundo, enquanto que o valor 1 indica que o ponto deverá ter a cor de frente.

Como definimos 10 para cor de frente e 12 para cor de fundo, o valor 0 indica que o ponto será aceso com a cor 12, enquanto que o valor 1 acende o ponto com a cor 10. Assim, o byte para a linha 0 da figura 1.3 será &B00001111 ou &HF ou ainda 15.

Para reproduzirmos esta linha na screen 2 falta ainda descobrirmos como localizar uma linha de um determinado caractere nas tabelas de padrões e cores na VRAM.

Suponhamos que eu deseje alterar a linha 4 do caractere 34 (vide figura 1.1). Este caractere se localiza na linha 1 e na coluna 2. Sabemos que cada linha ocupa exatamente 1 byte em cada tabela e que cada caractere possui 8 bytes. Dessa forma, necessitamos saber a ordem desse caractere na tabela, que no caso é 34, e multiplicá-lo por 8. Por fim, somamos o valor da linha do caractere que desejamos alterar.

Caso não saibamos a ordem do caractere na tabela, utilizamos suas coordenadas em linha e coluna para calculá-la.

$$pos\_caractere = linha \times 32 + coluna$$

No nosso exemplo, temos linha 1 e coluna 2: 1x32 + 2 = 34.

Devemos ressaltar que cada caractere da tabela de padrões e cores estão atrelados entre si, ou seja, não podemos mesclar um caractere da tabela de padrões com outro caractere da tabela de cores. Dessa forma, a fórmula para encontrar a linha de um caractere em ambas as tabelas na VRAM é:

$$endereço = inicio\_tabela + pos\_caractere \times 8 + linha\_caractere$$

Onde *inicio_tabela* é o inicio de cada tabela na VRAM. Temos:

```
Inicio da tabela de cores: 8192 ou &H2000
Inicio da tabela de padrões: 0 ou &H0000
```

Agora podemos desenhar no MSX a linha da figura 1.3 na posição de caractere 34.

```
10 SCREEN 2
20 VPOKE &H2000 + 34*8 + 4, &B10101100 ' Define a cor da linha
30 VPOKE 0 + 34*8 + 4, &B00001111 ' Define o padrao da linha
40 GOTO 40
```

Devemos ter em mente que, na realidade, as tabelas de cores e padrões da screen 2 definem formas primitivas através dos caracteres e que, ao final, formam um mosaico gigante com 768 caracteres.

1.1. A tabela de nomes

Após definirmos o formato dos caracteres, podemos controlar quais caracteres irão aparecer na tela e onde através da tabela de nomes. Dessa forma, podemos criar novos mosaicos a partir dos caracteres de 8x8 pontos presentes nas tabelas de padrões e cores.

A tabela de nomes possui o layout da figura 1.1 e define a correspondência entre uma posição física na tela e um caractere das tabelas de padrões/cores. Agora cada byte representa um caractere, onde a posição na tabela de nomes indica a posição física na tela,

enquanto que o valor do byte indica qual caractere das tabelas de padrões/cores deverá ocupar esse espaço.

Porém, nem tudo são flores. A tela é dividida verticalmente em três áreas iguais, e podemos acessar na tabela de nomes somente os caracteres das tabelas de padrões/cores presentes na mesma área física dela. Dessa forma, cada área possui 256 caracteres, numerados de 0 a 255.

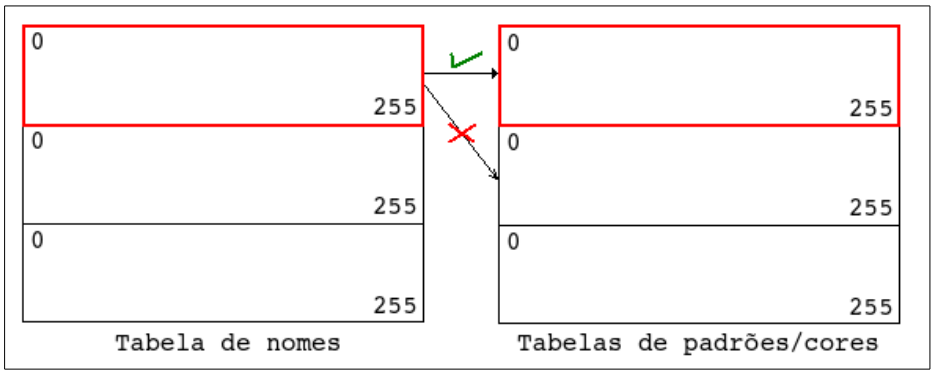

Figura 1.4. Correspondência de áreas das tabelas de nome e padrões/cores.

O exemplo a seguir irá ilustrar o uso da tabela de nomes. Vamos utilizar quatro caracteres do jogo Alcatraz, a Fuga Impossível [1].

```
10 COLOR 15,4,4:SCREEN 2
20 FOR E=0 TO 31
30 READ A$ : VPOKE E,VAL("&H"+A$)
40 NEXT E
50 FOR E=0 TO 31
60 READ A$ : VPOKE &H2000+E,VAL("&H"+A$)
70 NEXT E
80 GOTO 80
1000 ' Tabela de Padroes
1010 DATA 00,00,00,00,00,00,00,00
1020 DATA 01,01,01,FF,10,10,10,FF
1030 DATA 18,18,18,18,18,18,18,18
1040 DATA 18,18,3C,5A,99,18,24,24
1050 ' Tabela de Cores
1060 DATA 00,00,00,00,00,00,00,00
1070 DATA E6,E6,E6,E6,E6,E6,E6,E6
1080 DATA 10,40,10,40,10,40,10,40
1090 DATA 10,D0,10,10,10,10,10,10
```

O programa altera a tabela de padrões e cores para desenhar quatro caracteres na tela:

- Vazio – código 0
- Tijolo – código 1
- Grade – código 2
- Policial – código 3

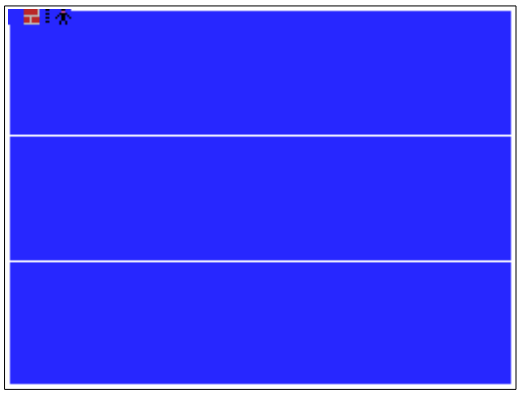

Figura 1.5. Criação de caracteres nas tabelas de padrões e cores.

Por *default*, a tabela de nomes é numerada de 0 a 255 nas três áreas, de forma a apresentar todos os caracteres na tela.

A posição inicial da tabela de nomes fica no endereço 6144 ou &H1800. Entretanto, cada caractere ocupa agora um byte em vez de 8. Assim, reescrevemos a fórmula de endereço das outras tabelas para:

$$endereço = inicio\_tabela + parte \times 256 + pos\_caractere$$

Vamos alterar o caractere na posição de tela 34 (vide figura 1.1) para o tijolo (caractere número 1). Para isso, acrescente a seguinte linha ao programa anterior.

```
75 VPOKE 6144 + 0 + 34, 1
```

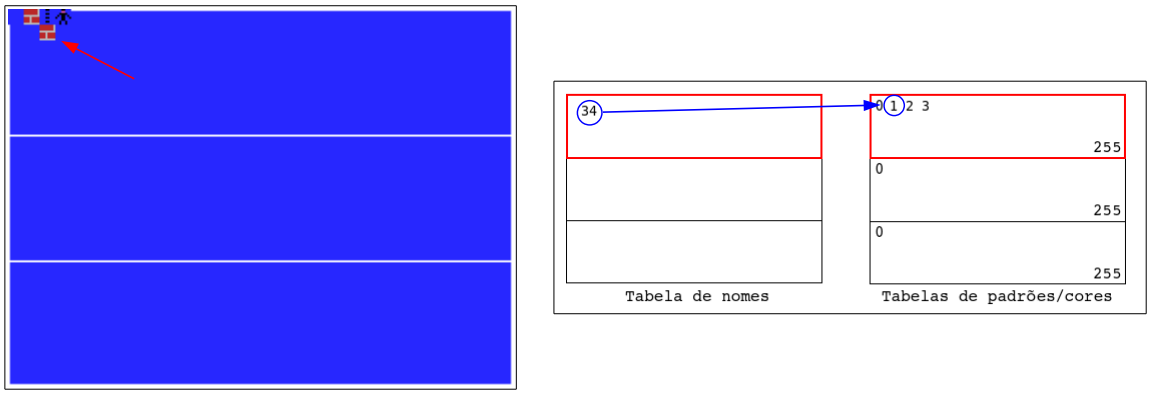

Figura 1.6. Adição de um tijolo na posição 34.

Conforme podemos observar na figura 1.6, a posição 34 da tabela de nomes da primeira parte (parte 0) da tela referencia o caractere 1 das tabelas de padrões/cores, também da primeira parte.

E se quiséssemos colocar o tijolo na segunda parte da tela?

```
75 VPOKE 6144 + 1*256 + 34, 1
```

Não desenhou o tijolo! Por quê? Porque estamos tentando desenhar um caractere da primeira parte da tela na segunda parte. Na realidade, quando acessamos a segunda parte da tabela de nomes, estamos desenhando o caractere 1 da segunda parte da tabela de padrões/cores que é um bloco vazio.

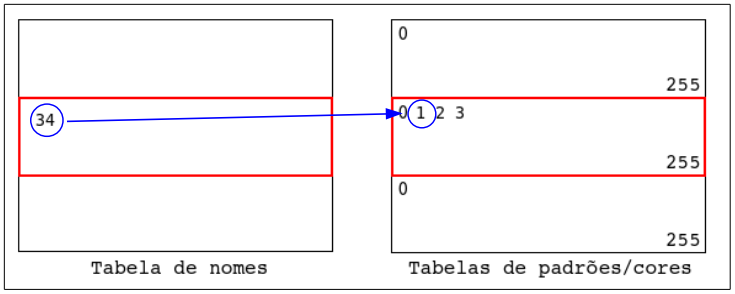

Figura 1.7. Desenhando na segunda parte da tela.

Para resolver o problema, devemos repetir os caracteres da primeira parte da tela na segunda parte. Se desejarmos o padrões em toda tela, devemos então repeti-los nas três partes.

Adicione as linhas:

```
35 VPOKE E+2048,VAL("&H"+A$) : VPOKE E+4096,VAL("&H"+A$)
...
65 VPOKE &H2000+E+2048,VAL("&H"+A$) : VPOKE &H2000+E+4096,VAL("&H"+A$)
```

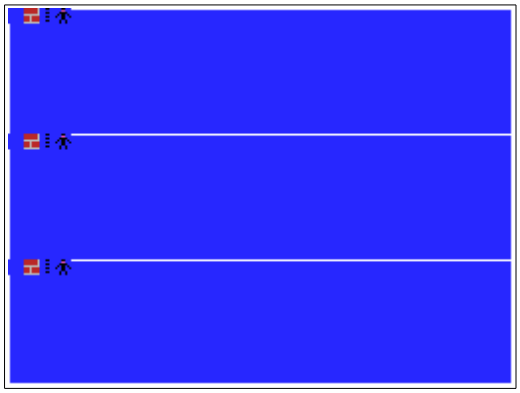

Figura 1.8. Padrões de caracteres nas três partes.

Teste agora:

```
75 VPOKE 6144 + 1*256 + 34, 1
```

Agora o bloco foi colocado no lugar.

Podemos criar várias coisas legais com isso: cenários, personagens gigantes, *scrolls* etc. Por exemplo, criaremos um labirinto na primeira parte da tela. Para isso, devemos configurar a tabela de nomes de acordo com a figura 1.9.

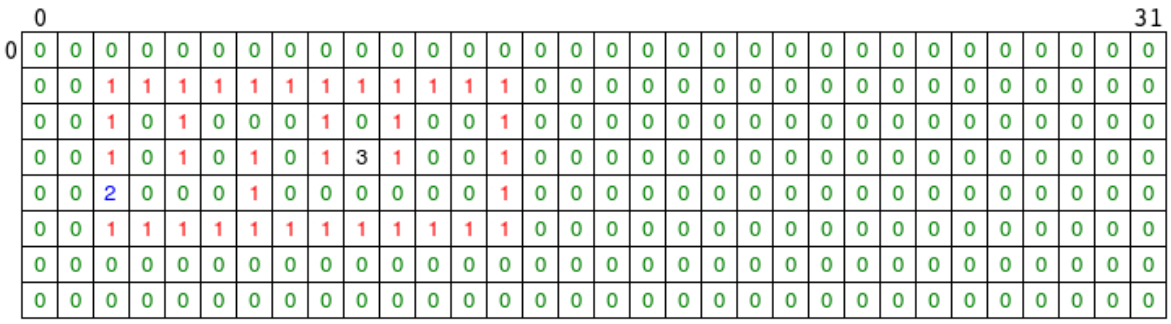

Figura 1.9. Configuração da tabela de nomes.

Em Basic:

```
10 COLOR 15,4,4:SCREEN 2
20 FOR E=0 TO 31
30 READ A$ : VPOKE E,VAL("&H"+A$)
40 NEXT E
50 FOR E=0 TO 31
60 READ A$ : VPOKE &H2000+E,VAL("&H"+A$)
70 NEXT E
80 FOR E=0 TO 255
90 READ A$ : VPOKE &H1800+E,VAL("&H"+A$)
100 NEXT E
120 GOTO 120
1000 ' Tabela de Padroes
1010 DATA 00,00,00,00,00,00,00,00
1020 DATA 01,01,01,FF,10,10,10,FF
1030 DATA 18,18,18,18,18,18,18,18
1040 DATA 18,18,3C,5A,99,18,24,24
1050 ' Tabela de Cores
1060 DATA 00,00,00,00,00,00,00,00
1070 DATA E6,E6,E6,E6,E6,E6,E6,E6
1080 DATA 10,40,10,40,10,40,10,40
1090 DATA 10,D0,10,10,10,10,10,10
1500 ' Tabela de nomes
1510 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1520 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1530 DATA 0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,0
1540 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1550 DATA 0,0,1,0,1,0,0,0,1,0,1,0,0,1,0,0
1560 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1570 DATA 0,0,1,0,1,0,1,0,1,3,1,0,0,1,0,0
1580 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1590 DATA 0,0,2,0,0,0,1,0,0,0,0,0,0,1,0,0
1600 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1610 DATA 0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,0
1620 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1630 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1640 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1650 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1660 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

O resultado no MSX pode ser visto na figura 1.10.

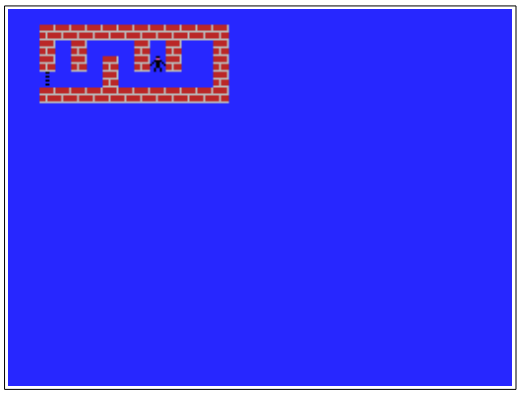

Figura 1.10. Criação de cenários através da tabela de nomes.

1.2. O "efeito paleta"

Podemos criar um efeito interessante na screen 2, quando definimos todos os espaços (ou uma região) na tabela de nomes com o mesmo caractere. Ao alterarmos a forma desse caractere nas tabelas de padrões/cores, a alteração é refletida imediatamente em todos os espaços definidos com esse caractere.

Veja o programa a seguir.

```
10 COLOR 15,4,4:SCREEN 2:OPEN"GRP:"AS#1
20 FOR E=6144 TO 6399 : VPOKE E,0 : NEXT E
30 PRESET(10,70),0:PRINT#1,"Tecle enter":A$=INPUT$(1)
40 FOR E=0 TO 7
50 READ A$ : VPOKE E,VAL("&H"+A$)
60 NEXT E
70 FOR E=0 TO 7
80 READ A$ : VPOKE &H2000+E,VAL("&H"+A$)
90 NEXT E
100 GOTO 100
1000 ' Tabela de Padroes
1020 DATA 01,01,01,FF,10,10,10,FF
1050 ' Tabela de Cores
1060 DATA E6,E6,E6,E6,E6,E6,E6,E6
```

Quando pressionado Enter, mudamos o caractere 1 das tabelas de padrões/cores de vazio para tijolo e toda a primeira parte da tela é imediatamente alterada.

Experimente o programa a seguir.

```
10 COLOR 15,4,4:SCREEN 2:OPEN"GRP:"AS#1
20 FOR E=6144 TO 6399 : VPOKE E,0 : NEXT E
30 FOR E=0 TO 7
40 READ A$ : VPOKE E,VAL("&H"+A$)
50 NEXT E
60 FOR E=0 TO 7
70 READ A$ : VPOKE &H2000+E,VAL("&H"+A$)
80 NEXT E
90 PRESET(10,70),0:PRINT#1,"Tecle enter":A$=INPUT$(1)
100 FOR E=0 TO 7 : VPOKE &H2000+E,&H46 : NEXT
105 FOR T=1 TO 100 : NEXT T
110 FOR E=0 TO 7 : VPOKE &H2000+E,&HE6 : NEXT
115 FOR T=1 TO 100 : NEXT T
120 GOTO 100
1000 ' Tabela de Padroes
1020 DATA 01,01,01,FF,10,10,10,FF
1050 ' Tabela de Cores
1060 DATA E6,E6,E6,E6,E6,E6,E6,E6
```

Nesse programa alteramos apenas a tabela de cores do tijolo, alternando as cores 4 e 14 (&HE) para o cimento entre os tijolos.

1.3. Saiba mais sobre a screen 2

Na página MarMSX, há diversos artigos falando sobre a arquitetura de telas do MSX 1. Por exemplo, o curso de Basic [2] possui os tópicos:

- As Tabelas de Nomes, Caracteres e Cores.
- Desenhando nas screens 0 e 1.
- Animações na screen 2.
- Estudo do jogo Alcatraz

O curso de criação de jogos [3] aborda profundamente o tema, explicando como criar cenários, personagens e animações. No artigo sobre Sprites e Gravidade [4] e no curso de Assembly [5], os sprites são abordados. Por fim, o projeto Screen 2 Show [6] apresenta mais um artigo sobre a arquitetura da screen 2.

Créditos e Bibliografia

Este artigo foi escrito por Marcelo Silveira.

Data: Janeiro de 2021
E-mail: flamar98@hotmail.com
Homepage: http://marmsx.msxall.com

Referências Bibliográficas:

[1] – Jogos de Habilidade, Autores Diversos. Editora Aleph, Brasil, 1986.
[2] – Curso de Basic. Em: http://marmsx.msxall.com/cursos/basic.
[3] – Curso de Jogos em Basic. Em: http://marmsx.msxall.com/cursos/basgame.
[4] – Sprites e Gravidade, Marcelo Silveira. Em: http://marmsx.msxall.com/artigos.
[5] – Curso de Assembly. Em: http://marmsx.msxall.com/cursos/assembly.
[6] – Screen 2 Show, Arquitetura da Screen 2. Em: http://marmsx.msxall.com.