MSX Article

A Memória do MSX (I)

Resumo

O objetivo deste artigo é mostrar como funciona o esquema de memória do MSX, que possui ROMs e RAMs compartilhadas em um espaço de 64 KB. Este é o primeiro de três artigos, no qual serão abordados os slots primários. No segundo artigo, serão analisados os slots secundários e também a Mapper e Megaram. O terceiro irá apresentar uma experiência prática na memória do MSX.

1- Introdução

Sabemos que a memória do MSX tem 64 Kb. Mas, se a ROM do sistema tem 32 KB e a RAM do usuário tem 64 KB, não são mais de 64 Kb? E quando um cartucho é inserido, não fica maior ainda? A resposta é sim. Então, o processador Z-80 pode acessar mais de 64 Kb? Pode, mas não diretamente.

O Z-80 só enxerga 64 Kb, devido ao fato do endereçamento de memória ser de 16 bits. Assim, só é possível representar números entre 0 e 65535, ou seja, 64 Kb. Para acessar outras memórias acopladas ao sistema, será necessário trocar o acesso à memória em uso para a memória que se deseje acessar. Por exemplo, o Z-80 deixa de enxergar a ROM e passa a enxergar a RAM. Entretanto, essa troca não é feita em toda a área de memória de uma só vez.

Os projetistas do MSX dividiram a memória em 4 partes iguais, chamadas de páginas. Assim, as páginas podem trocar o acesso a outras memórias, independentemente uma das outras, permitindo assim uma maior flexibilidade ao sistema. A figura 1 apresenta o esquema de páginas de memória do MSX.

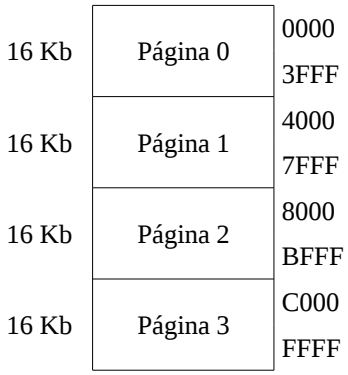

Figura 1. Divisão da memória do MSX em páginas.

As páginas são numeradas de 0 a 3, onde cada uma tem 16 Kb. Os limites de endereço de cada página são mostrados na figura 1.

Quantas memórias podem ser compartilhadas no MSX? É possível compartilhar até quatro memórias, com capacidade de até 64 Kb. Cada espaço desse contendo uma memória é chamado de slot, onde o número de identificação varia de 0 a 3.

A configuração de slots e páginas do Expert da Gradiente é apresentada na figura 2, de forma a ilustrar esse sistema. As áreas sombreadas correspondem às páginas ocupadas por cada memória. A configuração varia nos MSXs, mas sempre com a BIOS no slot 0.

	Slot 0 ROM - BIOS	Slot 1 Cartucho A	Slot 2 RAM	Slot 3 Cartucho B	
16 Kb	Página 0	Página 0	Página 0	Página 0	0000 - 3FFF
16 Kb	Página 1	Página 1	Página 1	Página 1	4000 - 7FFF
16 Kb	Página 2	Página 2	Página 2	Página 2	8000 - BFFF
16 Kb	Página 3	Página 3	Página 3	Página 3	C000 - FFFF

Figura 2. Divisão da memória do MSX em slots e páginas.

Uma memória não necessita ocupar todos os 64 Kb do slot, nem começar pela página 0. Um cartucho, por exemplo, pode ter a memória de 16 Kb e localizada na página 1.

2- O chaveamento de memória

Foi dito na seção anterior, que é possível trocar a memória em uso das páginas, de forma independente uma das outras. Assim, é possível ter, por exemplo, a página 0 acessando a memória do slot 0 e a página 2 acessando a memória do slot 3.

A PPI é a responsável por chavear a memória e configurar o que o processador Z-80 irá acessar em cada página, através da porta &HA8. A configuração dos dados enviados a esta porta é apresentada na figura 3.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Página	3		2		1		0	

Figura 3. Configuração dos dados enviados à porta &HA8.

O valor da porta é dividido em quatro partes, onde cada parte representa uma página da memória. Cada parte contém um valor de 2 bits, que corresponde a um número de slot. O exemplo a seguir, irá ilustrar como dados são configurados na porta &HA8.

Quando o Expert é iniciado, a memória é configurada de acordo com a figura 4. As páginas 0 e 1 estão com a ROM no slot 0 e as páginas 2 e 3 com a RAM no slot 2. As áreas sombreadas mostram o trecho de memória ativo, no qual é acessado pelo Z-80.

	Slot 0 ROM - BIOS	Slot 1 Cartucho A	Slot 2 RAM	Slot 3 Cartucho B	
16 Kb	Página 0	Página 0	Página 0	Página 0	0000 - 3FFF
16 Kb	Página 1	Página 1	Página 1	Página 1	4000 - 7FFF
16 Kb	Página 2	Página 2	Página 2	Página 2	8000 - BFFF
16 Kb	Página 3	Página 3	Página 3	Página 3	C000 - FFFF

Figura 4. Configuração inicial de memória do Expert.

De acordo com a configuração apresentada, tem-se:

Bit	7	6	5	4	3	2	1	0
Valor em Binário	1	0	1	0	0	0	0	0
Slot	2		2		0		0	
Página	3		2		1		0	

Então, o valor da porta &HA8 é 160 (&B10100000).

A porta &HA8 pode ser tanto lida, de forma a obter a configuração de slots, como escrita, de forma a modificá-los.

3- O cuidado na troca de slots

Quando uma página troca o slot, toda a região de memória relativa a essa página passa a acessar outra memória. Embora o conteúdo da memória anterior não seja perdido, ele se torna inacessível após a troca. Dessa forma, não se deve fazer o chaveamento de memória, caso o programa em execução estiver na mesma página que se deseje trocar, sob pena do programa perder o controle de execução.

De forma a ilustrar esse fato, suponha dois programas em Basic na mesmá página, mas localizados em slots diferentes. Suponha também que o stack pointer (SP) aponta para uma linha em Basic, em vez de um endereço de memória.

Programa no slot 2	Programa no slot 1
8000 PRINT"Vou trocar o slot" 8010 OUT &HA8, &B10010000 8020 END	8000 INPUT"Qual o seu nome";N$ 8010 INPUT"Qual a sua idade";I 8020 PRINT"Nome";N$ 8030 PRINT"Idade";I 8040 END

Ambos os programas se localizam na mesma região de memória. Ao executar o programa do slot 2, seria obtido o seguinte fluxo de execução, assinalado em azul:

Programa no slot 2	Programa no slot 1
8000 PRINT"Vou trocar o slot" 8010 OUT &HA8, &B10010000 8020 END	8000 INPUT"Qual o seu nome";N$ 8010 INPUT"Qual a sua idade";I 8020 PRINT"Nome";N$ 8030 PRINT"Idade";I 8040 END

O fluxo de execução passou do programa do slot 2 para o programa do slot 3, fazendo com que o programa principal no slot 2 perdesse o controle de execução.

Caso seja necessária uma troca de slots como no exemplo acima, a solução é desviar a execução do programa para outra página e fazer a troca de slots. Uma vez que o programa passa a executar em outra página, o controle de execução não é perdido. Esta estratégia foi utilizada no programa Slot View [1] e também é utilizada nas chamadas inter-slot [2].

Programa no slot 2	Programa no slot 1
8000 PRINT"Vou trocar o slot" 8010 GOTO C000 8010 END	8000 INPUT"Qual o seu nome";N$ 8010 INPUT"Qual a sua idade";I 8020 PRINT"Nome";N$ 8030 PRINT"Idade";I 8040 END
C000 OUT &HA8, &B10010000 C010 END	

4- As chamadas iter-slot

A BIOS é o sistema operacional do MSX, no qual contém várias instruções prontas, como escrever na tela, gerar sons e sprites, controlar joystick etc. Dessa forma, o usuário não precisa se preocupar em escrever esses códigos mais complexos, e somente focar em seu programa. A BIOS localiza-se na página 0 do slot 0.

O iterpretador Basic se localiza nas páginas 0 e 1 do slot 0 e é utilizado quando o MSX opera em modo Basic, juntamente com a BIOS. Um programa em Assembly não necessita do interpretador Basic, exceto quando são feitas chamadas a funções específicas do Basic. Assim, a página 1 fica livre para os programas. A página 0 pode até ser utilizada por um programa em Assembly, entretanto, o acesso direto ao sistema operacional da BIOS seria prejudicado. Tanto no caso do programa da página 0, como o da página 1, pode ser necessário a chamada de programas que se localizam em outros slots.

A BIOS do MSX possui sub-rotinas para chamar programas em outros slots, de forma segura e eficiente. Elas são as chamadas inter-slot.

Os casos mais comuns de chamadas a programas em outros slots são [2]:

- A BIOS é chamada a partir do MSX-DOS.
- A BIOS é chamada da SUB-ROM a partir do Basic (MSX 2)
- A BIOS é chamada a partir de um cartucho.

No caso do MSX-DOS, ele configura inicialmente as quatro páginas para a RAM. Com essa configuração, o acesso à BIOS não é possível. Então, quando uma chamada é feita à BIOS, é necessário trocar a página 0 para a ROM da BIOS e então realizar a chamada. Após as operações da BIOS, o estado original dos slots deverá ser restaurado.

Essa operação é realizada facilmente, quando o programa reside em uma página diferente de 0. Entretanto, problemas podem surgir quando o programa que realiza a chamada reside na página 0, no qual é a mesma página da BIOS. Assim, cuidados são necessários para previnir o programa de desaparecer e gerar resultados imprevisíveis. As chamadas inter-slot resolvem esse problema, através do desvio temporário do programa para a página 3, para a troca dos slots. Algumas chamadas inter-slots estão incluídas no próprio MSX-DOS [2].

5- O funcionamento dos cartuchos do tipo ROM

Os computadores MSX possuem, pelo menos, um slot externo e o hardware inserido ali é chamado de cartucho. Há cartuchos do tipo [2]:

- ROM – utilizados para jogos e aplicações.
- E/S – como disk-drives ou interface RS-232.
- RAM – expansão de memória RAM.
- Expansão – dispositivos para expandir o número de cartuchos.

Normalmente os jogos em Assembly de cartuchos utilizam a memória a partir da posição &H4000, ou seja, a página 1. Quando o cartucho contém um programa em Basic, ele se localiza em &H8000, na página 2.

Os cartuchos no formato ROM possuem um *header* de 16 bytes. Assim, quando o sistema é iniciado, o interpretador Basic analisa o *header* do cartucho e inicia o programa contido na ROM, baseado nessas informações. O formato do *header* é descrito na figura 5.

2 bytes	ID	+ 0000
2 bytes	INIT	+ 0002
2 bytes	STATEMENT	+ 0004
2 bytes	DEVICE	+ 0006
2 bytes	TEXT	+ 0008
6 bytes	Reserved	+ 000A

Figura 5. *Header* do cartucho ROM.

Descrição dos elementos do *header* [2]:

- ID - no caso de cartuchos de ROM, a identificação no formato texto é “AB” (&H41, &H42). Já os cartuchos do tipo SUB-ROM, a identificação é “CD”.
- INIT - contém o endereço inicial da rotina a ser executada, quando o cartucho é auto-executável. Caso contrário, possui o valor igual a 0.
- STATEMENT - quando o cartucho dispõe da função CALL para chamar uma rotina sua a partir do Basic, esses dois bytes contêm o endereço inicial dessa rotina. Caso não possua, o valor é igual a 0.
- DEVICE - endereço inicial da rotina de expansão de dispositivo. Caso não possua, o valor é igual a 0.
- TEXT - ponteiro para programa em Basic, quando o cartucho é auto-executável. Caso contrário, o valor é igual a 0.

6- Carregamento de ROMs a partir do Basic

Foi visto na seção anterior que os programas em Assembly de cartuchos do tipo ROM residem na página 1. Suponha agora que o conteúdo dessa ROM esteja em um arquivo e que se deseje rodá-lo a partir do Basic. O conteúdo da ROM terá que executar obrigatoriamente na página 1, e a partir de uma memória do tipo RAM. Deve-se lembrar que o modo Basic configura as páginas 0 e 1 para ROM da BIOS e as paginas 2 e 3 para a RAM.

A solução é copiar o arquivo da ROM para a página 1 da RAM, fazendo-o funcionar a partir do endereço inicial do programa ali contido. Entretanto, essa operação é difícil de ser realizada em Basic e deverá ser feita em Assembly. Uma estratégia para isso, é colocar um programa ao final do bloco do arquivo ROM, que tem como função trocar o slot da página 1 para RAM, copiar o conteúdo desse arquivo para a página 1 e depois iniciar a execução do programa. Um arquivo auto-executável pode realizar toda essa tarefa automaticamente.

As figuras 6 e 7 ilustram esse processo. Na figura 6, o arquivo da ROM mais o programa são carregados na página 2, e o programa responsável por copiar na página 1 (em verde) é posto em execução.

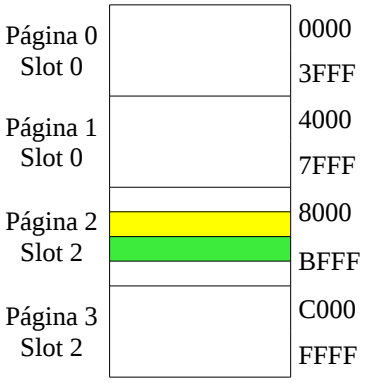

Figura 6. Carregamento de ROM na página 1.

Na figura 7, o bloco da ROM (em amarelo) é copiado para a página 1 da RAM e posto em execução.

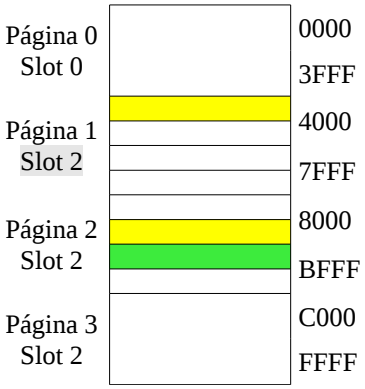

Figura 7. Processo finalizado.

7- Créditos

Este artigo foi escrito por Marcelo Teixeira Silveira, engenheiro de sistemas e computação formado pela UERJ, Universidade do Estado do Rio de Janeiro, Brasil.

Escrito em: maio de 2004.
Revisado em: julho de 2017.
e-mail: flamar98@hotmail.com
homepage: http://marmsx.msxall.com

Referências:

[1] – Slot View, MarMSX Development, em http://marmsx.mxsall.com
[2] – MSX 2 Technical Handbook, ASCII Corporation, 1987.
[3] – O Livro Vermelho do MSX, editora Mc Graw Hill.