+--------------+
¦ ASSEMBLY XVI ¦
+--------------+

Usando a memória Expandida (EMS).

Muitos modplayers hoje em dia usam a memória expandida para armazenar os samples. Neste texto veremos como funciona a memória expandida e como usá-la...

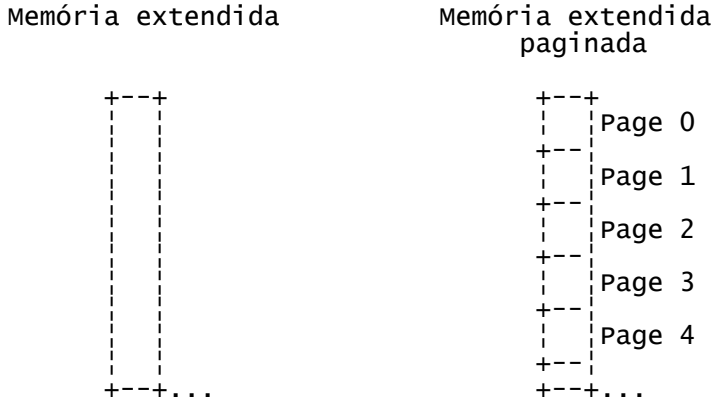
A maioria dos PC-ATs com mais de 1Mb de memória possui dois tipos de memória:

¦ Convencional - Na faixa de 0 até 1Mb
¦ Extendida: de 1Mb em diante.

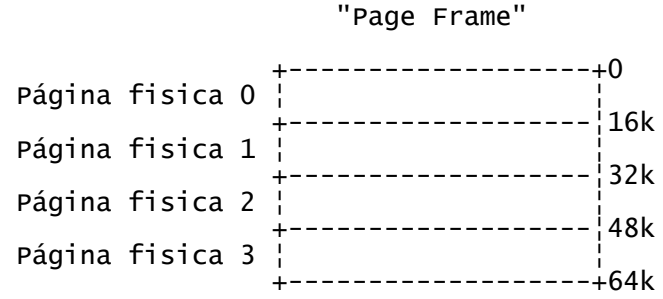
A memória extendida é facilmente manipulável quando um programa está em modo protegido e com toda a memória mapeada em um ou mais seletores. Os 386s permitem que um seletor acesse um segmento de até 4Gb de tamanho... Mas, näo é esse o nosso caso. Temos um pequeno programa rodando sob o MS-DOS, no modo real (modo nativo dos processadores Intel), que tem acesso somente à memória convencional. Podemos acessar a memória extendida através do driver HIMEM.SYS ou usando uma função de movimento de blocos da BIOS, mas isso aumentaria em muito a complexidade do software (e, por conseguência, seu tamanho).

A Lotus, Intel e Microsoft criaram a especificação EMS para esse caso. O programa EMM386.EXE, ou qualquer outro gerenciador de memória como o QEMM, emula a memória expandida da mesma forma que uma máquina com apenas este tipo de memória faria (A memória expandida por hardware näo fez muito sucesso nos EUA como a memória extendida!). A especificação EMS simplesmente usa um espaço da memória convencional (chamado de "Page Frame") para armazenar "páginas" de 16kb da memória extendida. Isto é... divida a sua memória extendida em diversos blocos de 16k e terá o número de páginas (pages) que poderão estar disponíveis para uso.

O EMM (Expanded Memory Manager) simplesmente faz a cópia das páginas desejadas para o "Page Frame" para que o nosso software posssa lê-las e escrevê-las, copiando-as de volta para as páginas corretas quando fizermos a troca de páginas do "Page Frame". No "Page Frame" cabem, normalmente, 4 páginas... fazendo um total de 64kb (ou seja, exatamente o tamanho de um segmento!). Considere a figura abaixo:



Ok... a memória extendida foi dividida em 'n' páginas de 16k. O "Page Frame" fica na memória convencional. Por exemplo, suponha que o "Page Frame" esteja localizado no segmento 0C000h:



Do offset 0 até 16k-1 fica a primeira página do "Page Frame", do offset 16k até 32k-1 a segunda, e assim por diante. A especificação EMS nos permite colocar apenas 4 páginas no "Page Frame". Assim, o nosso programa escolhe cada uma das quatro "páginas lógicas" que serão copiadas da memória extendida para cada uma das quatro "páginas fisicas" do Page Frame.

Vale a pena lembrar que o Page Frame está sempre em algum lugar da memória convencional, portanto acessível aos programas feitos para MS-DOS, que normalmente trabalham em modo real.

A interrupção 67h é a porta de entrada para as funções do EMM (EMM386, QEMM, 386MAX, entre outros). Mas antes de começarmos a futucar o EMM precisamos saber se ele está presente... Eis a rotina de detecção do EMM p/ os compiladores C da BORLAND:

```
---%<-----------------------------------%<---------------------------
#include <io.h>
#include <fcntl.h>
#include <dos.h>

#define CARRY_BIT   (_FLAGS & 0x01)

/* Obtém a maior versão do EMM - definida em outro módulo! */
extern int emm_majorVer(void);

/* Testa a presença do EMM
   Retorna 0 se EMM não presente ou versão < 3.xx
   Retorna 1 se tudo ok! */
int isEMMpresent(void)
{
    int handle;

    /* Tenta abrir o device driver EMMXXXX0 para leitura! */
    if ((handle = open("EMMXXXX0", O_BINARY | O_RDONLY)) == -1)
        return 0;   /* Não tem EMM! */

    /* Verifica se é um arquivo ou dispositivo
       Usa IOCTL para isso! */
    _BX = handle;
    _AX = 0x4400;
    geninterrupt(0x21);
    if (!(_DX & 0x80))
        return 0;   /* É um arquivo!!! Não é o EMM! */

    /* Verifica o dispositivo está ok */
```

```
    _BX = handle;
    _AX = 0x4407;
    geninterrupt(0x21);
    if (CARRY_BIT || !_AL) return 0; /* Näo está ok */

    /* Verifica a versäo do EMM.
       Para nossos propósitos tem que ser >= que
       3.xx */
    if (emm_majorVer() < 3) return 0; /* Näo é ver >= 3.xx */

    /* Tudo ok... EMM presente */
    return 1;
}
---%<---------------------------------%<---------------------------
```

No próximo texto mostrarei como usar o EMM.

See ya...