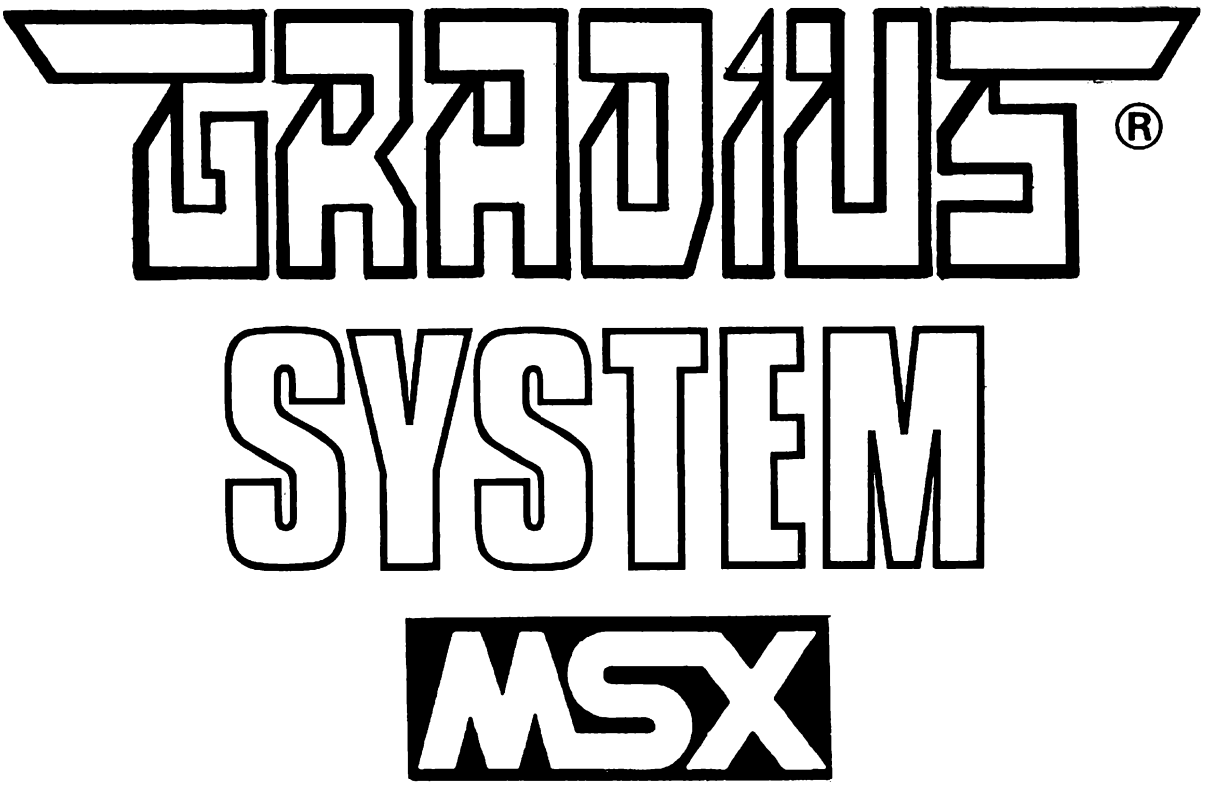
GRADIUS®
SYSTEM
MSX

INTRODUÇÃO

Antes de mais nada, gostaríamos de agradecer-lhe por adquirir este exemplar original do GRADIUS SYSTEM. Este seu procedimento nos encoraja a desenvolver novos produtos, contratar mais programadores e descobrir novos talentos; lançar mais novidades no mercado e principalmente fortalecer ainda mais o padrão MSX no campo da informática nacional.

Nas páginas deste manual de instruções como também fora delas, através de consultas que poderão ser solicitadas ao nosso departamento técnico, tentaremos esclarecer ao máximo a correta utilização deste software e seus acessórios, para que o mesmo possa ser explorado em sua total potencialidade.

É de grande importância o perfeito domínio do MSX BASIC para aqueles que desejarem se aprofundar no que vamos apresentar adiante. Recomendamos uma atenta leitura ao manual do micro-computador, principalmente no que se refere aos comandos e funções que utilizaremos.

Neste manual, e principalmente nos programas da linha GRADIUS SYSTEM usaremos termos da língua inglesa que fazem parte do mundo da informática em geral. Traduzir alguns destes nomes muitas vezes seria impossível e em outras ocasiões poderia confundir ainda mais o usuário. Os termos em inglês utilizados, bem como os próprios comandos do MSX BASIC, fazem parte da linguagem natural da máquina. A língua é ainda reconhecida internacionalmente como padrão para designar termos técnicos ligados a informática.

Outro detalhe de muita importância é que usaremos somente a notação hexadecimal quando nos referirmos a posições de memória e seus respectivos conteúdos. Se o leitor desconhece esta notação, sugerimos uma consulta ao manual de instruções do seu micro-computador.

CARREGAMENTO

Acompanha este manual de instruções, um disco contendo o GRADIUS SYSTEM: G-BASIC, G-DESK e seus respectivos programas integrantes. Este disco será denominado, daqui em diante, "MASTER DISK". Para carregar os programas na memória do seu micro-computador, basta inserir o disco no disk-drive e esperar alguns segundos. Será apresentada uma tela de abertura e em seguida será executado o aplicativo G-DESK. Antes de manuseá-lo leia atentamente as instruções que se seguem.

SISTEMA GRADIUS 1.0

O SISTEMA GRADIUS é um conjunto de programas destinado a ajudar os usuários na confecção e na sofisticação de software em BASIC, a versátil linguagem que acompanha o seu micro-computador.

Entre as vantagens que encontramos no MSX BASIC, destacamos a facilidade de programação (quase como escrevermos um texto em inglês), a depuração simplificada de erros ocorridos (inclusive com mensagens de que tipo de erro ocorreu e em que linha), a execução imediata do que programamos (sem a necessidade de compilação, "linkagem", etc.), etc..

As desvantagens encontradas quando utilizamos esta linguagem são basicamente a baixa velocidade de execução da maioria dos recursos disponíveis e a inexistência de alguns comandos e funções existentes em linguagens mais recentes.

O SISTEMA GRADIUS foi idealizado justamente para contornar algumas das deficiências existentes no MSX BASIC e para implementar novos comandos e funções a esta linguagem, proporcionando ao usuário, um melhor resultado nos programas por ele desenvolvidos. Para isso foram criadas, num novo ambiente visual com características "pós-iconográficas" inéditas, rotinas em linguagem de máquina e ferramentas de programação destinadas a ajudá-lo a deixar seus programas muito melhores do que eram.

A técnica utilizada para adicionar os recursos do SISTEMA GRADIUS aos seus programas pode variar de acordo com a versão do nosso software. A evolução deste conjunto híbrido pode, definitivamente, nos levar a criação de uma nova linguagem, propriamente dita, baseada nas vantagens do BASIC e nas implementações do SISTEMA GRADIUS. Esta nova linguagem poderá inclusive, ser utilizada em outros micro-computadores, aproveitando os recursos e vantagens que cada um tiver de melhor.

Nesta primeira versão do SISTEMA GRADIUS, optamos por utilizar a maneira mais simples e didática de se acrescentar rotinas em linguagem de máquina com as mais variadas funções aos seus programas em BASIC. As rotinas serão posicionadas na memória RAM do computador e acionadas com o comando "defusr" do MSX BASIC. Parâmetros porventura necessários a estas rotinas entrarão através de comandos "poke" ou serão lidos através do comando "peek". Em versões posteriores poderemos usar "A=USR(parâmetros)", "CALL COMANDO,(parâmetro 1),(parâmetro 2),... (parâmetro N)" ou até mesmo "COMANDO,(parâmetro 1),(parâmetro 2),... (parâmetro N)". O modo que utilizaremos nesta versão ajudará os programadores menos experientes a entender melhor o funcionamento do seu MSX e introduzirá aqueles que dominam o MSX BASIC no mundo da linguagem de máquina e do micro-processador Z80.

. Para aqueles que desejam desde já usar a sintaxe que utilizaremos nas próximas versões deste software, recomendamos a leitura do livro GUIA DO PROGRAMADOR MSX de Eduardo A. Barbosa editado pela Editora Ciência Moderna (Eduardo A. Barbosa é o autor dos "best-sellers" MSX-DOS TOOLS PLUS e SISTEMA OPERACIONAL HELLO!, softwares consagrados, ambos lançados pela NEMESIS INFORMATICA LTDA. no ano de 1988).

O SISTEMA GRADIUS 1.0 divide-se em 3 sub-grupos distintos: G-FILES (novos comandos, funções e rotinas prontas que podem ser acrescentadas em seus programas), G-TOOLS +1 (ferramentas de programação, que poderão auxiliá-lo nas mais diversas ocasiões) e G-MAKER (gerador automático de programas; muito útil para quem nunca programou e não entende nada de BASIC, mas desejaria criar um programa a seu gosto). Cada um destes grupos é composto por programas desenvolvidos para funcionar no ambiente GRADIUS.

O AMBIENTE GRADIUS

A grande maioria dos aplicativos existentes para os micro-computadores MSX foi desenvolvida em telas de texto ("screen 0") como é o caso dos editores de texto, DBASE II, SUPERCALC 2, etc; ou em modo gráfico ("screen 2") como os editores gráficos, MSX PAGE MAKER, EASY GRAPH e outros. Para o SISTEMA GRADIUS, escolhemos um modo distinto, que podemos chamar de modo texto-gráfico ou "screen mista", que nada mais é que uma mistura da "screen 1" (de texto) com a "screen 2". Esta técnica é amplamente usada nos fantásticos jogos de ação japoneses, inclusive nos incríveis MEGAROM. A preferência da "screen 1" deve-se à grande velocidade e facilidade de manuseio, além da baixa quantidade de memória exigida para definir seus caracteres; e em conjunto com a "screen 2", podem ser utilizadas as 16 cores em recursos gráficos que exploram ao máximo as potencialidades do seu MSX.

Do ponto de vista técnico, esta combinação pode ser facilmente obtida passando-se o computador para a "screen 1" e depois disso mudar o VDP (Vídeo Display Processor) para a "screen 2" sem alterar as informações da memória. Isto pode ser feito através da rotina "SETGRP" da BIOS com o comando "defusr=&h7E:u=usr(0)". Faltará apenas redesenhar todos os caracteres para este novo modo de tela. Esta é basicamente, a função do "MASTER DISK"; todas estas operações estarão prontas no ambiente que denominamos GRADIUS BASIC.

Na "screen mista" temos 24 linhas de 32 caracteres divididas em 3 terços de 8 linhas cada, para os quais podemos definir 256 caracteres formados por 8x8 pontos. No total, podemos definir 768 caracteres distintos (256 x 3). Cada caracter pode ser redefinido com até 16 cores simultaneamente.

O GRADIUS BASIC possui um banco de caracteres com formas e cores muito especiais, pois são peças fundamentais na construção do visual que é responsável pela integração do software com o usuário. Para visualizar os caracteres do SISTEMA GRADIUS, digite e execute o programa abaixo:

```
10 FOR N = 1 TO 3
20 FOR C = 0 TO 31
30 PRINT CHR$(1)+CHR$(64+C)
40 NEXT C
50 FOR C = 32 TO 255
60 PRINT CHR$(C)
70 NEXT C
80 NEXT N : LOCATE 0,0 : END
```

O usuário poderá redefinir seus próprios caracteres usando o utilitário G-CHRD10.TLS do GRADIUS TOOLS #1 ou diretamente do BASIC, colocando dados na VRAM através de comandos "vpoke". Os endereços dos caracteres na VRAM para a "screen mista" são:

```
&h0000 a &h07FF - forma dos caracteres do 1º terço da tela;
&h0800 a &h0FFF - forma dos caracteres do 2º terço da tela;
&h1000 a &h17FF - forma dos caracteres do 3º terço da tela;
&h1800 a &h1AFF - código dos caracteres impressos na tela;
&h2000 a &h27FF - atributos de cores dos caracteres do 1º terço da tela;
&h2800 a &h2FFF - atributos de cores dos caracteres do 2º terço da tela;
&h3000 a &h37FF - atributos de cores dos caracteres do 3º terço da tela.
```

Baseados nas informações acima, concluímos que, querendo uma maneira alternativa para colocar uma letra "A" (número &h41 no código ASCII) no canto superior esquerdo da tela poderíamos comandar "vpoke &h1800,&h41".

Para que este mesmo "A" tivesse cor branca (&hF) com fundo em preto (&h1) bastaria comandar "vpoke &h2208,&hF1". O endereço utilizado foi conseguido através da fórmula &h2000+(&h41x8), ou seja: Endereço inicial dos atributos do 1º terço somado ao número da letra "A" no código ASCII (consulte a tabela em anexo), vezes 8 (lembre-se que o caracter é formado por 8x8 pontos). O valor colocado no endereço é &HF1 de frente branca (&hF) com fundo preto (&h1). Para redefinir por completo o caracter, é necessário um "vpoke" para cada uma de suas 7 linhas restantes. Para isso digite:

```
FOR C = 1 TO 7 : VPOKE &h2208+C,&hF1 : NEXT C
```

Para imprimir um caracter na tela basta digitá-lo da maneira habitual, através do teclado, ou em linhas do programa com o comando "print". Para utilizar caracteres especiais ou gráficos comande "PRINT CHR$(&h??)", onde ?? é o número do caracter no código ASCII (consulte a tabela em anexo). Para imprimir um caracter com número abaixo de &h20 comande "PRINT CHR$(1)+CHR$(&h??+?h40)".

Outra vantagem da "screen mista" é que podemos utilizar os "SPRITES", muito úteis para representação de ícones, desenhos ou figuras que se movimentem, etc. Para visualizar o banco de "SPRITES" que acompanha o GRADIUS BASIC, digite e execute o programa abaixo:

```
10 FOR C = 0 TO 63
20 PUT SPRITE 0,(100,100),15,C
30 IF INKEY$ = "" THEN 30
40 NEXT C : GOTO 10
```

O usuário poderá redefinir seus próprios "SPRITES" usando o utilitário G-SPRD10.TLS do G-TOOLS ǂ1, diretamente do BASIC, colocando dados na VRAM através de comandos "vpoke" ou com os comandos específicos para esta finalidade existentes no MSX BASIC. No SISTEMA GRADIUS adotamos o modo duplo (16x16) para os "SPRITES". Os endereços dos "SPRITES" na VRAM para a "screen mista" são:

&h3800 a &h3FFF - formas dos "SPRITES";
&h1B00 a &h1B7F - atributos dos "SPRITES" (cores, planos e posições).

INTERFACE COM O USUARIO

Um assunto importante a respeito do ambiente GRADIUS que devemos comentar é o estilo de programação e de apresentação de dados. Recomendamos aos programadores que escrevam seus programas com separação entre as palavras e linhas de comentários que esclareçam os comandos e funções empregadas, visando facilitar a depuração de erros ou adaptações posteriores que se façam necessárias. Quanto a apresentação dos mesmos, recomendamos que seja seguido como exemplo o visual adotado em nossos programas, proporcionando uma melhor interação do software.

Nos nossos programas, procuramos criar uma maneira mais simples de comunicação com o usuário. Isso pode ser realizado com facilidade através das rotinas de "janelas" e menus "pull-down"; e ainda fica mais simples se o usuário puder controlar um cursor ou selecionar opções através das teclas direcionais ou ainda utilizar um "joystick" ou "mouse".

Procure apresentar textos e opções dentro de "janelas" utilizando um fundo quadriculado (composto com o caracter &hB6) para ressaltá-las; faça entrada de dados ("accept") através de rotinas específicas para este fim; evite enfim, sair do estilo proporcionado pelo nosso programa, que constitui o ambiente GRADIUS.

Este ambiente tenta representar uma tridimensionalidade, colocando na tela algumas das características de HARDWARE. A tecla (ESC) por exemplo, está representada pela união dos caracteres &hCD e &hCE. Este sistema de simulação está sendo muito utilizado nos sofisticadíssimos softwares japoneses de última geração. Ele é denominado "pós-iconográfico", pois é um aperfeiçoamento ao modelo iconográfico lançado pelos micro-computadores MacIntosh da Apple Computer Corp. que revolucionaram infinitamente a integração entre o usuário e a máquina na década de 80.

PROGRAMANDO EM G-BASIC

Escrever um programa em GRADIUS BASIC, nada mais é que utilizar o próprio BASIC do seu micro-computador, podendo contar com o auxílio de rotinas em linguagem de máquina que poderão ser solicitadas dentro da própria listagem em BASIC que você estiver digitando. A sintaxe de chamada para cada uma das rotinas existentes está disponível nas respectivas fichas de instruções. Existem rotinas para as mais diversas utilidades como: Animação gráfica, "janelas" de variados tipos, rotinas de "scroll", de impressão, de entrada de dados (accept), de controle de "mouse", etc. Outras rotinas podem ser encontradas nos pacotes G-FILES e adicionadas a este software. É claro que rotinas em linguagem de máquina criadas pelo próprio usuário funcionarão perfeitamente desde que não "atropelem" as rotinas existentes. Veja mais adiante como isso pode ser evitado.

Outra vantagem de se programar em G-BASIC é poder contar com uma fantástica possibilidade de redefinição dos caracteres para criar imagens multi-coloridas, tridimensionais e com características gráficas exepcionais, dependendo unicamente do bom senso do programador. Para a criação de telas de abertura e de apresentação, também pode ser utilizado o aplicativo G-SCED10.TLS que acompanha o GRADIUS TOOLS #1.

Depois de pronto, grave o seu programa no disco do SISTEMA GRADIUS, com o nome "PROGRAMA.BAS". Desta maneira, ele será automaticamente executado, como o "AUTOEXEC.BAS" do DISK-BASIC. O arquivo "PROGRAMA.BAS" terá prioridade no carregamento, inibindo a entrada do G-DESK. Se por acaso não haja um "PROGRAMA.BAS" o G-DESK será executado. No caso de também não existir o G-DESK no disco, o controle será passado para o GRADIUS BASIC.

. Usuários que não estejam familiarizados com programação poderão elaborar seus próprios programas através do GRADIUS MAKER, o gerador automático de programas em GRADIUS BASIC que poderá ser adquirido separadamente no seu fornecedor habitual.

LINGUAGEM DE MAQUINA

Ao acrescentar um programa em linguagem de máquina ao GRADIUS BASIC, verifique cuidadosamente o endereço onde ele será colocado, de modo que não sobreponha alguma das rotinas existentes. As áreas ocupadas por estas últimas estão descritas em suas respectivas fichas. Também é importante que sejam observados os endereços da memória que contenham dados de processamento em geral, áreas de variáveis e de controle da interface de disco, etc.. Para evitar estas regiões, consulte o Mapa da Memória, no manual de instruções que acompanha o seu micro-computador.

Para que o sistema do "MASTER DISK" carregue uma rotina automaticamente, é necessário que o nome da mesma tenha terminação ".MLF" (Machine Language File). É importante notar que o sistema apenas carrega a rotina para o seu local na memória. Para que a mesma seja executada, será necessário comandar "DEFUSR=&h????:U=USR(0)", onde ???? é o endereço de entrada em notação hexadecimal da mesma.

Programar em linguagem de máquina é trabalhoso, mas conhecer ao menos um pouco dela, é muito importante para aqueles que desejam se dedicar um pouco mais ao seu equipamento e se aprofundar no seu funcionamento. Recomendamos a estes interessados, uma leitura aos seguintes livros: Aprofundando-se no MSX (Editora Aleph), Linguagem de Máquina para MSX (Editora Ciência Moderna), O Livro Vermelho do MSX (Editora McGraw Hill, Programação Avançada em MSX (Editora Aleph) e Sistemas Operacionais do MSX e suas Ferramentas (Editora Ciência Moderna).

ROTINAS DE IMPLEMENTAÇÃO

Existem rotinas de comandos e funções adicionais que já estão contidas no seu "MASTER DISK". Outras poderão ser adquiridas separadamente nos conjuntos de acessórios G-FILES. As incluídas neste pacote, podem ser rapidamente localizadas através do comando "FILES", destacando-se por suas terminações ".MLF" (Machine Language File).

O nome das rotinas pode inicialmente parecer um pouco confuso, mas garantimos que é a maneira mais correta e racional de denominá-las. "G-FRAN11.MLF", por exemplo: "G-" do SISTEMA GRADIUS; "FRAN" de FRame ANimator (animação de quadros); "11" de versão 1.1 e finalmente a terminação ".MLF" de Machine Language File (arquivo em linguagem de máquina).

Mais adiante vamos apresentar algumas fichas com as principais informações sobre as rotinas fornecidas. Ao adquirir outros conjuntos de rotinas, coloque as respectivas fichas de informações juntamente com as que aqui estão. As informações constantes nas fichas são:

(a) Nome da rotina (em inglês);
(b) Arquivo (nome com o qual elas são gravadas no disco - terminação ".MLF") e os seus endereços ("header" - endereço inicial, final e de execução);
(c) Função (exatamente o que ela faz);
(d) Aplicações (no que ela pode ser usada);
(e) Endereços importantes (endereços notáveis e conteúdos dos mesmos);
(f) Utilização (instrução detalhada do funcionamento, sintaxe e limitações);
(g) Exemplos (programas escritos em G-BASIC que exemplificam a rotina em questão);
(h) Informações Técnicas (Alguns comentários adicionais, normalmente dirigidos aos programadores mais experientes).

É de extrema importância que os exemplos de utilização de cada rotina sejam executados e compreendidos para que o usuário possa passar ao estudo de um próximo. Tente modificá-los e ver o que acontece; pesquise novas utilidades para cada um; procure entender o seu funcionamento, tente encadear duas ou mais rotinas num mesmo programa, para obter novos efeitos.

GERANDO UMA COPIA DE TRABALHO

Normalmente quando usamos um programa qualquer, corremos o risco de perdermos ou alterarmos acidentalmente uma parte ou a totalidade do conteúdo do disco que estávamos utilizando. Principalmente quando ainda não estamos familiarizados com o software que manipulamos. Para que isso não ocorra, é interessante trabalharmos com uma cópia, mantendo o original em local seguro.

O "MASTER DISK" fornecido pode ser facilmente duplicado para uso pelo próprio usuário que o adquiriu. Para isso, basta utilizar o utilitário G-MDIN!O.TLS que acompanha este pacote e que pode ser executado através da função "EXEC" do G-DESK (consulte as instruções deste aplicativo mais adiante).

Embora exista uma ferramenta especial para esta função, lembramos que outros copiadores setoriais também poderão ser utilizados. Entre eles, podemos citar o "DISKCOPY.COM" do pacote MSX-DOS TOOLS PLUS da Nemesis Informática Ltda.

A diferença entre copiar com um copiador setorial qualquer ou com o G-MDIN10.TLS é que este último grava na cópia apenas os programas e rotinas que irão interessar ao usuário para um determinado programa que o mesmo irá criar, enquanto o último gera uma cópia fisicamente idêntica ao original.

ATENÇÃO: As cópias geradas pelo usuário estarão restritas ao seu próprio uso. De outra forma poderá ser encarada como uma cópia ilegal.

Os arquivos que compõem o SISTEMA GRADIUS 1.0 são: G-SYSTEM.SYS (carregador de rotinas e inicializador da "screen mista"), G-SYSTEM.VRS (bloco de VRAM contendo a tela de abertura, o banco de caracteres redefinidos e o banco de "SPRITES") e G-SYSTEM.OVR (bloco em linguagem de máquina com funções indispensáveis ao funcionamento do sistema). As rotinas (arquivos com terminação ".MLF") são opcionais. No disco de trabalho podem constar apenas aquelas que iremos utilizar. O sistema carregará automaticamente as 100 primeiras rotinas que encontrar no disco. Em caso de erro na leitura, que pode ser causada por defeito físico ou lógico no disco, será apresentada uma mensagem e o programa será interrompido. Caso haja a possibilidade da falha ter ocorrido por erro do operador, verifique o que pode ter ocasionado o problema e comande "CALL SYSTEM" para que o sistema tente reiniciar. Em caso de defeito comprovado, consulte o nosso departamento técnico.

ROTEIRO DE PROGRAMAÇÃO

Para exemplificar os procedimentos necessários para se criar um programa em G-BASIC, vamos apresentar a construção de um programa completo para ilustrar tudo que já mostramos até agora.

Inicialmente será necessário gerar uma cópia de trabalho, ou seja, uma cópia do "MASTER DISK" com os blocos indispensáveis (G-SYSTEM.SYS, G-SYSTEM.VRS e G-SYSTEM.OVR) além de G-DYLD10.MLF e G-OPWD.MLF, as duas únicas rotinas que iremos utilizar neste exemplo. Se for usado um copiador setorial para gerar o disco de trabalho, poderemos apagar os arquivos desnecessários.

Colocando o disco de trabalho para ser executado, será apresentada a tela inicial do GRADIUS SYSTEM e logo após aparecerá o "cursor" indicando que poderemos começar a programar em G-BASIC.

O programa que iremos criar é simplesmente um carregador de rotinas escritas em linguagem de máquina, que poderá ser usado, com pequenas alterações, como um carregador automático de jogos, em substituição ao comando "bload".

Inicialmente faremos uso da rotina G-DYLD10.MLF para carregar na memória o diretório do disco:

```
100 REM **********************
110 REM *                      *
120 REM * A$(N) = DIRETORIO *
130 REM *                      *
140 REM **********************
150 DIM A$ (112) : CLS
160 LOCATE 11,10 : PRINT "AGUARDE !"
170 A = 1 : B = &HBA97
180 FOR C = B TO &HC877STEP32
190 POKE C , 0 : NEXT C
200 POKE &HBA38,0
210 DEFUSR = &HBA30 : U = USR(0)
220 IF PEEK(B) = 0 THEN 400
230 IF PEEK(B) = 229 THEN 320
240 FOR C = 0 TO 7
250 A$(A)=A$(A)+CHR$(PEEK(B+C))
260 NEXT C
270 A$(A) = A$(A) + "."
280 FOR C = 8 TO 10
290 A$(A)=A$(A)+CHR$(PEEK(B+C))
300 NEXT C
310 A = A + 1
320 B = B + 32
330 GOTO 220
```

Repare que na linha 200 colocamos o valor "0" no endereço "&hBA38" através de um comando "poke". Isto foi feito visando avisar para a rotina que usaremos o drive "A". Na linha 210 a rotina é executada. A seguir, o programa coloca cada nome encontrado no diretório numa célula da variável "A$", dimensionada anteriormente com o valor 112, pois este é o número máximo de arquivos que o disco pode comportar.

Se quisermos imprimir os nomes dos arquivos constantes no disco, incluiremos as linhas de 400 a 450.

```
400 REM ************************
410 REM *                      *
420 REM * IMPRIME OS ARQUIVOS  *
430 REM *                      *
440 REM ************************
450 CLS : FOR C = 1 TO A : PRINT A$(C) : NEXT C
```

Estas últimas linhas apresentam o diretório da maneira mais simples, mas poderemos fazer algo muito melhor com as linhas que se seguem:

```
400 REM ****************************
410 REM *                          *
420 REM * 2 PRIMEIROS e 2 ULTIMOS  *
430 REM *                          *
440 REM ****************************
450 N = 3
460 A$ (1) = "              "
470 A$ (2) = "              "
480 A$ (A) = "              "
490 A$ (A+1) = "                "
500 REM ********************
510 REM *                  *
520 REM * MENU "ROLODEX"   *
530 REM *                  *
540 REM ********************
550 CLS : X = 8 : Y = 8
560 LOCATEX+1,Y : PRINTA$(N-2)
570 LOCATEX+1,Y+1 : PRINTA$(N-1)
580 LOCATE X,Y+2 : PRINT CHR$(&HC4) ;
590 FOR C = 1 TO 12
600 PRINT CHR$(ASC(MID$(A$(N),C,1))+164);
610 NEXT : PRINT CHR$(&HC4)
620 LOCATEX+1,Y+3 : PRINTA$(N+1)
630 LOCATEX+1,Y+4 : PRINTA$(N+2)
640 S = STICK(0)
650 IF STRIG(0) =-1 THEN 700
660 IF S=1 AND N>3 THEN N=N-1
670 IF S=5 AND N<A THEN N=N+1
680 IF N = A THEN 640
690 GOTO 560
```

As linhas de 500 a 690 executam um menu "rolodex", muito usado em sistemas iconográficos, os arquivos são "rolados" para cima e para baixo com o auxílio das setas direcionais. O arquivo escolhido é aquele cujo nome se encontra invertido sob a faixa central quando pressionamos a "barra de espaço". O nome do arquivo estará contido na variável "A$(N)".

As linhas de 400 a 490, juntamente com as alterações abaixo, que devem ser executadas na primeira parte do nosso programa, fazem com que o menu "rolodex" funcione perfeitamente.

```
150 DIM A$ (116) : CLS
170 A = 3 : B = &HBA97
```

Estas alterações acrescentam mais 4 nomes (em branco) ao nosso diretório, para que o nome do primeiro e do último arquivo existente no disco possam ser posicionados sob a faixa de inversão.

Para que o menu seja apresentado numa "janela", acrescentamos:

```
491 REM ******************
492 REM *                  *
493 REM * ABRE A "JANELA" *
494 REM *                  *
495 REM ******************
496 X=8:Y=8:H=5:L=16:T=4:V=1
497 POKE&HD90F,T:POKE&HDA6E,V: POKE&HDA7C,X
498 POKE&HDA7B,Y:POKE&HDA7D,H:POKE&HDA7E,L
499 DEFUSR = &HD900 : U = USR(0)
```

Deste modo, poderemos eliminar a linha 550 que tinha como função a limpeza da tela e as antigas posições do menu.

A linha 496 contém as coordenadas, a velocidade e o tipo da "janela", que são transmitidas para a rotina pelas linhas 497 e 498. A linha 499 executa a rotina que abrirá a "janela".

O arquivo selecionado será carregado pelas seguintes linhas:

```
700 REM ******************
710 REM *                  *
720 REM * CARREGA ROTINA *
730 REM *                  *
740 REM ******************
760 BLOAD A$(N)
770 CLS : LOCATE 0,0 : END
```

Poderemos ainda acrescentar ao nosso programa, uma rotina de tratamento de erros para evitar uma interrupção indesejável durante a execução do mesmo. Será também necessária uma linha com a condição de desvio:

```
750 ON ERROR GOTO 830

780 REM ************************
790 REM *                      *
800 REM * TRATAMENTO DE ERROS *
810 REM *                      *
820 REM ************************
830 POKE&HDA6E,1 : POKE&HDA7B,10 : POKE&HDA7C,4
840 POKE&HDA7D,1 : POKE&HDA7E,25 : POKE&HD90F,4
850 DEFUSR = &HD900 : U = USR(0)
860 IF ERR=53 THEN A$="ARQUIVO NÃO ENCONTRADO!" ELSE A$="ERRO NO PROGRAMA!"
870 IF ERR=56 THEN A$="NOME INVALIDO!" ELSE IF ERR=61 THEN A$="NÃO EXECUTAVEL!"
880 IF ERR=67 THEN A$="ERRO NA FAT!" ELSE IF ERR=69 THEN A$="ERRO DE E/S!"
890 IF ERR=70 THEN A$="SEM DISCO!"
900 LOCATE (32-LEN(A$))/2,10
910 PRINT A$ : DEFUSR = &H156 : U = USR(0)
920 IF INKEY$ = "" THEN 940
930 CLS : RESUME 491
```

No caso de se tentar carregar um programa que não seja uma rotina (motivo pelo qual foi desenvolvido este programa) ou causada por um erro qualquer no carregamento, aparecerá no vídeo uma nova "janela" com uma mensagem de que erro ocorreu, para que se possa tentar novamente.

Nesta última parte do nosso programa não apresentamos novidades exceto a chamada à rotina da BIOS iniciada em "&h0156" da linha 910. Usamos tal artifício para apagar o "buffer" do teclado e eliminar o "eco" da última tecla pressionada.

Depois de tudo digitado, gravamos nosso programa com o nome "PROGRAMA.BAS" no disco de trabalho, para que ele seja executado automaticamente pelo GRADIUS SYSTEM.

Para que este programa seja usado como carregador automático de jogos, a linha 930 deve ser alterada para: 760 BLOAD A$(N) , R

Os jogos a serem carregados devem estar no disco de trabalho. O programa somente carregará de maneira correta jogos que originalmente seriam carregados com o comando "bload".

GRADIUS FRAME ANIMATOR 1.1

Arquivo: G-FRAN11.MLF (&hD000,&hD03E,&hD000)

Função: Armazena até 21 telas de caracteres com acesso aleatório;

Aplicações: Animação gráfica (quadro a quadro);
Avanço e retrocesso em níveis de "janelas".

Endereços importantes: &hD000 (Endereço de entrada);
&hD013 (0 Armazena e 1 Mostra);
&hD015 (Número da tela - de 1 a 21);
&hD03E (Final da rotina).

Utilização:

A rotina de animação de quadros ("frame animator") é de extrema simplicidade de uso, embora seja capaz de realizar deslumbrantes efeitos de animação gráfica quando bem utilizada.

Seu funcionamento realiza-se da seguinte maneira: cria-se uma primeira tela, prepara-se a rotina para armazenamento, colocando-se "0" no endereço "&hD013" (poke &hD013,0), "1" (de primeira tela) no endereço "&hD015" (poke &hD015,1) e executando-a no endereço "&hD000" com "defusr=&hD000:u=usr(0)".

Em seguida cria-se uma segunda tela, repetindo-se a sequência anterior com o cuidado de colocar "2" (de segunda tela) no endereço "&hD015".

Podem ser armazenadas até 21 telas no "buffer" de 16 Kbytes, tendo sempre o cuidado de colocar no endereço "&hD015" o número da tela correspondente.

Para que as telas armazenadas sejam apresentadas no vídeo, prepara-se a rotina para apresentação, colocando-se "1" no endereço "&hD013" (poke &hD013,0) e o número da tela que queremos de volta no endereço "&hD015" (poke &hD015,1) e executando a rotina no endereço "&hD000" com "defusr=&hD000:u=usr(0)".

Para montar um "set" de movimentação, será necessário criar diversas telas (21 no máximo) com pequenas diferenças entre as mesmas, de modo que quando mostradas em sequência, apresentem um deslocamento ou um desenvolvimento previamente elaborado.

O efeito de animação gráfica (ilusão de movimento) é obtido graças a velocidade com que a rotina exibe as telas no vídeo.

Exemplos:

O programa abaixo exemplifica o armazenamento de uma tela:

```
10 CLS
20 FOR C = 1 TO 20
30 LOCATE 1,C
40 PRINT STRING$(30,182)
50 NEXT C
60 POKE &HD013,0
70 POKE &HD015,1
80 DEFUSR &HD000 : U=USR(0)
```

O programa seguinte apresenta 21 telas previamente armazenadas:

```
10 FOR C = 1 TO 21
20 POKE &HD013,1
30 POKE &HD015,C
40 DEFUSR=&HD000 : U=USR(0)
50 NEXT C
```

Informações técnicas:

A rotina G-FRAN11.MLF tem "&h3E" bytes e é perfeitamente relocável para outras áreas da memória (manualmente ou com o auxílio do utilitário G-MLM010.TLS do GRADIUS TOOLS #1), desde que não coincida com regiões reservadas para outras rotinas em utilização (consulte a tabela de mapeamento da memória para maiores informações).

Esta rotina pode ser utilizada fora do SISTEMA GRADIUS 1.0, desde que sejam observadas algumas regras:
SCREEN 0 - Armazenamento de até 18 telas de texto, desde que sejam efetuadas as seguintes alterações:
Poke &hD015,192, poke &hD016,3, poke &hD022,0, poke &hD023,0, poke &hD026,192, poke &hD027,3, poke &hD02D,0, poke &hD02E,0, poke &hD031,192, poke &hD032,3.
SCREEN 1 - Armazenamento de até 21 telas de texto sem alterações;

Devido a baixa quantidade de memória disponível no "buffer" de armazenamento de telas, torna-se impossível utilizar a rotina com os modos gráficos "SCREEN 2" e "SCREEN 3", assim como os modos adicionais existentes nos MSX2 e MSX2+.

GRADIUS DIRECTORY LOADER 1.0

Arquivo: G-DYLD10.MLF (&hBA30,&hBA96,&hBA30)

Função: Carrega o diretório do disco na memória do micro,
em um "buffer" a partir do endereço "&hBA97".

Aplicações: Utilização em programas onde seja necessária a
apresentação dos nomes dos programas constantes
no disco.

Endereços importantes: &hBA30 (Endereço inicial);
&hBA38 (Drive utilizado);
&hBA96 (Final da rotina).
&hBA97 (Endereço inicial do "buffer");
&hC877 (Endereço final do "buffer");

Utilização:

Nos programas do SISTEMA GRADIUS 1.0, a apresentação de informações em geral são impressas dentro de molduras para destacarem-se na tela. É o que chamamos de "janelas".

Quando a mensagem que se deseja apresentar é o conjunto de arquivos constantes no disco, será necessário ler o diretório do disco, e associar cada arquivo a uma variável (ou uma variável múltipla ou indexada) para que possamos imprimi-la no vídeo através do comando "PRINT".

A rotina G-DYLD10.MLF, quando acionada através de "defusr=&hBA30:u=usr(0)", faz a leitura dos setores do disco que contém o diretório, colocando os nomes dos arquivos a partir do endereço "&hBA97". Com uma pequena rotina adicional em BASIC, relacionamos os nomes dos arquivos com uma variável indexada.

Com os nomes dos arquivos relacionados com as variáveis, fica fácil manipulá-los, podendo imprimilos, selecioná-los, etc.

É bom lembrar que um disco pode conter no máximo 112 arquivos, logo dimensionaremos a variável com este valor. Quando utilizamos mais de um disk-drive, podemos selecioná-lo através do endereço "&hBA38" da seguinte maneira: poke &hBA38,0 (para drive "A:"), poke &hBA38,1 (para drive "B:"), poke &hBA38,2 (para drive "C:") e assim por diante.

Exemplos:

O programa abaixo seleciona o drive "A:" (linha 140) e executa a leitura do diretório. Da linha 160 até a linha 270 atribue-se cada arquivo do disco à uma das 112 células da variável dimensionada.

```
100 DEFINT A-Z :DIM A$ (112)
110 A = 1 : B = &HBA97
120 FOR C = B TO &HC877STEP32
130 POKE C , 0 : NEXT C
140 POKE &HBA38,0
150 DEFUSR = &HBA30 : U=USR(0)
155 '
160 IF PEEK(B) = 0 THEN 280
170 IF PEEK(B) = 229 THEN 260
180 FOR C = 0 TO 7
190 A$(A)=A$(A)+CHR$(PEEK(B+C))
200 NEXT C
210 A$(A) = A$(A) + "."
220 FOR C = 8 TO 10
230 A$(A)=A$(A)+CHR$(PEEK(B+C))
240 NEXT C
250 A = A + 1
260 B = B + 32
270 GOTO 160
275 '
280 FOR C = 1 TO A : PRINT A$(C) : NEXT C
```

Na linha 280, um "loop" se encarrega de imprimir o nome dos arquivos na tela. Lembre-se que esta impressão poderá ser controlada através do comando "LOCATE". Acrescente as seguintes linhas:

```
310 CLS : LOCATE 1,1
320 PRINT "O 3º arquivo do disco é:"
330 LOCATE 10,10 : ARQUIVO = 3 : PRINT A$(ARQUIVO)
```

Informações técnicas:

Esta rotina pode ser utilizada dentro e fora do SISTEMA GRADIUS 1.0, com qualquer sistema de drives e interfaces controladoras existentes para o padrão MSX.

GRADIUS WINDOWS ROUTINE 1.0

Arquivo: G-OPWD10.MLF (&hD900,&hDA90,&hD900)

Função: Abre uma "janela" (window) baseado nas coordenadas pré-determinadas.

Aplicações: Apresentação de menus;
Instalação de plataformas;
Limpeza de uma determinada parte da tela;
Superposição em geral, de ítens no vídeo.

Endereços importantes: &hD900 (Endereço de execução);
&hD90F (Tipo da "janela" - 1 a 5);
&hDA6E (Velocidade - 1 a 255);
&hDA7B (Coordenada "Y");
&hDA7C (Coordenada "X");
&hDA7D (Altura);
&hDA7E (Largura).

Utilização:

As "janelas" constituem um método sofisticado para apresentação através de superposições no vídeo, de informações em geral, destacando mensagens para melhor percepção pelo usuário.

Uma vez colocadas as coordenadas nos endereços correspondentes, escolhida e colocado o tipo da "janela" utilizada, a velocidade (cujo valor dependerá também do tamanho da "janela"), bastará comandar "defusr=&hD900:u=usr(0)" para que a rotina seja ativada.

Exemplo:

O programa ilustrado abaixo exemplifica o uso desta rotina:

```
10 CLS
20 Y=16:X=20:H=4:L=10:V=1,T=4
30 POKE &HDA7B,Y : POKE &HDA7C,X
40 POKE &HDA7D,H : POKE &HDA7E,L
50 POKE &HD90F,T : POKE &HDA6E,V
60 DEFUSR=&HD900 : U=USR(0)
70 DATA "UM","DOIS","TRES","QUATRO"
80 FOR C = 1 TO 4
90 READ A$ : LOCATE X+1,Y+C
100 PRINT A$ : NEXT C
```

Informações Técnicas:

Esta rotina não funciona adequadamente fora do SISTEMA GRADIUS 1.0; logo não é possível utilizá-la com o modo de texto "SCREEN 0" (funciona em "SCREEN1" desde que sejam usados caracteres redefinidos) e nos modos gráficos, além dos adicionais existentes nos MSX2 e MSX2+.

GRADIUS PULL-DOWN ROUTINE 1.0

Arquivo: G-MNPD10.MLF (&hD300,&hD3E2,&hD300)

Função: Monta um menu de barra ("pull-down") baseado nas coordenadas pré-determinadas.

Aplicações: Seleção de opções em qualquer parte durante a execução de um programa.

Endereços importantes: &hD300 (Endereço de execução);
&hD301 (Coordenada "Y");
&hD304 (Coordenada "X");
&hD310 (Altura);
&hD379 (Largura);
&hD3AA (Largura);
&hD3BF (Velocidade);
&hD3CD (Número da opção);
&hD3EC (Nome da opção).

Utilização:

O menu de barra ("pull-down") é um método sofisticado para escolha, de uma entre várias opções, durante a execução de um programa; facilitando inclusive, sua utilização pelo usuário.

Uma vez colocadas as coordenadas nos endereços correspondentes, escolhida e colocada a velocidade (cujo valor dependerá também do tamanho da largura do menú), bastará comandar "defusr=&hD300:u=usr(0)" para que o menu seja ativado.

A opção poderá ser selecionada por intermédio das teclas direcionais e acionada através da "barra-de-espaço".

Com uma simples leitura em (PEEK(&hD3CD)+1), obtém-se o número (de cima para baixo, como as coordenadas da tela) da opção selecionada; e a partir do endereço "&hD3EC" até a largura da "barra" (peek(&hD379)), encontra-se o texto ou o nome da opção que está sob a "barra" do menu (em cor invertida).

Exemplo:

O programa ilustrado abaixo exemplifica o uso desta rotina:

```
10 CLS
20 Y=16:X=20:H=4:L=10:V=20
30 DATA "UM","DOIS","TRES","QUATRO"
40 FOR C = 1 TO 4
50 READ A$ : LOCATE X+1,Y+C
60 PRINT A$ : NEXT C
70 POKE &HD301,Y : POKE &HD304,X
80 POKE &HD310,H : POKE &HD3AA,L
90 POKE &HD379,L : POKE &HD3BF,V
100 DEFUSR=&HD300 : U=USR(0)
110 A = (PEEK(&HD3CD)+1)
120 ON A GOTO 130,140,150,160
130 PRINT "ESCOLHEU UM!" : END
140 PRINT "ESCOLHEU DOIS!" : END
150 PRINT "ESCOLHEU TRES!" : END
160 PRINT "ESCOLHEU QUATRO!" : END
```

Para obter o texto que está sob a "barra", programa-se:

```
200 FOR C = 1 TO PEEK(&HD379)
210 B$ = B$ + CHR$(PEEK(&HD3EB)+C)
220 NEXT C : PRINT B$
```

Informações Técnicas:

Esta rotina não funciona adequadamente fora do SISTEMA GRADIUS 1.0; logo não é possível utilizá-la com os modos de texto e gráficos, além dos adicionais existentes nos MSX2 e MSX2+.

GRADIUS SCREEN BUFFER 1.1

Arquivo: G-SCBF11.MLF (&hEA60,&hEAB9,&hEA60)

Função: Armazena a tela ou o conteúdo do vídeo para utilização posterior.

Aplicações: Restauração de uma tela após a abertura de
uma "janela";
Criação do comando "undo" para editores gráficos.

Endereços importantes: &hEA60 (Armazena a tela);
&hEA8D (Exibe a tela armazenada);
&hEAB9 (Final da rotina).

Utilização:

A rotina de armazenamento de tela ("screen buffer") é de utilização extremamente simples e é praticamente indispensável em programas que utilizam recursos de "janelas".

Vamos supor que estamos numa planilha de cálculo, e precisamos chamar uma calculadora, que aparecerá no vídeo superpondo uma área da tela. Depois de sua utilização, seria interessante que a mesma desaparecesse, deixando o vídeo como se encontrava anteriormente. É uma aplicação clássica para a rotina de armazenamento de tela:

Basta armazenar o conteúdo do vídeo antes de chamar a calculadora com o comando "defusr=&hEA60:u=usr(0)" e depois que a mesma tiver sido deixada de lado, restaurar o vídeo com a tela previamente armazenada com o comando "defusr=&hEA8D:u=usr(0)".

Exemplos:

O programa ilustrado abaixo cria, armazena e apaga uma tela:

```
10 CLS
20 FOR C = 1 TO 20
30 LOCATE 1,C
40 PRINT STRING$(30,182)
50 NEXT C
60 DEFUSR=&HEA60
70 U=USR(0)
80 CLS
```

O programa seguinte reapresenta a tela previamente armazenada:

```
100 DEFUSR=&HEA8D
110 U=USR(0)
```

Informações técnicas:

A rotina G-SCBF11.MLF tem "&h59" bytes e é perfeitamente relocável para outras áreas da memória (manualmente ou com o auxílio do utilitário G-MLM010.TLS do GRADIUS TOOLS #1), desde que não coincida com regiões reservadas para outras rotinas em utilização (consulte a tabela de mapeamento da memória para maiores informações).

Esta rotina pode ser utilizada dentro e fora do SISTEMA GRADIUS 1.0, em qualquer um dos modos de texto ou gráficos do MSX1.

Devido a baixa quantidade de memória disponível no "buffer" de armazenamento de telas, torna-se impossível utilizar a rotina com os modos gráficos adicionais existentes nos MSX2 e MSX2+.

GRADIUS ACCEPT ROUTINE 1.0

Arquivo: G-FLAC10.MLF (&hA250,&hA33B,&hA250)

Função: Aceita a entrada via teclado de uma palavra de 12 letras.

Aplicações: Utilizada quando necessita-se informar ao programa,
o nome de um arquivo a ser manipulado.

Endereços importantes:

&hA250 (Endereço inicial);
&hA251 (Coordenada "Y");
&hA253 (coordenada "X");
&hA33B (Final da rotina);
&hDEC3 (área de "buffer").

Utilização:

Esta rotina de "accept" foi desenvolvida especialmente para ser utilizada no programa G-DESK10.PGM, entretanto poderá ser aproveitada em programas que manipulam arquivos constantes num disco. Sua extensão é fixa em 12 caracteres (oito para o nome, 1 para o ponto e 3 para a extensão do arquivo).

Quando a rotina é executada, acionando "defusr=&hA250:u=usr(0)" aparece nas coordenadas previamente definidas pelos endereços "&hA251" e "A253", o nome do programa que está no "buffer" que se inicia no endereço "&hDEC3". Tal nome está sob o "cursor", possibilitando a edição do mesmo. Se teclar-mos (RETURN) sairemos do modo de edição. A palavra editada poderá ser lida na tela através de uma simples rotina de "scanner" do conteúdo do vídeo, naquelas coordenadas.

Exemplo:

No exemplo abaixo, colocaremos a "string" "12345678.123" no "buffer" (linhas de 100 a 140); acionaremos a rotina de edição na coordenada 17,24 para que se possa modificar o texto (linhas de 150 a 230). Ao finalizarmos a edição pressionando (RETURN), será executada uma rotina de leitura do vídeo nas mesmas coordenadas, que nos apresentará o novo texto já editado (linhas de 240 a 280):

```
100 A$ = "12345678.123"
110 FOR C = 1 TO 12
120 A = ASC(MID$(A$,C,1))
130 POKE &HDEC3+C,A
140 NEXT C
145
150 CLS : POKE &HFCAB , 1
160 LOCATE 17 , 11
170 PRINT "Modifique:"
180 X=17 : Y=14
190 DEFUSR=&H156 : U=USR(0)
200 POKE &HA251 , Y
210 POKE &HA253 , X
220 DEFUSR=&HA250 : U=USR(0)
230 POKE &HFCAB , 0
235 '
240 A$ = "" : FOR C = 0 TO 11
250 A$=A$+CHR$(VPEEK((X-1)-(Y-1)*32+(6144+C)))
260 NEXT C
270 LOCATE 13 , 15
280 PRINT "Novo: " ; A$
```

Informações técnicas:

Quando a rotina for executada, será necessário que o micro esteja com "CAPS-LOCK" acionado. Para garantir o perfeito funcionamento, devemos travar as maiúsculas através do endereço "&hFCAB" com o valor "1" (poke &hFCAB,1). Depois de executada a rotina, poderemos retomar a normalidade com o valor "0" (poke &hFCAB,0).

Esta rotina pode ser utilizada fora do SISTEMA GRADIUS 1.0 desde que estejamos em "SCREEN 1".

GRADIUS ACCEPT ROUTINE 1.0

Arquivo: G-FLAC10.MLF (&hA250,&hA33B,&hA250)

Função: Aceita a entrada via teclado de uma palavra de 12 letras.

Aplicações: Utilizada quando necessita-se informar ao programa,
o nome de um arquivo a ser manipulado.

Endereços importantes:

&hA250 (Endereço inicial);
&hA251 (Coordenada "Y");
&hA253 (coordenada "X");
&hA33B (Final da rotina);
&hDEC3 (área de "buffer").

Utilização:

Esta rotina de "accept" foi desenvolvida especialmente para ser utilizada no programa G-DESK10.PGM, entretanto poderá ser aproveitada em programas que manipulam arquivos constantes num disco. Sua extensão é fixa em 12 caracteres (oito para o nome, 1 para o ponto e 3 para a extensão do arquivo).

Quando a rotina é executada, acionando "defusr=&hA250:u=usr(0)" aparece nas coordenadas previamente definidas pelos endereços "&hA251" e "A253", o nome do programa que está no "buffer" que se inicia no endereço "&hDEC3". Tal nome está sob o "cursor", possibilitando a edição do mesmo. Se teclar-mos (RETURN) sairemos do modo de edição. A palavra editada poderá ser lida na tela através de uma simples rotina de "scanner" do conteúdo do vídeo, naquelas coordenadas.

GRADIUS DISK FORMATER 1.0

Arquivo: G-DKFM10.MLF (&hD200,&hD258,&hD200)

Função: Formata um disco seguindo formato previamente selecionado

Aplicações: Preparação de um disco para utilização, sem
necessidade de se utilizar o comando "FORMAT".

Endereços importantes:

&hD200 (Endereço inicial);
&hD20B (Formato do disco);
&hD239 (Endereço de entrada);
&hD258 (Final da rotina).

Utilização:

Nos programas do SISTEMA GRADIUS 1.0, a apresentação de informações em geral são impressas dentro de molduras para destacarem-se na tela. É o que chamamos de "janelas".

Para que possamos "formatar" um disco para gravação, sem ter que utilizar o comando "FORMAT" que não poderia ser executado dentro de "janelas", foi criada esta rotina de formatação alternativa.

O tipo de formatação escolhida (40 ou 80 trilhas em face simples ou dupla) deve ser colocada no endereço "&hD20B".

Exemplo:

```
100 CLS
110 DATA "(1) 40 TRILHAS FS"
120 DATA "(2) 40 TRILHAS FD"
130 DATA "(3) 80 TRILHAS FS"
140 DATA "(4) 80 TRILHAS FD"
150 FOR C = 10 TO 13
160 LOCATE 7,C
170 READ A$ : PRINT A$
180 NEXT C
190 LOCATE 8 , 15
200 PRINT "Sua escolha ? " ;
210 A$ = INPUT$(1) : PRINT A$
220 POKE &HD20B , VAL(A$)
230 A$ = "" : OUT(&HD4),1
240 A = INP(&HDO) : OUT(&HD4),0
250 A$ = BIN$(A)
260 IF MID$(A$,2,1) = "1" THEN END
270 DEFUSR=&HD239 : U=USR(0)
280 DEFUSR=&HD200 : U=USR(0)
290 LOCATE 10 , 17
300 PRINT "FORMATADO !"
```

Informações técnicas:

Esta rotina pode ser utilizada dentro e fora do SISTEMA GRADIUS 1.0, com qualquer sistema de drives e interfaces controladoras existentes para o padrão MSX.

GRADIUS MOUSE CONTROLER 1.1

Arquivo: G-MOCT11.MLF (&hA400,&hA482,&hA400)

Função: Controla o periférico "mouse".

Aplicações: Utilização em programas onde seja necessária
a movimentação de um "cursor" pela tela.

Endereços importantes: &hA400 (Endereço inicial);
&hA3FE (Contém coordenada "Y");
&hA3FF (Contém coordenada "X");
&hA482 (Final da rotina).

Utilização:

A rotina de controle do periférico "mouse" substitui perfeitamente às rotinas de controle pelas "setas direcionais" ou por "joystick" em editores gráficos ou programas que utilizam recursos iconográficos em geral.

Foram utilizados para teste, um periférico nacional e um importado, ambos desenvolvidos para utilização em micro-computadores MSX.

Os endereços "&hA3FE" e "&hA3FF contêm valores relativos às coordenadas do "cursor", e a função "PDL" retorna o valor "0" (zero) quando um dos botões acionadores do periférico são pressionados.

Exemplo:

```
100 FOR C = 1 TO 8 : READ A$
110 B$ = B$+CHR$(VAL("&b"+A$))
120 NEXT C : SPRITE$(0) = B$
130 X = 10 : Y = 10 : CLS
140 DATA 00011000
150 DATA 00011000
160 DATA 00011000
170 DATA 11100111
180 DATA 11100111
190 DATA 00011000
200 DATA 00011000
210 DATA 00011000
220 DEFUSR=&HA400 : U=USR(0)
230 A = PEEK(&HA3FF)
240 B = PEEK(&HA3FE)
250 IF A>127 THEN A = A - 256
260 IF B>127 THEN B = B - 256
270 X = X - A : Y = Y - B
280 IF PDL(9)=0 THEN PLAY "A"
290 IF PDL(11)=0 THEN PLAY "B"
300 PUTSPRITE 0,(X,Y) : GOTO220
```

Informações técnicas:

A rotina G-MOCTF11.MLF tem "&h482" bytes e é perfeitamente relocável para outras áreas da memória (manualmente ou com o auxílio do utilitário G-MLM010.TLS do GRADIUS TOOLS #1), desde que não coincida com regiões reservadas para outras rotinas em utilização (consulte a tabela de mapeamento da memória para maiores informações). Devem ser consideradas entretanto, as posições "&hA3FE" e "&hA3FF" que são fixas.

Esta rotina pode ser utilizada dentro e fora do SISTEMA GRADIUS 1.0, em "SCREEN 1" e em qualquer um dos modos gráficos do MSX1, além da "SCREEN 8" nos MSX2 e MSX2+.

GRADIUS SCREEN PRINTER 1.1

Arquivo: G-SCPR11.MLF (&hA500,&hA75F,&hA515)

Função: Imprime a tela ou o conteúdo do vídeo na
impressora gráfica matricial.

Aplicações: Simples "HARDCOPY" a qualquer momento durante
a execução de um programa.

Endereços importantes: &hA500 (Endereço inicial);
&hA515 (Endereço de execução);
&hA75F (Final da rotina).

Utilização:

A rotina de impressão ("screen printer") é de utilização extremamente simples e é indispensável quando precisamos de cópias impressas em papel do conteúdo do vídeo.

Em qualquer parte do programa basta comandar "defusr=&hA515:u=usr(0)" para que seja realizada a impressão no tamanho 110 x 67 mm.

Se for comandado "defusr=&hA500:u=usr(0)", a impressão se efetuará quando a tecla (ESC) for pressionada.

Exemplos:

O programa ilustrado abaixo cria uma tela e executa a impressão:

```
10 CLS
20 FOR C = 1 TO 20
30 LOCATE 1,C
40 PRINT STRING$(30,65)
50 NEXT C
60 DEFUSR=&HA515
70 U=USR(0)
```

Com as seguintes alterações, o programa anterior aguarda que a tecla (ESC) seja pressionada para iniciar a impressão:

```
60 DEFUSR=&HA500
80 GOTO 80
```

Informações técnicas:

A rotina G-SCPR11.MLF tem "&h75F" bytes e é perfeitamente relocável para outras áreas da memória (manualmente ou com o auxílio do utilitário G-MLM010.TLS do GRADIUS TOOLS #1), desde que não coincida com regiões reservadas para outras rotinas em utilização (consulte a tabela de mapeamento da memória para maiores informações).

Esta rotina pode ser utilizada fora do SISTEMA GRADIUS 1.0, para "HARDCOPY" de telas em modo gráfico "SCREEN 2".

O endereço "&hFCAF" retorna o modo de tela em uso, podendo ser útil para prevenir impressões fora do modo "SCREEN 2".

Não é possível utilizar a rotina com os modos gráficos adicionais existentes nos MSX2 e MSX2+.

GRADIUS SCREEN PRINTER 1.1

Arquivo: G-SCPR11.MLF (&hA500,&hA75F,&hA515)

Função: Imprime a tela ou o conteúdo do vídeo na
impressora gráfica matricial.

Aplicações: Simples "HARDCOPY" a qualquer momento durante
a execução de um programa.

Endereços importantes: &hA500 (Endereço inicial);
&hA515 (Endereço de execução);
&hA75F (Final da rotina).

Utilização:

A rotina de impressão ("screen printer") é de utilização extremamente simples e é indispensável quando precisamos de cópias impressas em papel do conteúdo do vídeo.

Em qualquer parte do programa basta comandar "defusr=&hA515:u=usr(0)" para que seja realizada a impressão no tamanho 110 x 67 mm.

Se for comandado "defusr=&hA500:u=usr(0)", a impressão se efetuará quando a tecla (ESC) for pressionada.

GRADIUS SCROLL UP ROUTINE 1.0

Arquivo: G-SLUP10.MLF (&hE000,&hE046,&hE000)

Função: Executa o efeito de "scroll" no sentido de baixo para cima.

Aplicações: Utilizado quando necessita-se deslocar a tela
para simular um movimento ou simplesmente limpá-la.

Endereços importantes: &hE000 (Endereço inicial);
&hE046 (Final da rotina).

Utilização:

A rotina de "scroll" para cima ("scroll up") é muito fácil de se utilizar. Basta acioná-la com "defusr=&hE000:u=usr(0)" para que todo o conteúdo do vídeo corra uma linha para cima. Para que a tela toda corra para cima, basta colocar o comando de acionamento num "loop" que o ative 24 vezes (o número de linhas da tela).

Exemplo:

```
10 CLS
20 FOR C = 1 TO 22
30 LOCATE 1,C
40 PRINT STRING$(182,30)
50 NEXT C
60 FOR C = 1 TO 23
70 DEFUSR=&HE000 : U=USR(0)
80 NEXT C
90 GOTO 20
```

Informações técnicas:

Esta rotina pode ser utilizada fora do SISTEMA GRADIUS 1.0 desde que estejamos em "SCREEN 1".

GRADIUS SCROLL RIGHT ROUTINE 1.0

Arquivo: G-SLRG10.MLF (&hEOB2,&hEOF1,&hEOB2)

Função: Executa o efeito de "scroll" no sentido da esquerda para a direita.

**Aplicações: Utilizado quando necessita-se deslocar a tela
para simular um movimento o simplesmente limpá-la.**

**Endereços importantes: &hEOB2 (Endereço inicial);
&hEOF1 (Final da rotina).**

Utilização:

A rotina de "scroll" para a direita ("scroll right") é muito fácil de se utilizar. Basta acioná-la com "defusr=&hEOB2:u=usr(0)" para que todo o conteúdo do vídeo corra uma linha para direita. Para que a tela toda corra para direita, basta colocar o comando de acionamento num "loop" que o ative 32 vezes (o número de colunas da tela).

Exemplo:

```
10 CLS
20 FOR C = 1 TO 22
30 LOCATE 1,C
40 PRINT STRING$(182,30)
50 NEXT C
60 FOR C = 1 TO 23
70 DEFUSR=&HEOB2 : U=USR(0)
80 NEXT C
90 GOTO 20
```

Informações técnicas:

Esta rotina pode ser utilizada fora do SISTEMA GRADIUS 1.0 desde que estejamos em "SCREEN 1".

TRUQUES DE PROGRAMAÇÃO

Enumeramos abaixo, uma seleção de posições úteis da memória e truques de programação em geral, que certamente terão utilidade para os usuários que iniciam no BASIC.

```
Poke &hF247,N - Muda o drive em uso (N é o número do drive desejado);
Poke &hF3DB,N - Se N=0 desabilita o "clic" de teclado. Se N=1 reabilita;
Poke &hF417,N - Se N=0 a impressora imprime em ABNT. Se N=1 imprime em MSX;
Poke &hFBB1,N - Se N=1 desabilita o (CONTROL)+(STOP). Se N=0 reabilita;
Poke &hFCAB,N - Se N=1 trava as maiúsculas ("CAPS-LOCK"). Se N=0 destrava;
Poke &hFF07,N - Se N=&hC7 qualquer comando "reseta" o micro;
Poke &hFF89,N - Se N=&hC1 desabilita o comando "list". Se N=&hC8 reabilita;

Peek (&hF346) - Se retorna "0", não foi carregado o sistema operacional;
Peek (&hF347) - Retorna o número de drives lógicos do sistema;
Peek (&hF348) - Retorna o "slot" onde se encontra a interface de drive;
Peek (&hF3DC) - retorna a coordenada Y do cursor de texto;
Peek (&hF3DD) - retorna a coordenada X do cursor de texto;
Peek (&hF3E9) - retorna a cor de frente (não é válido para o GRADIUS BASIC);
Peek (&hF3EA) - retorna a cor do fundo (não é válido para o GRADIUS BASIC);
Peek (&hF3EB) - retorna a cor da borda ou moldura da tela;
Peek (&hFCAF) - Retorna o modo atual da tela ("screen" em operação);

DEFUSR=&H003E:U=USR(0) - Inicializa as teclas de função;
DEFUSR=&h0041:U=USR(0) - Desativa mostrador de tela;
DEFUSR=&h0044:U=USR(0) - Ativa mostrador de tela;
DEFUSR=&H0069:U=USR(0) - Elimina todos os "SPRITES" do vídeo;
DEFUSR=&h007E:U=USR(0) - Coloca o VDP em "screen 2";
DEFUSR=&h00C0:U=USR(0) - Executa o "beep";
DEFUSR=&h00CC:U=USR(0) - Elimina as teclas de função ("KEY OFF");
DEFUSR=&h00CF:U=USR(0) - Apresenta as teclas de função ("KEY ON");
DEFUSR=&H0156:U=USR(0) - Limpa o "buffer" de teclado;

N=INP(&h90) - Se X=&h7F a impressora está desconectada;

LOCATE (32-LEN(A$))/2,12 - Centraliza "A$" (palavra) na tela de 32 colunas.
```

Os programas que apresentamos em seguida foram elaborados para serem executados em GRADIUS BASIC, portanto, a maioria deles não funcionará adequadamente fora deste ambiente.

```
100 CLS : V = 1
110 X = INT (RND(1)*15)+1
120 Y = INT (RND(1)*12)+2
130 H = INT (RND(1)*10)+1
140 L = INT (RND(1)*14)+4
150 T = INT (RND(1)*5)+1
160 POKE&HDA6E,V : POKE&HDA7B,Y : POKE&HDA7C,X
170 POKE&HDA7D,H : POKE&HDA7E,L : POKE&HD90F,T
180 DEFUSR = &HD900 : U=USR(0) : GOTO110
```

O programa acima abre infinitas "janelas" de diferentes tipos e tamanhos em posições aleatórias da tela. Modificando as variáveis e teremos novos resultados.

```
100 ON ERROR GOTO 130
110 BLOAD "12345678.123"
120 LOCATE 0,0 : END
130 GOSUB 220
140 IF ERR = 53 THEN A$ = "file not found"
150 IF ERR = 56 THEN A$ = "invalid name"
160 IF ERR = 66 THEN A$ = "disk is full"
170 IF ERR = 67 THEN A$ = "no directory"
180 IF ERR = 68 THEN A$ = "disk protect"
190 IF ERR = 69 THEN A$ = "disk i/o error"
200 IF ERR = 70 THEN A$ = "disk offline"
210 LOCATE (32-LEN(A$))/2,12 : PRINT A$ : RESUME NEXT
220 POKE&HDA6E,1 : POKE&HDA7B,12 : POKE&HDA7C,9
230 POKE&HDA7D,1 : POKE&HDA7E,16 : POKE&HD90F,4
240 DEFUSR=&HD900 : U=USR(0) : RETURN
```

Este último exemplo é uma útil rotina de tratamento de erros para programas que utilizem entrada e saída de dados em disco. Lembre-se que esta rotina também poderá ser modificada para apresentar a ocorrência de outros tipos de erro no seu programa.

As rotinas de tratamento de erros são extremamente importantes para evitar uma interrupção durante a execução de um programa.

```
100 X=10 : Y=10 : L=12 : CLS : GOSUB 120
110 LOCATE 2,16 : PRINT A$ : END
120 LOCATE X,Y : PRINT CHR$(&HC4) : A$="" : C=0
130 B$ = INKEY$ : IF B$="" THEN 130
140 IF B$ = CHR$(&HD) THEN 190
150 IF (B$<>"" "ORB$<>CHR$(&H84)) AND B$<>CHR$(8) OR B$=CHR$(&H7F) THEN130
160 IF B$ = CHR$(8) AND C<>0 THEN C=C-1 : A$ = LEFT$(A$,LEN(A$)-1) : LOCA
TE X,Y : PRINTA$ CHR$(&HC4) " " : GOTO130 ELSE IF B$=CHR$(8) THEN 130
170 IF C=L THEN 130
180 C=C+1 : A$=A$+B$ : LOCATE X,Y : PRINT A$ CHR$(&HC4) " " : GOTO130
190 RETURN
```

Acima temos uma rotina de "accept" para entrada de dados. Esta rotina poderá ser muito útil na maioria dos programas criados pelo usuário, substituindo com vantagens o comando "input" do BASIC.

```
100 CLS
110 FOR C = 1 TO 22
120 LOCATE 1,C
130 PRINT STRING$ (30,182)
140 NEXT C
150 FOR C = 1 TO 7
160 VPOKE 1455+C,&B00000001
170 VPOKE 1455+2048+C,&B00000001
180 VPOKE 1455+4096+C,&B00000001
190 NEXT C
200 FOR C = 1 TO 7
210 VPOKE 1455+C,&B11111110
220 VPOKE 1455+C+2048,&B11111110
230 VPOKE 1455+C+4096,&B11111110
240 NEXT C
250 IF INKEY$ = "" THEN 150
260 END
```

Este último programa gera um efeito especial que demonstra um pouco das potencialidades da "screen mista". Neste exemplo em questão, foi criado um efeito de movimento modificando os atributos do caracter "&hB7". Pode-se fazer o mesmo com outros caracteres ou conseguir novos efeitos mexendo-se nos números binários apresentados.

```
100 CLS : X=5 : Y=5
110 IF S=-1 THEN GOTO 190
120 S=STRIG(0) : IF S=0 THEN S=STRIG(1)
130 Z=STICK(0) : IF Z=0 THEN Z=STICK(1)
140 X=X-1*(Z=2ORZ=3ORZ=4)+1*(Z=6ORZ=7ORZ=8)
150 Y=Y-1*(Z=4ORZ=5ORZ=6)+1*(Z=8ORZ=1ORZ=2)
160 IF X>31 THEN X=0 ELSE IF X<0 THEN X=31
170 IF Y>22 THEN Y=0 ELSE IF Y<0 THEN Y=22
180 LOCATE X,Y : PRINT CHR$(&HE0) : GOTO 110
190 CLS : END
```

Acima temos uma rotina de controle de uma seta pelas teclas do cursor ou "joystick". Para melhor resultado, recomendamos utilizar uma "sprite" no lugar do caracter "&hE0". Ao ser pressionada a "barra de espaço" ou um "tiro" do "joystick", o programa desvia para a linha 190, terminando sua execução.

```
100 CLS
110 C=INT(RND(1)*14)+1
120 X=INT(RND(1)*30)+1
130 Y=INT(RND(1)*22)+1
140 LOCATE X,Y : PRINT CHR$(&HB5+C)
150 GOTO 110
```

Este último exemplo é um novo efeito especial. Troque a linha 150 e verifique os novos resultados:

```
150 LOCATE 0,23 : PRINT : GOTO 110
```

Experimente outras alterações.

```
10 A$ = " NEMESIS INFORMATICA "
20 B$ = "" : FOR C=1 TO LEN(A$)
30 B$ = B$ + CHR$(ASC(MID$(A$,C,1))+164)
40 NEXT C : PRINT B$
```

A rotina acima inverte a "string" A$ para destaques no texto. Ela funciona trocando cada caracter contido na expressão por seu "sósia" de cores invertidas. Para isso, basta somar 164 ao número do caracter e imprimir o resultado no video.

```
10 LOCATE 10,10
20 PRINT "12345678.123"
30 X = 10 : Y = 10 : L = 12 : A$ = "" : FOR C = 0 TO L
40 A$=A$+CHR$(VPEEK(X+Y*32+(6144+C)))
50 NEXT C : LOCATE 2,16 : PRINT A$
```

Nas linhas de 10 a 20 do programa acima, imprimimos a "string" "12345678.123" nas coordenadas 10x10. As linhas que se seguem exemplificam uma rotina de "scanner" do vídeo naquelas coordenadas. Note que o comprimento da "string" obtida pode variar de acordo com a variável "L".

```
100 C1 = 8 : C2 = 7 : C1$ = HEX$(C1) : C2$ = HEX$(C2) : C3$ = "&h"+C1$+C2$
110 FOR C = &H2100 TO &H22D0
120 VPOKE C,VAL(C3$) : VPOKE C+&H800,VAL(C3$) : VPOKE C+&H1000,VAL(C3$)
130 NEXT C
```

Com o programa acima, implementamos algo muito similar ao comando "color" ao ambiente GRADIUS. Em "C1" colocamos o número da cor dos caracteres e na variável "C2" a cor de fundo da tela. No exemplo utilizamos vermelho médio sobre azul claro, que equivaleria ao comando "color 8,7" em BASIC. Note que o processo é bastante lento, e ficaria muito melhor se usassemos uma rotina em linguagem de máquina.

```
100 DATA 21,D8,2A,1,C0,1,3E,94,CD,56,0,21,D8,2A,1
120 DATA C0,1,3E,84,CD,56,0,21,D8,2A,1,C0,1,3E,64,CD
140 DATA 56,0,21,D8,2A,1,C0,1,3E,84,CD,56,0,21,D8,2A
160 DATA 1,C0,1,3E,94,CD,56,0,21,D8,2A,1,C0,1,3E,F4
180 DATA CD,56,0,CD,D8,0,A7,CA,0,D4,C9
200 FOR C = &HD400 TO &HD449
210 READ A$
220 POKE C,VAL("&H"+A$)
230 NEXT C
240 CLS : LOCATE 6,10
250 PRINT "nemesis informatica"
260 DEFUSR = &HD400 : U = USR(0)
```

A rotina acima faz algo parecido com o penúltimo exemplo, mostrando entretanto, a velocidade do processo de variação de cores em linguagem de máquina. Este é um efeito interessante para a tela de abertura de programas.

CONSIDERAÇÕES FINAIS

Esperamos que este manual de instruções tenha cumprido o seu papel de orientar o usuário na utilização deste software. E esperamos ainda mais que o próprio programa tenha alcançado as suas espectativas. Em todo caso, gostaríamos de receber suas críticas e sugestões que nos ajudarão a incrementar ainda mais as futuras versões do mesmo, e de outros lançamentos da linha.

ATENÇÃO

GARANTIA E SUPORTE TÉCNICO

A NEMESIS INFORMATICA LTDA. garante por 5 (cinco) anos o perfeito funcionamento deste software e de seus acessórios, bem como de outros produtos desenvolvidos pela empresa. O suporte técnico é gratuito. Pedimos o favor de não solicitar informações técnicas por telefone. Consultas deverão ser feitas através de carta registrada, onde o cliente deverá mencionar o número de sua cópia, data e local de aquisição.

Toda correspondência deverá ser enviada para:

NEMESIS INFORMATICA LTDA.
Caixa postal 4.583 Cep 20.001
Rio de Janeiro - RJ - Brasil.