

# **MSX-BASIC**

## **GUIA DE CONSULTA RÁPIDA**

Escrito e editorado por:  
Edison Antonio Pires de Moraes  
eapmoraes@msxall.com  
www.msxtop.msxall.com

22/05/2004



## GUIA DE CONSULTA RÁPIDA DO MSX-BASIC

### **ABS** (função, 1)

Formato: X = ABS (<exprN>)

Função: Retorna em X o valor absoluto (módulo) de <exprN>.

### **AND** (operador lógico, 1)

Formato: <exprA1> AND <exprA2>

Função: Efetua operação lógica AND entre <exprA1> e <exprA2>.

### **ASC** (função, 1)

Formato: X = ASC (<expr\$>)

Função: Retorna em X o código ASCII do primeiro caractere de expr\$.

### **ATN** (função, 1)

Formato: X = ATN (<exprN>)

Função: Retorna em X o valor do arcotangente de exprN (exprN deve ser expresso em radianos).

### **AUTO** (comando, 1)

Formato: AUTO [numlinha, [incremento]]

Função: Gera automaticamente números de linha, iniciando com [numlinha] e incrementado com o valor de [incremento].

### **BASE** (variável de sistema, 1-2-3)

Formato: X = BASE (<n>) | BASE (<n>) = <exprN>

Função: Retorna em X ou define os endereços de início das tabelas na VRAM para cada modo de tela. <n> é um número inteiro que segue a seguinte tabela:

		MODOS DE TELA											TABELA DE	
		SC0	SC1	SC2	SC3	SC4	SC5	SC6	SC7	SC8	SC10	SC11		SC12
VALOR	BASE	0	5	10	15	20	25	30	35	40	50	55	60	Nomes dos padrões
		6	11	16	21	26	31	36	41	51	56	61	Cores	
	2	7	12	17	22	27	32	37	42	52	57	62	Geradora de padrões	
		8	13	18	23	28	33	38	43	53	58	63	Atributos dos sprites	
		9	14	19	24	29	34	39	44	54	59	64	Geradora de sprites	

### **BEEP** (declaração, 1)

Formato: BEEP

Função: Gera um beep.

### **BIN\$** (função, 1)

Formato: X\$ = BIN\$(<exprN>)

Função: Converte o valor de <exprN> em uma string de códigos binários e retorna o valor obtido em X\$.

**BLOAD** (comando, 1-D)

Formato: BLOAD "<nomearq>"[,R[,<offset>]]

BLOAD "<nomearq>"[{,R | ,S}][, <offset>]] (D)

Função: Carrega um bloco binário na RAM ou na VRAM (,S). Se especificado [,R], executa programa em código de máquina.

**BSAVE** (comando, 1-D)

Formato: BSAVE "<nomearq>",<endini>,<endfim>[,<endexec>]

BSAVE "<nomearq>",<endini>,<endfim>[,<endexec>[,S]] (D)

Função: Salva em disco ou fita um bloco binário. Se especificado ,S salva um bloco da VRAM.

**CALL** (declaração, 1-2-3-4-D-M)

Formato: CALL <comando estendido> [( <argumento> [,argumento> ... ] )]

Função: Executa comandos estendidos através de cartuchos de ROM.

**CDBL** (função, 1)

Formato: X# = CDBL(<exprN>)

Função: Converte o valor de <exprN> em um valor de dupla precisão e retorna o valor obtido em X#.

**CHR\$** (função, 1)

Formato: X\$ = CHR\$(<exprN>)

Função: Retorna em X\$ o caractere cujo código ASCII é expressado em <exprN>.

**CINT** (função, 1)

Formato: X% = CINT(<exprN>)

Função: Converte o valor de <exprN> em um valor inteiro e retorna o valor obtido em X%.

**CIRCLE** (declaração, 1-2)

Formato: CIRCLE {(X,Y) | STEP(X,Y)},<raio>[,<cor>[,<ângulo inicial> [,< ângulo final>[,<proporção>]]]]

Função: Desenha uma circunferência com ponto central em (X,Y). Se for especificado STEP, as coordenadas serão calculadas a partir da atual. <ângulo inicial> e <ângulo final> devem ser especificados em radianos. <proporção> é a relação para elipse, sendo <1> circunferência perfeita.

**CLEAR** (declaração, 1)

Formato: CLEAR [<tamanho área string>[,limite superior memória>]]

Função: Inicializa as variáveis do BASIC e seta o tamanho da área para string e o limite superior de memória usado pelo BASIC.

**CLOAD** (comando, 1)

Formato: CLOAD ["nome arq"]

Função: Carrega um programa BASIC de fita cassete.

**CLOAD?** (comando, 1)

Formato: CLOAD? ["nome arq"]

Função: Compara um programa BASIC na fita cassete com o da memória.

**CLOSE** (comando, 1-D)

Formato: CLOSE [[#]<nº arquivo>[,[#]<nº arquivo>...]]

Função: Fecha os arquivos especificados. Se não for especificado nenhum arquivo, fecha todos os arquivos abertos.

**CLS** (declaração, 1)

Formato: CLS

Função: Limpa a tela.

**CMD** (comando, 1)

Formato: Sem formato definido.

Função: Reservado para implementação de novos comandos.

**COLOR** (declaração, 1-2)

Formato: COLOR [<cor frente>[,<cor fundo>[,<cor borda>]]] (1-2)

Função: Especifica as cores da tela. Os valores de cor podem variar de 0 a 15 para as screens 0 a 7 e 10 e de 0 a 255 para as screens 8, 11 e 12.

**COLOR =** (declaração, 2)

Formato: COLOR = (<nº paleta>,<nível verm.>,<nível verde>,<nível azul>)

Função: Especifica as cores da paleta. <nº de paleta> pode variar de 0 a 15 e os níveis de cor podem variar de 0 a 7.

**COLOR = NEW** (declaração, 2)

Formato: COLOR [= NEW]

Função: Restaura a paleta de cores para os valores iniciais.

**COLOR = RESTORE** (declaração, 2)

Formato: COLOR = RESTORE

Função: Copia o conteúdo da paleta de cores armazenada na VRAM para os registradores de paleta do VDP.

**COLOR SPRITE** (declaração, 1-2)

Formato: COLOR SPRITE (<nº do plano do sprite>)=<cor>

Função: Especifica a cor dos sprites. <nº do plano do sprite> pode variar de 0 a 31 e a cor é a da paleta (0 a 15).

**COLOR SPRITE\$** (declaração, 2)

Formato: COLOR SPRITE\$ (<nº do plano do sprite>)=<expr\$>  
 onde <expr\$> = CHR\$(cor 1ª linha) + CHR\$(cor 2ª linha) ...

Função: Especifica a cor de cada linha dos sprites. A cor é a da paleta (0 a 15).

**CONT** (comando, 1)

Formato: CONT

Função: Continua a execução de um programa que foi interrompido.

**COPY** (declaração, 1-2-D)

Formato: COPY "nomearq1" [TO "nomearq2"] (1-D)

Função: Copia o conteúdo de <nomearq1> para <nomearq2>.

Formato: COPY (X1,X2)-(Y1,Y2) [,<página fonte>] TO (X3,Y3) [,<página destino>[,<operação lógica>]] (2)

Função: Copia uma área retangular da tela para outra.

Formato: COPY (X1,X2)-(Y1,Y2) [,<página fonte>] TO {<variável matriz | <nomearq>} (2-D)

Função: Copia o conteúdo de uma área retangular da tela para uma variável matriz ou para um arquivo em disco.

Formato: COPY {<variável matriz> | <nomearq>} [,<direção>] TO (X3,Y3) [,<página destino>[,<operação lógica>]] (2-D)

Função: Copia o conteúdo de uma variável matriz ou de um arquivo em disco para uma área retangular na tela.

Formato: COPY <nomearq> TO <variável matriz> (2-D)

Função: Copia o conteúdo de um arquivo para uma variável matriz.

Formato: COPY <variável matriz> TO <nomearq> (2-D)

Função: Copia o conteúdo de uma variável matriz para um arquivo.

**COPY SCREEN** (declaração, 2, opcional)

Formato: COPY SCREEN [<modo>]

Função: Escreve os dados do Color Bus na VRAM.

**COS** (função, 1)

Formato: X = COS (<exprN>)

Função: Retorna em X o valor do cosseno de <exprN> (exprN deve ser expresso em radianos).

**CSAVE** (comando, 1)

Formato: CSAVE "<nomearq>" [,<baud rate>]

Função: Salva um programa BASIC na fita cassete.

**CSNG** (função, 1)

Formato: X! = CSNG(<exprN>)

Função: Converte o valor de <exprN> em um valor de precisão simples e retorna o valor obtido em X!.

**CSRLIN** (variável de sistema, 1)

Formato: X = CSRLIN

Função: Contém a posição vertical do cursor.

**CVD** (função, D)

Formato: X# = CVD (<string de 8 bytes>)

Função: Converte a string em um valor de dupla precisão e armazena o valor obtido em X#.

**CVI** (função, D)

Formato: X% = CVI (<string de 2 bytes>)

Função: Converte a string em um valor inteiro e armazena o valor obtido em X%.

**CVS** (função, D)

Formato: X! = CVS (<string de 4 bytes>)

Função: Converte a string em um valor de precisão simples e armazena o valor obtido em X!.

**DATA** (declaração, 1)

Formato: DATA <constante>[,<constante> ...]

Função: Armazena uma lista de dados para o comando READ.

**DEF FN** (declaração, 1)

Formato: DEF FN <nome> [(<argumento>[,<argumento>...])] = <expressão definidora de função de usuário>

Função: Define uma função do usuário.

**DEFDBL** (declaração, 1)

Formato: DEFDBL <faixa de caracteres>[,<faixa de caracteres>...]

Função: Declara as variáveis especificadas como dupla precisão.

**DEFINT** (declaração, 1)

Formato: DEFINT <faixa de caracteres>[,<faixa de caracteres>...]

Função: Declara as variáveis especificadas como inteiras.

**DEFSNG** (declaração, 1)

Formato: DEFSNG <faixa de caracteres>[,<faixa de caracteres>...]

Função: Declara as variáveis especificadas como precisão simples.

**DEFSTR** (declaração, 1)

Formato: DEFSTR <faixa de caracteres>[,<faixa de caracteres>...]

Função: Declara as variáveis especificadas como strings.

**DEFUSR** (declaração, 1)

Formato: DEFUSR[<número>] = <endereço>

Função: Define um endereço inicial para execução de programa assembly a ser chamado pela função USR.

**DELETE** (comando, 1)

Formato: DELETE {<linha inicial>-<linha final> | <linha> |-<linha final>}

Função: Apaga as linhas especificadas do texto BASIC.

**DIM** (declaração, 1)

Formato: DIM <variável> (<índice máximo>[,<índice máximo>...])

Função: Define uma variável matriz e aloca espaço na memória.

**DRAW** (macro declaração, 1)

Formato: DRAW <expr\$>

Função: Desenha uma linha de acordo com <expr\$>. Os comandos válidos para <expr\$> são os seguintes:

Un - para cima	Dn - para baixo	Ln - para esquerda
Rn - para direita	En - cima e direita	Fn - baixo e direita
Gn - baixo e esq.	Hn - cima e esq.	B - mov. sem desenho
N - volta origem	Mx,y - vai p/ X,Y	An - gira n*90 graus
Sn - escala n/4	Cn - cor n	Xsérie - macro em série

**DSKF** (função, D)

Formato: X = DSKF(<nº drive>)

Função: Retorna o espaço livre no drive especificado em clusters.

**EOF** (função, 1-D)

Formato: X = EOF(<nº do arquivo>)

Função: Retorna -1 (verdadeiro) caso o fim de arquivo seja detectado.

**ERASE** (declaração, 1)

Formato: ERASE <variável matriz>[,<variável matriz>...]

Função: Deleta as variáveis matriz especificadas.

**EQV** (operador lógico, 1)

Formato: <exprA1> EQV <exprA2>

Função: Efetua operação lógica EQV entre <exprA1> e <exprA2>.

**ERL** (variável de sistema, 1)

Formato: X = ERL

Função: Contém o número de linha onde o último erro ocorreu.

**ERR** (variável de sistema, 1)

Formato: X = ERR

Função: Contém o código de erro do último erro ocorrido.



**ERROR** (declaração, 1)

Formato: ERROR <código de erro>

Função: Coloca o programa na condição de erro.

**EXP** (função, 1)

Formato: X = EXP (<exprN>)

Função: Retorna em X o valor da potenciação natural de <exprN>.

**FIELD** (declaração, D)

Formato: FIELD [#]<nº arq>,<tamanho do campo> AS <nome var. string>[,<tamanho do campo> AS <nome var. string>...]

Função: Define a <var. string> para acesso aleatório ao disco.

**FILES** (comando, D)

Formato: FILES ["<nome arq>"]

Função: Apresenta os nomes de arquivos do disco de acordo com <nome arq>. Se <nome arq> for omitido, apresenta os nomes de todos os arquivos presentes no disco.

**FIX** (função, 1)

Formato: X = FIX(<exprN>)

Função: Retorna em X a parte inteira de <exprN>, sem arredondar.

**FOR** (declaração, 1)

Formato: FOR <nome variável> = <valor inicial> TO <valor final> [STEP <incremento>]

Função: Repete a execução do trecho entre o FOR e o NEXT.

**FRE** (função, 1)

Formato: FRE (0 | "")

Função: Retorna o tamanho da memória restante para o texto BASIC (0) ou para as variáveis string ("").

**GET** (declaração, D)

Formato: GET [#]<nº arq>[,<nº registro>]

Função: Lê um registro de um arquivo de acesso aleatório.

**GET DATE** (declaração, 2)

Formato: GET DATE <variável string> [,A]

Função: Retorna uma string com a data atual na <variável string>.

**GET TIME** (declaração, 2)

Formato: GET TIME <variável string> [,A]

Função: Retorna um string com a hora atual na <variável string>.

**GOSUB** (declaração, 1)

Formato: GOSUB <nº linha>

Função: Chama um subrotina que inicia na linha <nº linha>.

**GOTO** (declaração, 1)

Formato: GOTO <nº linha>

Função: Salta para a linha <nº linha>.

**HEX\$** (função, 1)

Formato: X\$ = HEX\$(<exprN>)

Função: Converte o valor de <exprN> em uma string hexadecimal e retorna o valor obtido em X\$.

**IF** (declaração, 1)

Formato: IF <condição> THEN {<comando> | <nº linha>} [ELSE  
{<comando> | <nº linha>}]

IF <condição> GOTO <nº linha> [ELSE <nº linha>]

Função: Executa comandos de acordo com a <condição>.

**IMP** (operador lógico, 1)

Formato: <exprA1> IMP <exprA2>

Função: Efetua operação lógica IMP entre <exprA1> e <exprA2>.

**INKEY\$** (função, 1)

Formato: X\$ = INKEY\$

Função: Retorna em X\$ um caractere quando a tecla está sendo pressionada; caso contrário, retorna uma string nula.

**INP** (função, 1)

Formato: X% = INP(<número da porta>)

Função: Lê uma porta de I/O do Z80 e retorna seu valor em X%.

**INPUT** (declaração, 1)

Formato: INPUT ["<prompt>";] <nome variável>[,<nome variável>...]

Função: Lê uma entrada de dados pelo teclado e armazena o(s) valor(es) obtido(s) na(s) variável(is) respectiva(s).

**INPUT#** (declaração, 1)

Formato: INPUT #<nº arq>, <nome variável>[,<nome variável>...]

Função: Lê dados do arquivo especificado e armazena o(s) valor(es) obtido(s) na(s) variável(is) respectiva(s).

**INPUT\$** (função, 1)

Formato: X\$ = INPUT\$ (<nº caracteres>[,<#>,<nº arq>])

Função: Lê o número especificado de caracteres do teclado ou de um arquivo e armazena o valor obtido em X\$.

**INSTR** (função, 1)

Formato: X = INSTR ([<exprN>],<expr\$1>,<expr\$2>)

Função: Procura a ocorrência de <expr\$2> em <expr\$1> a partir da posição <exprN> e retorna o valor obtido em X.

**INT** (função, 1)

Formato: X = INT (<exprN>)

Função: Retorna em X a parte inteira de <exprN>, arredondando.

**INTERVAL** (declaração, 1)

Formato: INTERVAL {ON | OFF | STOP}

Função: Ativa, desativa ou suspende interrupção por tempo.

**IPL** (comando, 1)

Formato: Sem formato definido.

Função: Reservado para implementação de novos comandos.

**KEY** (comando/declaração, 1)

Formato: KEY <número de tecla>,<expr\$>

Função: Redefine o conteúdo da tecla de função especificada.

Formato: KEY (<número de tecla>) {ON | OFF | STOP}

Função: Ativa, desativa ou suspende interrupção de tecla de função.

Formato: KEY {ON | OFF}

Função: Liga ou desliga a apresentação do conteúdo das teclas de função na última linha da tela.

**KEY LIST** (comando, 1)

Formato: KEY LIST

Função: Lista o conteúdo das teclas de função.

**KILL** (comando, D)

Formato: KILL "<nomearq>"

Função: Apaga arquivos no disco conforme especificado em <nomearq>.

**LEFT\$** (função, 1)

Formato: X\$ = LEFT\$ (<expr\$>,<exprN>)

Função: Retorna em X\$ os <exprN> caracteres esquerdos de <expr\$>.

**LEN** (função, 1)

Formato: X = LEN(<expr\$>)

Função: Retorna em X o número de caracteres de <expr\$>.

**LET** (declaração, 1)

Formato: [LET] <nome variável> = <exprA>

Função: Armazena na variável o valor de <exprA>.

**LFILES** (comando, 1)

Formato: LFILES ["<nomearq>"]

Função: Lista os nomes dos arquivos do disco na impressora de acordo com <nomearq>. Se <nomearq> for omitido, lista os nomes de todos os arquivos presentes no disco.

**LINE** (declaração, 1-2)

Formato: LINE [{(X1,Y1) | STEP(X1,Y1)}] - {(X2,Y2) | STEP(X2,Y2)}  
[,<cor>[, {B | BF} [, <operação lógica>]]]

Função: Desenha uma linha, um retângulo vazio (,B) ou um retângulo pintado (,BF). X1-Y1 são as coordenadas iniciais (ou um vértice do retângulo e X2-Y2 são as coordenadas finais (ou o outro vértice do retângulo). Se STEP for especificado, desenha relativamente à última coordenada. Os códigos de operação lógica estão listados no final deste guia.

**LINE INPUT** (declaração, 1)

Formato: LINE INPUT ["<prompt>";]<variável string>

Função: Lê uma seqüência de caracteres do teclado e armazena o valor lido na <variável string>.

**LINE INPUT #** (declaração, 1-D)

Formato: LINE INPUT #<nº arq>,<variável string>

Função: Lê uma seqüência de caracteres de um arquivo e armazena o valor lido na <variável string>.

**LIST** (comando, 1)

Formato: LIST [[<linha inicial>] - [<linha final>]]

Função: Lista na tela o programa BASIC que está na memória.

**LLIST** (comando, 1)

Formato: LLIST [[<linha inicial>] - [<linha final>]]

Função: Lista na impressora o programa BASIC que está na memória.

**LOAD** (comando, 1-D)

Formato: LOAD "<nomearq>" [,R]

Função: Carrega um programa na memória e opcionalmente o executa.

**LOC** (função, D)

Formato: X = LOC (<nº arq>)

Função: Retorna em X o número do último registro acessado do arquivo.

**LOCATE** (declaração, 1-2)

Formato: LOCATE [<cood. X>[,<cood. Y>[,<tipo cursor>]]]

Função: Posiciona o cursor nas telas de texto.

**LOF** (função, D)

Formato: X = LOF (<nº arq>)

Função: Retorna em X o tamanho do arquivo especificado.

**LOG** (função, 1)

Formato: X = LOG (<exprN>)

Função: Retorna em X o logaritmo natural de <exprN>.

**LPOS** (variável de sistema, 1)

Formato: X = LPOS

Função: Armazena a localização horizontal da cabeça da impressora.

**LPRINT** (declaração, 1)

Formato: LPRINT [<exprA>[{: | ,}<exprA>...]]

Função: Envia para a impressora os caracteres correspondentes às expressões <exprA>. “;” não alimenta filla de linha. “,” move para a posição de tabulação seguinte (em incrementos de 14 posições)

**LPRINT USING** (declaração, 1)

Formato: LPRINT USING <“forma”>;<exprA>[{: | ,}<exprA>...]

LPRINT USING <“forma expr\$”>

Função: Envia para a impressora os caracteres correspondentes às expressões <exprN> ou <expr\$>, formatando. Os caracteres usados para formatar a saída são os seguintes:

*Formatação numérica:*

- # Espaço para um dígito
- . Inclui ponto decimal
- + Indica + ou -; usado antes ou depois do número
- Indica -; usado depois do número
- \$\$ Coloca \$ à esquerda do número
- \*\* Substitui espaços à esquerda por asteriscos
- \*\*\$ Coloca um \$ à esquerda precedido por asteriscos
- ^^^ Apresenta o número em notação científica

*Formatação alfanumérica:*

- \ \ Espaço para caracteres
- ! Espaço para um caractere
- & Espaçamento variável
- \_ Próximo caractere é impresso normalmente
- outro Imprime caractere

**LSET** (declaração, D)

Formato: LSET <variável string> = <expr\$>

Função: Armazena o conteúdo de <expr\$> à esquerda na variável string definida pela declaração FIELD.

**MAXFILES** (declaração, 1-D)

Formato: MAXFILES = <número de arquivos>

Função: Define o número máximo de arquivos que podem ser abertos ao mesmo tempo.

**MERGE** (comando, 1-D)

Formato: MERGE "<nomearq>"

Função: Intercala o programa na memória com um programa salvo no formato ASCII em disco ou fita.

**MID\$** (função/declaração, 1)

Formato: X\$ = MID\$ (<expr\$>, <exprN1>[, <exprN2>])

Função: Retorna, em X\$, <exprN2> caracteres a partir do caractere <exprN1> de <expr\$>.

Formato: MID\$ (<variável string>, <exprN1>[, <exprN2>]) = <expr\$>

Função: Define <expr\$> usando <exprN2> caracteres a partir da posição <exprN1> da <variável string>.

**MKD\$** (função, D)

Formato: X\$ = MKD\$ (<valor de dupla precisão>)

Função: Converte um valor de dupla precisão em uma string de 8 bytes e a armazena em X\$.

**MKI\$** (função, D)

Formato: X\$ = MKI\$ (<valor inteiro>)

Função: Converte um valor inteiro em uma string de 2 bytes e a armazena em X\$.

**MKS\$** (função, D)

Formato: X\$ = MKS\$ (<valor de precisão simples>)

Função: Converte um valor de precisão simples em uma string de 4 bytes e a armazena em X\$.

**MOTOR** (declaração, 1)

Formato: MOTOR [{ON | OFF}]

Função: Liga ou desliga o motor do cassete.

**NAME** (comando, D)

Formato: "<nomearq1>" AS "<nomearq2>"

Função: Renomeia o arquivo <nomearq1> com <nomearq2>.

**NEW** (comando, 1)

Formato: NEW

Função: Deleta o programa da memória e limpa as variáveis.

**NEXT** (declaração, 1)

Formato: NEXT [<nome da variável>[,<nome da variável>...]]

Função: Indica o fim do laço FOR.

**NOT** (operador lógico, 1)

Formato: NOT (<exprA>)

Função: Efetua a negação de <exprA>.

**OCT\$** (função, 1)

Formato: X\$ = OCT\$ (<exprN>)

Função: Converte o valor de <exprN> em uma string octal e retorna o valor obtido em X\$.

**ON ERROR GOTO** (declaração, 1)

Formato: ON ERROR GOTO <número de linha>

Função: Define a linha inicial da rotina para manipulação de erro.

**ON GOSUB** (declaração, 1)

Formato: ON <exprN> GOSUB <nº linha>[,<nº linha>...]

Função: Executa a subrotina em <nº linha> de acordo com <exprN>.

**ON GOTO** (declaração, 1)

Formato: ON <exprN> GOTO <nº linha>[,<nº linha>...]

Função: Salta para a linha <nº linha> de acordo com <exprN>.

**ON INTERVAL GOSUB** (declaração, 1)

Formato: ON INTERVAL = <tempo> GOSUB <nº linha>

Função: Define o intervalo e o número da linha para interrupção de tempo.

**ON KEY GOSUB** (declaração, 1)

Formato: ON KEY GOSUB <nº linha>[,<nº linha>...]

Função: Define os números de linha para interrupção de teclas de função.

**ON SPRITE GOSUB** (declaração, 1)

Formato: ON SPRITE GOSUB <nº linha>

Função: Define o número de linha para interrupção por colisão de sprites.

**ON STOP GOSUB** (declaração, 1)

Formato: ON STOP GOSUB <nº linha>

Função: Define o número de linha para interrupção pelo pressionamento das teclas CTRL+STOP.

**ON STRIG GOSUB** (declaração, 1)

Formato: ON STRIG GOSUB <nº linha>[,<nº linha>...]

Função: Define os números de linha para interrupção pelo pressionamento dos botões de disparo do joystick.

**OPEN** (declaração, 1-D)

Formato: OPEN "<nomearq>" [FOR {INPUT | OUTPUT}] AS #<nº arq>  
[LEN=<tamanho do registro>]

Função: Abrir um arquivo em fita ou disco.

**OR** (operador lógico, 1)

Formato: <exprA1> OR <exprA2>

Função: Efetua operação lógica OR entre <exprA1> e <exprA2>.

**OUT** (declaração, 1)

Formato: OUT <nº da porta>, <exprN>

Função: Escreve o valor de <exprN> em uma porta de I/O do Z80.

**PAD** (função, 1-2)

Formato: X% = PAD (<exprN>)

Função: Examina o estado do mouse, trackball, caneta ótica ou tablete digitalizador conectados nas portas de joystick e retorna o valor obtido em X. O valor de retorno pode variar de 0 a 255 e <exprN> pode variar de 0 a 3 para touch pad conectado na porta A e 4 a 7 para touch pad conectado na porta B, 8 a 11 para caneta ótica, 12 a 14 para mouse conectado na porta A ou 16 a 18 para mouse conectado na porta B (15 e 19 não são usados. Para os botões, deve ser usada a função STRIG) de acordo com a tabela abaixo:

0, 4, 8, 12, 16 - retorna o status de atividade (255 para ativo e 0 para não ativo).

1, 5, 9, 13, 17 - retorna a coordenada X (horizontal).

2, 6, 10, 14, 18 - retorna a coordenada Y (vertical).

3, 7, 11 - retorna o status de tecla (255 se estiver sendo pressionada e 0 se não estiver).

**PAINT** (declaração, 1-2)

Formato: PAINT {(X,Y) | STEP(X,Y)} [, <cor> [, <cor da borda>]]

Função: Preenche a área delimitada por uma linha com a cor <cor da borda> com a cor <cor>.

**PDL** (função, 1)

Formato: X% = PAD (<nº paddle>)

Função: Retorna em X% o estado do paddle especificado. O valor retornado pode variar de 0 a 255 e o <nº paddle> pode variar de 1 a 12, conforme tabela abaixo:

1, 3, 5, 7, 9, 11 - paddles ligados na porta A de joystick;

2, 4, 6, 8, 10, 12 - paddles ligados na porta B de joystick;

**PEEK** (função, 1)

Formato: X = PEEK (<endereço>)

Função: Retorna em X o conteúdo do byte especificado por <endereço>.





**Obs.:** Tn, Vn, @Vn, Rn, X, =x; e . são idênticos aos outros instrumentos.

O valor <n> pode ser:

0 Toca somente o PSG (igual a PLAY)

1 Toca através da interface MIDI.

2 ou 3 Toca através do PSG e do OPLL (as 9 primeiras vozes são do OPLL e as três últimas do PSG).

### **POINT** (função, 1)

Formato: X% = POINT (X,Y)

Função: Retorna em X o código de cor do ponto (X,Y) da tela gráfica.

### **POKE** (declaração, 1)

Formato: POKE <endereço>,<exprN>

Função: Escreve no <endereço> de memória um byte de dados. <exprN> deve resultar um valor numérico entre 0 e 255.

### **POS** (variável de sistema, 1)

Formato: X = POS(0)

Função: Armazena a posição horizontal do cursor no modo texto.

### **PRESET** (declaração, 1-2)

Formato: PRESET {(X,Y) | STEP(X,Y)} [,<cor> [,<operação lógica>]]

Função: Apaga o ponto especificado por (X,Y) na tela gráfica.

### **PRINT** (declaração, 1)

Formato: PRINT [<exprA>[{: | ,}<exprA>...]]

Função: Apresenta na tela os caracteres correspondentes às expressões <exprA>.

### **PRINT#** (declaração, 1-D)

Formato: PRINT#<nº arq>,<exprA>[{: | ,}<exprA>...]]

Função: Escreve o valor de <exprA> no arquivo especificado.

### **PRINT USING** (declaração, 1)

Formato: PRINT USING <"formato">,<exprN>[{: | ,}<exprN>...]]

PRINT USING <"formato expr\$">

Função: Apresenta na tela os caracteres correspondentes às expressões <exprN> ou <expr\$>, formatando. Os caracteres usados para formatar a saída estão descritos na página seguinte.

*Formatação numérica:*

# Espaço para um dígito

. Inclui ponto decimal

+ Indica + ou -; usado antes ou depois do número

- Indica -; usado depois do número

\$\$ Coloca \$ à esquerda do número

\*\* Substitui espaços à esquerda por asteriscos

\*\*\$ Coloca um \$ à esquerda precedido por asteriscos

^^^ Presenta o número em notação científica

*Formatação alfanumérica:*

\ \ Espaço para caracteres  
! Espaço para um caractere  
& Espaçamento variável  
\_ Próximo caractere será impresso normalmente  
outro Imprime caractere

**PRINT# USING** (declaração, 1-D)

Formato: PRINT#<nº arq> USING <"forma">;<exprA>[{: | ,}<exprA>...]

Função: Escreve o valor de <exprA> no arquivo especificado, formatando. Os caracteres de formatação são os mesmos de PRINT USING.

**PSET** (declaração, 1)

Formato: PSET {(X,Y) | STEP(X,Y)} [,<cor> [,<operação lógica>]]

Função: Desenha o ponto especificado por (X,Y) na tela gráfica.

**PUT** (declaração, D)

Formato: PUT [#]<nº arq> [,<nº registro>]

Função: Grava um registro em um arquivo aleatório.

**PUT KANJI** (declaração, 1-2-K)

Formato: PUT KANJI [(X,Y)],<código JIS>[,<cor>[,<operação lógica> [,<modo>]]]

Função: Apresenta um caractere Kanji na tela.

**PUT SPRITE** (declaração, 1-2)

Formato: PUT SPRITE <plano do sprite>[,{(X,Y) | STEP(X,Y)} [,<cor> [,<nº do sprite>]]]

Função: Apresenta um sprite na tela.

**READ** (declaração, 1)

Formato: READ <nome variável>[,<nome variável>...]

Função: Lê os dados do comando DATA e os armazena nas variáveis.

**REM** (declaração, 1)

Formato: REM <comentários>

Função: Colocar comentários no programa.

**RENUM** (comando, 1)

Formato: RENUM [<novo nº linha>[,<nº linha antigo>[,<incremento>]]]

Função: Renumeras as linhas de programa.

**RESTORE** (declaração, 1)

Formato: RESTORE [<nº de linha>]

Função: Especifica o número de linha DATA inicial a ser lido por READ.

**RESUME** (declaração, 1)

Formato: RESUME { [0] | NEXT | <nº de linha> }

Função: Finaliza rotina de tratamento de erros.

**RETURN** (declaração, 1)

Formato: RETURN [<nº de linha>]

Função: Retorna de uma subrotina.

**RIGHT\$** (função, 1)

Formato: X\$ = RIGHT\$ (<expr\$>,<exprN>)

Função: Retorna em X\$ os <exprN> caracteres direitos de <expr\$>.

**RND** (função, 1)

Formato: X = RND [(<exprN>)]

Função: Retorna em X um número aleatório entre 0 e 1.

**RSET** (declaração, D)

Formato: RSET <variável string> = <expr\$>

Função: Armazena o conteúdo de <expr\$> à direita na variável string definida pela declaração FIELD.

**RUN** (comando, 1-D)

Formato: RUN [{<nº linha> | "nomearq" [,R]]

Função: Executa um programa na memória ou carrega um programa do disco e o executa.

**SAVE** (comando, 1-D)

Formato: SAVE "<nomearq>" [,A]

Função: Salva em disco ou fita o programa da memória.

**SCREEN** (declaração, 1-2-3)

Formato: SCREEN <modo tela> [,<tamanho sprite> [,<click teclas> [,<taxa cassete>[,<tipo impressora>[,<interlace>]]]]]

Função: Seleciona modo de tela e outros valores. <modo tela> pode variar de 0 a 3 para MSX1, 0 a 8 para MSX2 ou 0 a 8 e 10 a 12 para MSX2+ ou turbo R; <tamanho do sprite> pode ser 0 (8x8), 1 (8x8 ampliado), 2 (16x16) ou 3 (16x16 ampliado); <click teclas> pode ser 0 (desabilitado) ou 1 (habilitado), <taxa cassete> pode ser 0 (1200) ou 1 (2400); <tipo impressora> pode ser 0 (não MSX) ou 1 (MSX) e interlace pode ser 0 (normal), 1 (entrelaçado), 2 (não entrelaçado com alternância de 2 páginas de vídeo) ou 3 (entrelaçado com alternância de 2 páginas de vídeo).

**SET** (comando, 1-2-3-4)

Formato: SET <comando>,[<argumento>]

Função: Seta diversas configurações para o micro. Ver “SEQÜÊNCIA SET” para maiores detalhes.

**SGN** (função, 1)

Formato: X = SGN (<exprN>)

Função: Retorna o resultado do sinal de <exprN> em X.

**SIN** (função, 1)

Formato: X = SIN (<exprN>)

Função: Retorna em X o valor do seno de <exprN> (exprN deve ser expresso em radianos).

**SOUND** (declaração, 1)

Formato: SOUND <nº registrador>,<dado>

Função: Escreve no registrador do PSG o valor de <dado>.

**SPACE\$** (função, 1)

Formato: X\$ = SPACE\$ (<exprN>)

Função: Retorna em X\$ uma string com <exprN> espaços.

**SPC** (função, 1)

Formato: PRINT SPC (<exprN>)

Função: Imprime <exprN> espaços.

**SPRITE** (declaração, 1)

Formato: SPRITE {ON | OFF | STOP}

Função: Habilita/desabilita/suspende interrupção por colisão de sprites.

**SPRITE\$** (variável de sistema, 1)

Formato: X\$ = SPRITE\$ (<nº sprite>) | SPRITE\$ (<nº sprite>) = <expr\$>  
<expr\$> = chr\$(<padrão 1ª linha>)+chr\$(<padrão 2ª linha>)...

Função: Define ou lê o padrão dos sprites. <nº sprite> pode variar de 0 a 63 para sprites 16x16 e de 0 a 255 para sprites 8x8.

**SQR** (função, 1)

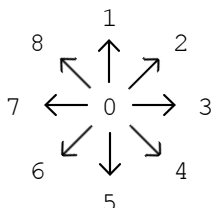
Formato: X = SQR(<exprN>)

Função: Retorna em X o valor da raiz quadrada de <exprN>.

**STICK** (função, 1)

Formato: X% = STICK (<nº porta joystick>)

Função: Examina a direção do joystick e retorna o resultado em X%. O resultado pode variar de 0 a 8, conforme a ilustração da página seguinte.

**STOP** (declaração, 1)

Formato: STOP

Função: Paralisa a execução de um programa.

Formato: STOP {ON | OFF | STOP}

Função: Habilita, desabilita ou suspende interrupção pelo pressionamento das teclas CTRL+STOP.

**STRIG** (função/declaração, 1)

Formato: X = STRIG (<nº porta joystick>)

Função: Examina a o estado dos botões de disparo e retorna o resultado em X.

Formato: STRIG (<nº porta joystick>) {ON | OFF | STOP}

Função: Habilita, desabilita ou suspende interrupção pelo pressionamento dos botões de disparo.

**STR\$** (função, 1)

Formato: X\$ = STR\$(<exprN>)

Função: Converte o valor de <exprN> em uma string decimal e retorna o valor obtido em X\$.

**STRING\$** (função, 1)

Formato: X\$ = STRING\$ (<exprN1>,{<expr\$> | <exprN2>})

Função: Retorna em X\$ uma string de comprimento <exprN1>, onde todos os caracteres são iguais, formada pelo primeiro caractere de <expr\$> ou pelo caractere cujo código ASCII está representado por <exprN2>.

**SWAP** (declaração, 1)

Formato: SWAP <nome variável>,<nome variável>

Função: Troca o conteúdo das duas variáveis.

**TAB** (função, 1)

Formato: PRINT TAB(<exprN>)

Função: Produz <exprN> espaços para as instruções PRINT.

**TAN** (função, 1)

Formato: X = TAN (<exprN>)

Função: Retorna em X o valor da tangente de <exprN> (exprN deve ser expresso em radianos).

**TIME** (variável de sistema, 1)

Formato: X = TIME | TIME = <exprN>

Função: Variável continuamente incrementada 60 vezes por segundo.

**TROFF** (comando, 1)

Formato: TROFF

Função: Desliga o rastreamento de linhas do programa em execução.

**TRON** (comando, 1)

Formato: TRON

Função: Liga o rastreamento de linhas do programa em execução.

**USR** (função, 1)

Formato: X = USR[<número>] (<argumento>)

Função: Executa uma rotina em assembly.

**VAL** (função, 1)

Formato: X = VAL (<expr\$>)

Função: Converte <expr\$> em um valor numérico e o armazena em X.

**VARPTR** (função, 1-D)

Formato: X = VARPTR (<nome variável>)

Função: Retorna em X o endereço onde a variável está armazenada.

Formato: X = VARPTR (#<nº arq>)

Função: Retorna em X o endereço do FCB do arquivo especificado.

**VDP** (variável de sistema, 1-2-3)

Formato: X = VDP(<nº registrador>) | VDP(<nº registrador>) = <dado>

Função: Lê ou escreve um dado em um registrador do VDP. <dado> deve ser um valor numérico entre 0 e 255.

**VPEEK** (função, 1-2)

Formato: X% = VPEEK (<endereço>)

Função: Retorna em X% o conteúdo do byte da VRAM especificado por <endereço>.

**VPOKE** (declaração, 1-2)

Formato: POKE <endereço>, <exprN>

Função: Escreve no <endereço> da VRAM um byte de dados. <exprN> deve resultar um valor numérico entre 0 e 255.

**WAIT** (declaração, 1)

Formato: WAIT <nº porta>, <exprN1>[, <exprN2>]

Função: Paralisa a execução do programa até que o valor da porta especificada coincida com o valor de <exprN1> ou <exprN2>.

**WIDTH** (declaração, 1-2)

Formato: WIDTH <número>

Função: Especifica a número de caracteres por linha nos modos texto.

**XOR** (operador lógico, 1)

Formato: <exprA1> XOR <exprA2>

Função: Efetua operação lógica XOR entre <exprA1> e <exprA2>.

**SEQÜÊNCIA CALL****ANK** (declaração, 1-2-K)

Formato: CALL ANK

Função: Sai do modo Kanji.

**BGM** (declaração, M)

Formato: CALL BGM(n)

Função: Seta execução de comandos enquanto a música está sendo tocada. <n> pode ser 0 ou 1, conforme abaixo:  
0 - nenhum comando pode ser executado durante a música.  
1 - comandos podem ser executados durante a música (default).

**CHDIR** (declaração, D2)

Formato: CALL CHDIR (<expr\$>)

Função: Troca subdiretório de acordo com o caminho <expr\$>.

**CHDRV** (declaração, D2)

Formato: CALL CHDRV (<expr\$>)

Função: Troca o drive de acordo com <expr\$>.

**CLS** (declaração, K)

Formato: CALL CLS

Função: Limpa a tela no modo Kanji.

**FORMAT** (comando, D)

Formato: CALL FORMAT

Função: Formata um disquete.

**KANJI** (declaração, K)

Formato: CALL KANJI [<n>]

Função: Ativa o modo Kanji. <n> pode variar de 0 a 3, mas os modos 1 a 3 só funcionam em um MSX2 ou superior.



**MDR** (declaração, 4, opcional)

Formato: CALL MDR

Função: Ativa a saída do MSX-MUSIC para a interface MIDI.

**MEMINI** (declaração, 2)

Formato: CALL MEMINI [(tamanho da RAM disk)]

Função: Ativa a RAM disk nos 32K inferiores de memória.

**MFILES** (declaração, 2)

Formato: CALL MFILES

Função: Lista os arquivos da RAM disk dos 32K inferiores de memória.

**MKDIR** (declaração, D2)

Formato: CALL MKDIR (&lt;expr\$&gt;)

Função: Cria um subdiretório com o nome especificado por &lt;expr\$&gt;.

**MKILL** (declaração, 2)

Formato: CALL MKILL (“&lt;nomearq&gt;”)

Função: Apaga o arquivo &lt;nomearq&gt; da RAM disk dos 32K inferiores de memória.

**MNAME** (declaração, 2)

Formato: CALL MNAME (“&lt;nomearq1&gt;” AS “&lt;nomearq2&gt;”)

Função: Renomeia o arquivo &lt;nomearq1&gt; com &lt;nomearq2&gt; na RAM disk dos 32K inferiores de memória.

**MUSIC** (declaração, M)

Formato: CALL MUSIC [ (&lt;n1&gt;[,0[,&lt;n3&gt;...[,n9]]]]] ) ]

Função: Inicia o MSX-MUSIC e determina quais vozes serão usadas e de que forma. &lt;n1&gt; pode ser:

0 - seleciona modo melodia puro (n3~n9 podem ser especificados)

1 - seleciona modo melodia + bateria (n3~n6 podem ser especificados).

&lt;n3&gt; até &lt;n9&gt; podem ser:

1 - seleciona melodia

2 - seleciona bateria

**PALETTE** (declaração, 3)

Formato: CALL PALETTE (&lt;nº paleta&gt;,&lt;R&gt;,&lt;G&gt;,&lt;B&gt;)

Função: Especifica as cores para a paleta.

**PCMPLOY** (declaração, 4)

Formato: CALL PCMPLOY (@&lt;endini&gt;,&lt;endfim&gt;,&lt;samp.rate&gt;[,S])

Função: Reproduz dados PCM armazenados na RAM ou VRAM. &lt;samp.rate&gt; pode ser 0 a 3 (ver PCMREC), &lt;endini&gt; e &lt;endfim&gt; são os endereços inicial e final para a reprodução. (variam de &amp;H0000 a &amp;HFFFF) e [,S] lê da VRAM.

**PCMREC** (declaração, 4)

Formato: CALL PCMREC (@<endini>,<endfim>,<samp.rate>,  
[[<nível de disparo>],[<salvamento>],S])

Função: Grava dados PCM na RAM ou VRAM. <endini> e <endfim> são os endereços inicial e final para gravação e podem variar de 0000H a FFFFH, <samp.rate> pode ser 0 (15,75 KHz), 1 (7,875 KHz), 2 (5,25 KHz) ou 3 (3,9375 KHz), <nível de disparo> de 0 a 127 e <salvamento> pode ser 0 ou 1 (1=salva na RAM, 0=não salva). [,S] grava na VRAM.

**PITCH** (declaração, M)

Formato: CALL PITCH (<n>)

Função: Ajuste fino do som. <n> pode variar de 410 a 459, sendo que o valor default é 440 (nota LÁ central).

**PLAY** (declaração, M)

Formato: CALL PLAY (<n>,<variável numérica>)

Função: Retorna na <variável numérica> o estado da voz <n> do OPLL (tocando[-1] ou não [0]). <n> pode variar de 0 a 9. Se for 0, todas as vozes são checadas; se for 1 a 9 é checada a voz respectiva.

**RAMDISK** (declaração, D2)

Formato: CALL RAMDISK (<exprN1>,[<exprN2>])

Função: Cria uma RAMDISK com tamanho máximo <exprN1> e opcionalmente retorna o tamanho efetivamente criado em <exprN2>. A RAMDISK é acessada através do drive H:.

**RMDIR** (declaração, D2)

Formato: CALL RMDIR (<expr\$>)

Função: Remove o subdiretório especificado por <expr\$>.

**STOPM** (declaração, M)

Formato: CALL STOPM

Função: Interrompe a música tocada pelo MSX-MUSIC.

**SYSTEM** (comando, D)

Formato: CALL SYSTEM

Função: Chama o MSXDOS ou MSXDOS2.

**TEMPER** (declaração, M)

Formato: CALL TEMPER (<n>)

Função: Define o modo bateria para o OPLL. <n> pode variar de 0 a 21, cujo significado está descrito na página seguinte.

0 - Pythagoraph	11 - Ritmo puro Cis+ (B-)
1 - Mintone	12 - Ritmo puro D+ (H-)
2 - Welkmeyster	13 - Ritmo puro Es+ (C-)
3 - Welkmeyster (ajustado)	14 - Ritmo puro E+ (Cis-)
4 - Welkmeyster (separado)	15 - Ritmo puro F+ (D-)
5 - Kilanbuger	16 - Ritmo puro Fis+ (Es-)
6 - Kilanbuger (ajustado)	17 - Ritmo puro G+ (E-)
7 - Velotte Young	18 - Ritmo puro Gis+ (F-)
8 - Lamour	19 - Ritmo puro A+ (Fis-)
9 - Ritmo perfeito (default)	20 - Ritmo puro B- (G-)
10 - Ritmo puro C+ (A-)	21 - Ritmo puro H- (Gis-)

**TRANSCOPE** (declaração, M)

Formato: CALL TRANSCOPE (<n>)

Função: Muda de clave. <n> pode variar de -12799 a +12799, sendo que 100 unidades correspondem a meio tom. O valor default é 0.

**VOICE** (declaração, M)

Formato: CALL VOICE ([@<n1>],[@<n2>], ..... [@<n9>])

Função: Especifica os instrumentos que serão usados em cada voz. <nx> pode variar de 0 a 63. O valor default é 0.

**VOICE COPY** (declaração, M)

Formato: CALL VOICE COPY (@<n1>,-<n2>)

Função: Copia dados referentes aos instrumentos de/para uma variável matriz tipo DIM A%(16). <n1> é a fonte e <n2> o destino. <n1> pode variar de 0 a 63 e <n2> só pode ser 63, ou <n1> e <n2> podem ser uma variável matriz.

**SEQÜÊNCIA SET****ADJUST** (declaração, 2)

Formato: SET ADJUST (<coordenada X>,<coordenada Y>)

Função: Muda a localização da tela. X e Y podem variar de -7 a 8.

**BEEP** (declaração, 2)

Formato: SET BEEP <timbre>,<volume>

Função: Seleciona o tipo e o volume do beep. Os valores válidos variam de 1 a 4.

**DATE** (declaração, 2)

Formato: SET DATE <expr\$> [,A]

Função: Altera a data do relógio. [,A] altera a data do alarme. <expr\$> deve conter uma especificação de data válida.

**PAGE** (declaração, 2)

Formato: SET PAGE <página apresentada>, <página ativa>

Função: Seleciona páginas de vídeo. <página apresentada> é a página a ser apresentada na tela e <página ativa> é a página na qual serão executados os comandos.

**PASSWORD** (declaração, 2)

Formato: SET PASSWORD <expr\$>

Função: Ativa a senha. <expr\$> deve conter uma senha de no máximo 6 caracteres.

**PROMPT** (declaração, 2)

Formato: SET PROMPT <expr\$>

Função: Ativa um novo prompt para o BASIC. <expr\$> deve conter o novo prompt com no máximo 6 caracteres.

**SCREEN** (declaração, 2)

Formato: SET SCREEN

Função: Grava na SRAM do relógio os dados da declaração SCREEN.

**TIME** (declaração, 2)

Formato: SET TIME <expr\$> [,A]

Função: Altera a hora do relógio. [,A] altera a hora do alarme. <expr\$> deve conter uma especificação de hora válida.

**TITLE** (declaração, 2)

Formato: SET TITLE <expr\$> [, <cor do título>]

Função: Define o título e a cor da tela inicial. <expr\$> deve conter o título com 6 caracteres no máximo. <cor do título> pode variar de 1 a 4

**VIDEO** (declaração, 2, opcional)

Formato: SET VIDEO [<modo>[, <Ym>[, <CB>[, <sync>[, <voz>[, <entrada de vídeo>[, <controle AV>]]]]]]]]

Função: Define superimposição e outros modos.

## CARACTERES ESPECIAIS

### ABREVIações DE INSTRUções

REM           ´  
 PRINT       ?  
 CALL        \_

### CóDIGOS DE OPERação LóGICA

PSET        TPSET<sup>1</sup>    Usa a cor especificada (default)  
 PRESET     TPRESET<sup>1</sup> Faz "NOT (cor especificada)"  
 XOR         TXOR<sup>1</sup>    Faz "(cor destino) XOR (cor especificada)"  
 OR          TOR<sup>1</sup>     Faz "(cor destino) OR (cor especificada)"  
 AND         TAND<sup>1</sup>    Faz "(cor destino) AND (cor especificada)"

### TABELAS VERDADE

		AND	OR	XOR	EQV	IMP
0	0	0	0	0	1	1
0	1	0	1	1	0	1
1	0	0	1	1	0	0
1	1	1	1	0	1	1

### NOTAções

&B    Precede uma constante na forma binária  
 &O    Precede uma constante na forma octal  
 &H    Precede uma constante na forma hexadecimal  
 %     Assinala variável como inteira  
 !     Assinala variável como precisão simples  
 #     Assinala variável como precisão dupla  
 \$     Assinala variável como alfanumérica  
 -     Operador matemático para subtração  
 +     Operador matemático para adição  
 /     Operador matemático para divisão  
 \*     Operador matemático para multiplicação  
 ^     Operador matemático para potenciação  
 =     Denota igualdade e atribui valores  
 <>    Denota diferença

---

**Nota 1:** Quando a operação lógica for precedida por "T", nenhuma operação será feita quando a cor for transparente.

## CONVENÇÕES USADAS

NOME DA INSTRUÇÃO (tipo da instrução, versão do BASIC)

Formato: Formatos válidos para a instrução.

Função: Forma de operação da instrução.

Há cinco tipos de instruções, a saber: declarações, comandos, funções, variáveis de sistema e operadores lógicos.

A versão do BASIC assinala a versão para a qual a instrução está implementada. Valores separados por “-” indicam que há diferenças de sintaxe ou comportamento para versões diferentes.

1~4 Versão do MSX-BASIC  
 M MSX-MUSIC BASIC  
 K Necessário Kanji-ROM  
 D Disk-BASIC 1.0  
 D2 Disk-BASIC 2.0

## NOTAÇÕES DE FORMATO

<exprA> variável, constante ou expressão string ou numérica.  
 <exprN> variável, constante ou expressão numérica.  
 <expr\$> variável, constante ou expressão string.  
 <n> é um número definido. Quando entre parênteses pode ser uma expressão ou variável numérica.  
 [ ] delimita parâmetro opcional.  
 | significa que apenas um dos itens pode ser utilizado.  
 { } delimita opção.  
 X variável qualquer.  
 X% variável inteira qualquer.  
 X! variável de precisão simples qualquer.  
 X# variável de precisão dupla qualquer.  
 X\$ variável alfanumérica qualquer.

Caracteres entre parênteses após múltiplos formatos para uma instrução indicam a versão do BASIC na qual aquele formato da instrução está disponível.



