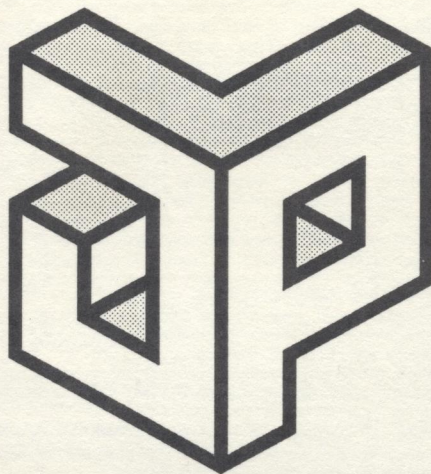


TopSpeed Modula-2



QUICK-REFERENCE

Copyright © 1987, 1988 Jensen & Partners International. All rights reserved.

Contents

Main Environment Commands	1
Compiler Directives	2
Predefined Function Procedures	3
Predefined Proper Procedures	3
Library Overview	3
The Environment	4
The Redirection File	4
Batch Compiler and Linker	4
Batch Compiler Switches	4
Batch Linker Switches	4

Main Environment Commands

Command	Invokes
Alt F	Files menu.
Alt E	Editor.
Alt C	Compiler.
Alt M	Make utility.
Alt L	Linker.
Alt R	Compiler, make utility, linker, in succession.
Alt O	Options menus for Compiler, Linker, Editor, Run, Environment Configuration, and the Make facility.
Alt I	Information about the current environment.

Compiler Directives (Defaults Highlighted)

\$A+/-	Enable/disable aliased behavior on global variables. The compiler will assume that variables declared with the directive <i>disabled</i> cannot be accessed directly <i>and</i> indirectly at the same time.
\$B+/-	Include/exclude a control-break handler in the program. The handler can be dynamically enabled and disabled with library procedures. The directive must be in the main module.
\$C h	Specifies which registers will be preserved by procedures. The hexadecimal number, <i>h</i> , specifies the registers, based on the following values: AX=1, CX=2, DX=4, BX=8, DS=10, ES=20, SI=40, DI=80. The BP register is always preserved. Default configuration is F0 = DS+ES+SI+DI, specified as: (*\$C F0*)
\$D n	Specifies the name, <i>n</i> , of the <i>data</i> segment in which to put the global variables declared by the module. The default is to use the module name. In any case, the name specified is prefixed by D. If used, this directive must appear before the MODULE keyword in <i>both</i> definition and implementation files.
\$E+/-	Enable/disable relaxed alias treatment of variant records. Because of overlapping, the compiler will normally consider fields of variant records volatile, causing them to be kept in memory.
\$F	Procedures will be called with FAR calls. Likewise, procedure types are 32-bit. This is the default, and applies only to global procedures; local procedures are always called with NEAR calls.
\$G+/-	Enable/disable module prefixes in external names.
\$H+/-	Enable/disable treating constant aggregates as variables, thereby allowing them to be modified.
\$I+/-	Enable/disable index checking. If enabled, accessing a non-existent array element will produce a run-time error. All variables are assumed to hold legal values corresponding to their type (see \$R below). Likewise, type transfer on indices is assumed not to cause problems.
\$J+/-	Enable/disable interrupt procedures, by generating IRET returns instead of the usual RET instruction.
\$K+/-	Enable/disable the C language calling convention for procedures. This specifies that <i>caller</i> (not called) pops parameters. Across calls, the DS register is set to the group named 'DGROUP.'
\$M n	Specifies the name, <i>n</i> , of the <i>code</i> segment in which to put code. The default is to use the module name. In any case, the name is prefixed by C. When used, this directive must appear before the MODULE keyword in <i>both</i> definition and implementation files.
\$N	When used, procedures will be called with NEAR calls. This requires callers to be in the same code segment. Likewise, procedure types are 16-bit.
\$O+/-	Enable/disable overflow checking on <i>whole</i> number operations. If this directive is enabled, a numeric overflow will cause a run-time error.
\$P+/-	Enable/disable generating external names for local procedures. Enabling eases debugging but can cause name clashes.
\$Q+/-	Enable/disable procedure tracing. If enabled, procedures will execute an INT 60H instruction on entry and INT 61H on exit. Handlers for these interrupts must be installed explicitly — e.g., using library module ProcTrace.
\$R+/-	Enable/disable subrange checking. If enabled, a run-time error is generated by assignment or parameter passing if the value is outside the bounds of the receiver's type.
\$S h	The hex number, <i>h</i> , specifies amount of stack allocated for program. Directive must be in main module.
\$S+/-	Enable/disable stack overflow checking. If this directive is enabled, a run-time error is generated if stack space is exhausted.
\$V+/-	Enable/disable copying of open array value parameters. Disabling such copying increases efficiency but is potentially incorrect.
\$W+/-	Enable/disable use of volatile variables. Volatile variables are <i>not</i> kept in registers across statements. The ability to use volatile variables can be essential if concurrent processes communicate via shared global variables, and can make debugging easier. The /V command line option selects this globally.
\$X+/-	Enable/disable 8087 stack spilling for procedures. Spilling is necessary if nested function calls exhaust the 8087 stack. Spilling involves saving excess values from floating point computations on the hardware stack — because there is no longer room on the 8087 stack.
\$Y+/-	Enable/disable coinciding variant fields in a record. If enabled, it's legal to use the same name for fields in distinct alternatives, provided these fields have the same type and are at the same offset in the record.
\$Z+/-	Enable/disable checks for dereferencing of NIL pointers, which then generate a run-time error. When enabled, all local variables are initialized to zero. If enabled in main module, global variables are also zeroed.

Predefined Function Procedures

ABS (X)	Absolute value of numeric operands.
ADR (X)	The physical ADDRESS of object X.
CAP (C)	Character C, changing 'a'..'z' to 'A'..'Z'.
CHR (X)	The CHAR with ordinal value X.
FLOAT (C)	REAL value of CARDINAL C.
HIGH (A)	The upper index bound of open array A.
MAX (T)	Maximum value of ordinal/real type T.
MIN (T)	Minimum value of ordinal/real type T.
ODD (X)	TRUE if ordinal value X is not even.
ORD (X)	CARDINAL, short ordinal value of X.
SIZE (T)	Size in bytes of type or object T.
TRUNC (R)	CARDINAL, truncated value of real R.
VAL (T, X)	The value X converted to type T.
VSIZ (R, F)	Size of record type R if it contained just the fields up to and including field F.

Predefined Proper Procedures

DEC (X)	Decrement ordinal object X.
DISPOSE (X)	When the compiler encounters the DISPOSE procedure, This is replaced by a call to DEALLOCATE (X, SIZE (X^)).
DEC (X, N)	Decrement ordinal object X by amount N.
EXCL (S, E)	Exclude element E from set object S.
HALT	Terminate program execution successfully.
INC (X)	Increment ordinal object X.
INC (X, N)	Increment ordinal object X with amount N.
INCL (S, E)	Include element E in set object S.
NEW (X)	This is replaced by a call to ALLOCATE (X, SIZE (X^)) when the compiler encounters the NEW procedure.

Library Overview

SYSTEM	Support for the specifics of the 8086 family of processors and for concurrent processes.
AsmLib	An auxiliary module which you need not use directly.
MATHLIB	Logarithmic, trigonometric and hyperbolic functions on reals. Conversion of reals to/from binary coded decimals. 8087 specifics.
Str	String concatenation, appending, insertion, deletion, comparison, searching, matching. String conversion to and from numeric types.
Lib	Sorting, random number generation. DOS-interrupt calls, command line access. Memory block operations, long jumps, address arithmetic, break and error handling, and sound procedures.
Storage	Allocation and deallocation of storage; information about storage availability.
Process	Starting up time-sliced processes; synchronization using semaphores.
Graph	Selecting a graphics board, selecting screen modes. Writing and reading of single pixels; drawing of lines and circles; drawing of filled circles and polygons.
FIO	Access to MS-DOS files. Creating, deleting, renaming, opening, closing files. Reading, writing, seeking on files. Creating, deleting, changing directory; directory scanning. Reading and writing of characters, booleans, integers, cardinals, reals, strings.
IO	Formatted input/output to the standard I/O devices. Reading and writing of characters, booleans, integers, cardinals, reals, strings. Redirection to any device. Reading characters directly from keyboard. Testing for key-ready.
Window	Creating, disposing, opening, closing windows. Setting frame, color, title on windows. Rearranging the size, position, layering of windows. Cursor control inside windows. Inserting and deleting lines in windows. Palette windows. Writing to windows using the IO module.
FloatExc	Enabling/disabling of 8087 exceptions.
ProcTrace	Monitoring procedure calls. Determining the current procedure. Monitoring the value of variables

The Environment

M2	"Hot start" system, from where last session ended.
M2/N	"Cold start" system, without reading M2.SES file.
F1	Provides on-line, context sensitive help.
Esc	Closes the current window or menu.
ScrollLock	Move, resize, and recolor windows.

The Redirection File

The contents of the currently active redirection file (M2.RED) specify where the system will write files and from where it will read files. For example, suppose the redirection file contains:

```
*.DEF = .;\M2\MYLIB; \M2\LIB
*.OBJ = \M2\OBJ; \M2\LIB
```

Then, whenever the system needs a .DEF file it will search first the current directory (.); then, if the file is not found, the system will search \M2\MYLIB and, finally, \M2\LIB.

Similarly, whenever the system needs to create a new .DEF file, it will be created in the first directory that matches the definition for .DEF files. In the example, this is the current directory.

Likewise, the system will search for .OBJ files in \M2\OBJ and \M2\LIB, in that order. The system will write new .OBJ files in the directory \M2\OBJ.

Redirection is applied only to files without a full path name. For example, the specification

```
\M2\NEW\TEST.MOD
```

would not cause redirection.

Batch Compiler and Linker

The TopSpeed Modula-2 Compiler and Linker may be invoked from the DOS command line instead of the environment.

The command format to run the batch compiler is:

```
M2/C filename /options
```

The command format to run the batch linker is:

```
M2/L filename /options/option/option
```

Batch Compiler Switches

/B	Turns run-time (\$I, \$O, \$R, \$S, \$Z) checking ON by default.
/D	Tells compiler to generate a debug information file for module being compiled.
/F	Allows file names to differ from module names.
/H	Used together with Make (see /M) to show which files need recompilation.
/J	Suppresses smart linking — that is, the automatic generation of libraries. Non-library OBJ-files are created. Although this results in larger files, this file format is necessary for some linkers.
/L	Generates screen log of information about compilation speed for definition and for implementation modules.
/M	Invokes the Make utility. The file specified to the compiler must contain a main module. Make will, based on the imports specified in that file, find all modules required by the program. For each of these, Make checks whether the corresponding OBJ-file is up-to-date, based on the time/date values of the files used. Modules that need it are recompiled. Only files whose module implementation is accessible are considered.
/N	Include line numbers in the OBJ-file. This enables a program such as a debugger to determine the correspondence between code addresses and source program lines.
/O	Disables reorganization of expressions. This ensures that expressions are evaluated left-to-right, but results in suboptimal code.
/P	Tells compiler to display the name of each procedure as it is compiled.
/R	Used together with Make to recompile everything, regardless of checks. This option may be necessary because the Make facility only considers the actual time/dates values of files; it does not read OBJ-files to find the version-records.
/V	Makes all variables volatile — kept in memory, instead of trying to keep them in registers. This can help when debugging.

Batch Linker Switches

/M	Generate map file
/I	Initialize all segments to zero
/S	Detailed map of segments
/N	No default libraries
/T	Trace first reference to module
/W	Suppress warning messages
/C	Lower case significant in symbols