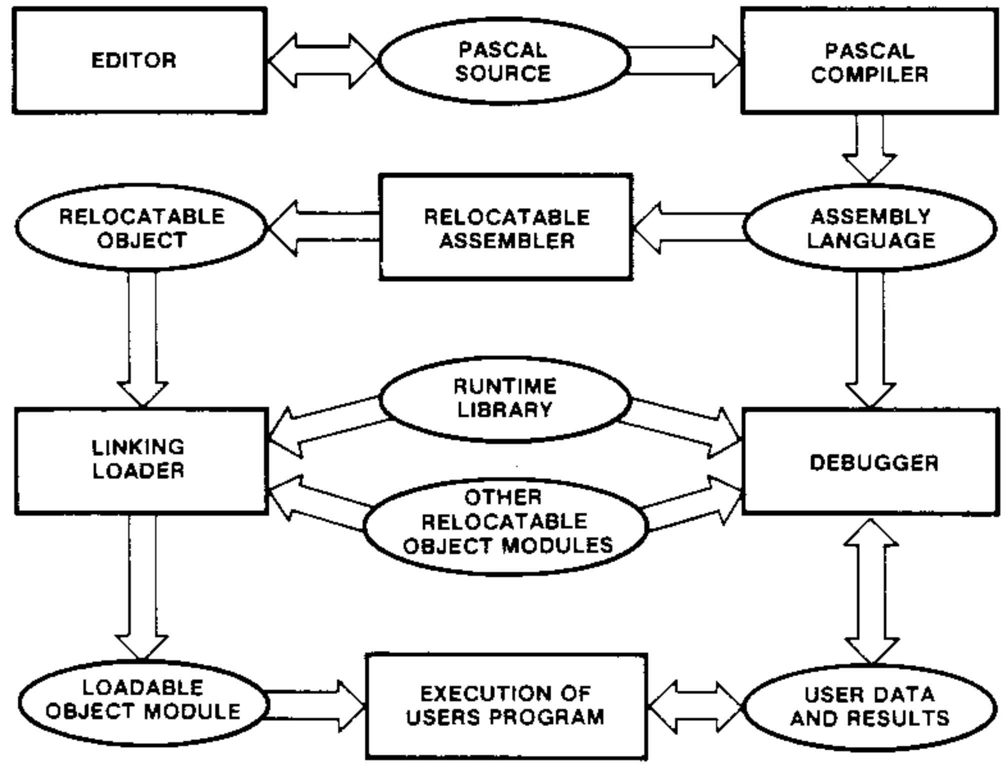
OmegaSoft Version 2 Pascal Configuration Manual For FLEX



THE OMEGASOFT 'PASCAL MACHINE'

SERIAL 10078

OMEGASOFT VERSION 2

PASCAL CONFIGURATION MANUAL

FOR FLEX

OMEGASOFT

P.O. BOX 70265 SUNNYVALE CA 94086

Second printing, March 1982

The information in this document has been carefully checked and is believed to be reliable, however, no responsibility is assumed for inaccuracies. OmegaSoft reserves the right to change specifications without notice.

Printed in the United States of America

FLEX is a trademark of TSC

CONTENTS

INSTALLATION

The host system must run the FLEX operating system, have at least two disk drives, and have at least 40k of contiguous memory starting at address $0000. More memory will increase the compiler symbol table space and increase the area available in the debugger for the user program.

If the system has normal or high capacity disks (not 5" single density single sided) it is preferable to put all of the required files on the system disk. These would include your editor, the compiler (PC), the debugger (DB), the debugger overlay (DBOV), the linkage creator (LC), the runtime library (RL), the relocatable assembler (RA), and the linking loader (LL). The relocatable assembler and linking loader are available from OmegaSoft.

If the system has low capacity disks (5" single density single sided) then all of those files won't fit. In this case we suggest you put the editor, the compiler, and the debugger on one system disk to develop and debug your program. After you've gotten them bugs out then you can build the loadable object module, this will require a different system disk with the linkage creator, the runtime library, and the assembler and linker.

COMPILER

Command line format

PC { <delim> <modifier> } where <delim> are one or more space or comma characters and valid modifiers are :

<path	- redirect pascal source file
>path	- redirect assembly output file
>>path	- redirect listing file
O	- enable output file
L	- enable listing
P=nn	- set page size
W=nn	- set page width
D	- enable debugger comments
S	- enable source comments
H	- halt listing on error
F	- use form feed for top of form
T	- put time in heading
'text'	- use text as title

The following path names are valid :

TERM	- system terminal (CRT)
P	- system printer
others	- disk file

The following default suffixes are used :

Source file	- .TXT (text)
Output file	- .CO (compiler output)
Listing file	- .CL (compiler listing)

Defaults are :

path	- TERM
O	- off
L	- off
P	- 0 (no pagination)
W	- system WD parameter if > 10, else 80
H	- off
F	- off (uses line feeds)
T	- off
'text'	- null string

If page size is zero (no pagination) then a signon message will be displayed after the program is loaded otherwise this is silenced to maintain proper pagination. The D option inserts special comments into the output code as instructions to the Pascal debugger. A CMPX #$XXXX instruction is also generated at the start of each executable statement and is used to encode the line number information and for breakpoints and traces. The S option inserts the pascal source lines as comments into the output code. This is useful for final documentation and as an aid in finding those really tough problems. The H option will stop the compiler after a pascal line in error is displayed. A carriage return will continue the compilation, a Q followed by a carriage return will terminate the compilation. If the TTYSET DP parameter is non zero and PS=Y then the end of screen pause will function when generating a listing to the screen. If pagination is enabled line feeds are normally used to advance the paper to the top of form. On some printers this is quite slow and the F option is provided to send a form feed to the printer instead. The T option will insert the operating system date string into the heading. Any characters between single quotes in the command line will be inserted into the page heading if pagination is enabled.

At any point in the compilation (with or without a listing) the operating system screen pause and abort keys will function. This version of the compiler provides two options for screen pause and abort. At location $010A in the compiler is a three byte key table. If the first byte is zero then the normal FLEX method is used : The escape character is used to pause and continue from a pause and the carriage return is used to abort if entered during a pause. If the first byte is non-zero then this character value is used as the abort key. Note that you need not be paused to abort. The second byte is used as the pause character and the third byte is used as the continue character. This feature wass added since the FLEX method was found to be inconvenient to use. We suggest setting the abort key code to ASCII ESC, the pause to control W ($17) and the continue to W ($57) as this consistant with a number of other operating systems. The compiler is shipped in the FLEX mode.

Current restrictions

Longintegers are not supported in current releases. These will be added later as a new revision. The standard identifier Longinteger and any of the other identifiers that are associated with longintegers will not be recognized by the compiler.

Structures may not include devices as part of the structure. This includes arrays of devices or records containing devices. This will be changed when longintegers are added. If this type of structure is desired it is possible to define the array or record as containing pointers to a device and using the Addr function to set the pointer.

RUNTIME

Input device

The supplied device driver for Input (DEV00) is located in file SD1. In addition to the 6 byte device descriptor and the 1 byte element buffer there is a 2 byte pointer and a 118 byte line buffer. Keyboard entries are buffered until a carriage return is entered before being processed. If a mistake is made in typing the system defined key for backspace can be used to back up on the screen and in the buffer. If you desire to delete the whole line the system defined key for line delete can be used. Control Z is used as an eof flag and is normally used as the last character before the carriage return to set the Eof function true. Each time a key is entered the line count in the operating system is cleared, this corresponds to the normal action required to facilitate end of screen pause.

The break function will wait if the pause character has been entered. If indeed it was then it will stay waiting till either the continue character is entered or if the abort character is entered. If the continue character is entered then the result of the break function will be false. If the abort character is entered then the result of the break function will be true. In addition if the abort character was entered during a screen pause (see Output device) then calling break will return true without checking further. Note that this device driver is configured identical to the compiler in regard to abort, pause, and continue keys. See the compiler section for exact details on the action of these keys.

Output device

The supplied device driver for Output (DEV01) is located in file SD1. In addition to the 6 byte device descriptor and the 1 byte element buffer there is a 1 byte column position flag and a 1 byte space for possible use as a line counter (for line feeds as opposed to using a form feed character). The column position flag is zero if no characters have been output since the last carriage return or one otherwise. This flag is checked when doing a page operation to make sure the carriage is at column zero before the page operation is performed.

When a writeln is performed on the Output device it will also increment the operating system line counter, allowing screen pause to function. If this counter reaches the TTYSET DP parameter (if non zero) and if PS=Y then Output will wait for the continue character or the abort character to be entered. In either case waiting will cease but in addition if the abort character was entered then the next call to the break function will return true.

Auxout device

The supplied device driver for Auxout (DEV02) is located in file SD4. It is configured to drive the system printer in this implementation. In addition to the 6 byte device descriptor and the 1 byte element buffer there is a 1 byte column position flag and a 1 byte space for possible use as a line counter (for line feeds as opposed to using a form feed character). The column position flag is zero if no characters have been output since the last carriage return or one otherwise. This flag is checked when doing a page operation to make sure the carriage is at column zero before the page operation is performed. Note that this device driver assumes that the printer drivers are located at their correct addresses. To be sure of this the OmegaSoft supplied program PRINTSET should be included in your startup file.

Keyboard device

The supplied device driver for Keyboard (DEV03) is located in file SD1. Keyboard entries are not buffered in any way and no tests are done to check for end of file. The keyboard entry will be read, its parity bit stripped off, and the operating system line counter will be cleared (for pause).

File device

The supplied device driver for Files (DEV04) is located in file SD2. In addition to the 6 byte device descriptor and the element buffer there is a 320 byte buffer to hold the operating systems File Control Block (FCB) and the sector buffer. Random access files are not supported in the current release. This facility will be added later in new revisions or possibly as an application note. There are no restrictions in the compiler itself, the lack of random access capability is in the device driver.

There are only two of the possible 6 modes available for opening a file in the current release. The first is Open for Input (Reset) which will open an existing file and allow it to be read. If a seek is done to record zero then the open input file is rewound. A seek to any other record will result in an error.

The second mode is Create for Output (Rewrite) which will create a new file, set it to zero length, and allow write operations. If the file specified already exists then it will be deleted (automatic overwrite). No seek operations are permitted in this mode.

If the file is of type text then space compression will be used and the default suffix will be ".TXT". If it is not of type text then it will be a non-compressed binary file and the default suffix will be ".DAT".

The last parameter of an Open or Create procedure is not currently used.

Error printing device

The supplied device driver for errors (DEV05) is located in file SD3. Error number 255 results in return to the warmstart of the operating system. Errors less than 100 are displayed as : "RUN TIME ERROR #X" and then a jump to warmstart. Non-pascal errors (disk errors) are stored in the error byte of the device descriptor as 100 plus the system error number. Therefore when the an error number over 100 is sent to the error device driver, 100 is subtracted and then that is displayed as a system error using the normal operating system routine before returning to warmstart. Note that if you are going to check for I/O errors using the Deverr function that all disk errors with the exception of end of file (converted to pascal error number 18) will be stored in this manner. For instance to check for the FLEX error for file not found the FMS error code is 4, but you would need to compare the result of Deverr to 104 to check for that error.

The supplied driver is of course unsuitable for most target systems and will need to be modified to fit your requirements.

DEBUGGER

The command line format for the debugger is :

DB <filename> { <delim> <option> } where delim is one or more space or comma characters and valid options are :

```
   D=<hex>   - set debugger table size in bytes
   H=<hex>   - set heap start address
```

If the D option is not used then 10000 bytes are allocated to the debugger table. If the H option is not used then the heap will start past the end of the user program. The default suffix for the filename is ".CO".

Memory usage

The debugger starts loading the kernal at location $100. The operating system defined value for end of memory is used and all memory available is used.

Custom versions

Full source code and "chain" files are provided so that custom versions can be created by the user. After linking the overlay you need to determine the amount of the linked code actually required. Look at the load map and find the address for the start of the module DBOV - this is the start of the area required. In the load map should also be the address for the end of PSCT - this is the end of the area required. Do a get of DBOV.CM and then do a save into DBOV.BIN on the system disk with the start and end address noted earlier, no transfer address is required. The most common change to the debugger would be different I/O in the target system or a change to the runtime library.

LINKAGE CREATOR

Command line format

LC { <delim> <modifier> } where <delim> is one or more space or comma characters and valid modifiers are :

>path	- redirect summary file
L	- enable summary
P=nn	- set summary page size
D	- delete files

The following path names are valid :

TERM	- system terminal (CRT)
P	- system printer
others	- disk file (default suffix = ".TXT")

Defaults are :

path	- TERM
L	- off
P	- 0 (no pagination)
D	- off

Suffixes

Regardless of the command line option used, the suffixes used will be :

Output of compiler	- CO
Output of compiler after assembly	- CA
Setup source	- PS
Setup after assembly	- PA
Other reloctable files	- RO
Loadable object module	- BIN
"Chain" file	- CF

Auto setup

If the Auto setup option is used then you will be prompted for the amount of memory to be used for the system stack - in most cases 128 bytes is sufficient (80 hex). This will start at the top of memory as defined by the operating system variable. Below the system stack will be the data stack which will grow towards the heap. The heap will start right past the end of the program and grow towards the data stack.

The operating system command line pointer address will be stored in the global stack frame to allow access by the Cline function. Whether or not the Auto setup option is used execution will go to the operation system warmstart after completion of the pascal program.

Load map

After you are asked for additional library files you will be asked "Load options" and the entered string will be added to the linker command line.

Running the "Chain"

Use the OmegaSoft supplied CHAIN command with the filename as its parameter.

Custom versions

Full source code and "chain" files are provided so that custom versions can be created by the user. The most common change to the linkage creator would be to support a different assembler and linker.

RE-ENTER

There is a utility command called RE (re-enter) that will simply jump to location $100. This is useful to run an OmegaSoft program already in memory without encountering the very slow load times of the FLEX operating system. This is not useful for re-running the debugger as it is not re-entrant (because of the overlay).

CHAIN

There is a utility command called CHAIN that is used when building a loadable object module using the information from the linkage creator. This command is similar to the FLEX EXEC command with the following exceptions :

a) you can see what is going on.

b) the "chain file" is also read for single character requests in addition to he command line processing. This allows the data in the chain file to be used with the linking loader.

c) the default suffix for the chain file is ".CF".