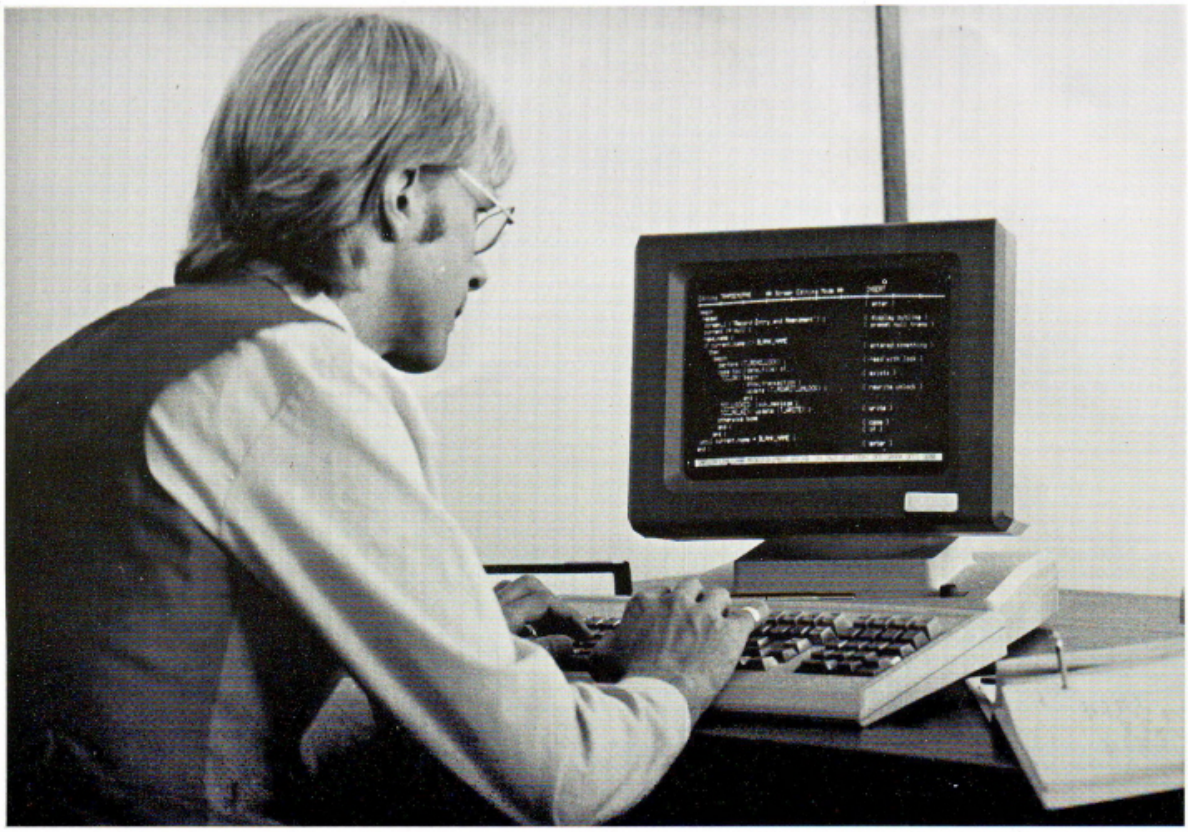
Swifte Pascal

ISO Pascal For MC68000 Microcomputers



Swifte Pascal — Th

Swifte Pascal

Swifte Pascal is a comprehensive implementation of the programming language Pascal for MC68000-based microcomputers running under the powerful multi-user operating system MIRAGE.

Why Pascal? Innovations in computer hardware are giving improved performance over older systems in terms of speed and cost. Similar benefits accrue by using modern programming languages, of which Pascal is the most outstanding example.

The drawbacks of the older languages (e.g. Fortran, Cobol) were manifold; they were verbose, clumsy, difficult to maintain and suffered from limitations, inconsistencies and irregularities. In the same way that Homo Sapiens evolved from Neanderthal Man, Pascal has surpassed older languages because it has learnt from its ancestors' mistakes making it compact, clean, easy to maintain and therefore more suited to the demands of today's environment.

When discussing languages the phrase 'horses for courses' crops up frequently. It seems that certain languages are intended for scientific use, others are for business use, whilst still others are systems programming langauges. Pascal, however, is a multi-role language, so now you can use a single language for all your programming tasks.

Portability of programs is becoming more and more important to software producers and consumers alike because it allows a software product to run, with few or no alterations, on computers made by various manufacturers. Portability means software developers can gain access to a much wider marketplace; similarly, software buyers have a wider selection of products to choose from.

Portability of programs is achieved by writing such programs in a language which adheres to an internationally agreed standard. Swifte Pascal adheres closely to the ISO/BSI Level 0 and ANSI/IEEE Standards on Pascal to permit interchange. A facility exists to compile programs strictly to the ISO/BSI Standard, rejecting all extensions. In addition, because Swifte Pascal implements IEEE-standard Floating Print Arithmetic, data can be interchanged between different systems, and different language processors.

Large software projects require a language processor built to cope with the complexity of these undertakings. Development and maintenance of programs, whether large or small, is eased by the self-documenting nature of Pascal; creation and updating of programs is performed using EDIT (MIRAGE's standard full-screen text editor). Using Swifte Pascal, a programmer or team can compile segments of programs separately which can be linked together at a later time. Of course, Pascal programs can be linked with other high-level languages or MC68000 assembler routines. Embedded compiler directives enable inclusion of library source files, page ejects on the listing file, run-time range checking and debugging information.

Using Swifte Pascal, time wasted debugging is dramatically reduced due to comprehensive compile-time and run-time checking. When a run-time error occurs, a back-trace by procedure name and statement number is produced. A comprehensive system of error (exception) handling is provided, enabling the selective trapping of error conditions. Use of the exception handling facilities allow 'firewalls' to be built around modules to localise error recovery. Exceptions can also be explicitly generated by the user's program (via the RAISE statement) to facilitate the testing of exception handling code and to quit deeply-nested procedures in a clean manner. Exception handlers may be nested; if a handler cannot deal with the exception condition, it can percolate up to an enclosing handler.

Swifte Pascal is a true compiler; its output is assembly language which is assembled to give MC68000 machine code. The code generated is fast, re-entrant, re-locatable and re-usable. Object programs interface to a separate, shareable run-time support package in order to conserve space on disc and in memory when more than one user is running Pascal programs; since the compiler was written in Pascal, it too is shareable.

Direct access to the host operating system is desirable if full use is to be made of its advanced features such as networking, keyed-access files, print spooling, time and date, screen management; naturally, Swifte Pascal provides a full set of such facilities.

ıe Natural Selection

Standard Facilities

- Integer type: – 2147483647 to + 2147483647 (4 byte); subranges in 0..255 are stored in 1 byte.
- Real type: 16 digit precision, IEEE format, 1E-308 to 1E308 (8 byte).
- Char type — 128/256 ISO character set.
- Boolean type — (False, True).
- Enumerated types — up to 256 members.
- Arrays, Records (including variants) and Files are fully implemented.
- Sets — members may range in 0..255 (32 bytes).
- All expression forms are fully implemented.
- All standard statement forms are fully implemented, including GOTO's across blocks.
- Recursive procedures and functions fully implemented.
- All parameter-passing modes available, including procedural and functional parameters.
- Programs may be up to 16 Mbytes in size.

Extended Facilities

- Full access to advanced features of MIRAGE — i.e. Networking Filing System, Spooler, Time and Date, Screen Management.
- Separate compilation — Sections of a program can be compiled independently and linked together later using the standard MIRAGE linker.
- Interface to Assembly Language — Special purpose or ultra-fast routines written in MC68000 Macro Assembler can be easily interfaced.
- Dynamic string handling — to overcome some of the restrictions of standard Pascal fixed-length strings, a predeclared dynamic string type and its associated utility functions is defined.
- Otherwise/Else clause on CASE statement — The specification of all possible values of the case index is not required.
- Exception Handling — User-controllable error trapping permits 'bullet-proof' programming and simplifies programming of exceptional conditions.
- Direct access to the TRAP Transaction Processor, the multi-user keyed-access record manager with full locking control.
- Random (Contiguous) File access — Useful for programming where the direct access (by record number) method is required.
- Two-pass Compiler — Forward declarations of procedures and functions are unnecessary, simplifying program layout.
- Hexadecimal constants permitted, e.g. $0FF.
- Underscores permitted in identifiers, e.g. on_line.

Built-in Procedures and Functions

File Predicates — eof, eoln, exists, ioerror

File Manipulation — assign, reset, rewrite, close, erase, opena, openr, create, seek

File Input/Output — get, put, read, readln, write, writeln, page

TRAP Access — build, key, create, tcc, opent, close, locker, section, trap, readhd

Arithmetic — sin, cos, exp, ln, arctan, tan, arcsin, arccos, log10, alog10, sqrt, abs, sqr, trunc, round, odd

Ordinal — ord, chr, pred, succ

Heap Management — new, dispose, memavail, mark, release

Storage Access — addr. sizeof, moveleft, moveright

Strings — pack, unpack, length, concat, copy, insert, delete, pos, str, lcs, ucs, movps, movsp

Miscellaneous — exit, halt, chain, spool, crt, time, date, getcc, getchar, sleep, getcmdl

Documentation

Swifte Pascal comes complete with a comprehensive Language Reference Manual and Installation Guide. The documentation may be purchased separately from your supplier.

Hardware Requirements

The Swifte Pascal Compiler requires an MC68000-based microcomputer running the MIRAGE Operating System (V1.2 or later) with at least 150K user memory and one disc drive. Generated programs require at least 25K user memory.

Notes:

ISO — International Standards Organisation
BSI — British Standards Institute
ANSI — American National Standards Institute
IEEE — Institute of Electrical and Electronic Engineers

Swifte Pascal and the MIRAGE Operating System were written in the U.K. by Swifte Computer Systems Ltd.

Swifte Computer Systems Ltd.

Southampton House, 192/206 York Road, London SW11. Tel. 01-223 0346