PL/I-80 two-pass compiler from Digital Research

By Frank J. Derfler, Jr. and William H. Greene

There are several powerful languages now available for microcomputer systems, but most are limited versions of their mainframe "cousins." The PL/I-80 compiler from Digital Research is an excellent example of a powerful mainframe language successfully adapted to microcomputer use.

PL/I was originally developed in 1964 as a joint project of IBM and its two user groups: SHARE and GUIDE. It has been frequently called "IBM's answer to COBOL and ALGOL."

PL/I is a powerful language for both business and scientific uses, and this version is no exception.

Features

PL/I-80 is a two-pass compiler with a third-pass linker designed to operate on 8080, 8085 and Z80 microprocessors using the CP/M operating system. (A compiler is a software system that generates the object code that can be run directly by a computer's operating system. Once a program has been compiled and the subroutines linked together, the higher-order language is not needed to actually run the program. A compiled program takes up less space than an interpreted program and runs more quickly.)

The compatibility of the PL/I-80 compiler with CP/M is assured by the fact that Digital Research produced both packages. A 32K system is required as a minimum, but 48K is really needed to do serious software development. Obviously, the smaller the system, the smaller the subroutines allowed in building the program.

PL/I-80 is based on the ANSI (American National Standards Institute) General Purpose (G) subset specified by the PL/I standardization committee. This subset does not include the statements: DEFINED, FLOAT DECIMAL, LIKE PICTURE, FILE (except for OPEN statements); or the built-in functions ATANH, DATE, STRING, TIME and VALID.

The subset adds: %REPLACE, ASCII and RANK; READ and WRITE forms for variable-length ASCII records; control characters in string constants; and GET EDIT extended to full record length.

Anyone with PL/I programming experience knows that this structure-oriented language has all the right programming features: DO WHILE-END, IF-THEN-ELSE, BEGIN-END and so on. Embedded IFs with an ELSE at each level are allowed. Embedded DOs are constrained only by a 512-byte stack (which can be increased to 2048 bytes with a simple option). While some programs may require an increase in the compiler's internal stack, we have embedded loops six deep and have yet to need an increased stack size.

PL/I-80 has an answer for the programmer writing a program normally thought of as too big to run on a microcomputer. It has overlay features that allow module references to be embedded up to five deep. If you can allow the time for frequent disk references, there are no limits to the size of the program you can develop.

The linker gives you great flexibility in defining which disk drive will hold relative files and where to place both the object and overlay modules. You can generate symbol tables and direct them to the disk, the console or the printer. The symbol tables are useful for cross-referencing variable names in programs that use many subroutines and overlays.

The linker can combine as many module names as can be keyed into 128 bytes. The only file name you must specify is the primary file name. The compiler provides software switches to control the source and destination devices during the link process. You can specify as many as 19 logical devices for up to five different types of files. The Link-80 portion of the PL/I-80 package exceeds the needs of most users.

Performance

Simply stated, the PL/I package worked as advertised from the first minute we inserted the diskette into the drive.

We developed and tested software packages using PL/I-80 that were easily converted for use on larger machines. This compiler proved to be sufficiently compatible with standard PL/I compilers for large machines; thus, few or no changes were needed to run the compiled program on mainframe systems.

This feature can be extremely useful to large-machine programmers who can use a microcomputer (perhaps with a communications capability) to do software development while leaving the big system free for production work. The program could pay for itself in just a few days of programming when used in this way.

InfoWorld
Software Report Card

PL/I-80 compiler

	Poor	Fair	Good	Excellent
Performance	☐	☐	☐	☑
Documentation	☐	☐	☐	☑
Ease of Use	☐	☐	☐	☑
Error Handling	☐	☐	☐	☑

System Requirements

- 8080-, Z80- or 8085-based computer
- CP/M
- 32K RAM
- One disk drive

Price: ***$500***

Digital Research
P.O. Box 579
Pacific Grove, CA 93950

Ease of use

PL/I-80 is about as easy to use as a compiler can be, even though it is sophisticated. While the package does not include a built-in editor—it is a pure compiler and linker—it does compile programs built with the CP/M editor. Standard CP/M files created by other editors can, of course, be used too.

The process of compiling is easy to direct. Default options are included for most common conditions. The options consist of the ability to direct object and print modules, variable tables and other constructs to various output devices.

You can use other features to increase the embedded layers within a program. With one valuable feature, you can expand each PL/I construct into assembly language for modification and debugging. Users can also bypass the print- or object-generation functions during a compile pass to save time.

The linker has options also. The linking process requires a great deal of memory for building tables and resolving variable references. If the program is too large, you may place the tables on a diskette. This slows the linking process, but it greatly increases the size of the modules you can link.

Options in both the compiler and linker are easy to specify. Everything is contained on one command line with the options in brackets, but without tricky punctuation.

Error handling

The PL/I-80 package is not for the raw novice. Anyone who has written even a couple of programs in a higher-order language should have no trouble learning to use PL/I-80, however. Experienced PL/I programmers will love it.

Error handling is performed in the standard form defined by PL/I subset G. You can use either of two methods to engage the error-intercept features. ON ERROR (procedure call) will cause a user-defined error procedure to be used when an error is detected. ON ERROR (error code) will return a specific code when the error is found. Examples include OVERFLOW, FIXEDOVERFLOW, ZERODIVIDE, ENDFILE and UNDEFINEDFILE.

Of the 256 possible codes allowed, 50% are reserved for the definition of the programmer. In general, using the error features in PL/I-80 is simple.

Documentation

Digital Research provides a language manual, an applications manual and an operators' guide with PL/I-80. The language manual describes the syntax of the language in sufficient detail for most users to grasp easily. It includes a quick reference guide to the instructions and functions. A pocket-sized version of the command summary is provided that includes the compile and link options and error codes.

The applications manual is designed to pass on advice about the efficient use of PL/I-80. It has programming examples of I/O conventions, exception processing, string and list processing and the use of recursion. Since the more common languages do not allow recursion, this guide is a valuable aid in getting the most power out of the language.

The operators' guide contains a detailed explanation of the compile and link procedures. It also includes assembly-language listings of the various library routines PL/I-80 provides. Programmers can use these routines in their original form or modify them for special purposes.

Digital Research took more than a little criticism for the documentation of CP/M. The company apparently learned a lesson from that experience and has provided PL/I-80 with excellent guides to the theory, operation and application of the language.

Summary

The number of CP/M-capable machines is growing rapidly and Digital Research should find a strong market for the powerful PL/I-80 compiler. It is a fine tool for anyone wishing to develop or market fast-running compiled programs. ■