gmFortran: FORTRAN to C Translator

What is Great Migrations Fortran

The Great Migrations FORTRAN to C Translator (gmFortran) is a tool that translates FORTRAN source code to correct (compilable) and clean (readable and maintainable) C source code. The C code produced can then be compiled with almost any C compiler on almost any platform to produce executable code, which in turn will produce results on the target C platform that are identical to the results produced on the source FORTRAN platform. What’s more, the C code is clean and readable so that, if you wish, you can start maintaining your application in C rather than in FORTRAN. And, unlike the original FORTRAN, the C code is portable so you can move it to multiple “open” platforms. In addition, the tool is flexible and configurable to allow extensive user control over both the input FORTRAN dialect accepted and the look of the C output generated.

This tool is intended for those who wish to convert their FORTRAN codes to C automatically; 100 percent conversion rates are achieved routinely. The conversions yield reproducible results and are repeatable and verifiable. A number of input FORTRAN dialects are supported via a dialect definition flag, and the form of the output C code is controlled by a multitude of command-line switches.

The translations are so complete and correct that the tool may also be used in conjunction with a standard C Compiler as a general-purpose, multi-dialect and portable FORTRAN compiler. It runs on multiple platforms and supports both the ANSI FORTRAN 66 and ANSI FORTRAN 77 standard dialects, as well as a large number of extensions found in various commercial FORTRAN compilers. The following FORTRAN dialects are supported: VAX, PDP, SUN, HP, CRAY, PRIME, Data General, UNISYS, and Honeywell. Some FORTRAN 90 extensions are also supported. With this product, existing FORTRAN applications will compile on “open” platforms as is, i.e., without making any changes to the original FORTRAN source code.

Validated by the GSA FORTRAN Compiler Validation Test Suite, the compiler works by translating FORTRAN source code to C source code (from where it derives its portability). It requires a C compiler/linker to produce executable code and is the ideal processor for hybrid FORTRAN/C applications. Symbolic FORTRAN debugging is supported by the symbolic debugger of the host C platform (e.g., the dbx tool for UNIX platforms).

No matter how old or how extended your FORTRAN dialect is, gmFortran will process it by first translating it to the more versatile and more portable C language. Your long-established FORTRAN programs do not have to be maintenance burdens running inefficiently on old platforms; with gmFortran, you can give them new life on contemporary platforms where you can take advantage of new technology options, including the option of program maintenance in either FORTRAN or C.

gmFortran offers a number of advantages over other FORTRAN to C translators:

1. Multi-Dialect, Multi-Platform Processing. gmFortran translates both standard and nonstandard dialects of FORTRAN. Both the ANSI FORTRAN 66 and the ANSI FORTRAN 77 dialects are supported, and several Fortran 90 extensions are included as well. Several commercial, extended dialects of FORTRAN – such as VAX, PDP, PRIME, Honeywell, Data general, and SUN – are also supported. The C source code generated can be ported to any platform that supports a standard C compiler. In addition, the translator can be configured to process dialect extensions not included in the default dialect definition.

2. Completeness. The syntax processor handles almost all features of standard and extended FORTRAN dialects, including structures and pointer variables. On the execution side, its comprehensive runtime library covers the full functionality of all of the above FORTRAN dialects (full FORTRAN I/O, complex arithmetic, NAMELIST, DECODE/ENCODE, DEFINE FILE, B-FORMAT, %VAL, %REF, %DESCR, STRUCTURE, RECORD, embedded and inline comments, etc.).

3. Correctness. Since gmFortran compiles FORTRAN to C, i.e., performs both a syntactic and a semantic transformation of FORTRAN to C, the translated C code is both compilable and “correct,” that is, it yields reproducible results - which is not the case with other translation tools which are limited to simple syntax transformations that yield partial or incorrect translations. The compiler has been tested on several platforms with Version 2.0 of the FORTRAN compiler Validation System from the FederalSoftwareTestingCenter and passed the test at the full validation level on all platforms.

4. Software Conversion. gmFortran is a highly efficient portation tool. It can convert thousands of lines of code per minute, depending on the platform, and achieves 100 percent conversion rates routinely, “right out of the box.” By removing the need to make any manual source changes during the conversion, it makes the porting process repeatable and allows for rigorous port verification.

5. Language Migration. gmFortran is a language migration tool. It helps you migrate from maintaining your applications in FORTRAN to maintaining them in C. You can treat the C code generated by gmFortran as an intermediate step in a compilation or as the end product of an automatic translation process.

6. Documentation. According to a Computer Language review (October, 1988), “the documentation for gmFortran is excellent. It meticulously details the translation process, how to create your own dialects, all the runtime library routines, the inner pseudocode into which gmFortran translates your FORTRAN, how this pseudocode translates back to C, and more.” In addition, the language reference component of the documentation describes the actual FORTRAN dialect that gmFortran supports.

7. Error Processing. When a FORTRAN error occurs, an error message is issued and the user has the option to exit or continue processing. A command-line switch allows the user to select from five different error handling options:

- Stop translation when the first error is encountered
- Translate all errors into non-compilable FORTRAN ERROR statements
- Translate all errors into non-linkable FORTRAN ERROR function calls
- Translate all errors into executable print-error-message statements
- Translate all errors into warning comments.

The C code produced by gmFortran is clean, readable and maintainable. The code generator actually offers three major output options (or biases) as well as a multitude of detailed command-line switches:

The FORTRAN bias generates C output which is as close to the original FORTRAN as possible and is aimed at easing the transition of those users who are presently FORTRAN programmers but wish to (or must) become C programmers.

The C bias generates C output which looks much like a standard C program and is aimed at those users who are C programmers but must now take over the maintenance of a FORTRAN code.

The optimized bias generates C output which is designed to compile as quickly as possible and to produce an efficient as possible executable module. This output is not very readable and is aimed at those users who wish to continue to program in FORTRAN. For these users, the C output is of no importance as such. It is merely an intermediate step and serves as input to the C compiler.

How Does It Work?

gmFortran translates FORTRAN code to C code which can then be compiled via any standard C compiler and linked with the gmFortran runtime library to produce efficient executable code. The resultant executable code produces the same results on a target platform as the original code does on the source platform.

In designing gmFortran we took the position that the only difference between a translator and a compiler should be that a compiler converts the source code into machine language while a translator takes it to a higher level language. gmFortran compiles the FORTRAN source language into a low level pseudo-code. This pseudo-code is much like the output produced by the first, or second, pass of contemporary compilers. Second, it optimizes that code using the same techniques as used during the optimization pass of a compiler. Third, it does code generation; but the code generated is not machine code, it is C.

A more detailed description of the design and methodology of gmFortran appeared in a series of three technical papers in the Journal of C Language Translation:


More Information

The gmFortran manual is available as PDF [here](#).