Chapter 3

Compilers

This chapter describes the compilers supported by IRIX and the programming language standards adhered to in the implementation of these compilers. Read "Creating Executable Files" on page 37 to learn how to compile and debug programs written in these languages. Refer to Table 2-2 for a list the manuals you can read to learn more about the topics discussed in this chapter.

C and C++

Silicon Graphic provides compilers for both the C and C++ programming languages.

The IRIX C compiler conforms to the ANSI C standard as well as "traditional C", the dialect of C defined by Kernigan and Ritchie in *The C Programming Language*. Compiler options allow you to compile programs written in "traditional C", pure ANSI C, or ANSI C with Silicon Graphics extensions. ANSI C is part of the IRIS Developer's Option (IDO).

The Silicon Graphics implementation of C++ conforms to the standard as defined in *The Annotated C++ Reference Manual* by Margaret Ellis and Bjarne Stroustrup.

The IRIX operating system supports two versions of the C++ compiler.

NCC This is a 32-bit native compiler that implements all the features of the language described in *The Annotated C++ Reference Manual* by Margaret Ellis and Bjarne Stroustrup.

DCC This is the Delta/C++ compiler that is available as part of the CASEVision/Workshop Pro C++ package. Delta C++ is an extension to C++. It is a native compiler (not a preprocessor). It supports dynamic classes, which minimizes the need for recompilation if you modify classes.

Fortran 77

Fortran, as implemented on Silicon Graphics IRIS workstations, contains the full ANSI Programming Language Fortran (X3.9-1978). It has extensions that provide full VMS Fortran compatibility to the extent possible without the VMS operating system or data representation. It also contains extensions that provide partial compatibility with programs written in VMS Fortran and Fortran 77.

The Parallelizing Compilers

If your application runs on a multiprocessor platform and performance is a critical issue, you may want parts of the program to run concurrently. Two Silicon Graphics compilers, C and Fortran, have preprocessors that analyze source code and produce, where possible, object code that utilizes the multiprocessor environment.

Power C

IRIS Power C allows your program to make efficient use of Silicon Graphics multiprocessor platforms by generating code segments that execute concurrently. Power C consists of the standard C compiler and a preprocessor that automatically analyzes sequential code to determine where loops can run in parallel. The preprocessor generates a modified version of the source code with multiprocessing directives added. The C compiler, when it compiles the modified source code, interprets the directives and produces object code that uses multiple processors. An advantage of Power C is that you can use it to recompile existing serial C programs so that they run efficiently on multiprocessor computers without hand recoding.

The IRIS Power C Analyzer (PCA) is the C code optimization preprocessor that detects potential parallelism in C code. It also performs other optimizing tasks. The Power C Analyzer can

- direct C code to run in parallel
- determine data dependencies which might prevent code from running concurrently
- distribute well-behaved loops and certain other code across multiprocessors
- optimize source code

You can use PCA either as a standalone tool or as a phase of the C compiler. You can also enter the directives that produce concurrent code directly into your program rather than using the PCA. Figure 3-1 illustrates the role of the PCA in producing an executable module that can utilize more than one processor on a multiprocessor system.

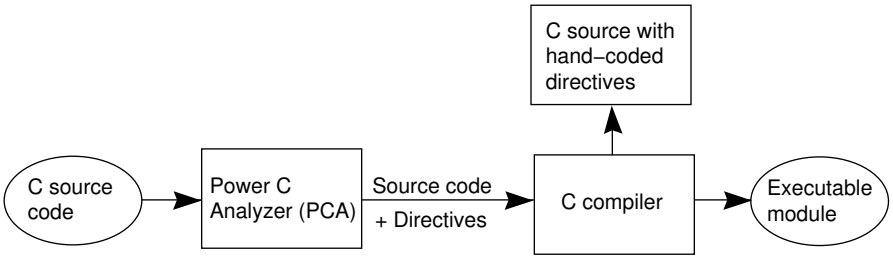

Figure 3-1 Using Power C to Produce a Parallelized Program

Power C can produce a listing containing information about the loops that it parallelizes and those that it cannot. Using this information, you may be able to modify your source so that a subsequent Power C compilation produces more efficient code. You can select a PCA compilation of your source code by specifying the *pca* compiler driver option when you compile your program.

Power Fortran Accelerator

The Power Fortran Accelerator (PFA) is a source-to-source preprocessor that enables you to run existing Fortran 77 programs efficiently on the Silicon Graphics POWER Series™ multiprocessor systems.

PFA analyzes a program and identifies loops that don't contain data dependencies. It is a preprocessor that automatically inserts special compiler directives into a Fortran program to produce a modified copy of the source. The Silicon Graphics Fortran 77 compiler can then interpret these directives to generate code that can run across all available processors. Because the directives inserted by PFA look like standard Fortran 77 comment statements, PFA does not affect the portability of the code to non-Silicon Graphics systems. Figure 3-2 illustrates the role of PFA in producing an executable module that can utilize more than one processor on a multiprocessor system.

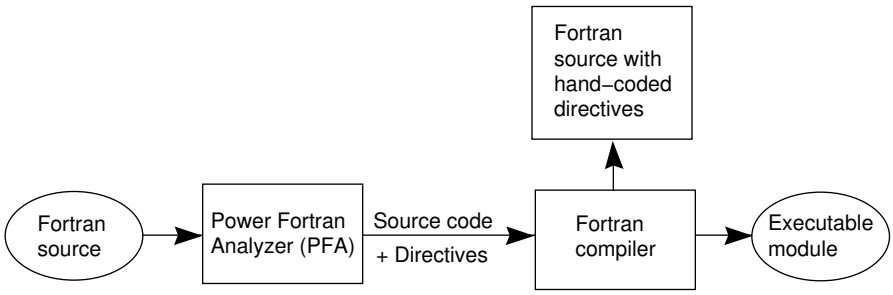

Figure 3-2 Using PFA to Produce a Parallelized Program

PFA can, if you request it, produce a listing file explaining which loops were parallelized and if not, why not. You may be able to use this information to modify your application for more efficient use of multiple processors.You can select a PFA compilation of your source code by specifying the *pfa* compiler driver option when you compile your program.

Pascal

The Pascal language supported by the Silicon Graphics Pascal compiler is an implementation of ANSI Standard Pascal (ANSI/IEEE770X3.97-1983). This implementation complies with ANSI requirements except for some extensions. These extensions include:

- names—allows underscores in identifiers and use of lowercase for public names
- constants—four extensions for constants
- statement—seven statement extensions
- declaration—six declaration extensions
- predefined procedures—four new procedures
- predefined functions—20 new functions
- predefined data types and predefined data type attributes
- extensions that affect the compile process

Ada

The Silicon Graphics AXM (Ada X/Motif) Ada Development System provides an Ada development environment for Silicon Graphics platforms. It consists of

- Ada compiler (ANSI/MIL-STD-1815A)
- X11 and OSF/Motif
- Bindings to the IRIS Graphics Library
- Non-intrusive, symbolic debugger

MP/Ada 6.2 is a multiprocessor Ada development system. It's built using the POSIX-compliant threads model for Ada tasking. If your application runs on a multiprocessor platform, MP/Ada6.2 allows Ada tasks in this application to run concurrently.