

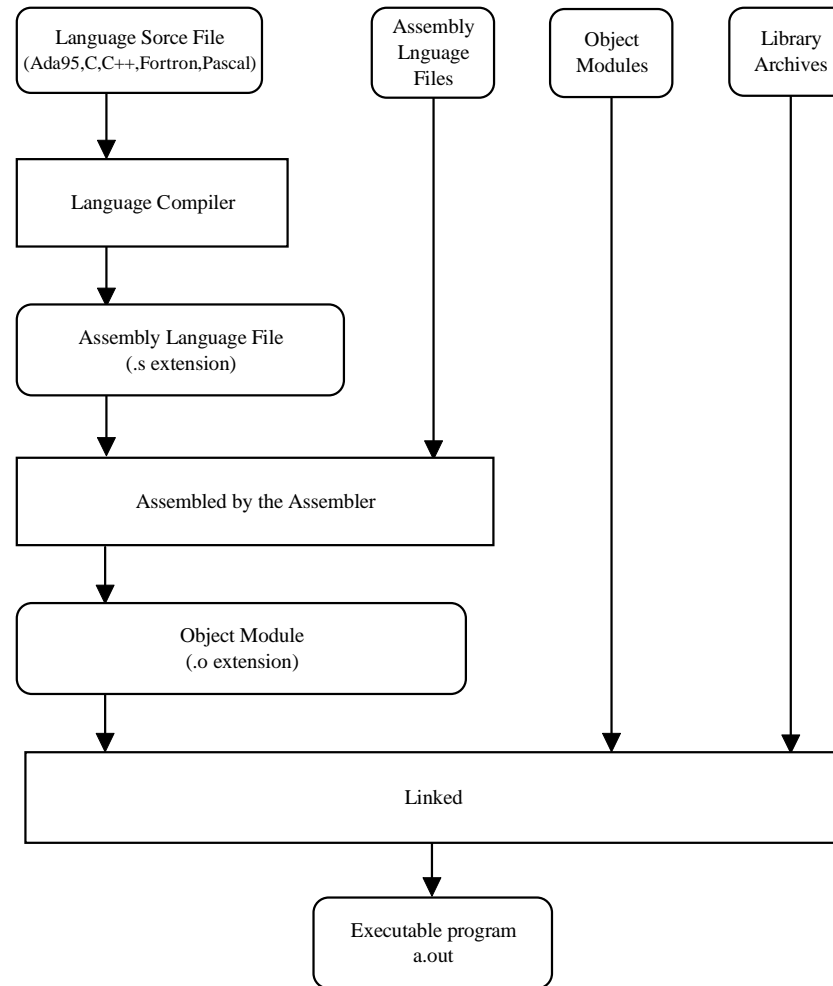
GHS

Compiler Driver

Advanced Data Controls Corp.



Flow of Source Files Through the Tool Chain



Compiler Drivers

- **A compiler driver** is a program which invokes the other components of the tool set to process a program. There is a separate compiler driver for each source language. The drivers use command line arguments and source file extensions to determine which compiler or assembler to invoke for each source file, then sequence the resulting output through the subsequent linker and conversion utilities, relieving the user of the burden of invoking each of these tools individually.

Compilers

- **Each Green Hills optimizing compiler** is a combination of a language-specific front end, a global optimizer, and a target-specific code generator. Green Hills provides compilers for five languages: Ada, C, C++, FORTRAN, and Pascal, including all major dialects. All languages for a target use the same subroutine linkage conventions. This allows modules written in different languages to call each other. The compilers generate assembly language.

Assembler

- **The relocatable macro assembler** translates assembly language statements and directives into a relocatable object file containing instructions and data.

Librarian

- **The Librarian** combines object files created by assembler or Linker into a library file. The linker can search library files to resolve internal references.

Linker

- **The Linker** combines one or more ELF object modules into a single ELF relocatable object or executable program.

Behavior of the Compiler Drivers(1)

- You can build an executable program in one command line, using the Green Hills compiler drivers.
 - Compiles source language files
 - Assembles assembly language files
 - Links together object files and libraries

Behavior of the Compiler Drivers(2)

- You can invoke the driver from the command line. The driver calls the appropriate compiler with the correct default options. By default, the driver uses the linker to link in object files from the appropriate libraries.

Behavior of the Compiler Drivers(3)

- A following lists are file name of a Compiler Dreiver.
 - **ccXXX** Compiler driver for C
 - **cxXXX** Compiler driver for C++
 - **fcXXX** Compiler driver for FORTRAN
 - **pcXXX** Compiler driver for Pascal
 - **adaXXX** Compiler driver for Ada (XXX is a type of processor)

Behavior of the Compiler Drivers(4)

- The driver begins with a list of input file types and what compilation steps to perform on that file. For example, a file with **a .c extension** is **a C source file**, and a file with **a .f or .for extension** is a **FORTRAN source file**. Each file needs to be compiled with the appropriate language compiler, assembled, and then linked.

Behavior of the Compiler Drivers(5)

- The syntax for the compiler driver command is:
 - *driver_name [options] file(s)*
 - *driver_name*
 - » The name of the driver for the language that you are using.
 - *options*
 - » Represent any combination of compiler driver options.
There options may be places either before or after filename(s) on the command line.
 - **file(s)**
 - » Represents the source file or files you wish to compile, assemble, or link. A space is required between each filename.

Behavior of the Compiler Drivers(6)

- The compiler driver recognizes certain filename extensions listed in a following table. It determines each file type from the extension and processes the file accordingly.

Extension	Assumed file type
.o	object file
.a	library file
.s	assembly language file
.inf	Inline and dependency intermediate files
.ada .adb .ads	Ada source file
.c .i	C source file
.cxx .C .cpp .cc	C++ source file
.f .for	FORTRAN source file
.p .pas	Pascal source file

- » If a extension is processor type, it indicates a assembly language with C preprocessor directives.

Behavior of the Compiler Drivers(7)

- To build an executable from a C source file called demo.c, enter the following command:

```
% ccXXX demo.c
```

- The driver recognizes **demo.c** as a valid C source file by its **.c extension** and invokes the C compiler. The compiler produces an object file which is sent to the linker and linked.
- If no errors occur, an executable file called **a.out** is created in the current directory. You can rename the output file with -o option.

Behavior of the Compiler Drivers(8)

- When more than one file is given to the driver, any object files that are created in the process are not deleted. This is convenient when only one of the files must be recompiled.

```
% ccXXX demo.c file1.c file2.c
```

Behavior of the Compiler Drivers(9)

- You then discover that `demo.c` must be modified. After editing `demo.c`, you can build the executable with this command.

```
% ccXXX demo.c file1.o file2.o
```

- Since `file1.c` and `file2.c` have not been modified, it is possible to use the object modules `file1.o` and `file2.o` created by the previous compilation.

Behavior of the Compiler Drivers(10)

- Assembly source files may be input to the compiler driver if the name of the file ends with .s For example:

```
% ccXXX demo.s
```

- For assembly source files, the compiler driver first invokes the assembler, then the linker produces **a.out** in the current directory as an executable file.

Behavior of the Compiler Drivers(11)

- If you use the **-c** option, the compiler driver stops after creating an object file for each source file on the command line.
- The following command line produces two relocatable object modules, called `demo.o` and `file.o`, in the current directory:

```
% ccXXX -c demo.c file.s
```

Behavior of the Compiler Drivers(12)

- If only one object file is created, you can use the **-o** option with the **-c** option to rename the object file. The new name must contain the suffix **.o**, For example, the following command line creates the relocatable object module **newdemo.o** in the current directory:

```
% ccXXX -c demo.c -o newdemo.o
```

Behavior of the Compiler Drivers(13)

- The compiler driver links relocatable object modules into an executable program. If all of the names of the inputs files to the driver end in .o, the compiler driver invokes the linker only. The following command line links the demo.o, file1.o, and file2.o object modules to produce the file a.out:

```
% ccXXX demo.o file1.o file2.o
```

Behavior of the Compiler Drivers(14)

- Compiler driver automatically links a startup module(`crt0.o`), GHS libraries. You can see a detail link process by using a `-v` option like as following.

```
% ccXXX -v demo.o file1.o file2.o
```

```
/green/lx ¥
```

```
@/green/{CPU_DIR}/default.lnk -sda -e ¥
```

```
__start -Y UL,/green/{CPU_DIR} ¥
```

```
/green/{CPU_DIR}/crt0.o demo.o file1.o ¥
```

```
file2.o -lansi -lind -lsys -larch
```

Behavior of the Compiler Drivers(15)

- You can disable this behavior by using a **-nostdlib** option.

```
% ccXXX -v -nostdlib demo.o file1.o file2.o
```

```
/green/lx ¥
```

```
@@/green/{CPU_DIR}/default.lnk -sda ¥
```

```
demo.o file1.o file2.o
```

Behavior of the Compiler Drivers(16)

- Compiler driver uses a default linker directives file which were provided in the target library directories of the distribution. The filename are:
 - default.lnk ; Used for normal,absolutely-located programs
 - pic.lnk ; Appropriate for PIC(Position Independent Code)
 - pid.lnk ; Appropriate for PID(Position Independent Data)
 - picpid.lnk ; Appropriate when using

Behavior of the Compiler Drivers(17)

- When you specify a linker directives file as a input file of a compiler driver then compiler driver use it instead of a default one. So a suggested method for customizing a linker directives file is to copy the default link map and make appropriate changes to it,such as altering the addresses of text and data,specifying additional user-defined sections.

```
% ccXXX -v -nostdlib demo.o file1.o file2.o demo.lnk
```

```
/green/lx @demo.lnk -sda demo.o file1.o file2.o
```

Behavior of the Compiler Drivers(18)

- When you don't need to link a startup code(`crt0.o`) but you want to link GHS libraries then you should specify a `-L` and `-l` options to do that.
 - `-Ldir`
 - Specifies a directory to be searched for libraries specified `-llib`.
 - `-lnam`
 - Looks for a library named **libnam.a** in each of the directories specified in `-Ldir` options and then in the primary and secondary library search directories.

```
% ccXXX -v -nostdlib demo.o file1.o file2.o demo.lnk  
-L/green/{CPU DIR} -lansi lind -lsys -larch
```

Behavior of the Compiler Drivers(19)

- Following options are useful one.

-D*name* ; Defines the argument *name* for the preprocessor.
-S ; Produces only an assembly file from the source file.
-I*directory* ; Adds directory to the list of directories to search when processing **#include** directives.
-list ; Produces a listing file.
-passsource ; This option is specified with **-S** or **-list** option to insert C statements as a comments in a assembly source or a listing file.
-map ; Produces a map file.