Compiler (1A)

Young Won Lim
9/29/14

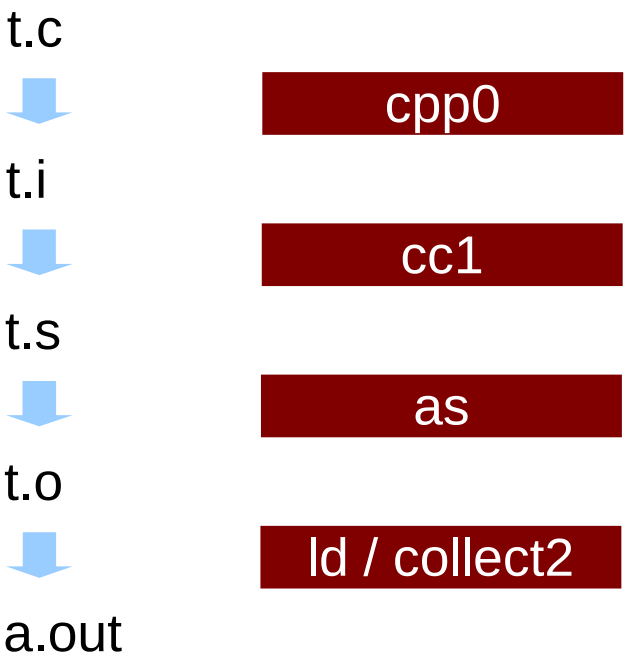
Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

GNU Compiler

- cpp0 pre processor
- cc1 c compiler
- cc1obj objective-c compiler
- cc1plus c++ compiler
- f771 fortran compiler
- jc1 java compiler
- collect2 linker

GCC Compile Process



gcc

1. Program Files

c++, c++filt, cc (link to gcc), cc1, cc1plus, collect2, cpp, cpp0, g++, gcc, gccbug, gcov and tradcpp0

2. Descriptions

2.1 cc, cc1, cc1plus, gcc

These are the C compiler.

The C++ language provides function overloading, which means that it is possible to write many functions with the same name (providing each takes parameters of different types). All C++ function names are encoded into a low-level assembly label (this process is known as mangling).

2.2 c++, cc1plus, g++

These are the C++ compiler, the equivalent of cc and gcc etc.

2.3 c++filt

The c++filt program does the inverse mapping: it decodes (demangles) low-level names into user-level names so that the linker can keep these overloaded functions from clashing.

2.4 collect2

collect2 assists with the compilation of constructors.

http://www.faqs.org/docs/linux_scratch/appendixa/gcc.html

2.5 cpp, cpp0
cpp pre-processes a source file, such as including the contents of header files into the source file. Simply add a line, such as #include <filename>, to your source file. The preprocessor will insert the contents of the included file into the source file.

2.6 gccbug
gccbug is a shell script which is used to simplify the creation of bug reports.

2.7 gcov
gcov analyzes programs to help create more efficient, faster running code through optimization.

2.8 tradcpp0
No description is currently available.

http://www.faqs.org/docs/linux_scratch/appendixa/gcc.html

cpp0

- gcc -v –save-temps -o test test.c

cc1

- gcc -da t.c

as

- as -V -Qy -o t t.s

collect2

Static Library

- mkdir libttt
- cd libttt
- cp /usr/lib/libc.a ./
- ar -x libttt.a

Dynamic Library

-

Compiling

gcc -Wall hello.c -o hello

/usr/lib/libm.a

gcc main.o hello_fn.o -o hello

gcc -Wall calc.c /usr/lib/libm.a -o calc

gcc -Wall calc.c -lm -o calc

gcc -Wall -lm calc.c -o calc (incorrect order)

gcc -Wall -I/opt/gdbm-1.8.3/include dbmain.c -lgdbm

gcc -Wall -I/opt/gdbm-1.8.3/include
  -L/opt/gdbm-1.8.3/lib dbmain.c -lgdbm

Compiling

```
$ C_INCLUDE_PATH=/opt/gdbm-1.8.3/include
$ export C_INCLUDE_PATH

$ CPLUS_INCLUDE_PATH=/opt/gdbm-1.8.3/include
$ export CPLUS_INCLUDE_PATH

$ LIBRARY_PATH=/opt/gdbm-1.8.3/lib
$ export LIBRARY_PATH

$ C_INCLUDE_PATH=.:/opt/gdbm-1.8.3/include:/net/include
$ LIBRARY_PATH=.:/opt/gdbm-1.8.3/lib:/net/lib

$ gcc -I. -I/opt/gdbm-1.8.3/include -I/net/include
  -L. -L/opt/gdbm-1.8.3/lib -L/net/lib .....
```

Gcc options

-E
-S
-c
-v
-- save-temps

cpp0 options

- -I
- -include
- -D
- -D[macro]=[macro val]
- -U[macro]
- -M
- -MM
- -nostdinc
- -C
- -Wp,[options]

cc1 options (1)

- -ansi
- -std=[c89, c99, gnu89, gnu99]
- -traditional
- -fno-asm

- -W, -Wall
- -w
- -Werror
- -pedantic
- -pedantic-errors
- -Wtraditional

cc1 options (2)

- -O0
- -O1
- -O2
- -O3
- -Os

- -g
- -pg

as options

- -Wa,[as options]
  - -al
  - -as
  - -l[pass]
  - -W, --no-warn
  - -march=[arch string]

collect2 options (1)

- -L[library directory]
- -l[library name]
- -shared
- -static
- -nostdlib
- -nostartfiles

collect2 options (2)

- -W,[link options]
  - -s
  - -x
  - -n
  - -r
  - -e [name]
  - -oformat [format]

ELF

gcc -g t t.c

objdump -S t

gcc -v –save-temps -o t t.c

- t.i (the preprocessed output)
- t.s (the assembly file)

-save-temps

-save-temps
-save-temps=cwd
Store the usual "temporary" intermediate files permanently;
gcc -c -save-temps foo.c produces foo.i, foo.s, foo.o.
This creates a preprocessed foo.i output file even though
the compiler now normally uses an integrated preprocessor.

objdump

objdump [-a | --archive-headers]
[-b bfdname | --target=bfdname]
[-d | --disassemble] [-D | --disassemble-all]
[-f | --file-headers]
[-h | --section-headers | --headers] [-i | --info]
[-j section | --section=section]
[-l | --line-numbers]
[-m machine | --architecture=machine]
[-r | --reloc] [-R | --dynamic-reloc]
[-s | --full-contents] [--stabs]
[-t | --syms] [-T | --dynamic-syms] [-x | --all-headers]
[--version] [--help] objfile...

objdump displays information about one or more object files. The options control what particular information to display. This information is mostly useful to programmers who are working on the compilation tools, as opposed to programmers who just want their program to compile and work.

objfile... are the object files to be examined. When you specify archives, objdump shows information on each of the member object files.

The long and short forms of options, shown here as alternatives, are equivalent. At least one option besides `-l' must be given.

-g

-g Produce debugging information in the operating system's native format (stabs, COFF, XCOFF, or DWARF 2). GDB can work with this debugging information.

-g

-v Verbose mode. Print out GNU CPP's version number at the beginning of execution, and report the final form of the include path.

GCC Options

gcc -W -Wall -O2 -o t t.c

-E
-S
-c
-v
--save-temps

CC1 Options

-ansi
-std
-traditional
-fno -asm
-W
-Wall
-O0
-O1
-O2
-O3

-Os

Debug Options

-g
-pg

Collect2, ld Options

-L
-l
-shared
-static
-nostdlib
-nostartfiles
-Wl
-s
-x
-n
-r
-e
-M
-oformat

ELF

gcc -da t.c

as -V -Qy -o t.o t.s

Static Library

gcc -c t1.c t2.c

ar rscv libttt.a t1.o t2.o

gcc -o t t.c -l./ -lttt

ar rus [lib name] [obj files]

ar ds [lib name] [obj files]

ar x [lib name]

ar t [lib name]

Dynamic Shared Library

gcc -ipic -c t1.c t2.c

gcc -shared -wl,soname,libttt.so.0 -o libttt.so.0.0.0 t1.o t2.o

-ln -s libttt.so.0.0.0 libttt.so

-ln -s libttt.so.0.0.0 libttt.so.0

ldconfig

gcc -o t t.c -l./ -lttt

binutil

addr2line

ar

as

gprof

ld

nm

objcopy

objdump

ranlib

readelf

size

strip

addr2line

addr2line -fe t 0x80484a0

profile

gcc -g -o t t.c

gprof -a ./t

nm

A
B
C
D
G
I
N
R
S
T
U
V
W
-
?

nm

-a
-D
-g
-v
-s
-u

objcopy

objcopy t t.new
objcopy -o binary t t.new
objcopy -r,note -r,comment t t.new
objcopy -s t t.new

Objdump

Objdump -d t.o
Objdump -S t.o

-a
-x
-t
-T
-r
-R

strip

Strip R,note -R,comment -s t

References

[1] An Introduction to GCC, B. Gough, http://www.network-theory.co.uk/docs/gccintro/
[2] Unix, Linux Programming Indispensable Utilities, CW Paik