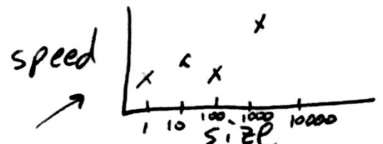
Sheaff

Nicholas LaJoie
ECE 331
1/24/17

- Globbing is weird
  - `ls [4-7]` == `ls {4..7}` — extension in bash
    - ↳ Use w/ double digits
  - ".." operator (range) "Range Operator"
- #5) Formatting



if it's 10,000 b/s, put kb/s → semilogx scale

(*tree)

Filesystem

(*Old days: bin, boot, etc on seperate drive to boot)

- Standard across linux

/ - root directory
bin - binary files - essential (gets system booted)
dev - device special files
boot - boot files (kernel)
lib - essential libraries
proc - virtual filesystem - interface to process data (progs. running on pi)
root - user "root" home directory ("privilage user")
  ⇒ Cannot kill zombies
run - runtime data → tmpfs (RAM) (not on SD Card))
sbin - essential admin commands
sys - interface to configure hardware
var - variable data
  ⇒ package management ...
usr - keep usr files here → commands, docs, libs, afterboot
home - user's home dir's

(*arm-linux-gnueabihf)

*'ls usr' mirrors root a bit
usr/share - docs, man pages, static data
usr/src - linux kernel

(*libraries don't need to be installed system-wide)

etc - configuration
usr/local - can install own programs (i.e. avrdude can be in local or elsewhere)

Navigation:

".." - parent directory
"." - current directory (a special file that contains directory info)
"/" - root
"pwd" - print working directory "FHS" - file heirarchy standard

Compiler

- invokes preprocessor
- produces text-based assembly
- assembler creates object file
- invoke a linker (resolves symbols)
- produces executable

(*'pic' code
position independent... code?)

*Intel Fortran Compiler

`gcc -E -o x.cpp x.c`

`gcc -S -o x.cpp x.c //assembly!`

elf
"executable linker format"

`gcc -c -o x.o x.c`

`file x.o` ⇒ LSB, ARM, etc.

-Wall ("almost all warnings")

`gcc -v -Wall -g -o p x.c`

libm.so ← libm.a
-lm

-g debug
-L/usr/local/lib

*Optimize -O2

-I./include (if header files are in a weird spot)

File Permissions

Permission - files have:
- USER (owner)
- group
- other

./ to run stuff (dot usually isn't in a path)

USER
- read
- write
- execute

group

other

filetype (relative to filesys)

-rw-rw----

user group other

- means denied
- r-read
- w-write
- x-execute

u + r
g - w
o = x
a

- c - char device
- B - block device
- l - symbolic link
- - - reg. file
- d - directory
- s - socket
- p - named pipe

*symbolic link gives permissions for all (don't try to change it)

*directory permissions are different

dir
- ↳ read - list files
- ↳ write - mkdir
- ↳ clear exec - no changing directories
  - list files
  - can't see attributes
  - can't create files
- ↳ set exec - do all

*umask

*chmod "change mode"
- → symbolically
- → absolute

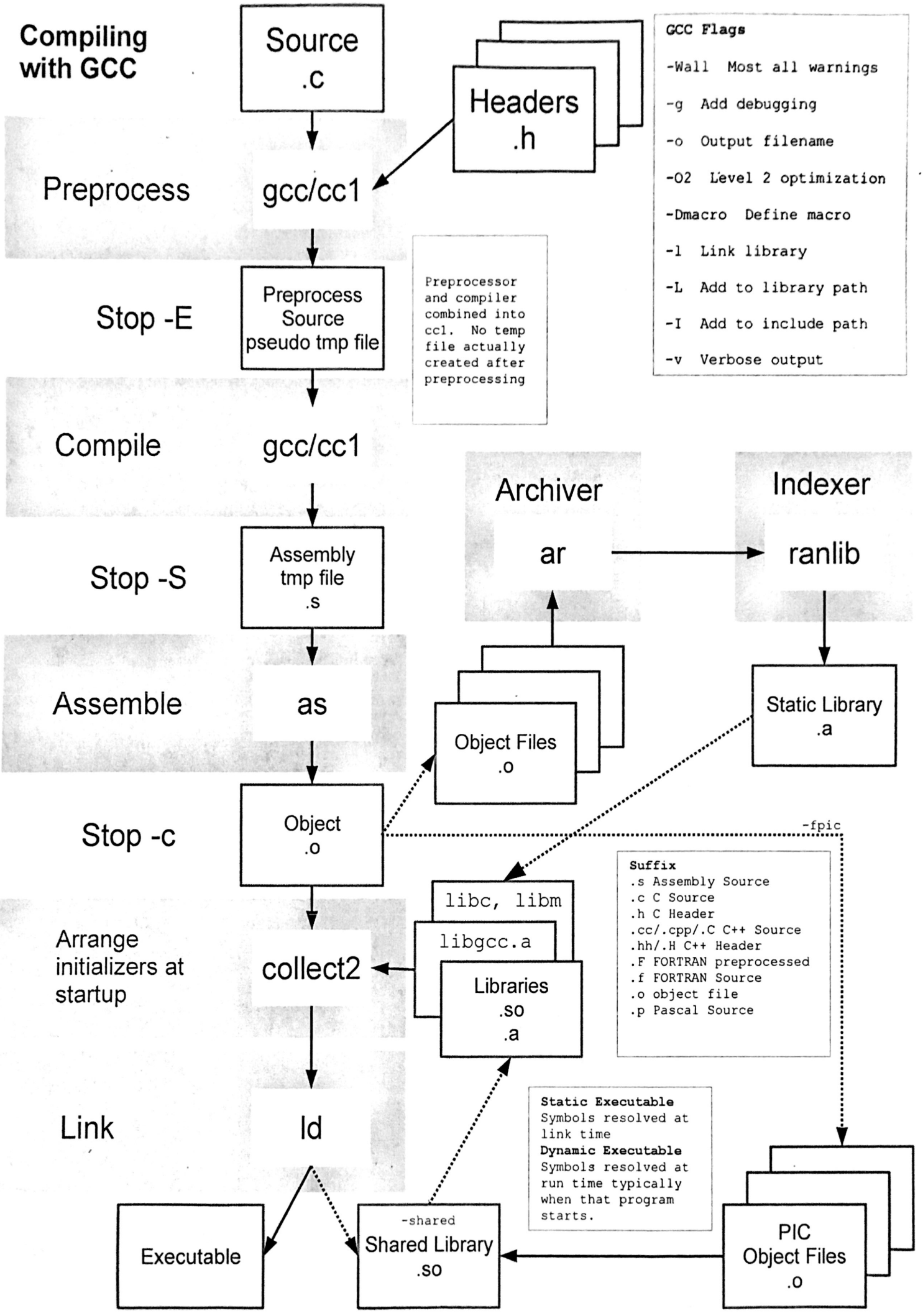
*Look up "octal"

'chmod 664 9'

* remove permissions w/ 000 (i.e. cat 8)

chmod u+r, a+x 8

user read all executable


Compiling with GCC
Source .c
Headers .h
GCC Flags
-Wall Most all warnings
-g Add debugging
-o Output filename
-O2 Level 2 optimization
-Dmacro Define macro
-l Link library
-L Add to library path
-I Add to include path
-v Verbose output
Preprocess
gcc/cc1
Stop -E
Preprocess Source pseudo tmp file
Preprocessor and compiler combined into cc1. No temp file actually created after preprocessing
Compile
gcc/cc1
Archiver
ar
Indexer
ranlib
Stop -S
Assembly tmp file .s
Assemble
as
Object Files .o
Static Library .a
Stop -c
Object .o
-fpic
libc, libm
libgcc.a
Libraries .so .a
Suffix
.s Assembly Source
.c C Source
.h C Header
.cc/.cpp/.C C++ Source
.hh/.H C++ Header
.F FORTRAN preprocessed
.f FORTRAN Source
.o object file
.p Pascal Source
Arrange initializers at startup
collect2
Link
ld
Static Executable
Symbols resolved at link time
Dynamic Executable
Symbols resolved at run time typically when that program starts.
Executable
-shared
Shared Library .so
PIC Object Files .o