CH3 GNU Software

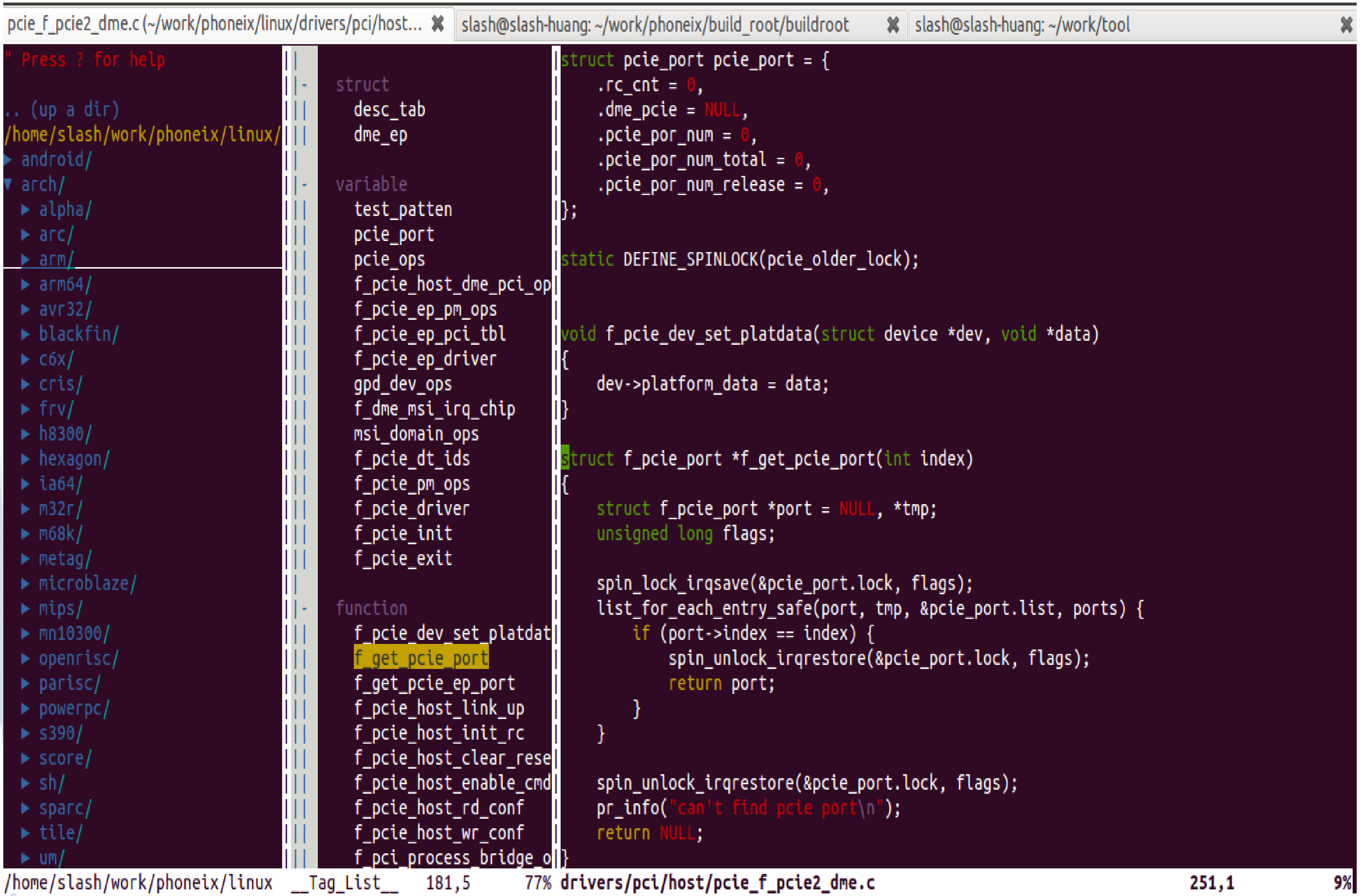
Open Source License

- GNU General Public License
  - 只要在一個軟件中使用 (" 使用 " 指類庫引用，修改後的代碼或者衍生代碼) GPL 協議的產品，則該軟件產品必須也採用 GPL 協議，既必須也是開源和免費． 這就是所謂的 " 傳染性 "
- BSD License
  - 基本上使用者可以 " 為所欲為 ", 可以自由的使用，修改源代碼，也可以將修改後的代碼作為開源或者專有軟件再發佈 .
- LGPL
  - LGPL 是 GPL 的一個為主要為類庫使用設計的開源協議 .LGPL 允許商 業軟件通過類庫引用 (link) 方式使用 LGPL 類庫而不需要開源商業軟件的代碼． 這使得採用 LGPL 協議的開源代碼可以被商業軟件作為類庫引用並發布和銷售 .

Develop Tool

Vim Tool

- Vim tools
  - Nerdtree
    - https://github.com/scrooloose/nerdtree
    - Nerdtree
  - Taglist
    - http://vim-taglist.sourceforge.net/
    - Command : Tlist



pcie_f_pcie2_dme.c (~/work/phoneix/linux/drivers/pci/host... | slash@slash-huang: ~/work/phoneix/build_root/buildroot | slash@slash-huang: ~/work/tool

```
" Press ? for help

.. (up a dir)
/home/slash/work/phoneix/linux/
▶ android/
▼ arch/
  ▶ alpha/
  ▶ arc/
  ▶ arm/
  ▶ arm64/
  ▶ avr32/
  ▶ blackfin/
  ▶ c6x/
  ▶ cris/
  ▶ frv/
  ▶ h8300/
  ▶ hexagon/
  ▶ ia64/
  ▶ m32r/
  ▶ m68k/
  ▶ metag/
  ▶ microblaze/
  ▶ mips/
  ▶ mn10300/
  ▶ openrisc/
  ▶ parisc/
  ▶ powerpc/
  ▶ s390/
  ▶ score/
  ▶ sh/
  ▶ sparc/
  ▶ tile/
  ▶ um/
```

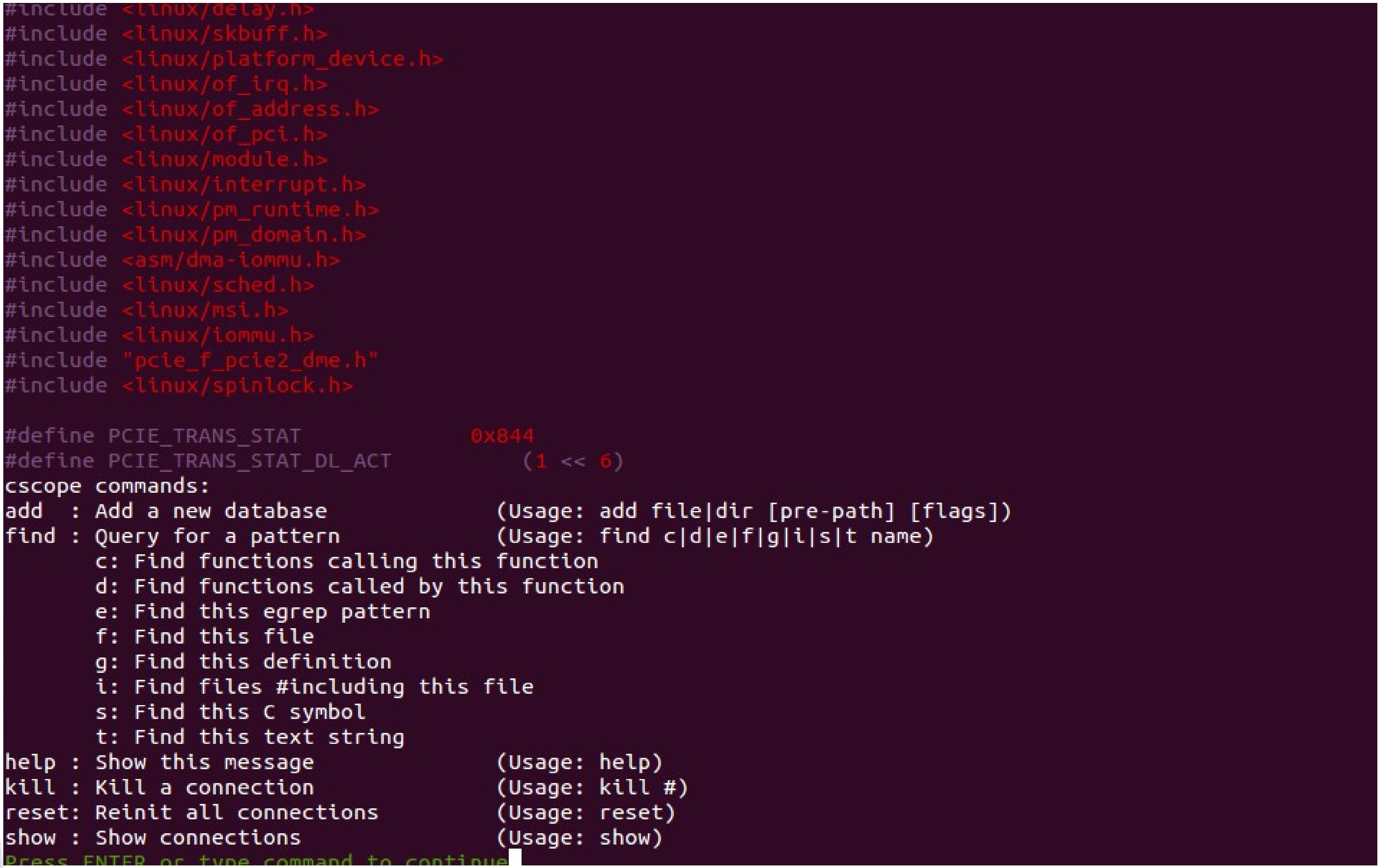
```
struct
  desc_tab
  dme_ep

variable
  test_patten
  pcie_port
  pcie_ops
  f_pcie_host_dme_pci_op
  f_pcie_ep_pm_ops
  f_pcie_ep_pci_tbl
  f_pcie_ep_driver
  gpd_dev_ops
  f_dme_msi_irq_chip
  msi_domain_ops
  f_pcie_dt_ids
  f_pcie_pm_ops
  f_pcie_driver
  f_pcie_init
  f_pcie_exit

function
  f_pcie_dev_set_platdat
  f_get_pcie_port
  f_get_pcie_ep_port
  f_pcie_host_link_up
  f_pcie_host_init_rc
  f_pcie_host_clear_rese
  f_pcie_host_enable_cmd
  f_pcie_host_rd_conf
  f_pcie_host_wr_conf
  f_pci_process_bridge_o
```

```
struct pcie_port pcie_port = {
    .rc_cnt = 0,
    .dme_pcie = NULL,
    .pcie_por_num = 0,
    .pcie_por_num_total = 0,
    .pcie_por_num_release = 0,
};

static DEFINE_SPINLOCK(pcie_older_lock);


void f_pcie_dev_set_platdata(struct device *dev, void *data)
{
    dev->platform_data = data;
}

struct f_pcie_port *f_get_pcie_port(int index)
{
    struct f_pcie_port *port = NULL, *tmp;
    unsigned long flags;

    spin_lock_irqsave(&pcie_port.lock, flags);
    list_for_each_entry_safe(port, tmp, &pcie_port.list, ports) {
        if (port->index == index) {
            spin_unlock_irqrestore(&pcie_port.lock, flags);
            return port;
        }
    }

    spin_unlock_irqrestore(&pcie_port.lock, flags);
    pr_info("can't find pcie port\n");
    return NULL;
}
```

```
/home/slash/work/phoneix/linux  __Tag_List__    181,5      77% drivers/pci/host/pcie_f_pcie2_dme.c                    251,1         9%
```

Tracking code tool :Cscope

- cscope -Rk
- make cscope, cscope -d cscope.out
- Vim command
  - cscope add ~/work/phoneix/linux/cscope.out
  - cscope find g f_get_pcie_ep_port
- Linux command
  - Find -name “*.c” | xarge grep -n “function name”



```
#include <linux/delay.h>
#include <linux/skbuff.h>
#include <linux/platform_device.h>
#include <linux/of_irq.h>
#include <linux/of_address.h>
#include <linux/of_pci.h>
#include <linux/module.h>
#include <linux/interrupt.h>
#include <linux/pm_runtime.h>
#include <linux/pm_domain.h>
#include <asm/dma-iommu.h>
#include <linux/sched.h>
#include <linux/msi.h>
#include <linux/iommu.h>
#include "pcie_f_pcie2_dme.h"
#include <linux/spinlock.h>

#define PCIE_TRANS_STAT                 0x844
#define PCIE_TRANS_STAT_DL_ACT              (1 << 6)
cscope commands:
add  : Add a new database             (Usage: add file|dir [pre-path] [flags])
find : Query for a pattern            (Usage: find c|d|e|f|g|i|s|t name)
       c: Find functions calling this function
       d: Find functions called by this function
       e: Find this egrep pattern
       f: Find this file
       g: Find this definition
       i: Find files #including this file
       s: Find this C symbol
       t: Find this text string
help : Show this message              (Usage: help)
kill : Kill a connection              (Usage: kill #)
reset: Reinit all connections         (Usage: reset)
show : Show connections               (Usage: show)
Press ENTER or type command to continue
```

Linux CodeStyle

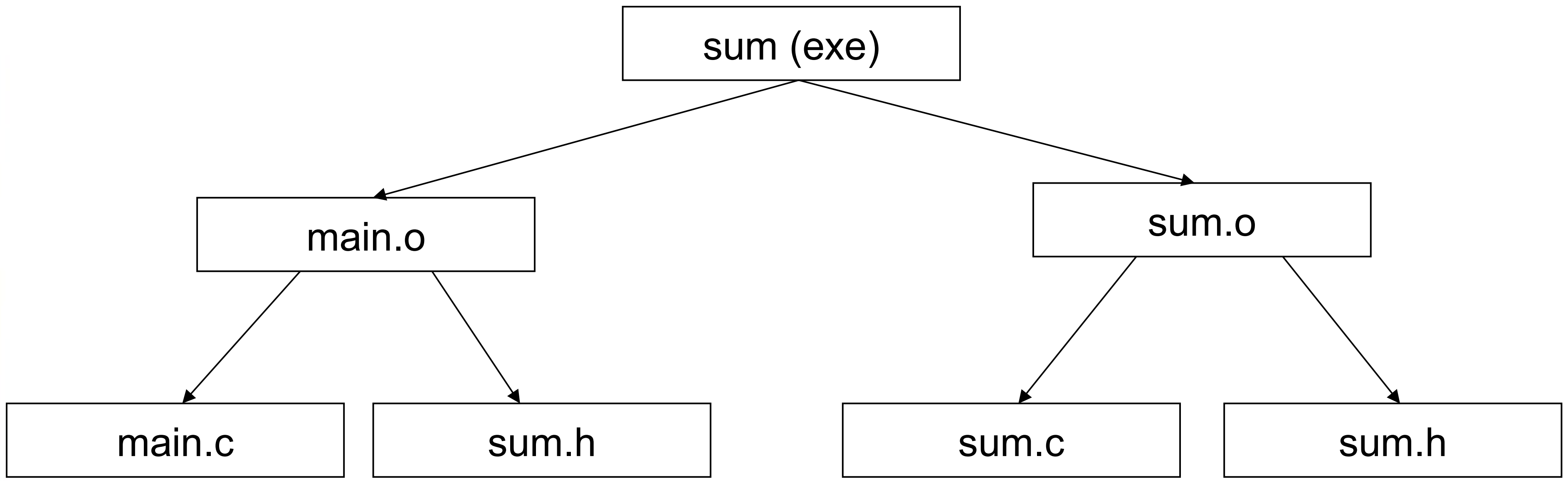
- 1. Indentation
- 2. Breaking long lines and strings
- 3. Placing Braces and Spaces
- 4. Naming
- 5. Typedefs
- 6. Functions
- 7. Commenting
- 10. Kconfig configuration files
- 11. Macros, Enums and RTL
- 12. Printing kernel messages
- 13. Function return values and names

Makefile

Makefile

- Simplify compile command
- Automation compile, linker program source
- It can update source in accordance with the dependence

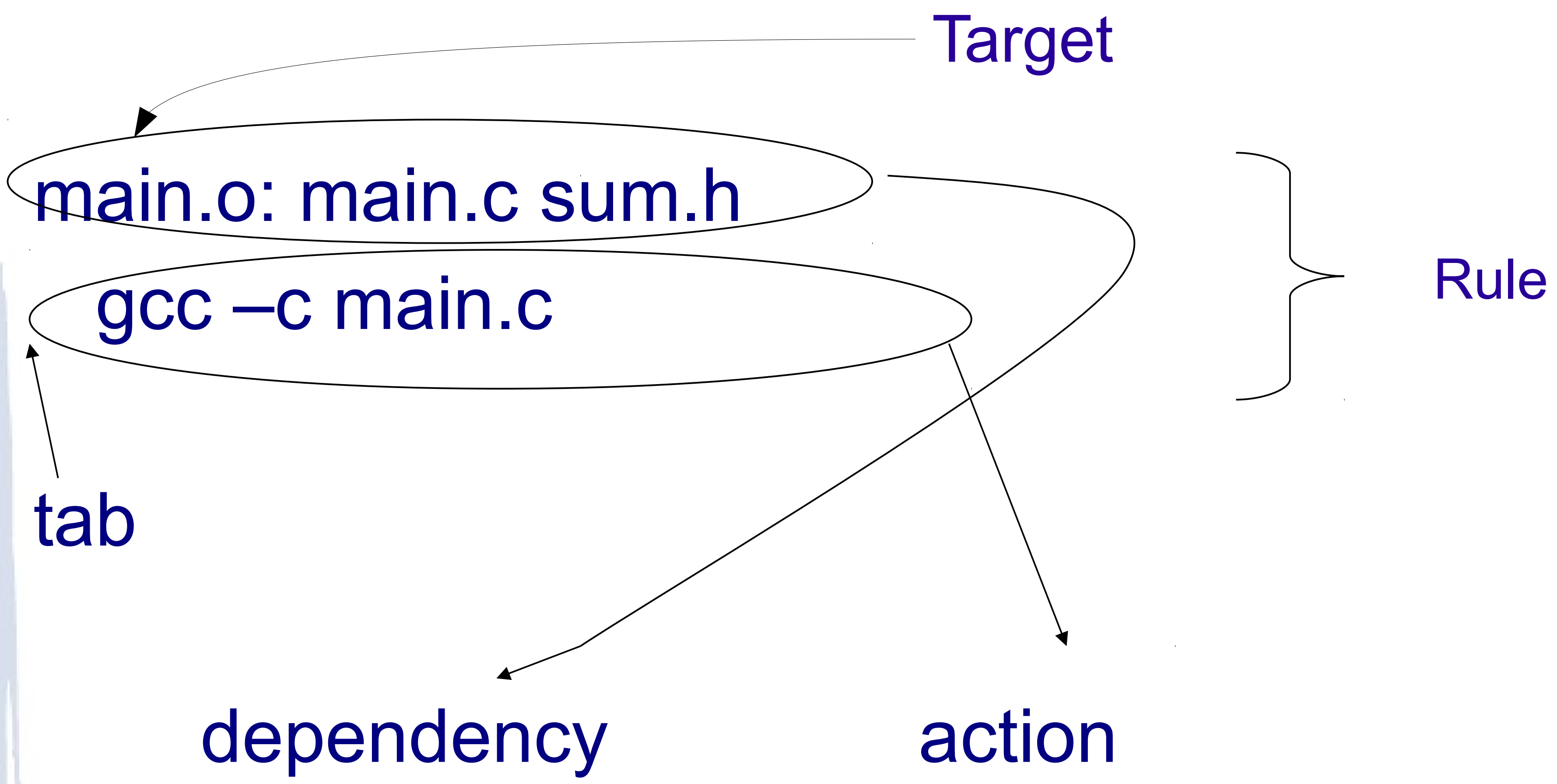
Makefile



Makefile

```
sum: main.o sum.o
	gcc –o sum main.o sum.o

main.o: main.c sum.h
	gcc –c main.c

sum.o: sum.c sum.h
	gcc –c sum.c
```

Rule syntax



```
main.o: main.c sum.h
	gcc –c main.c
```

Targets and Prerequisites

```
Target1 Target2 Target3: prereq1 prereq2 prereq3
Tab     commands
        commands
        commands
```

```
foo.o: foo.c foo.h
	gcc -c foo.c
```

Built-in Rules

- You don't have to tell make how to do standard operations like compiling an object file from C source
- The program has a built-in default rule for that operation, and for many others

```
CC = gcc
CFLAGS = -Werror -std=c99
OBJS =  circle.o circulararea.o

circle: $(OBJS) -lm
```

Double-Colon Rules

- They are handled differently from ordinary rules when the same target appears in more than one rule

```
CC = gcc
RM = rm -f
CFLAGS = -Wall -std=c99
DBGFLAGS = -ggdb -pg
DEBUGFILE = ./debug
SRC = circle.c circulararea.c

circle :: $(SRC)
	$(CC) $(CFLAGS) -o $@ -lm $^

circle :: $(DEBUGFILE)
	$(CC) $(CFLAGS) $(DBGFLAGS) -o $@ -lm $(SRC)

.PHONY : clean
clean  :
	$(RM) circle
```

Double-Colon Rules

```
$ make clean
rm -f circle
$ make circle
gcc -Wall -std=c99 -o circle -lm circle.c circulararea.c
$ make circle
make: `circle' is up to date.
$ touch debug
$ make circle
gcc -Wall -std=c99 -ggdb -pg -o circle -lm circle.c circulararea.c
$ make circle
make: `circle' is up to date.
$ make clean
rm -f circle
$ make circle
gcc -Wall -std=c99 -o circle -lm circle.c circulararea.c
```

Assignment Operators

- =
  - Defines a recursively expanded variable
- :=
  - Defines a simply expanded variable
- +=
  - Also called the append operator. Appends more characters to the existing value of a variable
- ?=
  - The conditional assignment operator. Assigns a value to a variable, but only if the variable has no value, otherwise keep original value

The Automatic Variables

- $@
  - The target filename.
- $<
  - The first prerequisite.
- $^
  - The list of prerequisites, excluding duplicate elements.
- $?
  - The list of prerequisites that are newer than the target.
- $*
  - The stem of the target filenamethat is, the part represented by % in a pattern rule
- $+
  - The full list of prerequisites, including duplicates.

The Automatic Variables

```
CC = gcc
CFLAGS = -Wall -g -std=c99
LDFLAGS = -lm

circle : circle.o circulararea.o
        $(CC) $(LDFLAGS) -o $@ $^

circle.o : circle.c
        $(CC) $(CFLAGS) -o $@ -c $<

circulararea.o: circulararea.c
        $(CC) $(CFLAGS) -o $@ -c $<
```

Phony Targets

- .PHONY
  - Any targets that are prerequisites of .PHONY are always treated as out of date.

```
#Naming our phony targets
.PHONY: clean install

#Removing the executable and the object files
clean:
   rm sample main.o example.o
   echo clean: make complete

#Installing the final product
install:
   cp sample /usr/local
   echo install: make complete
```

Command-Line Options

- -C dir, --directory= dir
  - make changes the current working directory to dir before it does anything else. If the command line includes multiple -C options, each directory specified builds on the previous one
- -I dir, --include-dir= dir
  - If a makefile contains include directives that specify files without absolute paths, search for such files in the directory.
- -j [number] , --jobs[= number]
  - Run multiple commands in parallel
- -o filename, --old-file= filename, --assume-old= filename
  - make treats the specified file as if it were up to date, and yet older than any file that depends on it

Build Linux Library

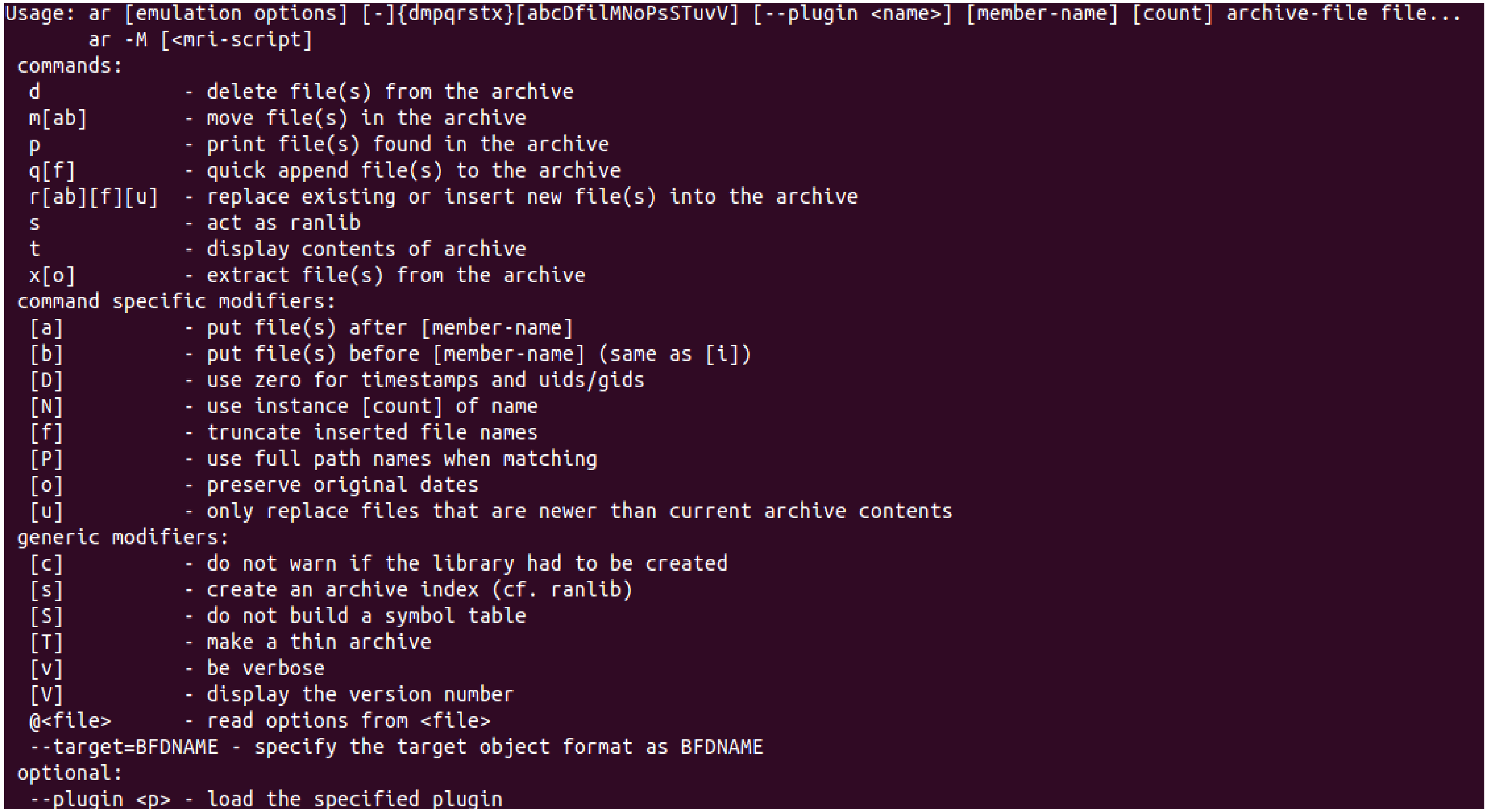
Linux Library

- Static Libraries
  - statically aware
- Dynamically Linked "Shared Object" Libraries
  - Dynamically linked at run time

Static Libraries

- static_lib_name.a
- Create static library with **ar**
  - **ar --help**
  - **ar** -cvq libctest.a test1.o test2.o
- Compile
  - gcc -o test main.c libctest.a
  - gcc -o test main.c -L/path/to/library-directory -lctest

ar



```
Usage: ar [emulation options] [-]{dmpqrstx}[abcDfilMNoPsSTuvV] [--plugin <name>] [member-name] [count] archive-file file...
       ar -M [<mri-script]
 commands:
  d            - delete file(s) from the archive
  m[ab]        - move file(s) in the archive
  p            - print file(s) found in the archive
  q[f]         - quick append file(s) to the archive
  r[ab][f][u]  - replace existing or insert new file(s) into the archive
  s            - act as ranlib
  t            - display contents of archive
  x[o]         - extract file(s) from the archive
 command specific modifiers:
  [a]          - put file(s) after [member-name]
  [b]          - put file(s) before [member-name] (same as [i])
  [D]          - use zero for timestamps and uids/gids
  [N]          - use instance [count] of name
  [f]          - truncate inserted file names
  [P]          - use full path names when matching
  [o]          - preserve original dates
  [u]          - only replace files that are newer than current archive contents
 generic modifiers:
  [c]          - do not warn if the library had to be created
  [s]          - create an archive index (cf. ranlib)
  [S]          - do not build a symbol table
  [T]          - make a thin archive
  [v]          - be verbose
  [V]          - display the version number
  @<file>      - read options from <file>
  --target=BFDNAME - specify the target object format as BFDNAME
 optional:
  --plugin <p> - load the specified plugin
```

Dynamically Linked "Shared Object" Libraries

- Dynamic_lib_name.so
- Create share library
  - gcc -**shared** -Wl,-soname,libctest.so.1 -o libctest.so.1.0 test1.o test2.o
  - ln -s libctest.so.1.0 libctest.so.1
  - ln -s libctest.so.1 libctest.so
- gcc -o test main.c -L/library_PATH/ -lctest
- export LD_LIBRARY_PATH=LIB_PATH:$LD_LIBRARY_PATH
- ./test