系統程式設計 System Programming 2021

電機四 406415089 宋 O 天

Hw06 mybusybox

<前置作業>

安裝 BusyBox

```
git clone https://git.busybox.net/busybox/
```

ncurses 是一個程式函式庫，

它提供了 API，可以允許程式設計師編寫獨立於終端的基於文字的使用者介面

```
sudo apt-get install libncurses5-dev
```

進入 busybox 目錄之後輸入

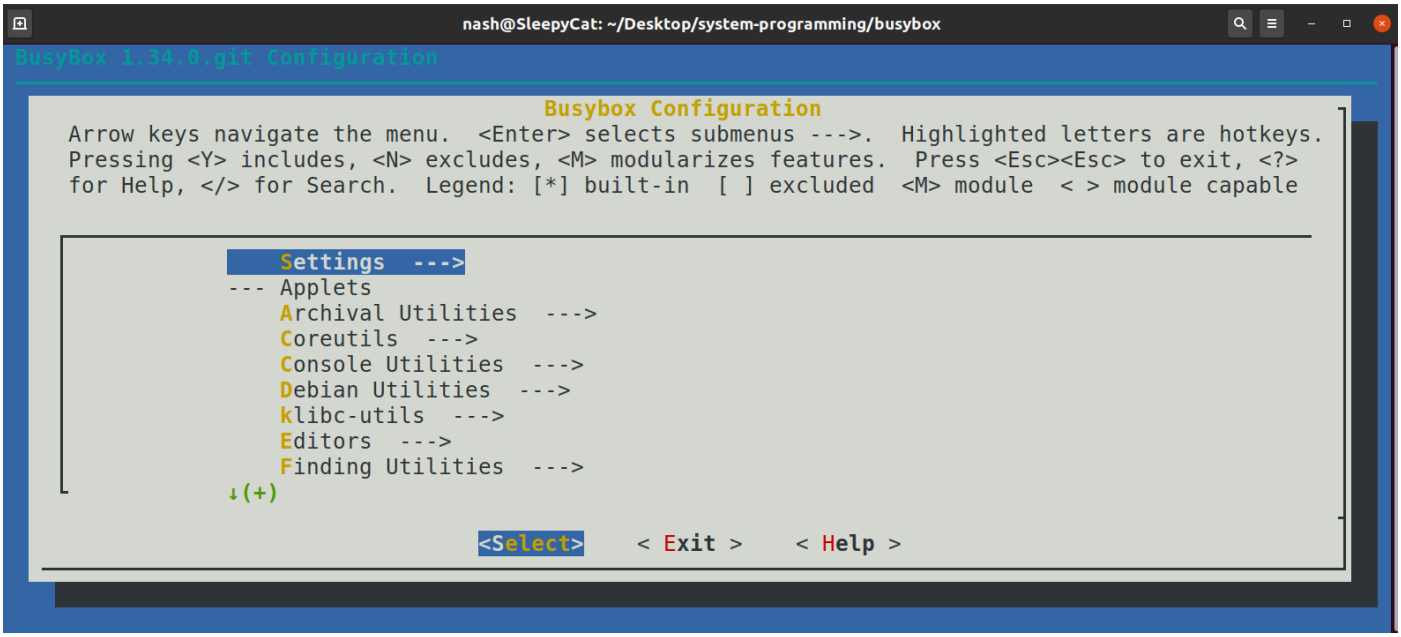
```
make defconfig
```

在 Linux kernel，編譯內核文件時，要先配置.config 文件，然後 makefile 在編譯時通過讀取.config 文件的配置來選擇要編譯的文件，選擇驅動的加載方式。

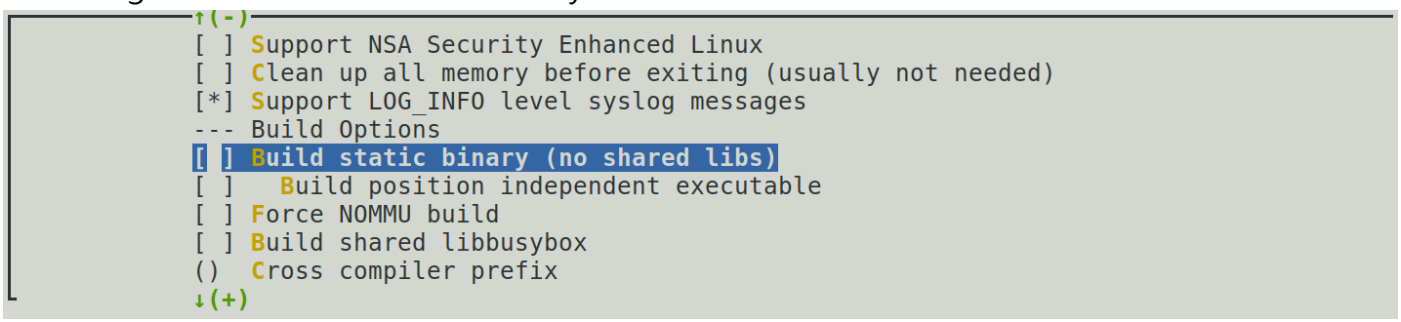
.config 文件生成可以用 make menuconfig ARCH=arm 或 make defconfig

接著輸入 `make menuconfig`

有可能會因為視窗太小不能成功開啟 UI，成功的話介面長這樣



選 settings 進入之後，啟用 static library



接著輸入 make

之後看一下資料夾內出現了甚麼

```
Final link with: m resolv
  DOC     busybox.pod
  DOC     BusyBox.txt
  DOC     busybox.1
  DOC     BusyBox.html
nash@SleepyCat:~/Desktop/system-programming/busybox$ ls
applets                 debianutils  loginutils              procps
applets_sh              docs         mailutils               qemu_multiarch_testing
arch                    e2fsprogs    Makefile                README
archival                editors      Makefile.custom         runit
AUTHORS                 examples     Makefile.flags          scripts
busybox                 findutils    Makefile.help           selinux
busybox_unstripped      include      make_single_applets.sh  shell
busybox_unstripped.map  init         miscutils               size_single_applets.sh
busybox_unstripped.out  INSTALL      modutils                sysklogd
Config.in               klibc-utils  networking              testsuite
configs                 libbb        NOFORK_NOEXEC.lst       TODO
console-tools           libpwdgrp    NOFORK_NOEXEC.sh        TODO_unicode
coreutils               LICENSE      printutils              util-linux
```

busybox 是執行檔，strip 指令可以將執行檔沒有用到的 symbol table 清除掉

接下來輸入 make install

這個指令會把東西放在_install 裡面，裡面長這樣

```
nash@SleepyCat:~/Desktop/system-programming/busybox$ cd _install
nash@SleepyCat:~/Desktop/system-programming/busybox/_install$ ls
bin  linuxrc  sbin  usr
```

可以發現在 bin 裡的檔案全部都是 symbolic link 指向 busybox

Busybox 實作的原理可以看參考資料: 向 busybox 中添加自己的 applet

```
nash@SleepyCat:~/Desktop/system-programming/busybox/_install/bin$ ls -l
total 2628
lrwxrwxrwx 1 nash nash       7  四   19 22:09 arch -> busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 ash -> busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 base32 -> busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 base64 -> busybox
-rwxr-xr-x 1 nash nash 2689504  四   19 22:09 busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 cat -> busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 chattr -> busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 chgrp -> busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 chmod -> busybox
lrwxrwxrwx 1 nash nash       7  四   19 22:09 chown -> busybox
```

接下來的 code 就是要利用類似的原理，實作出一個類似 Busybox 的功能

<Code>

makefile:

```
 7
 8 all: ${EXE}
 9
10 %:  %.c
11     ${CC} ${CFLAGS} $@.c ${LIB} -o $@
12     ln -s $@ ls
13     ln -s $@ cp
14     ln -s $@ cat
15     ln -s $@ chown
16
17 clean:
18     rm ${EXE}
19     rm ls
20     rm cp
21     rm cat
22     rm chown
```

myBusybox.c

就是透過字串比較來判斷輸入的指令是甚麼，然後利用 basename 拿掉路徑，sprintf 讓字串接起來存到一個 char buffer 內，最後在利用 system()呼叫

有限制輸入參數的數量

```c
#include <stdio.h>
#include <string.h>
#include <libgen.h>
#include <stdlib.h>

int main(int argc,char* argv[]){
    char  exebuf[100];
    char* exename = basename(argv[0]);

    if (strcmp(exename,"ls")==0 && argc == 2)
        sprintf(exebuf,"%s %s",exename,argv[1]);
    else if (strcmp(exename,"cp" )==0 && argc==3)
        sprintf(exebuf,"%s %s %s",exename,argv[1],argv[2]);
    else if (strcmp(exename,"cat")==0 && argc==2)
        sprintf(exebuf,"%s %s",exename,argv[1]);
    else if (strcmp(exename,"chown")==0 && argc==3)
        sprintf(exebuf,"%s %s %s",exename,argv[1],argv[2]);
    else{
        printf("Nothing Match\n");
        exit(EXIT_FAILURE);
    }
    system(exebuf);
```

執行結果大致上長這樣

```
nash@SleepyCat:~/Desktop/system-programming-hw$ ls
cat  chown  cp  ls  makefile  myBusybox  myBusybox.c
nash@SleepyCat:~/Desktop/system-programming-hw$ ./ls
Nothing Match
nash@SleepyCat:~/Desktop/system-programming-hw$ vim myBusybox.c
nash@SleepyCat:~/Desktop/system-programming-hw$ ./ls ~/Desktop/
practice  system-programming  system-programming-hw
nash@SleepyCat:~/Desktop/system-programming-hw$ ./cat makefile
SHELL = /bin/bash
CC = gcc
CFLAGS = -g
LIB = -lacl
SRC = $(wildcard *.c)
EXE = $(patsubst %.c, %, $(SRC))
```

<回答問題>

<1> readelf

readelf -d /usr/bin/ls

這個會列很多東西出來

readelf -d ./busybox/_install/bin/busybox

```
nash@SleepyCat:~/Desktop/system-programming$ readelf -d ./busybox/_install/bin/busybox

There is no dynamic section in this file.
```

因為選取 Build static binary

Linux readelf 指令用來顯示 elf 檔案格式裡的資訊，常用於顯示 symbols、headers、sections、segments，這在分析編譯器如何從原始碼生成二進制檔案時非常實用。

<2>執行 chroot ~/busybox/_install/bin/ash

這裡我做不出來，問題是 ash not a directory

<3>BUSYBOX 用途

檔案系統是 Linux 不可缺少的一部份，不過在嵌入式系統上資源有限，不可能建立像 PC 那麼大的檔案系統，因此系統工具程式必須經過精簡化將檔案縮小這樣才可節省嵌入式系統的資源，現在在 Embedded Linux 上最常用的檔案系統為 Busybox。Busybox 整合各種系統工具程式 (ex : ls, mkdir, mount, ifconfig...)成為一個單一執行檔，大大縮減了系統使用容量。

<參考資料>

https://blog.csdn.net/woshishui918/article/details/83997447

Linux 下的 Kconfig defconfig .config 和 Makefile 文件之间的联系

https://www.cnblogs.com/arnoldlu/p/10905698.html

向 busybox 中添加自己的 applet

https://zh-hant.hotbak.net/key/busybox%E8%83%BD%E5%B9%B9%E5%98%9B.html

busybox 的作用和功能

https://shengyu7697.github.io/linux-readelf/

readelf 用法與範例

https://blog.xuite.net/ian11832/blogg/33493241-Linux%E5%8B%95%E6%85%8B%26%E9%9D%9C%E6%85%8B%E5%87%BD%E5%BC%8F%E5%BA%AB%E8%A7%A3%E8%AA%AA

Linux 動態&靜態函式庫解說

<補充資料>

動態連結程式庫(Shared library)是在程式開始執行時才載入的，

其優點在於

(1)減少執行檔的大小

(2)更新程式庫而無需重新編譯其他程式

(3)甚至可在程式執行時更改程式庫

靜態函式庫的話

函式庫中的元件會連結到我們的執行檔中，此時執行檔的大小會比較大，

好處是當我們在執行程式時，就不需要再函式庫的配合

<致謝>

羅 O 五老師