作業系統-新增 System Call

蘇柏豪 A1015538

環境: ubuntu 13.10 ,kernel :3.10.73

1. Add New Kernel

新增一個新的kernel到:

/usr/src

遇到困難:

/usr/src/資料屬於有權限限制，所以使用chmod 777使權限開放到最大。

1. Add the System Call to the System Call Table

(1)

arch/avr32/kernel/syscall\_table.S

arch/m32r/kernel/syscall\_table.S

arch/microblaze/kernel/syscall\_table.S

再上面三個地方新增: long sys\_a1015538

(2)在syscall\_32.tbl (32-bits system)

新增351 i386 a1015538 sys\_a1015538

在syscall\_64.tbl (64-bits system)

新增314 common a1015538 sys\_a1015538

1. Register the System Call as aMacro

arch/sh/include/uapi/asm/unistd\_32.h

arch/sh/include/uapi/asm/unistd\_64.h

include/uapi/asm-generic/unistd.h

在上面三個地方新增

#define \_\_NR\_a1015538 369 (unistd\_32.h)

後面號碼隨著你資料裡最後的號碼做變化，而且要放在#define \_\_NR\_syscalls前，而#define \_\_NR\_syscalls號碼也跟著做相對應變化。

最後一個unistd.h則多加\_\_SYSCALL (\_\_NR\_a1015538, sys\_a1015538)。

1. Add the System Call to the Syscall Interface Header

include/linux/syscalls.h

在上面添加 asmlinkage long sys\_a1015538(int len,char \*a)

1. Add the System Call to the Makefile

在新kernel目錄下的Makefile找到kernel/ mm/ fs/ ipc/ security/ crypto/ block/

修改成kernel/ mm/ fs/ ipc/ security/ crypto/ block/ a1015538/

1. Create the Source File for the System Call

在新kernel目錄底下新增一個a1015538資料夾

內部建造一個a1015538.c。

內部code:

#include <linux/kernel.h>

asmlinkage long sys\_a1015538(int len, char \*a) {

char b[len+1];

b[0]=155-a[0];

int i=1;

for(i=1;i<len;i++)

{

b[i]=105-a[i];

}

b[i]='\0';

printk(KERN\_INFO "%s\n",b);

printk("Hello DaGG.\n");

return 0;

}

建造一個Makefile

內部code:

obj-y := a1015538.o

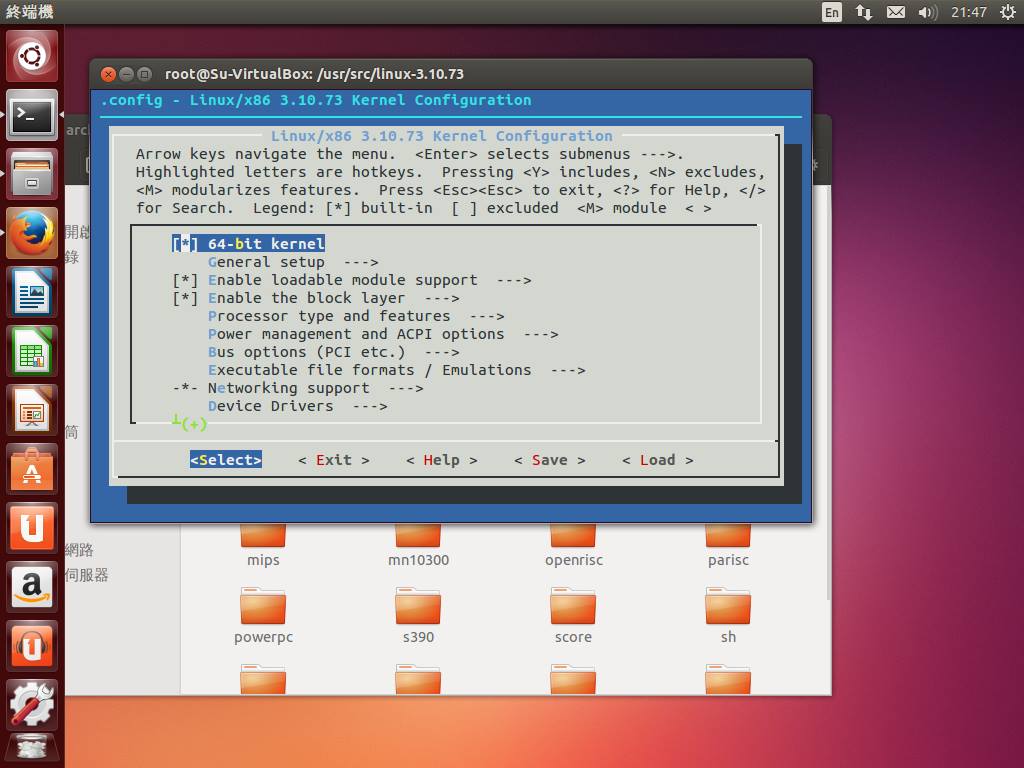
1. Build and Install the Modified Kernel(全在新kernel目錄下)

(1)製作config檔案

apt-get install libncurses-dev指令獲取正確的ncurses

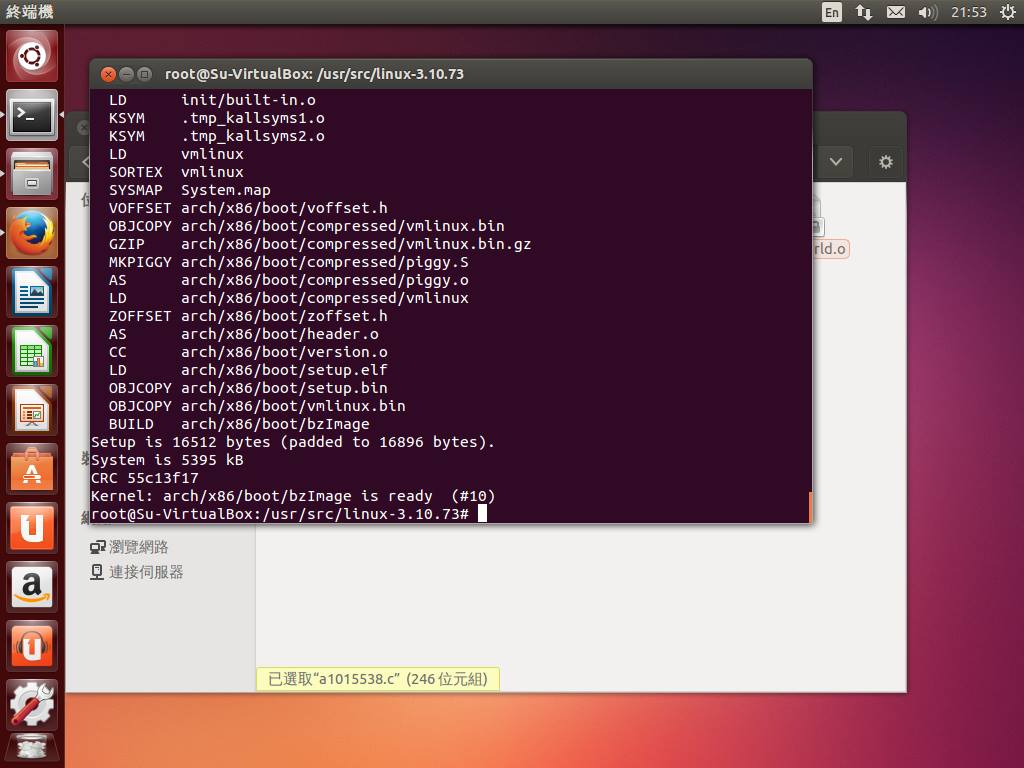
接下來輸入

make menuconfig



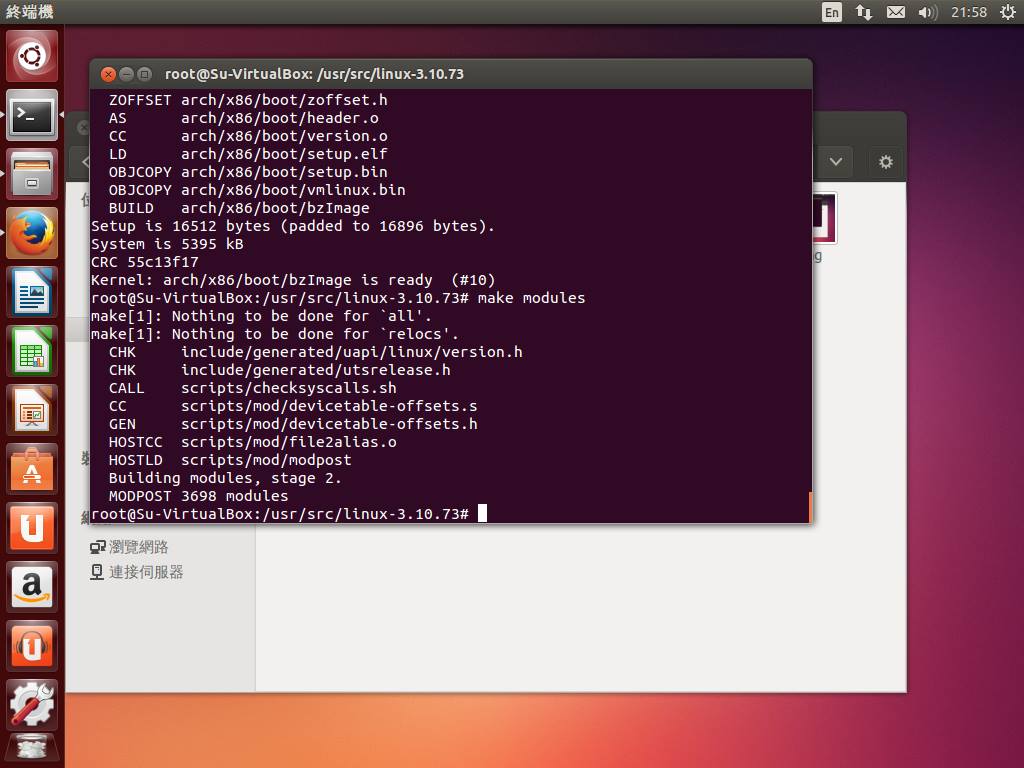
(2)編譯核心

make bzImage



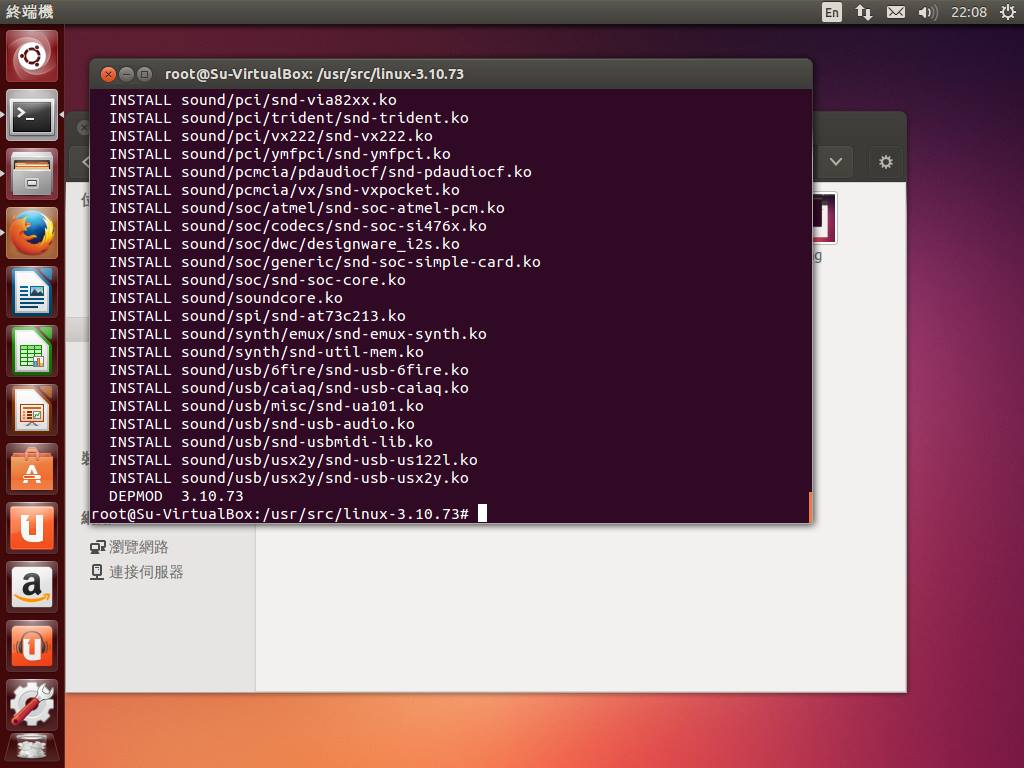
(3)編譯模組

make modules



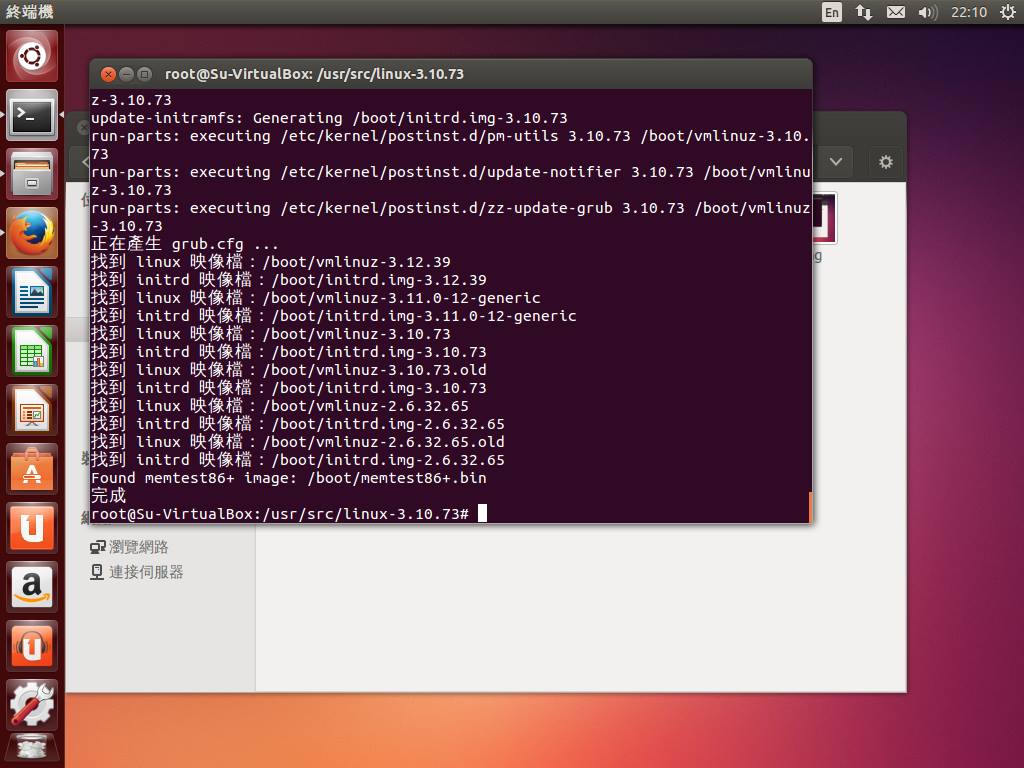
(4)模組安裝

make modules\_install



(5)移動編譯完檔案及更新新訊息

make install



1. Create a Program to Run the new System Call

在/usr/src新增一 “.c”測試檔(ex: test.c)

Code:

#include <sys/syscall.h>

#include <unistd.h>

#include <stdio.h>

#include <linux/kernel.h>

#include <stdlib.h>

#include <string.h>

int main(int argc, char \*argv[])

{

int len;

len=strlen(argv[1]);

//a = helloworld\_syscall(char); perror("msg");

printf("System call returned %ld\n", syscall(314, len, argv[1]));

//syscall(351);

return 0;

}

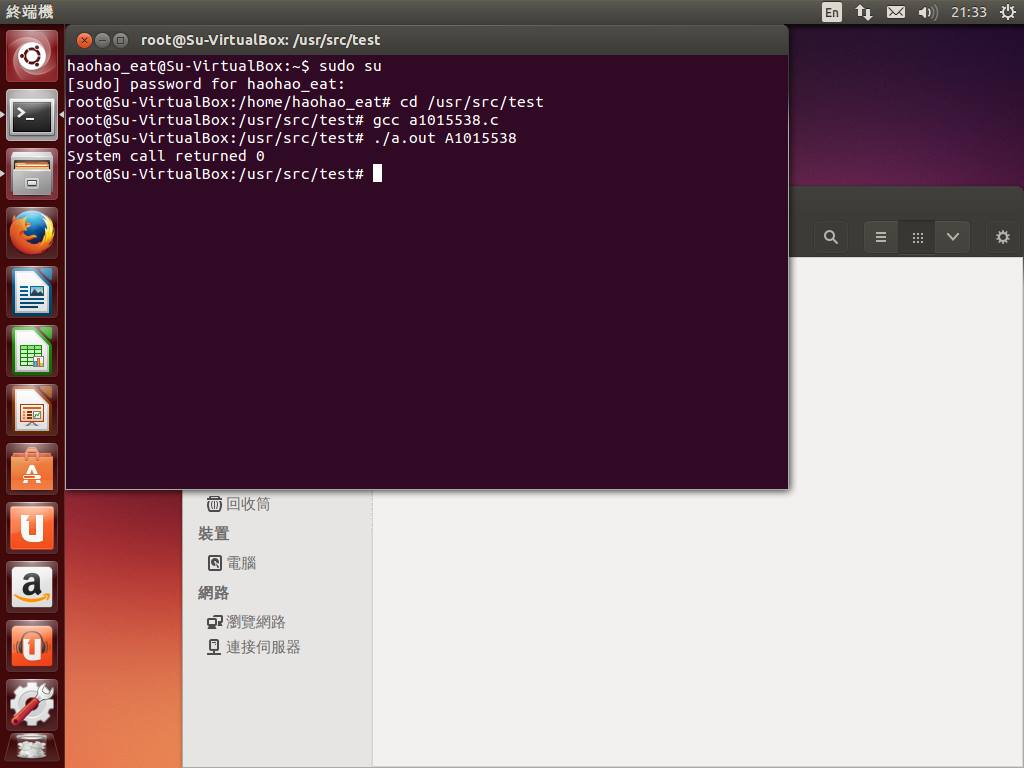
重新啟動電腦選擇新kernel(ubuntu開機時按Shift可選擇kernel)

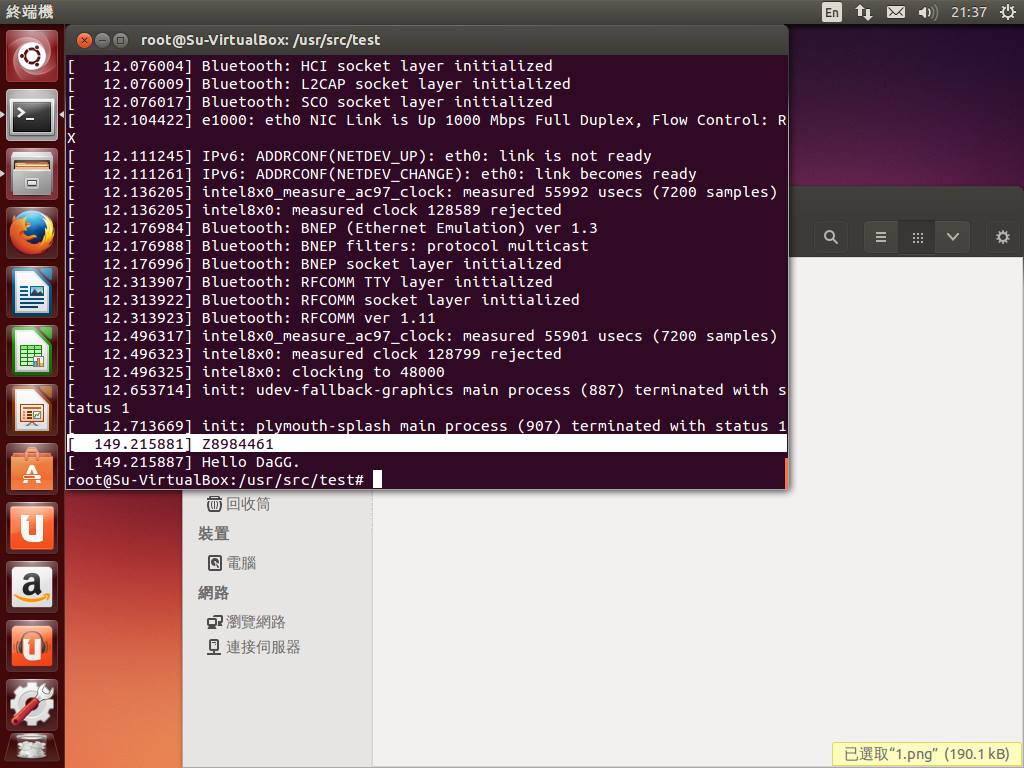
編譯:

gcc test.c

./a.out A1015538(字母設定為大寫)

顯示新System Call 輸入dmesg





心得:一半的時間都在編譯，還嘗試了4個Kernel，最後是用3.10.73版本，而找到的資料都是舊版的或不詳細，都太不成功，所以只能慢慢拼湊，新的Kernel慢慢嘗試順利裝好，但是因為裝的是64BITS而沒在相關檔案修改而呼叫不出新的system call，最後發現修改才成功，重點是還搞到linux作業系統開不起來，一開始不知道如何切換Kernel，所以亂修改開機程序，只能重灌，只是後來發現可以強制切換Kernel解救開機但是為時已晚，所還是不要亂用比較好~