

How to rebuild kernel to support PS/2 controller

By default, the PS/2 TouchKit controller connected with PS/2 auxiliary port always be directed to be as a standard PS/2 mouse device under Linux kernel 2.6.x or later. It can not be used as a char device for other devices such as touchscreen like kernel 2.4.x does. To make it possible to work with PS/2 touchscreen, it needs to rebuild the kernel for kernel later than 2.6.

Note: Some kernel version later than 2.6.15 (such as Ubuntu 6.06 or Fedora Core 6) already include the compiled kernel module *“serio_raw”*. The user can enter *“lsmod | grep serio_raw”* to check in a terminal window if this kernel module exist and be loaded or not. *If this kernel module was loaded already, it does not need to rebuild kernel. Otherwise, rebuild kernel is required.* User must append the following description into the file *“rc.local”* (It is renamed *boot.local* under SuSE Linux series) in order to support PS2 auxiliary port as a char device like kernel 2.4.x does. This file can be found in the */etc/rc.d*.

```
## SERIO_RAW section begin ##
  echo -n "serio_raw" > /sys/bus/serio/devices/serioX/drvctl
## SERIO_RAW section end ##
```

It needs user to check with serio port was for PS2 auxiliary port to assign correct above serioX value. If it is on serio0, the above setting should be

```
echo -n “serio_raw” > /sys/bus/serio/devices/serio0/drvctl
```

*The user can use **“cat /sys/bus/serio/devices/serio0/description”** to check in a terminal window. By default, the PS2 auxiliary port is assigned to serio0 under Ubuntu 6.06 and Fedora Core 6 Linux.*

*After reboot, the PS2 auxiliary port can be used as a char device like kernel 2.4.x does. **For touchscreen application, the user still needs to install the TouchKit driver to make sure touchscreen work.***

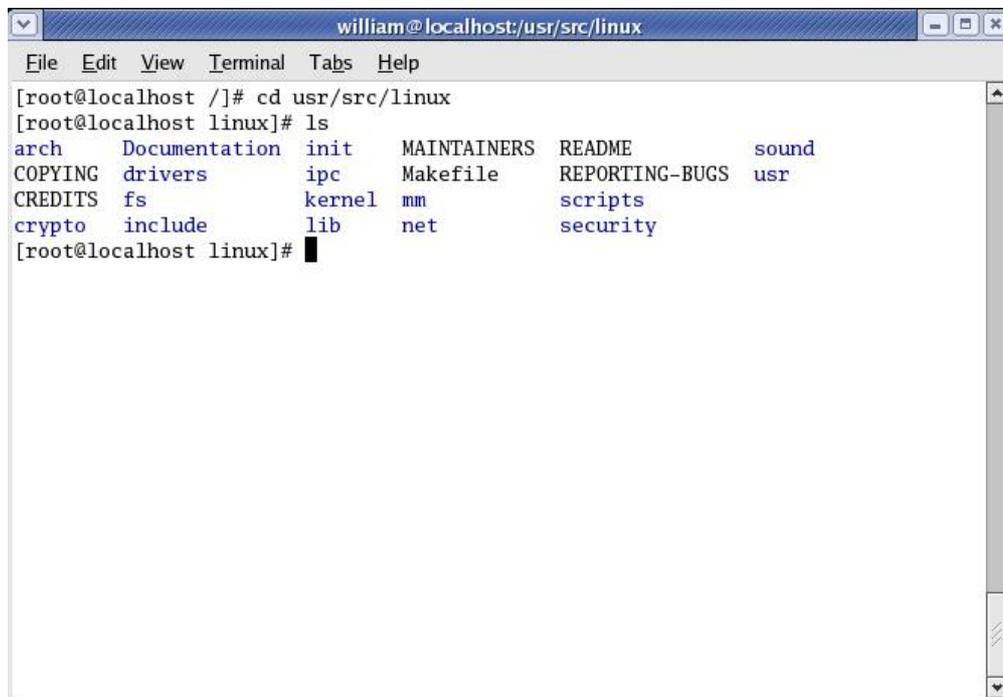
Please follow below steps to rebuild new kernel image to make it work with PS/2 TouchKit touchscreen. Below sample is based on the kernel source 2.6.9-1.667 for Fedora Core 3 Linux.

Warning: *Improper setting may cause bad performance of customized kernel.*

1. Make sure the kernel-source package is installed before you want to rebuild Linux kernel image. In addition, **the kernel-source version must be the same as your running kernel.**

The user can use **“uname -r”** instruction command to check the current kernel version and enter **“rpm -q kernel-source”** to check the kernel-source version in a terminal window.

2. Open a terminal window and change working directory to **/usr/src/linux** shown as Figure 1. **Note:** *The root permission is required to rebuild kernel.*



```
william@localhost:/usr/src/linux
File Edit View Terminal Tabs Help
[root@localhost /]# cd usr/src/linux
[root@localhost linux]# ls
arch      Documentation  init           MAINTAINERS  README        sound
COPYING  drivers        ipc           Makefile     REPORTING-BUGS  usr
CREDITS  fs             kernel        mm           scripts
crypto   include       lib           net          security
[root@localhost linux]#
```

Figure 1: kernel-source directory

3. Execute **“make mrproper”** to remove old compilation files. Then, execute **“make menuconfig”** to configure the target Linux kernel configuration shown as Figure 2.

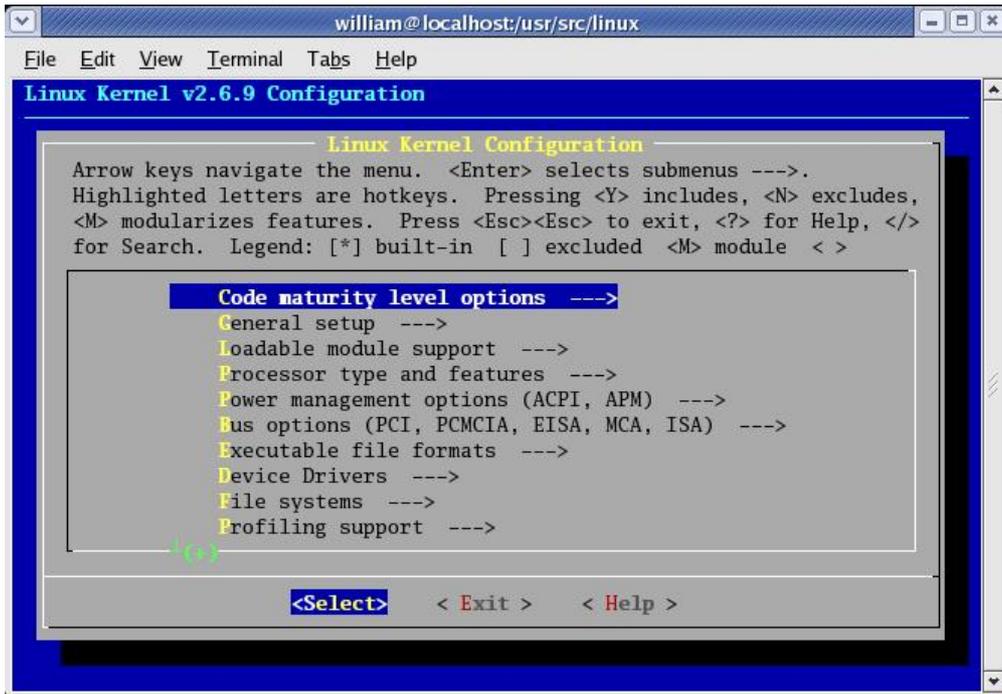


Figure 2: Target Linux kernel configuration

4. Enter submenu to [**Device Drivers -> Input device support**] shown as Figure 3.

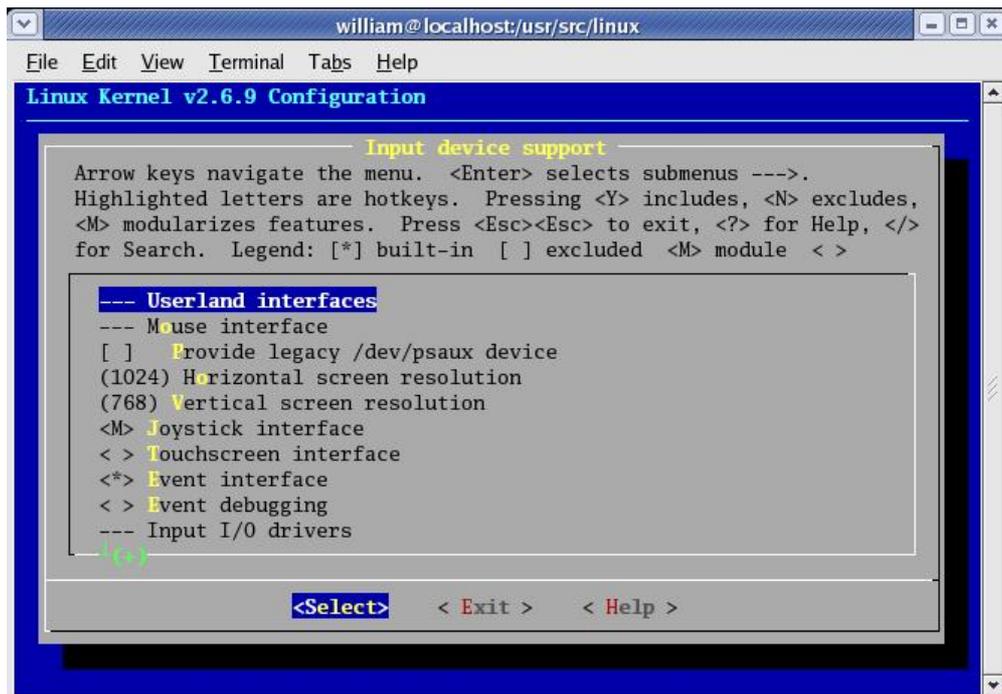


Figure 3: Input device support page

5. Uncheck [**Provide legacy /dev/psaux device**] option shown as Figure 4.

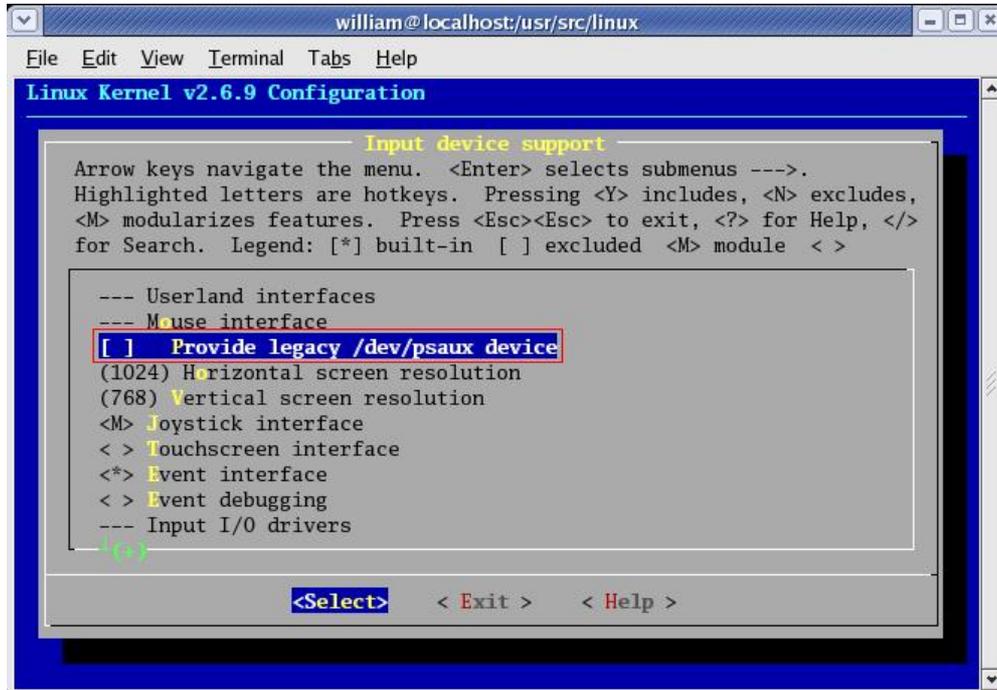


Figure 4: To uncheck [Provide legacy /dev/psaux device] option.

6. Select <*> for the option [**Raw access to serio ports**] shown as Figure 5.

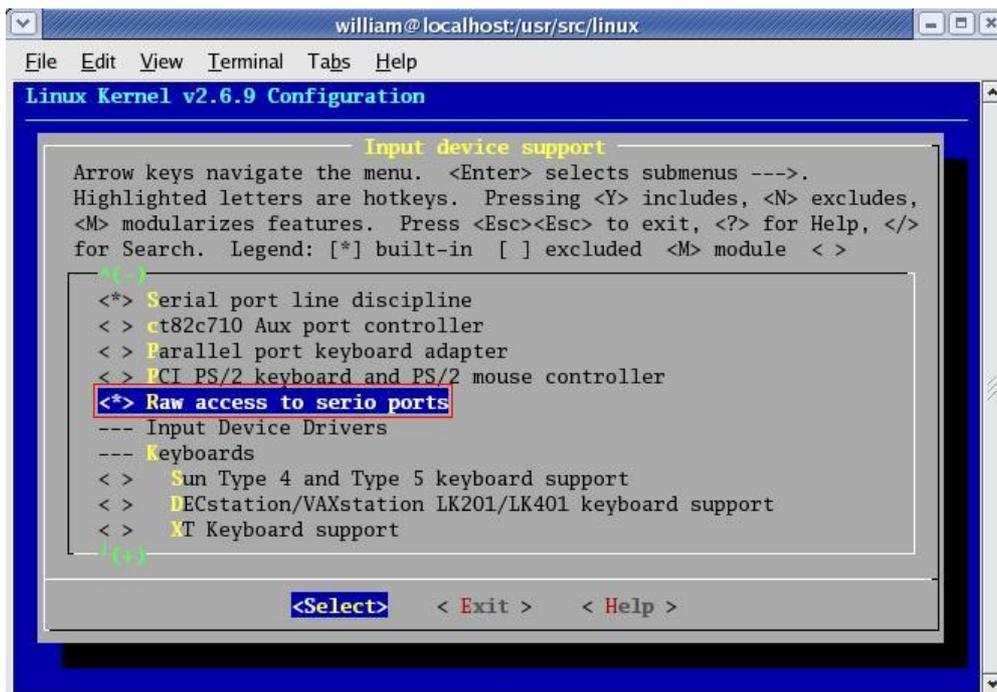


Figure 5: To select <*> for the option [Raw access to serio ports].

- Quit the Linux kernel configuration and save new kernel configuration shown as Figure 6.

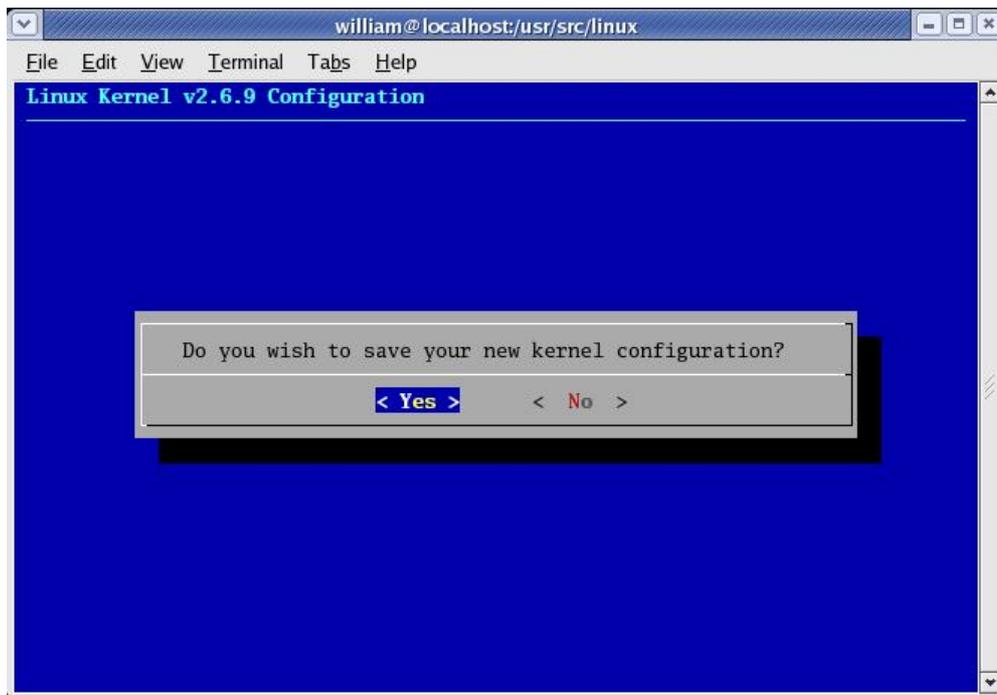


Figure 6: To save new kernel configuration.

- Execute **"make clean"** to delete old .o compilation files first. Then, execute **"make bzImage"** to build new target kernel shown as Figure 7.

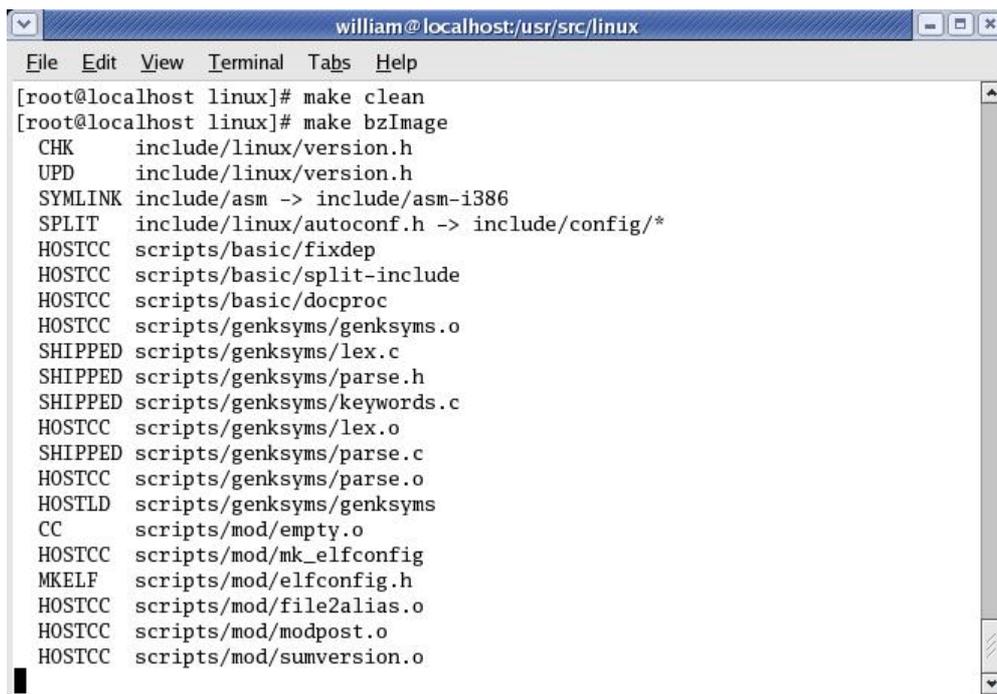
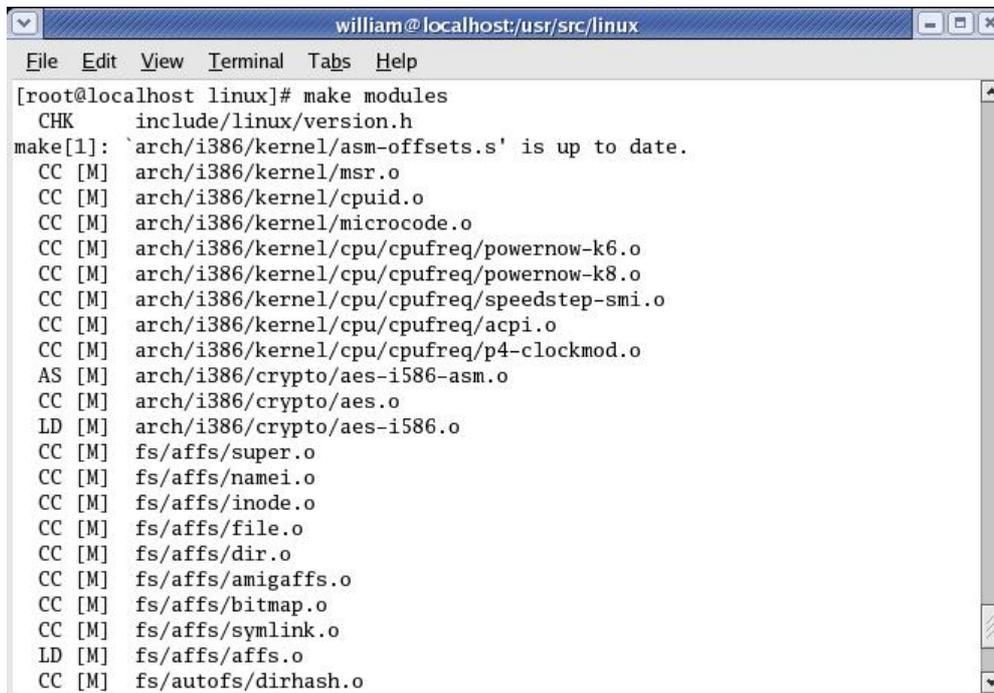


Figure 7: To build new target kernel.

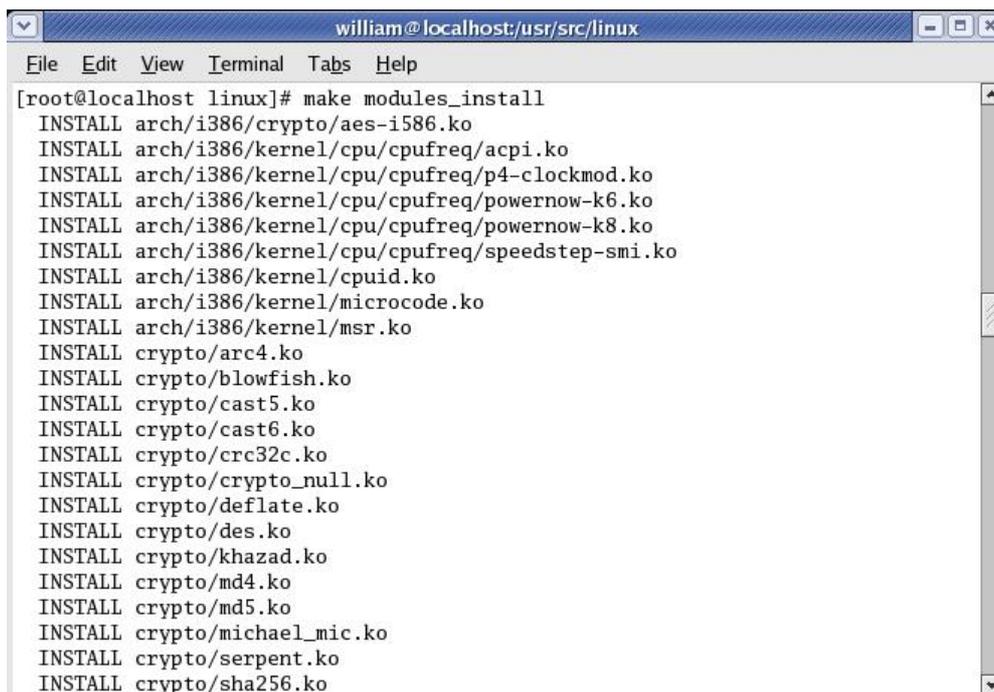
9. Execute **“make modules”** to build new modules shown as Figure 8.



```
william@localhost:/usr/src/linux
File Edit View Terminal Tabs Help
[root@localhost linux]# make modules
CHK include/linux/version.h
make[1]: `arch/i386/kernel/asm-offsets.s' is up to date.
CC [M] arch/i386/kernel/msr.o
CC [M] arch/i386/kernel/cpuid.o
CC [M] arch/i386/kernel/microcode.o
CC [M] arch/i386/kernel/cpu/cpufreq/powernow-k6.o
CC [M] arch/i386/kernel/cpu/cpufreq/powernow-k8.o
CC [M] arch/i386/kernel/cpu/cpufreq/speedstep-smi.o
CC [M] arch/i386/kernel/cpu/cpufreq/acpi.o
CC [M] arch/i386/kernel/cpu/cpufreq/p4-clockmod.o
AS [M] arch/i386/crypto/aes-i586-asm.o
CC [M] arch/i386/crypto/aes.o
LD [M] arch/i386/crypto/aes-i586.o
CC [M] fs/affs/super.o
CC [M] fs/affs/namei.o
CC [M] fs/affs/inode.o
CC [M] fs/affs/file.o
CC [M] fs/affs/dir.o
CC [M] fs/affs/amigaffs.o
CC [M] fs/affs/bitmap.o
CC [M] fs/affs/symlink.o
LD [M] fs/affs/affs.o
CC [M] fs/autofs/dirhash.o
```

Figure 8: To build new modules.

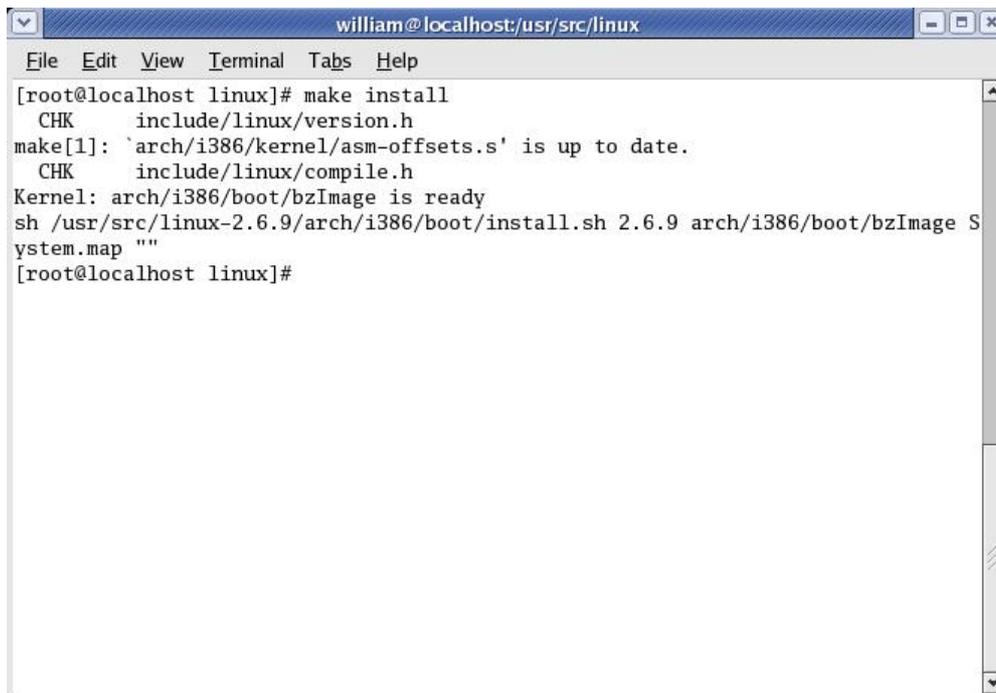
10. Execute **“make modules_install”** to install new modules into target directory **“/lib/modules/[kernel version]”** shown as Figure 9.



```
william@localhost:/usr/src/linux
File Edit View Terminal Tabs Help
[root@localhost linux]# make modules_install
INSTALL arch/i386/crypto/aes-i586.ko
INSTALL arch/i386/kernel/cpu/cpufreq/acpi.ko
INSTALL arch/i386/kernel/cpu/cpufreq/p4-clockmod.ko
INSTALL arch/i386/kernel/cpu/cpufreq/powernow-k6.ko
INSTALL arch/i386/kernel/cpu/cpufreq/powernow-k8.ko
INSTALL arch/i386/kernel/cpu/cpufreq/speedstep-smi.ko
INSTALL arch/i386/kernel/cpuid.ko
INSTALL arch/i386/kernel/microcode.ko
INSTALL arch/i386/kernel/msr.ko
INSTALL crypto/arc4.ko
INSTALL crypto/blowfish.ko
INSTALL crypto/cast5.ko
INSTALL crypto/cast6.ko
INSTALL crypto/crc32c.ko
INSTALL crypto/crypto_null.ko
INSTALL crypto/deflate.ko
INSTALL crypto/des.ko
INSTALL crypto/khazad.ko
INSTALL crypto/md4.ko
INSTALL crypto/md5.ko
INSTALL crypto/michael_mic.ko
INSTALL crypto/serpent.ko
INSTALL crypto/sha256.ko
```

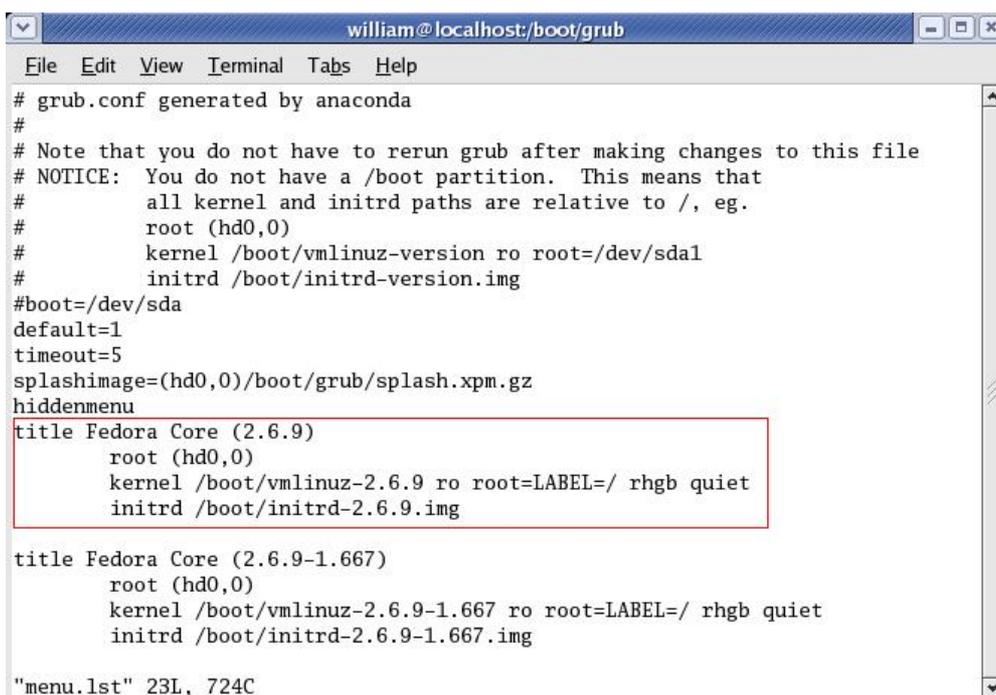
Figure 9: To install new modules.

11. Execute **“make install”** to install new kernel into target directory **“/boot”** and modify grub boot menu automatically. This grub boot menu is called **“menu.lst”** and can be found in **/boot/grub**. See Figure 11 below.



```
william@localhost:/usr/src/linux
File Edit View Terminal Tabs Help
[root@localhost linux]# make install
CHK    include/linux/version.h
make[1]: `arch/i386/kernel/asm-offsets.s' is up to date.
CHK    include/linux/compile.h
Kernel: arch/i386/boot/bzImage is ready
sh /usr/src/linux-2.6.9/arch/i386/boot/install.sh 2.6.9 arch/i386/boot/bzImage S
ystem.map ""
[root@localhost linux]#
```

Figure 10: To Install new kernel and modify grub boot menu.



```
william@localhost:/boot/grub
File Edit View Terminal Tabs Help
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE: You do not have a /boot partition. This means that
#         all kernel and initrd paths are relative to /, eg.
#         root (hd0,0)
#         kernel /boot/vmlinuz-version ro root=/dev/sda1
#         initrd /boot/initrd-version.img
#boot=/dev/sda
default=1
timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title Fedora Core (2.6.9)
    root (hd0,0)
    kernel /boot/vmlinuz-2.6.9 ro root=LABEL=/ rhgb quiet
    initrd /boot/initrd-2.6.9.img

title Fedora Core (2.6.9-1.667)
    root (hd0,0)
    kernel /boot/vmlinuz-2.6.9-1.667 ro root=LABEL=/ rhgb quiet
    initrd /boot/initrd-2.6.9-1.667.img

"menu.lst" 23L, 724C
```

Figure 11: The content of menu.lst file

Note: Please double check every option of the Linux kernel configuration according to error message if the compilation failed.

12. Modify the file of **rc.local** (It is renamed **boot.local** under SuSE Linux series.) to append below setting for PS/2 device. This file can be found in **/etc/rc.d**. Append the following description in this file.

```
## SERIO_RAW section begin ##  
    echo -n "serio_raw" > /sys/bus/serio/devices/serioX/driver  
## SERIO_RAW section end ##
```

Note: *It needs user to check with serio port was for PS2 auxiliary port to assign correct above serioX value. If it is on serio0, the above setting should be*

echo -n "serio_raw" > /sys/bus/serio/devices/**serio0**/driver

*The user can use "**cat /sys/bus/serio/devices/serio0/description**" to check in a terminal window. By default, the PS2 auxiliary port is assigned to serio0 under Fedora Core 3 Linux.*

13. Restart your system to validate the new kernel to support PS/2 touchscreen. After reboot to new kernel, the PS2 auxiliary port can be used as a char device like kernel 2.4 does. **For touchscreen application, the user still needs to install the TouchKit driver to make sure touchscreen work.**