Embedded System Programming LAB 7

Linux Kernel Compilation

Compiling the kernel is a relatively simple operation. The most recent releases of the kernel sources are available on ftp.kernel.org. This site is mirrored worldwide so there is probably a fast copy of it somewhere near you. You may be thinking "But why recompile it? It works fine as it is." There are three reasons for a recompile. Firstly, you may have some hardware that is so new that there's no kernel module for it in on your distribution CD. Secondly, you may have come across some kind of bug which is fixed in a revision of the operating system. Lastly, you may have some new software which requires a newer version of the operating system. Following is the list of commands required to compile Linux 2.6.28 kernal.

Commands to follow:

Get Linux 2.6.28 kernal source from soft folder and copy it under /usr/src directory. Extract the kernel using following command that will create folder ***linux-2.6.28.2*** under /usr/src directory.

```
ESP# tar –xvjf linux-2.6.28.2.tar.bz2

ESP# cp /boot/config-2.6.x.y.fcz.i686 /usr/src/linux-2.6.28.2/config
```

Last line copies the current configuration file to the new source. After it issue the following command to create configuration file.

Change directory to /usr/src/*linux-2.6.28.2* and issue the command.

```
ESP# make oldconfig
```

You can also user other configuration commands given below.

```
ESP# make menuconfig

ESP# make xconfig
```

After configuring you kernel next step is to check for the dependency in the kernel modules. This command builds the tree of interdependencies in the kernel sources. These dependencies may have been affected by the options you have choosen in the configure step.

```
ESP# make dep
```

Now create the image for the new kernel and build modules using following commands.

```
ESP# make bzImage

ESP# make modules
```

Next, you have to install the newly build kernel, for that you have to copy your image in to boot directory and modules in to /lib/modules. Following commands will install the kernel.

```
ESP# make install

ESP# make modules_install
```

At the next reboot, select the kernel 'new' in grub, and it will load the new kernel.

Linux Device Drivers

In the kernal compilation part we did two things created image of kernal and complied modules. The image normally file resides in the /boot directory and modules in the /lib/modules. In section we will explore the Linux Device Driver development part. Our focus is on how to write a loadable module which works as device driver. In this we will first develop the one simple module that called the helloWorld module and second we will develop the sum-module and go through keyboard device deriver.

Hello Word Module:

```
#define MODULE
#include <linux/module.h>

/* Loads a module in the kernel */

int init_module (void)
{
      printk("Hello kernel n");
      return 0;
}
/* Removes module from kernel */

void cleanup_module(void)
{
      printk("GoodBye Kerneln");
}
```

Make file for the module compilation, go to folder containing you hello.c file and use the following makefile.

```
# Makefile
Obj-m += hello.o
all:
      make –C /lib/modules/$(shell uname -r )/build M=$(PWD) modules
clean:
      make –C /lib/modules/$(shell uname -r )/build M=$(PWD) clean
```

This will generate kernal module named hello.ko. Now next task is to insert this module in the kernal, use following command for this.

```
ESP# insmod hello.ko
```

In your output locate the “Hello Kernel” and “GoodBye Kerenel” messages. If these messages don’t appear on your screen, modify the hello.c so that you can see the “Hello Kernel” and “GoodBye Kerenel” messages on your screen. Further you have to get the process ID of you running module and need to display it. Once you finish this part go for the sum and keyboard module. Follow the same steps to compile and load these modules as you did for hello module. These modules are complied for the older kernal, so it wont work with you new kernal make required modification in the source code to make it run with new kernal.

Sum-Module and Keyboard modules are provided on courses website.