Linux driver assignments ± 2 - static or module capability addition/deletion

1. use hello.c and hello1.c to generate modules and you may integrate your code into the kernel source tree in the directory of kernel source tree using the procedure described in ch8 of ELP provided as pdf.

   Please do not just copy what is given in the book. Analyse each step, modify as per requirements, add the required files to the kernel source tree. Also add a kernel parameter that you can test as needed.

   a) configure your module a module, compile the kernel and test it
   b) configure your module as static code, compile the kernel and test it.
   c) in the above cases, verify the linux kernel module and static kernel module using ªnmº and ªreadelfº.
   d) also prove that the cleanup routine of the module does work in the case of dynamically linked module code and not in the case of statically linked module code.

Linux driver assignments ± 2 - static or module capability addition/deletion

Hints for this assignment :

a) read chapter 4 of Embedded Linux primer ± specifically, section 4.3 is the relevant section for our assignment ± more specifically, do not read the other sections, now ± you may read them later
b) read chapter 8 of Embedded Linux primer ± in section 8.1.4 and further, use <KSRC>/drivers/misc/custom_test
c) in addition, do not use their example code ± let us use hello.c and hello1.c modules that we have
d) do not blindly copy what is given in chapter 8 ± section 8.1.4 ± modify appropriately as per our requirement

Linux driver assignments ± 2 - static or module capability addition/deletion

Hints for this assignment :

e) the process of doing this assignment can be summarized as below using
a traditional kernel developers© style:

```
while(1){
        step1 : add your source file(s) to the kernel source directory;
                make changes to relevant Kconfig(s) and Makefile(s)
        step2 : use make menuconfig to check whether your changes
                are updated and visible in the kernel configuration menu
                items
        step3 : select appropriate settings for your menu-item(module or
                static)
        step4 : recompile the kernel without errors ± reboot
        step5 : load the module using modprobe or check if module
                is loaded, if it is statically built into the kernel
        step6 : if above steps worked properly, break ; otherwise,
                continue
 }
```