```c
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/string.h>
#include <asm-generic/bitsperlong.h>

static int __init hello_module_init(void)
{
    char i = -1;

    char dog[10];
    char *p = &dog[1];
    unsigned long *l = (unsigned long *)p;

    int x = 1;

    memset(dog, 0, sizeof(dog));

    /* check arm core is in 32 or 64 bits */
#if BITS_PER_LONG == 32
    printk("32-bit ARM\n");
#else
    printk("64-bit ARM\n");
#endif

    /* check un-allignment access */
    *l = 1;

    /* check byte order */
    if (*(char *)&x == 1)
        printk("little endian\n");
    else
        printk("big endian\n");


    /* check char is unsigned or signed on ARM */
    if (i > 0)
        printk("char on arm is [unsigned]\n");
    else
        printk("char on arm is [signed]\n");


    return  0;
}

static void __exit hello_module_exit(void)
{
}

module_init(hello_module_init);
module_exit(hello_module_exit);
MODULE_LICENSE("GPL");
```

The output is as follow

```
root@granite2:~# insmod hello.ko
32-bit ARM
little endian
char on arm is [unsigned]
```

1. 使用BITS_PER_LONG来测试运行在32-bit / 64-bit是安全的。C standard规定了long type必须等于CPU's word.
2. 在Linux ARM上访问memory不对齐并不会crash
3. G2运行在little endian
4. ARM上char默认是unsigned的，与x86上不同。