EMBEDDED DEVICE DRIVERS

Linux Device Drivers on Beaglebone Black

LKM: IOCTL – What?

- IOCTL – *Input Output Control*
    - A system call
    - For device-specific operations
    - That cannot be clubbed as open/read/write/close

    - Examples
        - Sending an eject command to a CDROM drive
        - Changing the baud rate of a serial port
        - Lowering the volume of a speaker

- Linux kernel offers a separate mechanism
    - IOCTL call

LKM: IOCTL mechanics

- IOCTL has 2 components
  - Kernel space
    - IOCTL command has to be created in kernel space
      - Called ‘request number’
    - IOCTL function should have support in driver/module code
      - Usually handled by a switch statement
  - User space
    - User program should call this IOCTL
      - By using an IOCTL system call

- Thus, creating an IOCTL
  - Is as good as creating a new system call in Linux

LKM: IOCTL (kernel) – Step 1

- Define the IOCTL command

  #define <ioctl_name> _IOX(magic number, command number, argument type)

  - *ioctl_name*: Name of the IOCTL; used in user space
  - *IOX*: parameters
    - *IO: no parameters*
    - *IOR: read parameters (to user, hence use copy_to_user)*
    - *IOW: write parameters (from user, hence use copy_from_user)*
    - *IORW: both write and read parameters*
  - *magic number*: Unique number / character
  - *command number*: Number assigned to this IOCTL
  - *argument type*: datatype to be passed

- Header file:

  #include <linux/ioctl.h>

- Examples:

```
#define MY_HW_READ         _IOR('z', 123, unsigned int *);
#define MY_HW_WRITE        _IOW('z', 124, unsigned int *);
```

LKM: IOCTL (kernel) – Step 2

- Write the IOCTL function in the module/driver code
    - IOCTL prototype:

    long ***my_ioctl****(struct file *file, unsigned int cmd, unsigned long arg);*

    - *struct *file*: as defined earlier
    - *cmd*: Number of the command from user space
    - *arg*: argument to/from user space

- Insert this function entry in the file_operations struct

    fops->unlocked_ioctl *= my_ioctl;*

LKM: IOCTL (user) – Steps

- Step 1:
  - Define the same IOCTL command in user program
    - #define MY_HW_READ __IOX(“z”, 123, unsigned int *);

- Step 2:
  - Include (user space) header file

    #include <sys/ioctl.h>
  - Use this info in an IOCTL system call
    - System call prototype:

    int ioctl(int fd, unsigned int cmd, unsigned int *ioctl_arg);

    - *fd*: Device file descriptor
    - *cmd*: Number of the command
    - *ioctl_arg*: Argument (as defined in the #define)

LKM: IOCTL exercise

- Refer ***mod6*** directory
  - The file ***mod6.c*** contains the module code
    - We define 2 IOCTLs ***(MY_HW_READ, MY_HW_WRITE)***
    - And code in support for the same in ***my_ioctl()***
  - The file ***mod6_app.c*** contains user space code
    - We open the device created by our module
    - And issue both IOCTLs
      - ***MY_HW_WRITE*** followed by ***MY_HW_READ***
  - Compile the module and load on BBB
  - Compile the app file and get ***a.out***
  - Run ***a.out*** and observe ***dmesg*** output

THANK YOU!