Linux Kernel Programming: LED, Parameters & FIFO
Topics

1) How can the kernel drive an LED?
2) How can we pass configuration parameters to our driver when loading?
3) How can we create a queue data structure in the kernel?
Timing and LED Triggers
Delays in the Kernel

- Can delay within a system call to the kernel:
  ```c
  #include <linux/delay.h>
  int my_read(...) {
    msleep(100);
  }
  ```

  Kernel switches to other processes while waiting.

- Ex: Delay 200ms between characters
  ```c
  #include <linux/delay.h>
  int my_read(...){
    int i;
    printk(KERN_INFO);
    for (i = 0; i < count; i++) {
      printk("%c", buff[i]);
      msleep(200);
    }
    printk("\n");
  }
  ```

(Note: Poor use of `printk`!)

19-10-28
Kernel Driving LEDs

• Drivers Accessing LEDs
  – Could hard-code your driver to flash a specific LED.
  – Bad: ..

  Linux runs on virtually any hardware:
  We want to be able to re-use our drivers on any
  Beaglebone, Raspberry Pi, ...

• Decouple driver’s LED on/off decisions from a specific LEDs
  – ..
    In user space, each LED is mapped to a trigger.
    Device tree can specify default LED trigger.
Driving LEDs

- #include <linux/leds.h>
  DEFINE_LED_TRIGGER(my_trigger);

- **in init:**
  led_trigger_register_simple(
    "da-name", // New trigger name
    &my_trigger); // Previously defined trigger stuct

- **in exit:**
  led_trigger_unregister_simple(my_trigger);

- **drive LEDs:**
  led_trigger_event(my_trigger, LED_FULL);
  led_trigger_event(my_trigger, LED_OFF);
Kernel LED Demo

• demo_ledtrig.c
  - Cat led0's trigger:
    # cat /sys/class/leds/beaglebone....0/trigger
  - Load driver:
    # insmod demo_ledtrig
  - Cat led0's triggers again
  - Change trigger
    # echo demo > /sys/class/leds/beaglebone...0/trigger
  - Write text to demo_ledtrig
    # echo ‘hello world’ > /dev/demo_ledtrig
Module Parameters
Parameters to Modules

- Registration macros for each parameter
  - Place this code at the top of your module (outside of all functions)

```c
static int myvar = 123; // Give default value.
module_param(myvar, int, S_IRUGO);
MODULE_PARM_DESC(myvar, "Description here.");
```

- Validate the value at module-load in init to ensure the user-provided value is good.

- Passing value from command line
  ```
  # insmod drvname.ko myvar=50
  ```

- View parameter value
  ```
  # cat /sys/module/drvname/parameters/myvar
  ```
Module Parameter Demo

• View code: demo_paramdrv.c

• Steps:
  - Load without parameter.
    # insmod demo_paramdrv.ko
    # cat /dev/demoparam
  - Load with parameter.
    # insmod demo_paramdrv.ko myfavnumber=24

• Show:
  # cat /dev/demoparam
  # cat /sys/module/demo_paramdrv/parameters/myfavnumber
  # modinfo (or on host)
Kernel Data Structures:
FIFO Queue (kfifo)
Creating a KFifo

- Calls to driver's read/write execute in calling app's thread/process
- Can use a queue to pass data between threads/processes
  ```
  #include <linux/kfifo.h>

  // Create global variable for queue (up with globals)
  // size must be a power of 2!
  static DECLARE_KFIFO(my_fifo, char, 128);

  // Initialize from within your init function:
  // REMEMBER TO DO THIS!
  INIT_KFIFO(my_fifo);
  ```
KFifo Use

• Basic Use
  – // Insert a character:
    kfifo_put(&my_fifo, 'a');
  – // Read to populate buffer:
    kfifo_out(&my_fifo, buf, count);
  – Has functions to..
    see online documentation for kfifo.

• Concurrency
  – ..
  – Kernel locking beyond scope of this course.
KFifo Demo

• Code
  – echo.c

• Setup Viewer Shell
  # while true; do cat /dev/echo; done

• Setup Writer Shell
  # echo hello world > /dev/echo
Summary

• Use triggers to drive LEDs from the kernel
  - Have user-space setup mappings to actual LEDs (use device trees as needed)

• Create module parameters to allow your driver to be customized when it is loaded.

• Use a kfifo for a queue data structure.