driver assignment – device model and sysfs

- you must understand kobject_example1.c along with lecture notes and source code comments – first understand device model working and then , sysfs working – use dump_stack() to understand call backs of this example – you need to write user-space code similar to proc_seq_test.c to test sysfs files, call backs, and attributes !!!

- next, you must understand kset-example.c and kset-example_modified.c along with lecture notes and source code comments – first understand device model working and then, sysfs – use dump_stack() to understand call backs of this example – you need to write user-space code similar to proc_seq_test.c to test sysfs files, call backs, and attributes – you must write user-space code that is single threaded for initial testing of 3 devices and their attributes – you must write a multi threaded user-space code for further testing – this is a good opportunity to write multi threaded code – you must write re-entrant thread methods for managing 3 different devices – you can refer to thread lecture notes and examples provided Earlier

driver assignment – device model and sysfs - continued

- write an application that will use 1 thread per device – one thread writes to attributes and reads from attributes of a device – for 3 devices, you can use 3 threads

- your application must open all sysfs files in O_RDWR mode, before creating their threads – in addition, each thread must be passed appropriate argument(s) to tell which device must be handled in the respective thread (re-entrant thread functions/methods must be written)
- if your driver supports multiple devices, you must test all the devices simultaneously in your application using multi threading, as described above- meaning, open all 3 devices' sysfs files and handle at the same time in your application !!

- you may improvise your user-space code, if you have a better coding pattern – what is suggested above is just for basic user-space code using threads

---