GUJARAT TECHNOLOGICAL UNIVERSITY

ELECTRONICS & COMMUNICATION ENGINEERING (11)
DEVICE DRIVER & WRITING
SUBJECT CODE: 2181106
B.E. 8thSEMESTER

Type of course: NA

Prerequisite: Basic knowledge of Linux operating system and Embedded Systems.

Rationale: The course provide knowledge of design and implementation of device driver using linux kernel.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks						Total Marks
				Theory Marks			Practical Marks			
L	T	P	C	ESE (E)	PA (M)		ESE (V)		PA (I)	
					PA	ALA	ESE	OEP		
4	0	2	6	70	20	10	20	10	20	150

Content:

Sr. No.	Course Contents	Total Hrs	%Weightage
1	**An Introduction to Device Drivers and Char Drivers:** The Role of the Device Driver, Splitting the Kernel, Classes of Devices and Modules, Security Issues, Building and Running Modules. The Design of scull, Major and Minor Numbers, Some Important Data Structures, Char Device Registration, open and release, scull's Memory Usage, read and write.	12	25%
2	**Debugging Techniques:** Debugging Support in the Kernel, Debugging by Printing, Debugging by Querying, Debugging by Watching, Debugging System Faults, Debuggers and Related Tools	4	8%
3	**Concurrency and Race conditions:** Pitfalls in scull, Concurrency and Its Management, Semaphores and Mutexes, Completions, Spinlocks, Locking Traps, Alternatives to Locking.	4	8%
4	**Advanced Char driver operations:** Ioctl, Blocking I/O, poll and select, Asynchronous Notification, Seeking a Device, Access Control on a Device File.	3	5%
5	**Time, Delays and Deferred work:** Measuring Time Lapses, Knowing the Current Time, Delaying Execution, Kernel Timers, Tasklets, Workqueues.	3	5%
6	Allocating memory and communicating with hardware: Lookaside Caches, get_free_page and Friends, vmalloc and Friends, Per-CPU Variables, Obtaining Large Buffers, I/O Ports and I/O Memory,	6	12%

	Using I/O Ports, An I/O Port Example, Using I/O Memory.		
7	**Interrupt handling and data types in the kernel:** Preparing the Parallel Port, Installing an Interrupt Handler, Implementing a Handler, Top and Bottom Halves, Interrupt Sharing, Interrupt-Driven I/O, Use of Standard C Types, Assigning an Explicit Size to Data Items, Interface-Specific Types, Other Portability Issues, Linked Lists.	**6**	12%
8	**PCI drivers and USB drivers:** The PCI Interface, A Look Back: ISA, PC/104 and PC/104+ Other PC Buses, SBus, NuBus, External Buses, USB Device Basics, USB and Sysfs, USB Urbs, Writing a USB Driver, USB Transfers Without Urbs.	**10**	25%

Legends: R: Remembrance; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create and above Levels (Revised Bloom's Taxonomy)

Note: This specification table shall be treated as a general guideline for students and teachers. The actual distribution of marks in the question paper may vary slightly from above table.

Reference Books:

1. Linux Device Drivers by Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman, O'Reilly Media, U.S.A.
2. Linux Kernel Development by Robert Love, Pearson India.

Course Outcome:

After learning the course the students should be able to:

- Configure, compile, and install a Linux kernel/kernel module from sources
- Navigate and read the Linux kernel sources
- Understand and use correctly the Linux kernel internal services needed by devices drivers, including mutual exclusion, memory mapping, kernel memory allocation, interrupt handling, timekeeping, and scheduling.
- Design and implement a kernel module.
- Modify, or design and implement a device driver.

List of Experiments:

1. Experiments on the Driver specific Commands & Tools

- Setting up the Driver Development Environment
- Understanding Kernel's Build System
- Writing Makefile
- Adding a driver to kernel
- Writing "first" Driver
- Writing various Character Drivers
- Null Driver

- Memory Based Driver
- Multiple minors
- Module parameters
- UART (Hardware) Based Drivers
- Char drivers using UART to apply char driver concepts
- Implementing ioctl function
- Understanding the USB Device entries
- Interfacing with the USB Core
- Reading USB cdc adm class specification
- Writing a USB Driver
- USB Driver & Device Registration
- Hot-plug-ability: probe and disconnect
- Bulk Transfers & Various System Calls

2. Experiments on USB driver for peripherals

- Interrupt implementation using deferred execution
- Concurrency management using mutex, semaphores
- Memory based Block Driver

3. Experiments on Implementation a char based PCI driver

- Using kdb debugger
- Creating proc entries for a driver

Design based Problems (DP)/Open Ended Problem:

Few examples of design based problems:

1. USB card scanner driver / barcode scanner driver
2. Context sensitive file system
3. Monitoring system for each bio structure in device mapper
4. GPS Drift Correction
5. Autosync clock using GPS
6. Linux kernel module to simulate geographically large networks by introducing a lag to incoming traffic from targeted IPs.

Major Equipment: Computer with root access, Linux operating system.

List of Open Source Software/learning website:

1. Ubuntu for Linux OS

2. Linux device driver community.

ACTIVE LEARNING ASSIGNMENTS: Preparation of power-point slides, which include videos, animations, pictures, graphics for better understanding theory and practical work – The faculty will allocate chapters/ parts of chapters to groups of students so that the entire syllabus to be covered. The power-point slides should be put up on the web-site of the College/ Institute, along with the names of the students of the group, the name of the faculty, Department and College on the first slide. The best three works should submit to GTU.