-GUJARAT TECHNOLOGICAL UNIVERSITY

SUBJECT NAME : Real Time Operating System (RTOS) Design
SUBJECT CODE: 3715402
M.E. (Embedded Systems) SEM-I

Type of course: Program Elective - I

Prerequisite: Microprocessor/Microcontroller and Interfacing, Embedded System, Operating System basics and architecture

Rationale: Real Time Operating System (RTOS) Design course is essential for understanding and designing critical embedded system in automobile, medical and space application.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks				Total Marks
L	T	P	C	Theory Marks		Practical Marks		
				ESE(E)	PA (M)	PA (V)	PA (I)	
3	0	2	4	70	30	30	20	150

Content:

Sr. No.	Content	Total Hrs	% Weightage
1	**Introduction:** Overview of OS: Multirate Systems, Processes and Threads, Context Switching, Multi tasking, Cooperative Multi-tasking, Pre-emptive Operating Systems structure, Operating system function, Timing requirements on processes, Features of an Operating System	4	10%
2	**Real Time Task Scheduling:** Process state and scheduling, Clock driven and Event driven scheduling, Rate-Monotonic Scheduling, Earliest-Deadline First Scheduling, Fault-Tolerant Scheduling	4	10%
3	**Handling resource sharing and dependencies among real time tasks:** Resource sharing Protocols: Priority Inheritance Protocol, Highest locker protocol, priority ceiling protocol, Priority Inversion, Issues in resource sharing protocols	6	15%
4	**Evaluating and Optimizing Operating System Performance:** Effects of scheduling, Response-time Calculation, Interrupt latency, Time-loading, Memory Loading, Power Optimization Strategies for Processes, Advanced Configuration and Power Interface	6	15%
5	**Inter-process Communication:** Signals, Shared Memory Communication, Message-Based Communication	5	10%
6	**Memory & I/O Management:** Process Stack Management, Dynamic Allocation **I/O Management and Disk Scheduling**: Synchronous and Asynchronous I/O , Interrupt Handling, Details on Device Drivers	8	15%
7	**Examples of Real-time OS**: Features of RTOS, POSIX, Case studies: FreeRTOS. µCOS, RTx51 TinyOS, Building RTOS/EoS image for Target Hardware, Benchmarking	8	15%

	RTOS, VxWorks, CMX, Embedded Linux, Android, WinCE, Symbian		
8	**RTOS Application Domains** Comparison and study of RTOS: Vxworks and µCOS – Case studies: RTOS for Image Processing – Embedded RTOS for voice over IP – RTOS for fault Tolerant Applications – RTOS for Control Systems.	4	10%

Reference Books:

1. Kai Qian, David den Haring, Li Cao, "Embedded Software development with C", Springer
2. Wayne Wolf, "Computers as Components: Principles of Embedded Computing System Design," 2/e, Kindle Publishers, 2005.
3. Shibu K.V. "Introduction to Embedded Systems", McGraw Hill Education (India) Pvt. Ltd.
4. Tanenbaum,"Modern Operating Systems," 3/e, Pearson Edition, 2007.
5. Jean J Labrosse,"Embedded Systems Building Blocks Complete and Ready-to-use Modules in C," 2/e, 1999.
6. C.M.Krishna and G.Shin, "Real Time Systems," McGraw-Hill International Edition, 1997.
7. Philips A. Laplante, "Real-Time System Design and Analysis," Wiley-IEEE Press, 3rd Edition, 2004.
8. Rajib Mall, "Real Time System, Theory and Practice" Pearson Edition

Course Outcomes:

After learning the course the students should be able to:

- Understand Theoretical background and practical knowledge of real-time operating systems.
- Understand multitasking techniques in real-time systems.
- Understand the impact of real time operating systems on application area.
- Write programs using FreeRTOS code.

List of Experiments:

This list is general guideline. List of experiments may vary from institute to institute depending on availability of resources)

- Understanding POSIX and related functions
- POSIX thread creation, joining and destroying
- Write a FreeRTOS code to create two tasks to blink two LEDs at different rate.
- Write a FreeRTOS code to Create Two tasks Task1 & Task 2 with the priority 1 & 2 respectively
- Write a freeRTOS code which using the block state to create delay. Create delay using VTaskDelayUntil() API
- Write a freeRTOS code which is defining an Idle Task Hook Function
- Write a freeRTOS code for dynamically changing task priority
- Write a freeRTOS code for deleting task from other tasks
- Write freeRTOS code for binary semaphore to synchronize task with an interrupt
- Write freeRTOS code for counting semaphore to synchronize task with an interrupt
- Develop a linux program for the following environment: Create two thread in which 1st thread write A….Z in "data" file whereas 2nd thread write a……z in "data" file. Use the Mutex IPC to lock and unlock both the threads whenever it writes into "data" file.
- Write a freeRTOS code for ARM Cortex M3 or M4 boards that creates two tasks of the same priority and sets the time slice period to illustrate time slicing

Major Equipment:

- Linux Operating system
- ARM Development Board
- Arduino board for freeRTOS experiments

List of Open Source Software/learning website:

- NPTEL Lectures on "Real Time Systems"
- FreeRTOS Manual