17ES005 EMBEDDED PROGRAMMING

Hours Per Week :

L	T	P	C
3	-	3	5

Total Hours :

L	T	P
45	-	45

WA/RA	SSH/HSH	CS	SA	S	BS
15	30	-	5	5	-

Course Objectives:

This course is an exploration of various programming techniques and constructs used to develop reliable software systems capable of responding in real time. An overview of the platforms, tools, and processes used in developing software for embedded systems will empower you to successfully create software which consumes less memory usage. This course is highly intended for designers who want to use C language to program microcontrollers

Course Outcomes:

The student will be able to:

- Ability to do Embedded programming in C,
- Ability to understand Embedded control and applications

SKILLS :

- Trace out the problems if any through testing.
- Design fault tolerant systems.

UNIT-I

EMBEDDED PROGRAMMING: C and Assembly - Programming Style - Declarations and Expressions - Arrays, Qualifiers and Reading Numbers - Decision and Control Statements - Programming Process - More Control Statements - Variable Scope and Functions - Advanced Types – Simple Pointers - In-line Assembly.

UNIT-II

C PROGRAMMING TOOLCHAIN IN LINUX: C Preprocessor - Stages of Compilation - Debugging and Optimization- Introduction to GCC - Debugging with GDB - The Make utility - Building and Using Libraries -Profiling using gprof-Memory Leak Detection with valgrind.

UNIT-III

REAL TIME OPERATING SYSTEMS : Brief History of OS, Defining RTOS, The Scheduler, Objects, Services, Characteristics of RTOS, Defining a Task States, Scheduling, Task Operations, Structure, Synchronization, Communication and Concurrency, Defining Semaphores, Operations and Use, Defining Message Queue, States, Content, Storage, Operations and Use.

UNIT-IV

EMBEDDED C AND EMBEDDED OS : Adding Structure to 'C' Code: Object oriented programming with C, Header files for Project and Port, Examples. Meeting Real-time constraints: Creating hardware delays - Need for timeout mechanism - Creating loop timeouts - Creating hardware timeouts. Creating embedded operating system: Basis of a simple embedded OS, Introduction to sEOS, Using Timer 0 and Timer 1, Portability issue, Alternative system architecture, Important design considerations when using SEOS.

UNIT-V

Memory Management : Dynamic memory allocation, Fixed size memory management, Blocking vs Non Blocking Memory functions, Hardware memory management UNITs, synchronizing and communication. Case Studies of RTOS: VxWorks, Free RTOS.

ACTIVITIES:

- *Programming using C and C++ for different applications.*
- *Assembly Programming for real time applications*

TEXTBOOKS:

1. Jane W.S.Liu, Real Time Systems, Pearson Education.2012
2. Real Time Concepts for Embedded Systems – Qing Li, Elsevier, 2011

REFERENCE BOOKS:

1. Embedded /Real-Time Systems: concepts, Design and Programming— The Ultimate Reference, Prasad K.V.K.K, DREAMTECH PRESS, NEW DELHI . 2011
2. VxWorks Programmers Guide
3. VxWorks Reference Manual
4. Free RTOS Programmers Guide
5. Free RTOS Reference Manual

Embedded Programming Lab

Course Learning Outcomes:

- To be able to learn about various C Programming tools
- To be able to learn about Arduino in detail.

1. Introduction to C Programming tools
2. Using Standard I/O
3. Using Conditionals
4. Using Loops
5. Intro to Addresses, Pointers and Handles
6. Interfacing with Arduino
7. Arduino Digital Output
8. Arduino Digital Input
9. Arduino Analog Input
10. Arduino Reaction Timer
11. Arduino Reaction Timer Redux
12. Arduino Analog Output via PWM
13. Arduino Event Counter
14. Arduino Arbitrary Waveform Generator