Semester:	4				
Course Code:	EE2080				
Course Name:	Embedded Systems Design				
Credit Value:	3 (Notional hours: 150)				
Pre-requisites:	None				
Core/Optional	Core				
Hourly Breakdown	**Lecture**	**Tutorial**	**Practical**	**Assignment**	**Independent Learning & Assessment**
	24	3	18	18	87

Course Aim: To provide the students with the knowledge and skills on designing and implementing embedded systems.

Intended Learning Outcomes:

At the end of this course, students should be able to:

- **explain** the basic building blocks of embedded systems.
- **design** embedded systems for given applications.
- **optimize** Embedded Systems.

Course Content:

- **Basic Building blocks of embedded systems**

The concept of embedded systems, embedded system platforms; microcontroller, microprocessor and hybrid systems. Industrial embedded computers/single board computers, Different microcontroller and microprocessor architectures. Instruction set architectures.

- **Design of Embedded Systems**

Fundamental concepts in embedded systems design, tools used for modelling embedded systems, introduction to programming languages; Assembly, embedded C, Real-time programming languages (ADA, Realtime Java), timing analysis of program execution, microcontroller peripherals and interfacing, code optimization.

- **Advanced concepts in Embedded system design**

Energy management and low-power design, Real-time embedded systems, multitasking and multithreading, distributed embedded systems, networking and Internet of Things (IoT), safety and reliability in embedded systems.

- **Embedded System Design mini project**

Teaching /Learning Methods:

Lectures, Tutorials, Practical sessions, and projects

Assessment Strategy:

Continuous Assessment 50%	Final Assessment 50%		
Details:	Theory	Practical	Other (%)

Assignment 30% Practical work 20%	(%) 50	(%)	

Recommended Reading:

- Peter Marwedel, Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things
- Elecia White, Making Embedded Systems: Design Patterns for Great Software