GPU COMPUTING

II Semester: CSE								
Course Code	**Category**	**Hours / Week**			**Credits**	**Maximum Marks**		
		L	**T**	**P**	**C**	**CIA**	**SEE**	**Total**
BCSC20	**Elective**	3	0	0	3	30	70	100
Contact Classes: 45	**Total Tutorials: Nil**	**Total Practical Classes: Nil**				**Total Classes: 45**		

I. COURSE OVERVIEW:

The GPU accelerates applications running on the CPU by offloading some of the compute-intensive and time-consuming portions of the code. This course includes memory hierarchy, consistency, and debugging gpu programs.

II. COURSE OBJECTIVES:

The students will try to learn:

I. The concepts of parallel programming in problem solving.
II. The Debugging and profiling parallel programs.
III. The GPU synchronizations.

III. COURSEOUTCOMES:

After successful completion of the course, students should be able to

CO 1	Define terminology commonly used in parallel computing, such as efficiency and speedup.	Remember
CO 2	Explain common GPU architectures and programming models	Understand
CO 3	Identify efficient algorithms for common application kernels, such as matrix multiplication.	Apply
CO 4	Develop an efficient parallel algorithm to solve it.	Apply
CO 5	Identify an efficient and correct code to solve it, analyze its performance, and give convincing written and oral presentations explaining the achievements.	Apply

IV. SYLLABUS:

MODULE-I: INTRODUCTION (13)

History, Graphics Processors, Graphics Processing Units, GPGPUs. Clock speeds, CPU / GPU comparisons, Heterogeneity, Accelerators, Parallel programming, CUDA Open CL / Open ACC, Hello World Computation Kernels, Launch parameters, Thread hierarchy, Warps / Wave fronts, Thread blocks / Workgroups, Streaming multiprocessors, 1D / 2D / 3D thread mapping, Device properties, Simple Programs.

MODULE-I: MEMORY (08)

Memory hierarchy, DRAM / global, local / shared, private / local, textures, Constant Memory, Pointers, Parameter Passing, Arrays and dynamic Memory, Multi-dimensional Arrays, Memory Allocation, Memory copying across devices, Programs with matrices, Performance evaluation with different memories.

MODULE-III: SYNCHRONIZATION (08)

Memory Consistency, Barriers (local versus global), Atomics, Memory fence. Prefix sum, Reduction. Programs for concurrent Data Structures such as Work lists, Linked-lists. Synchronization across CPU and GPU

Functions: Device functions, Host functions, Kernels functions, Using libraries (such as Thrust), and

developing libraries.

MODULE-IV: SUPPORT AND STREAMS (09)

Debugging GPU Programs. Profiling, Profile tools, Performance aspects
Asynchronous processing, tasks, Task-dependence, Overlapped data transfers, Default Stream, Synchronization with streams. Events, Event-based- Synchronization - Overlapping data transfer and kernel execution, pitfalls.

MODULE-V: CASE STUDIES (05)

Image Processing, Graph algorithms, Simulations, Deep Learning.

V. TEXT BOOKS:

1. David Kirk, Wen-meiHwu, Morgan Kaufman, "Programming Massively Parallel Processors: A Hands- on Approach", 2010 (ISBN:978-0123814722).
2. Shane Cook, Morgan Kaufman "CUDA Programming: A Developer's Guide to Parallel Computing with GPUs", 2012 (ISBN:978-0124159334).

VI. REFERENCE BOOKS:

1. Dr Brian Tuomanen, "Hands-On GPU Programming with Python and CUDA", Packt, 2014.

VII. WEB REFERENCES:

1. http://www.sctie.iitkgp.ernet.in/
2. http://www.rkala.in/softcomputingvideos.php
3. http://www.sharbani.org/home2/soft-computing-1
4. http://www.myreaders.info/html/soft_computing.html

VIII. E-TEXT BOOKS:

1. https://www.books.google.co.in/books?id=bVbj9nhvHd4C
2. https://www.books.google.co.in/books?id=GrZHPgAACAAJ&dq=1.+J.S.R.Jang,+C.T.Sun+and+E
3. Mizutani,+Neuro,+Fuzzy+and+Soft+Computing,+PHI,+2004,Pearson+Education.