ADDITIONAL CONTROLLER OF EXAMINATIONS
AU
CHENNAI
600 025

Roll No. | | | | | | | | | | |

B.E (FT) END SEMESTER EXAMINATIONS – NOV / DEC 2024

Computer Science and Engineering
Seventh Semester
CS6011 GPU Computing
(Regulation 2018 - RUSA)

Time: 3 Hours | Answer ALL Questions | Max. Marks 100

PART-A (10 x 2 = 20 Marks)

1. How do streaming multiprocessors (SMs) execute multiple threads in CUDA?
2. Write a CUDA kernel along with kernel launch function to initialize an array of size 32 to all zeros in parallel.
3. Why is cudaMemcpyAsync preferred over cudaMemcpy for overlapping data transfer and computation?
4. What is APOD cycle?
5. List some of the commonly used CUDA error handling APIs.
6. How do #pragma omp critical and #pragma omp barrier differ?
7. How does MPI differ from shared memory models?
8. What is the role of MPI_Comm_rank in an MPI program?
9. Define the term 'SIMT' in GPU programming.
10. How does OpenACC handle data dependencies in loops?

PART – B (8 x 8 = 64 marks)
(Answer any 8 questions)

11. Describe in brief about GPU architecture and its components in detail.
12. Describe the execution model of a CUDA kernel and how thread blocks and warps are scheduled on the hardware.
13. Describe how chunking and parallelism are applied in multi-GPU programming.
14. Explain different memory spaces available on CUDA device along with memory constraints.
15. Describe CUDA error handling APIs and explain how they can be used for error checking.
16. Discuss the issues in parallel programming with suitable examples.
17. Discuss OpenMP data sharing and synchronization mechanisms with programs.
18. Discuss on Parallel patterns using Convolution.
19. Explain tasking in OpenMP with examples. Discuss task creation, task synchronization, and task dependencies. Include appropriate code examples.

20. Describe how MPI supports non-blocking communication.
21. Describe the role of non-default streams in CUDA programming.
22. Explain the difference between kernels and parallel constructs in OpenACC with examples.

PART – C (2 x 8 = 16marks)

23. Write a CUDA program for vector addition. Discuss how the program is parallelized using threads and blocks, and explain the process of launching the kernel.
24. Create a program using MPI_Bcast to broadcast a message from process 0 to all other processes. Explain how the message is shared across processes.