UNIVERSITI
TEKNOLOGI
PETRONAS

FINAL EXAMINATION
JANUARY 2025 SEMESTER

COURSE	**:**	**EDB4603 - DISTRIBUTED AND PARALLEL COMPUTING**
DATE	**:**	**8 APRIL 2025 (TUESDAY)**
TIME	**:**	**9.00 AM - 12.00 NOON (3 HOURS)**

INSTRUCTIONS TO CANDIDATES

1. Answer **ALL** questions in the Answer Booklet.
2. Begin **EACH** answer on a new page in the Answer Booklet.
3. Indicate clearly answers that are cancelled, if any.
4. Where applicable, show clearly steps taken in arriving at the solutions and indicate **ALL** assumptions, if any.
5. **DO NOT** open this Question Booklet until instructed.

Note :

i. There are **FIVE (5)** pages in this Question Booklet including the cover page .

ii. **DOUBLE-SIDED** Question Booklet.

iii. **Engineering Data & Formulae Booklet will be provided.**

1. Modern CPUs support parallel execution using multithreading (e.g., pthreads, OpenMP) and vectorization (e.g., SIMD instructions, automatic vectorization). These techniques improve performance by exploiting different levels of parallelism.

   a. Analyze the impact of multithreading and vectorization on modern CPU performance for different types of workloads (e.g., computationally intensive vs. memory-bound tasks). Provide a comparative discussion with real-world examples and illustrate your explanation with a diagram.

   [10 marks]

   b. Identify and discuss one advantage and one disadvantage of using behavioural modelling compared to structural modelling in Verilog.

   [5 marks]

   c. A dataset of 10 million floating-point numbers needs to be processed using OpenMP-based multithreading. Each element in the dataset must be squared and stored back in the same array. Write a C/C++ code snippet using `#pragma omp parallel for` to process the dataset efficiently. Ensure that your implementation avoids false sharing and correctly handles thread synchronization if necessary. Use an appropriate OpenMP scheduling strategy to optimize performance for different hardware architectures.

   [10 marks]

2. Amdahl's Law and Gustafson-Barsis Law are two fundamental models for analyzing the performance of parallel computing systems. Amdahl's Law focuses on the theoretical speedup limits due to the presence of a serial portion, while Gustafson-Barsis' Law provides an alternative perspective on speedup when problem size scales with the number of processors.

a. Compare Amdahl's Law and Gustafson-Barsis in terms of their role in predicting the speedup of a parallelized system.

[5 marks]

b. Given that a program spends 75% of its execution time on tasks that can be parallelized $P = 0.75$, compute the theoretical speedup using both Amdahl's Law and Gustafson-Barsis Law for $N = 8$ and $N = 32$ processors.

[10 marks]

c. Compare the results in part (c) between Amdahl's and Gustafson-Barsis. Discuss how increasing the number of processors affects speedup under each model. What does this imply for the scalability of parallel systems as N becomes very large?

[5 marks]

3. A large-scale weather simulation runs on a distributed cluster using MPI, where a 10,000 × 10,000 grid represents temperature variations over a geographic region. Each processor updates its assigned portion of the grid based on neighboring cell values and exchanges boundary data with adjacent processors in each iteration.

   a. Design an efficient parallel communication strategy for a large-scale weather simulation running on a distributed MPI cluster. Your strategy should integrate both collective operations (e.g., MPI_Reduce, MPI_Allgather) and point-to-point communication (e.g., MPI_Send, MPI_Recv) to optimize performance and scalability. Justify your design choices with respect to communication overhead, data consistency, and computational efficiency. Provide a diagram illustrating the data flow in your approach.

   [10 marks]

   b. Write a pseudocode using MPI to perform one iteration of the grid update, incorporating both collective computation and point-to-point communication.

   [15 marks]

4. Your autonomous drone's onboard vision system processes 16-Megapixel images (5000 × 3200 pixels) for real-time object detection. A 2D 7 × 7 convolution filter is applied to analyze image features. To accelerate computation, the processing is offloaded to a GPU using CUDA. The GPU has compute capability 8.0, supporting a maximum of 1024 threads per block.

a. Determine the optimal grid and block dimensions to efficiently utilize the GPU device with compute capability 8.0 for the given problem.

[5 marks]

b. Calculate the total number of multiplications performed in the 2D convolution operation for a single 16-Megapixel image.

[5 marks]

c. Implement the design by writing the CUDA kernel for the 2D convolution operation, using the function in **FIGURE Q4** below.

```
__global__ void 2D_conv(int *in, int *out, int *mask) {
    // 2D convolution code goes here
    // *in is the pointer to input image
    // *out is the pointer to the output image
    // *mask is the pointer to the 5x5 mask
    // Note that all pointers represent the 2D data
    // in 1D array format
}
```

FIGURE Q4

[20 marks]

-- END OF PAPER --