Design Document for Sync Project

1. INTRODUCTION

In this design document, we have tried to explain various design decisions and synchronizations used in the development of the project. We would cover the following topics:

The synchronization mechanism between the threads.

The synchronization mechanism between Main program and Computation program

The communication between Server and Client

2. SYNCHRONIZATION SORTING THREADS

We have implemented the parallel version of the merge sort. The sorting function takes unsorted list of the numbers and invokes number of threads to accomplish the task.

Each thread in turns call the recursively merge sort function to sort a certain section. The main thread waits for all the threads to complete.

Listing 1. Pseudo Code

```
Parallel_Sort(integer_Array, number_of_integers)
    create_thread(M)
    M_threads_process_sorting()
    wait_for_M_Threads()
    merge_M_Arrays()
    print_Sorted_output()
```

3. SYNCHRONIZATION MAIN AND SORTING

Between the main process and the forked process, we use Pipes to communicate. In order to main the queuing structure between the two process, we use semaphore based design. A shared semaphore is created and initialized with the value of Q. Before the parent process can write any command to the Pipe, it needs to acquire a semaphore. Once the child process reads the data from the pipe, it invokes the sort function. At the end of the sorting, semaphore is incremented.

Listing 2. Pseudo Code

```
parent_process:
    Acquire_semaphore()
    write_request_to_child()

Child_process:
    read_request_from_parent()
    invoke_parallel_merge()
    release_semaphore()
```

4. SERVER AND CLIENT

There is no unique synchronization required between the server and the client.