Dated :
Assessment No. : 7

Basic MPI Communication Routines

AIM

Consider the following program, called mpi_sample1.c. This program is written in C with MPI commands included.

The new MPI calls are to MPI_Send and MPI_Recv and to MPI_Get_processor_name. The latter is a convenient way to get the name of the processor on which a process is running. MPI_Send and MPI_Recv can be understood by stepping back and considering the two requirements that must be satisfied to communicate data between two processes:

1. Describe the data to be sent or the location in which to receive the data
2. Describe the destination (for a send) or the source (for a receive) of the data.

```
#include <stdio.h>
#include <string.h>
#include "mpi.h"
int main(int argc, char* argv[]){
        int  my_rank; /* rank of process */
        int  p;       /* number of processes */
        int source;   /* rank of sender */
        int dest;     /* rank of receiver */
        int tag=0;    /* tag for messages */
        char message[100];        /* storage for message */
        MPI_Status status ;   /* return status for receive */
        /* start up MPI */
        [redacted]
        /* find out process rank */
        [redacted]
        /* find out number of processes */
        [redacted]
                /* create message */
                [redacted]
                /* use strlen+1 so that '\0' get transmitted */
                [redacted]
        }
        else{
                [redacted]
                }
        }
        /* shut down MPI */
        [redacted]
        return 0;
}
```

1. Implement the above code
2. Build and Execute the logical scenario with few test cases
3. Depict the screenshots along with proper justification