**Hello world program**

nllabc2d22@nllabc2d-22:~/opt/openmpi/bin$ gedit hello.c

#include <stdio.h>

#include "mpi.h"

int main(int argc, char\* argv[])

{

int rank, size, len;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

printf("Hello, world, I am %d of %d\n",rank, size);

MPI\_Finalize();

return 0;

}

**Program to transfer data from core 0 to core 1**.

nllabc2d22@nllabc2d-22:~/opt/openmpi/bin$ gedit assi03.c

#include <stdio.h>

#include "mpi.h"

int main(int argc, char\* argv[])

{

int rank, size, len;

int num=10;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

if(rank == 0)

{

printf("Sending message containing: %d from rank %d\n", num,rank);

MPI\_Send(&num, 1, MPI\_INT, 1, 1, MPI\_COMM\_WORLD);

}

else

{

printf(" at rank %d\n",rank);

MPI\_Recv(&num, 1, MPI\_INT, 0, 1, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

printf("Received message containing: %d at rank %d\n", num,rank);

}

MPI\_Finalize();

return 0;

}

**Assignment program: Add 20 numbers in an array using 4 cores**

nllabc2d22@nllabc2d-22:~/opt/openmpi/bin$ gedit addition.c

#include <stdio.h>

#include "mpi.h"

int main(int argc, char\* argv[])

{

int rank, size;

int num[20]; //N=20, n=4

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

for(int i=0;i<20;i++)

num[i]=i+1;

if(rank == 0){

int s[4];

printf("Distribution at rank %d \n", rank);

for(int i=1;i<4;i++)

MPI\_Send(&num[i\*5], 5, MPI\_INT, i, 1, MPI\_COMM\_WORLD); //N/n i.e. 20/4=5

int sum=0, local\_sum=0;

for(int i=0;i<5;i++)

{

local\_sum=local\_sum+num[i];

}

for(int i=1;i<4;i++)

{

MPI\_Recv(&s[i], 1, MPI\_INT, i, 1, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

}

printf("local sum at rank %d is %d\n", rank,local\_sum);

sum=local\_sum;

for(int i=1;i<4;i++)

sum=sum+s[i];

printf("final sum = %d\n\n",sum);

}

else

{

int k[5];

MPI\_Recv(k, 5, MPI\_INT, 0, 1, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

int local\_sum=0;

for(int i=0;i<5;i++)

{

local\_sum=local\_sum+k[i];

}

printf("local sum at rank %d is %d\n", rank, local\_sum);

MPI\_Send(&local\_sum, 1, MPI\_INT, 0, 1, MPI\_COMM\_WORLD);

}

MPI\_Finalize();

return 0;

}