МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА
Факультет електроніки і комп’ютерних технологій

Звіт

Про виконання лабораторної роботи №5

“ Комунікації «точка-точка»: прості блоковані обміни”

Виконав:
Ст. групи ФеС-32
Молібожко Олександр

Львів 2024

Хід роботи:

Для вирішення даної задачі написано програму обміну даними між двома процесами A та B. Програма працює на основі технології MPI та забезпечує передачу вектору випадкових даних від процесу A до процесу B, далі процес B збільшує кожен елемент вектору у вказану константу x та повертає відредагований вектор процесу A. Кожен процес виводить на екран свій номер та номер процесу відправника даних.

Для компіляції програми використовується mpicc, а для її виконання - команда mpirun -np N prog, де N - кількість процесів, а prog - назва вашої програми.

Код програми:

```
decodemoli@DESKTOP-MLFCFA8:~$ cat lab5.c
#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv) {
    MPI_Init(&argc, &argv);

    int rank, size;
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    const int N = 10;
    int data[N];

    if (rank == 0) {
        for (int i = 0; i < N; i++) {
            data[i] = rand() % 100;
        }

        printf("Процес 0, надсилає вектор: ");
        for (int i = 0; i < N; i++) {
            printf("%d ", data[i]);
        }
        printf("\n");

        MPI_Send(data, N, MPI_INT, 1, 0, MPI_COMM_WORLD);

        MPI_Recv(data, N, MPI_INT, 1, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
        printf("Процес 0, отримав змінений вектор: ");
        for (int i = 0; i < N; i++) {
            printf("%d ", data[i]);
        }
        printf("\n");
    } else if (rank == 1) {
        MPI_Recv(data, N, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);

        printf("Процес 1, отримав вектор: ");
        for (int i = 0; i < N; i++) {
            printf("%d ", data[i]);
        }
        printf("\n");

        for (int i = 0; i < N; i++) {
            data[i] += 1;
        }

        MPI_Send(data, N, MPI_INT, 0, 0, MPI_COMM_WORLD);
    }

    MPI_Finalize();
    return 0;
}
```

Результат програми :

```
Процес 0, надсилає вектор: 83 86 77 15 93 35 86 92 49 21
Процес 0, отримав змінений вектор: 84 87 78 16 94 36 87 93 50 22
Процес 1, отримав вектор: 83 86 77 15 93 35 86 92 49 21
decodemoli@DESKTOP-MLFCFA8:~$
```

Висновок: У цій роботі було використано стандартну бібліотеку MPI (Message Passing Interface) для організації обміну даними між процесами, які функціонують на різних вузлах розподіленої обчислювальної системи. Завдяки MPI стало можливим створення паралельних програм, які можуть виконуватись на кластері або суперкомп'ютері, забезпечуючи розподіл обчислень між окремими процесами. Для передачі даних між процесами застосовувались функції MPI_Send та MPI_Recv.