

```
1 //-----
2 // 2B_Ping_pong_various_sends.cpp
3 //-----
4 #if (defined __linux__) || (defined _AIX)
5     #include <sys/types.h>
6     #include <sys/stat.h>
7     #include <unistd.h>
8 #elif (defined _WIN32) || (defined _WIN64)
9     #include <conio.h>
10    #include <direct.h>
11 #endif
12
13 #include "mpi.h"
14 #include <iostream>
15 using namespace std;
16
17 int main(int argc, char* argv[])
18 {
19     int i, j, rank, size, mode, first_process;
20     double time;
21     int array[10];
22
23     MPI_Request request;
24     MPI_Status status;
25
26     MPI_Init(&argc, &argv);
27
28     MPI_Comm_size(MPI_COMM_WORLD, &size);
29     MPI_Comm_rank(MPI_COMM_WORLD, &rank);
30
31     if (rank == 0)
32         cout << "There are " << size << " processes";
33
34     if (rank == 0) {
35         cout << "\n-----";
36         cout << "\nSelect a mode:\n 1. MPI_Send\n 2. MPI_Ssend (deadlock)\n"
37             << " 3. MPI_Ssend\n 4. MPI_Issend\n\n mode = ";
38         cin >> mode;
39         MPI_Ssend(&mode, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
40     }
41     else
42         MPI_Recv(&mode, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
43
44     if (rank == 0) {
45         cout << "\nSelect which process will operate as the first:\n\n"
46             << " first process = ";
47         cin >> first_process;
48         cout << "-----\n";
49         MPI_Ssend(&first_process, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
50     }
51     else
52         MPI_Recv(&first_process, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
53
```

```
54     for (i = 0; i < 10; i++)
55         array[i] = 0;
56
57     if (rank == 0) {
58         cout << "\nprocess " << rank << " - Initial values of array are:" << endl;
59         for (i = 0; i < 10; i++)
60             cout << array[i] << " ";
61         cout << endl;
62         std::fflush(stdout);
63     }
64
65     MPI_Barrier(MPI_COMM_WORLD);
66     time = -MPI_Wtime();
67
68     if (rank != first_process)
69         for (i = 0; i < 1000000; i++)           // zdrzujeme second_proces
70             for (j = 0; j < 1000000; j++)
71                 if (i <= j)
72                     i = j;
73
74     cout << "\nprocess " << rank << " is prepared" << endl;
75     std::fflush(stdout);
76
77     switch (mode) {
78     case 1:
79         for (i = 0; i < 1000000; i++) {
80             if (rank == 0) {
81                 MPI_Send(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD);
82
83                 MPI_Recv(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
84             }
85             if (rank == 1) {
86                 for (j = 0; j < 10; j++)
87                     array[j]++;
88                 MPI_Send(array, 10, MPI_INT, 0, 0, MPI_COMM_WORLD);
89
90                 MPI_Recv(array, 10, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
91             }
92         } // array[j] = 500 000, prikazy v if-och sa vykonaju hned a spolu naraz,
93         // necaka sa na synchronizaciu s druhym if-om.
94         // Ak MPI_Recv(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE)
95         // nenajde na sieti odpovedajuci MPI_Send (= matching), netrapi sa tym a
96         // ide dalej. O synchronizaciu sa musite postarat sami.
97         break;
98     case 2:
99         for (i = 0; i < 1000000; i++) {
100             if (rank == 0) {
101                 cout << "process " << rank << ": i=" << i << ", stage 0" << endl;
102                 MPI_Ssend(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD);
103                 cout << "process " << rank << ": i=" << i << ", stage 1" << endl;
104                 MPI_Recv(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
105                 cout << "process " << rank << ": i=" << i << ", stage 2" << endl;
106             }
```

```
107     if (rank == 1) {
108         for (j = 0; j < 10; j++)
109             array[j]++;
110         cout << "process " << rank << ": i=" << i << ", stage 3" << endl;
111         MPI_Ssend(array, 10, MPI_INT, 0, 0, MPI_COMM_WORLD);
112         cout << "process " << rank << ": i=" << i << ", stage 4" << endl;
113         MPI_Recv(array, 10, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
114         cout << "process " << rank << ": i=" << i << ", stage 5" << endl;
115     }
116 } // DEADLOCK !!!
117 break;
118 case 3:
119     for (i = 0; i < 1000000; i++) {
120         if (rank == 0) {
121             MPI_Ssend(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD);
122
123             MPI_Recv(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
124         }
125         if (rank == 1) { // proces 1 POCKA na zasielku od procesu 0
126             MPI_Recv(array, 10, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
127
128             for (j = 0; j < 10; j++)
129                 array[j]++;
130             MPI_Ssend(array, 10, MPI_INT, 0, 0, MPI_COMM_WORLD);
131         }
132     }
133     break;
134 case 4:
135     for (i = 0; i < 1000000; i++) {
136         if (rank == 0) {
137             MPI_Issend(array, 10, MPI_INT, 1, 0, MPI_COMM_WORLD, &request);
138             // Do_something_else_while Issend_happens();
139             // Now wait for send to complete
140             MPI_Wait(&request, &status);
141
142             MPI_Irecv(array, 10, MPI_INT, 1, 1, MPI_COMM_WORLD, &request);
143             // Do_something_else_while MPI_Irecv_happens();
144             // Now wait for receive to complete
145             MPI_Wait(&request, &status);
146         }
147         if (rank == 1) { // proces 1 POCKA na zasielku od procesu 0
148             MPI_Irecv(array, 10, MPI_INT, 0, 0, MPI_COMM_WORLD, &request);
149             // Do_something_else_while MPI_Irecv_happens();
150             // Now wait for receive to complete
151             MPI_Wait(&request, &status);
152
153             for (j = 0; j < 10; j++)
154                 array[j]++;
155             MPI_Issend(array, 10, MPI_INT, 0, 1, MPI_COMM_WORLD, &request);
156             // Do_something_else_while Issend_happens();
157             // Now wait for send to complete
158             MPI_Wait(&request, &status);
159         }
160     }
161 }
```

```
160     }
161     break;
162
163     default: break;
164 }
165
166 if (rank == 0) {
167     cout << "\nprocess " << rank << " - Final values of array are:" << endl;
168     for (i = 0; i < 10; i++)
169         cout << array[i] << " ";
170 }
171
172 time += MPI_Wtime();
173 if (rank == 0)
174     printf("\n\nTime = %8.6f s\n", time);
175
176 std::fflush(stdout);
177 MPI_Finalize();
178
179 return 0;
180 }
```