```cpp
//---------------------------------------------------------------------------
// synchronous blocking ring = DEADLOCK !!!!!!!!!!!!!!!
//---------------------------------------------------------------------------
#if (defined __linux__) || (defined _AIX) || (defined __APPLE__)
    #include <sys/types.h>
    #include <sys/stat.h>
    #include <unistd.h>
#elif (defined _WIN32) || (defined _WIN64)
    #include <conio.h>
    #include <direct.h>
#endif

#include<mpi.h>
#include<stdio.h>
#include <iostream>
using namespace std;

int main(int argc, char** argv)
{
    int i, n = 0, rank, size, dest, src, mess = 0;
    double time;

    MPI_Request* request = new MPI_Request();
    MPI_Status* status = new MPI_Status();

    MPI_Init(&argc, &argv);
    MPI_Barrier(MPI_COMM_WORLD);
    time = -MPI_Wtime();

    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    if (rank == 0) {
        printf("\nThere are %d processes.\n", size);
        mess = 333;
        cout << "Select how many times the message " << mess << " should be shifted"
            << " in the direction of a cycle of processes!\nIt should be n >= 0 and"
            << " n < " << size << ".\n  n = ";
        cin >> n;
        cout << endl;
        fflush(stdout);
    }
    MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);

    for (i = 0; i < n; i++) {
        dest = (rank + 1) % size;
        cout << "process " << rank << ": i=" << i << ", stage 0" << endl;
        MPI_Ssend(&mess, 1, MPI_INT, dest, 0, MPI_COMM_WORLD);
        cout << "process " << rank << ": i=" << i << ", stage 1" << endl;
        printf("send: %d->%d\n", rank, dest);

        src = (rank + size - 1) % size;
        MPI_Recv(&mess, 1, MPI_INT, src, 0, MPI_COMM_WORLD, status);
        printf("receive: %d->%d\n", src, rank);
    }

    printf("processor %d: sum of messages = %d\n\n", rank, mess);

    time += MPI_Wtime();
    if (rank == 0)
        printf("\nTime = %10.6f s\n", time);

    fflush(stdout);
    MPI_Finalize();

    delete status;
    delete request;

    return 0;
}
```