**Приложение А**

**СОЗДАНИЕ СРЕДЫ ДЛЯ РАБОТЫ**

**1. Установка**

Для разработки MPI-программ можно использовать различные реализации стандарта MPI. В данном приложении изложено описание работы с HPC PACK 2008 SDK. Заметим, что необязательно иметь Windows кластер или даже многоядерную/многопроцессорную рабочую станцию для разработки MPI-программ: любой настольный компьютер, который может работать с Windows XP может быть использован для разработки MPI-программ. Для корректной работы создаваемая среда для исполняемых программ требует, чтобы имя пользователя компьютера было на английском языке (проще всего создать нового) и путь к проекту не должен содержаться символы кириллицы.

**1.1. VisualStudio**

Первым шагом в подготовке к написанию MPI-программы является установка Microsoft Visual Studio. Стоит заметить, что от варианта издания Visual Studio будет зависеть дальнейшая работа с разрабатываемой MPI-программой, т.е. версии Express и Standard не поддерживают MPI-кластерную отладку, которая намного упрощает отслеживание ошибок. Но даже при её отсутствии существует возможность отлаживать MPI-программы, используя метод, который описан в данном приложении.

**1.2. HPCPack**

Для реализации кластерных MPI-программ Microsoft выпустил High Performance Computing Pack 2008 SDK. Также были разработаны Windows HPC Server 2008 или Microsoft Computer Cluster Server 2003, обеспечивающие поддержку развертывания MPI-программ. Для целей разработки MPI-программ достаточно использовать SDK.

**2. Настройка**

**2.1. Создание проекта**

Теперь, когда все установлено, необходимо создать проект. Для этого запустим VisualStudio, перейдем в меню File и выберем New→Project. Появится диалоговое окно, выберем консольное приложение, назовем его HelloWorld (желательно чтобы название не содержало пробелы), выберем папку для проекта и нажмем ОК. В следующем окне нажмем NEXT, так как следует убрать опцию Precompiled Header, это необходимо для того, чтобы в последующем можно было проще скомпилировать исходный код на другой платформе. Нажмем FINISH.

**2.2. Изменение кода**

Изменим код по умолчанию и скомпилируем программу, убедившись, что все работает правильно:

#include "stdafx.h"

#include <iostream>

using namespace std;

//int \_tmain(int argc, \_TCHAR\* argv[])

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

{

cout << "Hello World" << endl;

return 0;

}

Есть несколько причин, по которым необходимо изменить \_tmain на стандартный main. Во-первых, это было сделано для последующей совместимости на другой платформе. Во-вторых, нам позже понадобится argv как переменная типа char\*, а не TCHAR\*.

**2.3. Конфигурация проекта**

Теперь, когда программа скомпилирована и работает, нам необходимо её сконфигурировать, чтобы она включала в себя библиотеки и заголовочные файлы MPI. Выберем свойства (Properties), нажав на проект правой кнопкой в обозревателе решения (SolutionExplorer). В раскрывшемся окне выберем конфигурацию (Configuration) все конфигурации (All Configurations) и перейдем к пункту VC++ DirectoriesкатегорииConfiguration Properties. Далее в ниспадающем меню Include Directories выберем Edit. Добавим новую строку и укажем путь к директории C:\Program Files\Microsoft HPC Pack 2008 SDK\Include. Нажмём кнопку OK. Аналогичные действия проделаем для Library Directories, указав в независимости от разрядности путь C:\Program Files\Microsoft HPC Pack 2008 SDK\Lib\i386. Далее необходимо указать от чего зависит проект. Для этого перейдем в категорию Linker и выберем пункт Input. Добавим через Edit к Additional Dependencies msmpi.lib. Нажмем OK и для проверки скомпилируем программу.

**3. Запуск**

**3.1. Изменение кода**

Необходимо выполнить изменение кода, чтобы можно было проверить результат работы библиотеки. Функций, которые используются, описаны в следующих разделах:

#include "stdafx.h"

#include <iostream>

#include "mpi.h"

using namespace std;

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

{

int nTasks, rank;

MPI\_Init(&argc, &argv);

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

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

cout << "Number of threads = " << nTasks << endl

<< "Hello World from rank " << rank << endl;

MPI\_Finalize();

return 0;

}

**3.2. Запуск программы в несколько потоков**

Если теперь откомпилировать программу, она всегда будет показывать количество потоков равное 1, а ранг – 0. Для использования MPI необходимо запустить программу через приложение mpiexec.exe. Для этого запустим командную строку и перейдем к папке проекта. Введя mpiexecHelloWorld.exe мы скорее всего получим другой результат, который зависит от того какой процессор исполняет программу. Чтобы точно указать какое количество потоков мы хотим использовать, необходимо добавить спецификатор –n, т.е. mpiexec –n 8 HelloWorld.exe. Теперь программа будет выполняться в восемь потоков.

**4. Отладка**

Конечно же, запуск через командную строку не удобен и не позволяет отлаживать программы. Чтобы это исправить необходимо перейти к пункту Debugging категории ConfigurationProperties в свойствах проекта. Далее существует два способа начать отлаживать программу.

**4.1. Прикрепление к процессам**

В поле Command необходимо указать путь к программе C:\Program Files\Microsoft HPC Pack 2008 SDK\Bin\mpiexec.exe, в CommandArguments необходимо указать количество потоков и исполняемый файл, например вот так -n 8 "$(TargetPath)". Далее необходимо добавить код, который позволит остановить на время все потоки программы, и прикрепить отладчик к процессам. Добавим следующий код после MPI\_Comm\_rank (MPI\_COMM\_WORLD, &rank):

if (rank == 0)

{

cout << "Press any key" << endl;

cin.get();

}

MPI\_Barrier(MPI\_COMM\_WORLD);

Теперь, когда программа ожидает ввода пользователя можно прикрепить отладчик к запущенным процессам. Для этого запустим отладку, необходимо подтвердить намерение отлаживать запускаемую программу, далее перейдем в меню Debug и выберем пункт AttachtoProcess…, где можно указать, к какому процессу необходимо прикрепить отладчик. Далее указав точку останова в программе, можно её отлаживать. Еще очень полезно бывает видеть, какой процесс отлаживается, для этого необходимо открыть окно Debug→Windows→Processes.

**4.2. Использование MPI-кластерного отладчика**

Намного удобнее и проще использовать специально разработанный кластерный отладчик. Для его включения необходимо перейти к пункту Debugging и в ниспадающем меню Debuggertolaunch: выбрать MPIClusterDebugger. Далее в RunEnvironment можно указать узел и количество процессов. При запуске каждый процесс будет иметь свою собственную консоль.