УДК 519ю684

А. А. Романенко [1], **А. В. Снытников** [2], **И. В. Тимофеев** [3]

[1] *Новосибирский государственный университет*
ул. Пирогова, *2*, *Новосибирск*, *630090*, *Россия*

[2] *Институт вычислительной математики и математической геофизики СО РАН*
пр. Академика Лаврентьева, *6*, *Новосибирск*, *630090*, *Россия*

[3] *Институт ядерной физики СО РАН*
пр. Академика Лаврентьева, *11*, *Новосибирск*, *630090*, *Россия*

arom@ccfit.nsu.ru, *snytav@ssd.sscc.ru*, *timofeev@ngs.ru*

ТРЕХМЕРНЫЙ ГИБРИДНЫЙ КОД ДЛЯ МОДЕЛИРОВАНИЯ ГЕНЕРАЦИИ ВЫСОКОЧАСТОТНОГО ЭЛЕКТРОМАГНИТНОГО ИЗЛУЧЕНИЯ ТУРБУЛЕНТНОЙ ПЛАЗМЫ [*]

Существует необходимость моделирования излучения турбулентной плазмы в трехмерной постановке. В связи с этим разрабатывается код, позволяющий проводить трехмерное моделирование и ориентированный на использование гибридных суперЭВМ на основе GPU и ускорителей Intel Xeon Phi. Приведены результаты моделирования генерации электромагнитных волн при входе мощного электронного пучка в область, где находится плазма.

Ключевые слова: трехмерная модель, гибридные суперЭВМ, излучение, турбулентность.

Введение

Актуальность настоящей работы связана с необходимостью проведения моделирования генерации высокочастотного (терагерцового) излучения, наблюдаемого в экспериментах [1] на многопробочной магнитной ловушке ГОЛ-3 (ИЯФ СО РАН) в трехмерной постановке. Моделирование задачи в двумерной постановке проведено в 2015 году и описано в [2]. Стоит отметить, что в трехмерной постановке требуется проводить на порядки больше вычислений (количество узлов по третьей, координате, по Z должно быть более 100 в полномасштабной модели) и, таким образом существенно растут затраты по памяти (в данном случае на 2 порядка).

С точки зрения программирования актуальность работы связана с разработкой параллельного высокопроизводительного комплекса программ, использующего несколько уровней параллелизма (распараллеливание по узлам суперЭВМ, по отдельным процессорным элементам внутри узла и по отдельным ядрам процессора или ускорителя вычислений). При этой достижение высокой вычислительной производительности является принципиально важным для получения физически значимых результатов.

[*] Работа выполнена при поддержке гранта РНФ № 14-11-00485. Создание параллельной программы было поддержано грантами РФФИ № 14-07-00241, 16-07-00434, 16-01-00209.

Романенко А. А., *Снытников А. В.*, *Тимофеев И. В.* Трехмерный гибридный код для моделирования генерации высокочастотного электромагнитного излучения турбулентной плазмы // Вестн. Новосиб. гос. ун-та. Серия: Информационные технологии. 2016. Т. 14, № 3. С. 81–90.

ISSN 1818-7900. Вестник НГУ. Серия: Информационные технологии. 2016. Том 14, № 3

Математическая модель

Физическая система, состоящая из плазмы и электронного пучка, описывается системой уравнений, состоящей из уравнения Власова (отдельно для ионной и электронной компонент плазмы) и уравнений Максвелла [3]:

$$\frac{\partial f_{i,e}}{\partial t} + \boldsymbol{v}\frac{\partial f_{i,e}}{\partial \boldsymbol{r}} + \boldsymbol{F_{i,e}}\frac{\partial f_{i,e}}{\partial \boldsymbol{p}} = 0, \quad \boldsymbol{F_{i,e}} = q_{i,e}\left(\boldsymbol{E} + \frac{1}{c}[\boldsymbol{v}, \boldsymbol{B}]\right)$$

$$\boldsymbol{rot\ B} = \frac{4\pi}{c}\boldsymbol{j} + \frac{1}{c}\frac{\partial \boldsymbol{E}}{\partial t}$$

$$\boldsymbol{rot\ E} = \frac{1}{c}\frac{\partial \boldsymbol{B}}{\partial t}$$

$$\boldsymbol{div\ E} = 4\pi\rho$$

$$\boldsymbol{div\ B} = 0.$$

Здесь f – функция распределения (фазовая плотность), $\boldsymbol{r}$, $\boldsymbol{v}$, и $\boldsymbol{p}$ – координата, скорость и импульс, $\boldsymbol{E}$ и $\boldsymbol{B}$ – электрическое и магнитное поля, $\boldsymbol{j}$ – ток, ρ – плотность заряда. Эти уравнения решаются в безразмерной форме. Для перехода к безразмерным величинам используются следующие параметры:

- скорость света $c = 3\times10^{10}$ см/с;
- плотность плазмы $n_0 = 10^{14}$ см$^{-3}$;
- плазменная электронная частота $\omega_{pe} = 1{,}7\times10^{11}$ с$^{-1}$.

Граничные условия периодические по Y и Z, по X граничные условия непериодические для обеспечения вход и выход частиц пуска без создания нефизического потенциального барьера на границе.

Численные методы

Уравнение Власова решается методом частиц в ячейках [3, 4]. В рамках этого метода решаются уравнения движения модельных частиц, которые представляют собой уравнения характеристик для уравнения Власова:

$$\frac{\partial \mathbf{p}}{\partial t} = -\kappa(\mathbf{E} + [\mathbf{v}, \mathbf{B}])$$

$$\frac{\partial \mathbf{r}}{\partial t} = \mathbf{v},\ \mathbf{p} = \gamma\mathbf{v},\ \gamma^{-1} = \sqrt{\mathbf{1} - \boldsymbol{v^2}},$$

где $\boldsymbol{p}$ – импульс частицы, $\boldsymbol{E}$ и $\boldsymbol{B}$ – электрическое и магнитное поля в точке, где находится частица, $\boldsymbol{r}$ – координата частицы, $\boldsymbol{v}$ – скорость частицы, κ – отношение заряда частицы к массе, γ – релятивистский гамма-фактор (жирным шрифтом показаны векторные величины).

Моделирование прохождения пучка сквозь плазму

В данной работе рассматривается вариант непрерывного прохождения частиц пучка сквозь плазму. Поскольку реальные размеры установки (установка ГОЛ-3, ИЯФ СО РАН) [1] в рамках кинетической модели рассмотреть не представляется возможным, моделируется небольшой участок плазмы с частицами пучка. Для этого необходимо, чтобы частицы пучка входили в пространство моделирования и выходили из него. В программе на основе метода частиц это выполняется следующим образом. В левой части пространства моделирования

$$0 < x < L_x,\ L_{y1} < y < L_{y2},\ L_{z1} < z < L_{z2}$$

где L_x, L_{y1}, L_{y2}, L_{z1}, L_{z2} – границы буфера, при этом L_x подбирается экспериментально, но не более 10 % размера пространства моделирования. Размер буфера по Y и по Z выбирается из физических соображений (соотношение ширины пучка и плазмы).

В начальный момент времени частицы пучка распределены внутри буфера равномерно, так же, как и электроны и ионы плазмы, так что в целом плотность заряда равна нулю, как и ток в целом по области [5]. В дальнейшем с течением времени частицы пучка вылетают за пределы буфера, и для того, чтобы обеспечить непрерывное прохождение пучка сквозь плазму, в буфер добавляются новые частицы пучка, с распределением, аналогичным начальному. Так как на частицы пучка внутри буфера поле не действует, то распределение импульсов не меняется со временем.

Параллельная реализация

Распараллеливание выполнено методом декомпозиции расчетной области по координате Y, т. е. по направлению, перпендикулярному направлению движения электронного пучка (пучок летит вдоль X), как показано на (рис. 1).

Схематически показаны электроны пучка (голубым цветом), которые на начальном этапе движутся строго вдоль X, а также электроны и ионы плазмы, направление движения которых произвольно.

Используется смешанная эйлерово-лагранжева декомпозиция. Сетка, на которой решаются уравнения Максвелла, разделена на одинаковые подобласти по одной из координат. С каждой подобластью связана группа процессоров (в том случае, когда вычисления производятся на многоядерных процессорах, процессором для единообразия будет именоваться отдельное ядро). Далее, модельные частицы каждой из подобластей разделяются между процессорами связанной с этой подобластью группы равномерно, вне зависимости от координаты.

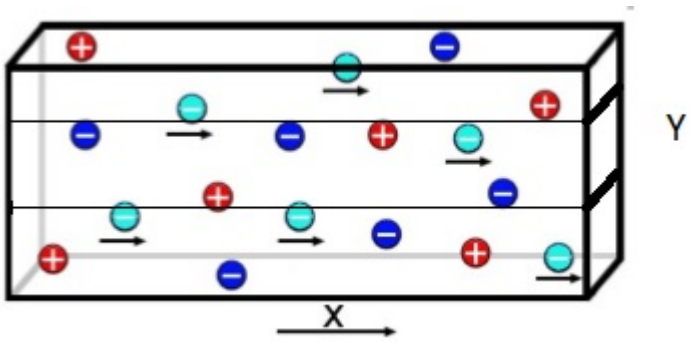


Рис. 1. Декомпозиция расчетной области (область делится по координате Y)

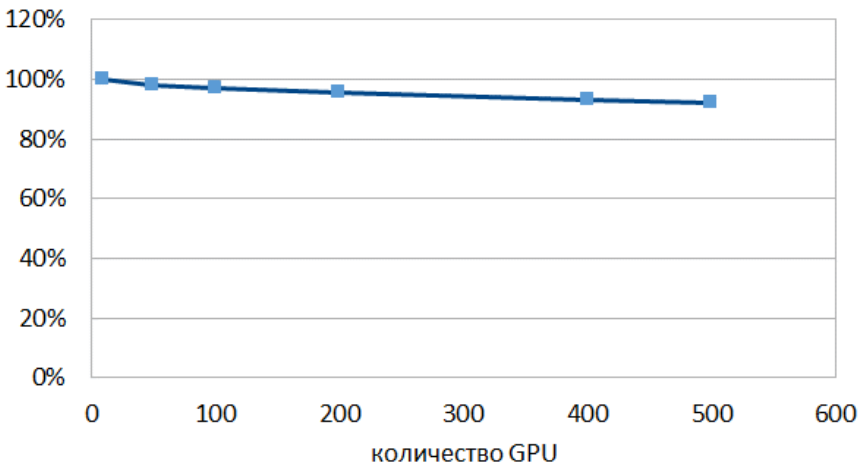


Рис. 2. Эффективность распараллеливания в слабом смысле.
Расчет проведен на суперкомпьютере «Ломоносов», НИВЦ МГУ

Каждый из процессоров группы решает уравнения Максвелла во всей подобласти. Далее решаются уравнения движения модельных частиц. После этого происходит суммирование значений тока по всей подобласти. Один из процессоров группы производит обмен граничными значениями тока и полей с соседними подобластями, и затем рассылает полученные граничные значения всем процессорам своей группы. В случае, если и уравнения Максвелла для подобласти, и уравнения движения всех частиц подобласти частиц решаются на одном процессоре, вычисления с частицами занимают в 10–20 раз больше времени. Эффективность распараллеливания в слабом смысле показана на рис.2.

Реализация на суперЭВМ на основе графических ускорителей

Программа была реализована на GPU (Graphical Processing Unit, графический процессор) с использованием технологии CUDA (Compute Uniform Device Architecture, программно-аппаратная архитектура параллельных вычислений на графических процессорах Nvidia) [6]. При этом сетка потоковых блоков CUDA соответствует вычислительной сетке, и на каждую ячейку сетки назначается один потоковый блок. Частицы обрабатываются параллельно, каждая своим отдельным потоком, если общее число частиц в ячейке не превышает число потоков в блоке, в противном случае каждый поток обрабатывает несколько частиц.

Частицы хранятся распределенными по ячейкам для ускорения доступа к ним (такой вариант обеспечивает ускорение в 2 раза по сравнению с хранением частиц в едином массиве для всей области). Но в таком случае возникает вопрос о передаче частиц для хранения в другую ячейку в случае ее перелета. При этом необходимо обойтись без синхронизации – иначе производительность катастрофически снизится. В связи с этим для передачи частиц используются буфера – в каждой ячейке по одному буферу для каждой из 26 соседних ячеек (количество соседних ячеек в одномерном случае – 2, в двумерном – 8, вместе с расположенными по диагонали, в трехмерном случае – 26), которые заполняются частицами, покидающими данную ячейку. Затем соседние ячейки независимо и одновременно считывают каждая свой буфер.

Достигнутое в результате ускорение (отношение времени счета на GPU и времени счета на Intel Xeon) показано на (рис. 3). Расчет произведен на графическом ускорителях Nvidia Kepler K40 Nvidia Tesla C2090 в сравнении с 4 ядрами процессора Intel Xeon.

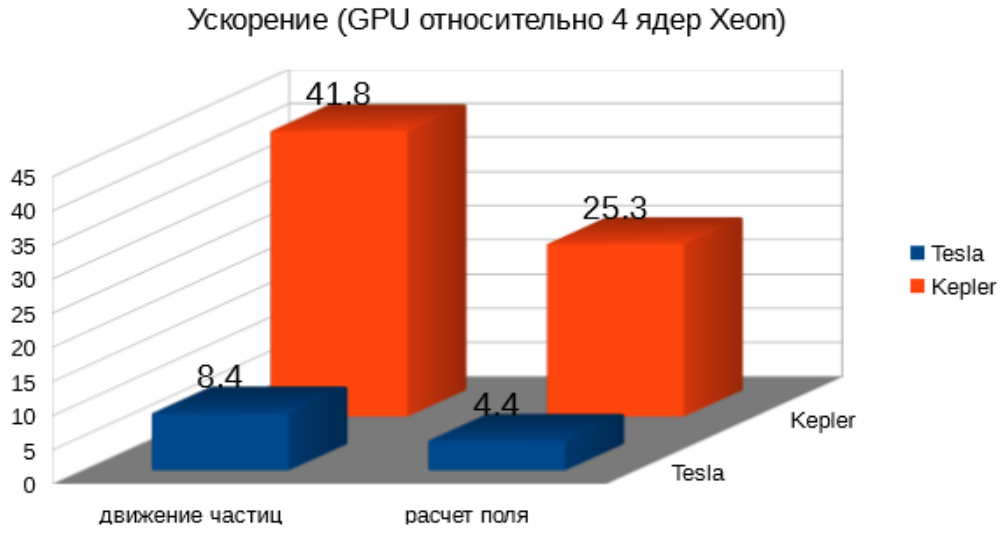


Рис. 3. Ускорение счета на графических ускорителях Nvidia Kepler K40 и Nvidia Tesla C2090 по сравнению с 4 ядрами процессора Intel Xeon

Переход на суперЭВМ на основе ускорителей Intel Xeon Phi

Некоторые из наиболее мощных суперЭВМ (в частности, № 2 в списке Top500 за ноябрь 2016 г. [1]) в настоящее время оснащаются ускорителями Intel Xeon Phi [7]. Таким образом, необходимо обеспечить возможность счета и на таких машинах. Есть несколько вариантов реализации программ под разные архитектуры. Мы решили сохранить возможность использовать разные ускоритель при минимальной модификации кода. Для этого мы выделили из разработанной на предыдущем этапе программы код, не зависящий от графического процессора, в один модуль, а весь специфический код – в другой. Архитектурно-зависимые фрагменты были оформлены в виде функций со специальной сигнатурой. Рассматривается перенос программы с архитектуры Nvidia CUDA на архитектуру Intel MIC (англ. Intel Many Integrated Core Architecture – архитектура многоядерной процессорной системы, разработанная Intel) и не рассматривается на данный момент вопрос оптимизации под ту или иную архитектуру.

Основные проблемы переноса с архитектуры CUDA на архитектуру MIC заключаются в следующем.

1. Необходимо обеспечить компиляцию функций, оформленных, как ядра CUDA и в особенности вызовов таких функций без компилятора Nvidia. Это можно сделать, например, с помощью директив условной компиляции, как будет показано ниже.
2. Операции копирования между различными видами памяти в CUDA должны быть пропущены при компиляции на MIC.
3. Необходимо определить типы данных и ключевые слова, входящих в расширение языка C, используемое в CUDA.

Основной принцип предлагаемой методики переноса программ: сделать такой непереносимый (т.е. пригодный для компиляции только с помощью компилятора Nvidia) участок кода (запуск ядра CUDA) по крайней мере единственным. Для это оформляется специальная процедура запуска вычислительных функций (рис. 4). Вычислительная функция передается этой процедуре в качестве параметра, также передается размерность сетки потоковых блоков и потокового блока технологии CUDA. Далее, в случае компиляции компилятором Nvidia, происходит запуск универсального ядра, которому опять же в качестве параметра передается вычислительная функция. Таким образом, в программе есть только одно ядро CUDA. Если же компиляция проводится с помощью другого компилятора (например, Intel) то переданная в качестве параметра процедурная переменная (вычислительная функция) вызывается внутри 6-мерного цикла, количество итераций которого определяется размерностью сетки потоковых блоков и потокового блока, как показано на (рис. 4).

Предложенный подход был реализован и протестирован на многоядерных процессорах Intel Xeon с использованием OpenMP и на ускорителях Intel Xeon Phi, результаты сравнения производительности показаны на рис. 5.

В данном случае рассматривается только лишь время счета движения модельныз частиц. Это наиболее затратная часть вычислительного алгоритма. В приведенном расчете использовано 6,4 млн модельных частиц, то есть в среднем 3,5–4 тыс. частиц на ячейку.

На рис. 5 видно, что расчет на графическом ускорителе Nvidia Kepler происходит значительно быстрее. Это может объясняться недостаточным на настоящий момент уровнем оптимизации кода для процессора Intel Xeon и для ускорителя Intel Xeon Phi. Как видно из рис. 4, никаких оптимизаций для OpenNP не использовано.

Можно также обратить внимание на сравнение времени работы для процессора Intel Xeon и для ускорителя Intel Xeon Phi. Ожидается, что расчет на Intel Xeon Phi будет происходить значительно быстрее. Тем не менее это не обязательно так, и обычно это требует серьезной дополнительной оптимизации именно Intel Xeon Phi, как описано, например, Nakashima.

Относительно показателей, приведенных на рис. 5, можно сказать, что время расчета движения частиц отличается приблизительно в три раза для Intel Xeon и Intel Xeon Phi, что вполне объяснимо с той точки зрения, что Intel Xeon имеет 6 ядер, а Intel Xeon Phi 61 ядро. При этом производительность одного ядра Intel Xeon Phi значительно меньше, чем одного

[1] Top500. Список 500 наиболее мощных компьютеров мира (ноябрь 2016). https://www.top500.org/lists/2016/11/

ядра Intel Xeon. Таким образом, предельное ускорение (как отношение числа ядер), при переходе с Intel Xeon на Intel Xeon Phi будет 10, с учетом более слабых ядер можно получить ускорение в 3–5 раз, что и было получено.

Также было проведено тестирование масштабируемости программы на кластере RSC Petastream в МСЦ РАН, результат показан на рис. 6. Видно, что в целом программа хорошо масштабируется, локальный максимум, видимый на графике при 5 ускорителях, может объясняться тем, что система на тот момент работала в тестовом режиме.

```
int Kernel_Launcher(Cell<Particle> **cells, KernelParams *params,
    unsigned int grid_size_x, unsigned int grid_size_y, unsigned int grid_size_z,
    unsigned int block_size_x, unsigned int block_size_y, unsigned int block_size_z,
    SingleNodeFunctionType h_snf)
{
#ifdef __CUDACC__
    dim3 blocks(grid_size_x, grid_size_y, grid_size_z),
        threads(block_size_x, block_size_y, block_size_z);

    GPU_Universal_kernel<<<blocks,threads>>>(cells,params,h_snf);
#else

#pragma omp parallel for
    for(int i = 0;i < grid_size_x;i++)
    {
#pragma omp parallel for
        for(int j = 0;j < grid_size_y;j++)
        {
#pragma omp parallel for
            for(int k = 0;k < grid_size_z;k++)
            {
#pragma omp parallel for
                for(int i1 = 0;i1 < grid_size_x;i1++)
                {
#pragma omp parallel for
                    for(int j1 = 0;j1 < grid_size_y;j1++)
                    {
#pragma omp parallel for
                        for(int k1 = 0;k1 < grid_size_z;k1++)
                        {
                            h_snf(cells,params,i,j,k,i1,j1,k1);
                        }
                    }
                }
            }
        }
    }
#endif
```

Рис. 4. Универсальная процедура запуска вычислительных функций

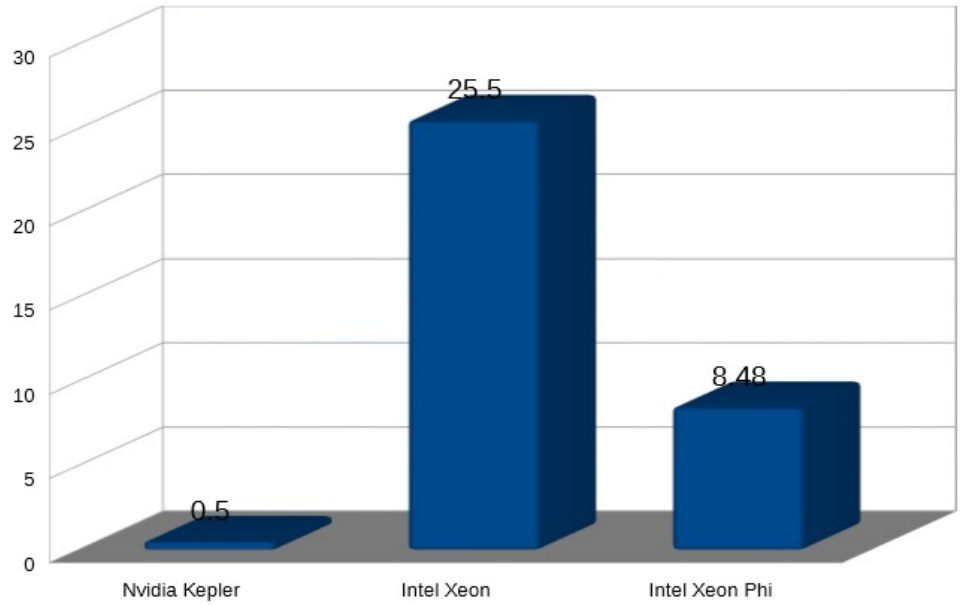


Рис. 5. Время (с) расчета движения модельных частиц в течение одного временного шага на графическом ускорителе Nvidia Kepler, на процессоре Intel Xeon и на ускорителе Intel Xeon Phi

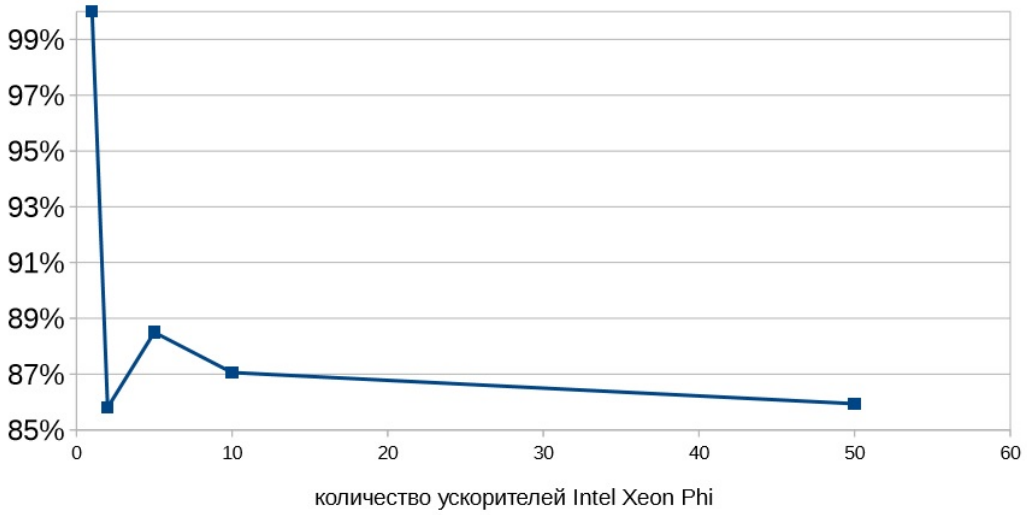
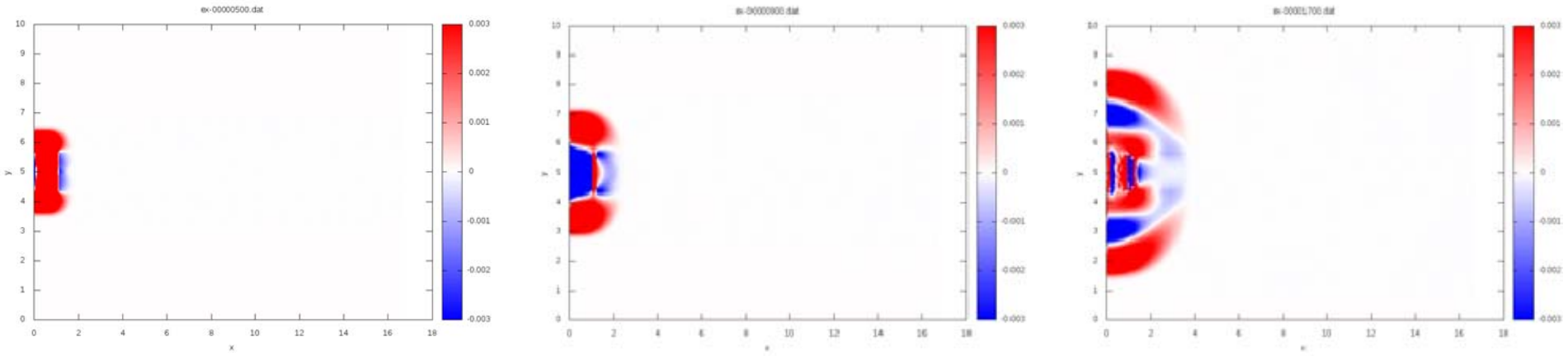


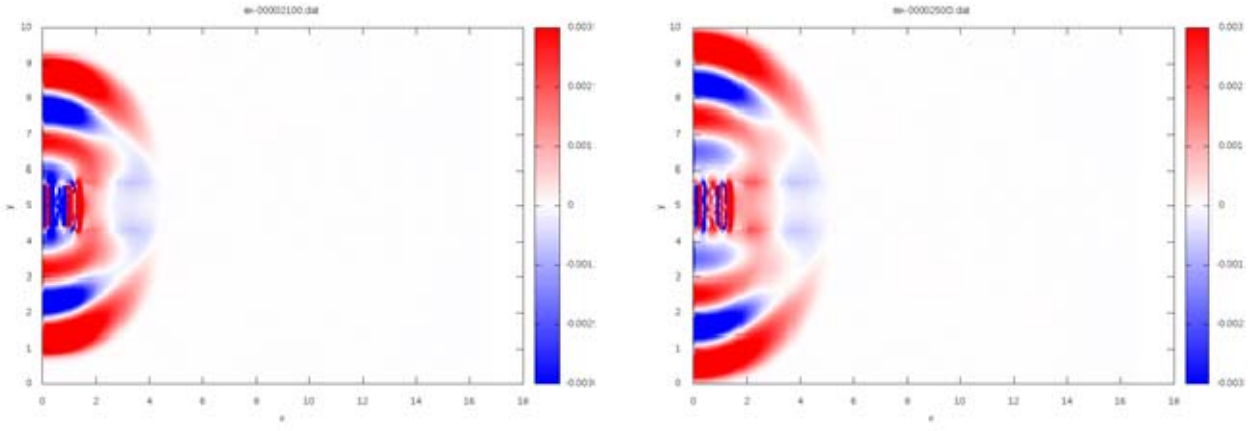
Рис. 6. Эффективность распараллеливания в слабом смысле с использованием Intel Xeon Phi. Расчет проведен на суперкомпьютере RSC Petastream, МСЦ РАН

Вычислительные эксперименты по моделированию излучения в системе «плазма – электронный пучок»

В работах [1, 2, 8] было показано наличие электромагнитного излучения турбулентной плазмы в терагерцовом диапазоне в экспериментах на установке ГОЛ-3 (ИЯФ СО РАН). Математическое моделирование излучения турбулентной плазмы представляет собой сложную вычислительную задачу, в частности в силу того, что требуется расчетная сетка с разрешением от 2000 × 500 × 500 узлов с количеством модельных частиц не менее 100 в каждой ячейке.



а б в



г д

Рис. 7. Поле E_x в различные моменты времени:
а – $t = 0{,}5$; *б* – $t = 1{,}0$; *в* – $t = 1{,}7$; *г* – $t = 2{,}1$; *д* – $t = 2{,}5$

При этом актуальным является именно трехмерное моделирование в силу того, что двумерное активно проводится и до известной степени исчерпало свои возможности. Проведенное в данной работе моделирование позволило получить результат, который в первом приближении соответствует наблюдаемому явлению.

На рис. 7 показано распределение одной из компонент электрического поля (в безразмерных единицах), E_x, в различные моменты времени, соответствующие вхождению электронного пучка в область.

Видно, что области максимума электрического поля, показанные красным цветом, постепенно сдвигаются от центра области к краям, и также ведут себя области минимума (синий цвет), т. е. имеет место распространение колебаний электромагнитного поля в пространстве, что по определению представляет собой волну. Таким образом, динамика изменений поля позволяет говорить о наличии в системе электромагнитного излучения. Размер сетки в плоскости XY 250 × 250 узлов, временной шаг 0,001, размер области 16,8 × 10,0 × 10,0 (длина выражена в c/ω_{pe}, где ω_{pe} – плазменная электронная частота). Проведенный расчет является полностью трехмерным, тем не менее на рисунках показаны только распределения поля в плоскости XY в силу сложности отображения трехмерных данных.

Заключение

В статье предложена методика перенесения программы с суперЭВМ гибридной архитектуры на основе GPU на суперЭВМ гибридной архитектуры на основе ускорителей Intel Xeon Phi. Продемонстрирована эффективность распараллеливания и реализации на GPU, а также масштабирование на суперЭВМ на основе Intel Xeon Phi. Получены предварительные результаты по моделированию электромагнитных волн при входе пучка в расчетную область.

Список литературы

1. *Astrelin V. T., Burdakov A. V., Postupaev V. V.* Generation of ion-acoustic waves and suppresion of heat transport during plasma heating by an electron beam // Plasma Physics Reports. 1998. Vol. 24 (5). P. 414–425.

2. *Annenkov V. V., Timofeev I. V., Volchok E. P.* Simulations of electromagnetic emissions produced in a thin plasma by a continuously injected electron beam // Physics of Plasmas. 2016. Vol. 23. P. 053101.

3. *Бэдсел Ч., Лэнгдон А.* Физика плазмы и численное моделирование. М.: Энергоатомиздат, 1989.

4. *Григорьев Ю. Н., Вшивков В. А., Федорук М. П.* Численное моделирование методами частиц в ячейках. Новосибирск: Изд-во СО РАН, 2004.

5. *Месяц Е. А., Снытников А. В., Лотов К. В.* О выборе числа частиц в методе частиц-в-ячейках для моделирования задач физики плазмы // Вычислительные технологии. 2013. Т. 18, № 6. С. 83–96.

6. *Боресков А. В., Харламов А. А.* Основы работы с технологией CUDA. ДМК Пресс, 2010.

7. *Gelas J. de.* Intel's Xeon Phi in 10 Petaflops supercomputer. AnandTech (11 сентября 2012). URL: http://www.anandtech.com/show/6265/intels-xeon-phi-in-10-petaflops-supercomputer.

8. *Burdakov A. V., Kotelnikov I. A., Erofeev V. I.* Explanation of Turbulent Suppression of Electron Heat Transfer in GOL-3 Facility at the Stage of Relativistic Electron Beam Injection // Fusion Science and Technology. 2005. Vol. 47 (1T).

Материал поступил в редколлегию 20.09.2016

A. A. Romanenko [1], **A. V. Snytnikov** [2], **I. V. Timofeev** [3]

[1] *Novosibirsk State University*
1 Pirogov Str., Novosibirsk, 630090, Russian Federation

[2] *Institute of Computational Mathematics and Mathematical Geophysics SB RAS*
6 Academician Lavrentiev Ave., Novosibirsk, 630090, Russian Federation

[3] *Budker Institute of Nuclear Physics*
11 Academician Lavrentiev Ave., Novosibirsk, 630090, Russian Federation

arom@ccfit.nsu.ru, snytav@ssd.sscc.ru, timofeev@ngs.ru

3D HYBRID CODE FOR SIMULATION OF HIGH-FREQUENCY ELECTROMAGNETIC RADIATION GENERATION IN TURBULENT PLASMA

The simulation of electromagnetic radiation generation in turbulent plasma should be carried out in 3D. Therefore a code is being developed that enables to conduct 3D simulations. The code is primarily for hybrid supercomputers equipped with either GPUs or Intel Xeon Phi accelerators. The results are given of the simulation of the electromagnetic waves excited by the powerful electron beam entering the plasma domain.

Keywords: 3D model, hybrid supercomputers, radiation, turbulence.

References

1. V. T. Astrelin, A. V. Burdakov, V. V. Postupaev, Generation of ion-acoustic waves and suppresion of heat transport during plasma heating by an electron beam // Plasma Physics Reports. 24(5) 414–425, 1998.

2. V. V. Annenkov, I. V. Timofeev, and E. P. Volchok. Simulations of electromagnetic emissions produced in a thin plasma by a continuously injected electron beam // Physics of Plasmas 23, 053101, 2016.

3. C. K. Birdsall, A. B. Langdon. Plasma Physics via Computer Simulation. CRC press, 2004.

4. Yu. N. Grigoryev, V. A. Vshivkov, M. P. Fedoruk. Numerical Particle-in-Cell Methods: Theory and Applications. VSP books. 2002.

5. K. V. Lotov, I. V. Timofeev, E. A. Mesyats, A. V. Snytnikov V. A. Vshivkov. Note on quantitatively correct simulations of the kinetic beam-plasma instability // Phys. Plasmas 22, 024502 (2015).

6. J. Sanders, E. Candrot. CUDA by example. Addison-Wesley, 2011.

7. Johan De Gelas. Intel's Xeon Phi in 10 Petaflops supercomputer. AnandTech (11 сентября 2012). http://www.anandtech.com/show/6265/intels-xeon-phi-in-10-petaflops-supercomputer.

8. A. V. Burdakov, I. A. Kotelnikov, V. I. Erofeev. Explanation of Turbulent Suppression of Electron Heat Transfer in GOL-3 Facility at the Stage of Relativistic Electron Beam Injection// Fusion Science and Technology 47(1T), 2005.