Литература

1. Nios II Processor Reference Handbook. [Electronic resource]. Mode of access: http://www.altera.com/literature/hb/nios2/n2cpu_nii5v1.pdf.

2. Hard Processor System Technical Reference Manual. [Electronic resource]. Mode of access: http://www.altera.com/literature/hb/cyclone-v/cv_5v4.pdf.

3. Cyclone V Device Overview. [Electronic resource]. Mode of access: http://www.altera.com/literature/hb/cyclone-v/cv_51001.pdf.

4. Booting and Configuration Introduction. [Electronic resource]. Mode of access: http://www.altera.com/literature/hb/cyclone-v/cv_5400A.pdf.

5. Configuration, Design Security, and Remote System Upgrades in Cyclone V Devices. [Electronic resource]. Mode of access: http://www.altera.com/literature/hb/cyclone-v/cv_52007.pdf.

6. Altera SoC Embedded Design Suite User Guide. [Electronic resource]. Mode of access: http://www.altera.com/literature/ug/ug_soc_eds.pdf.

7. FPGA-Adaptive Software Debug and Performance Analysis. [Electronic resource]. Mode of access: http://www.altera.com/literature/wp/wp-01198-fpga-software-debug-soc.pdf.

РЕАЛИЗАЦИЯ АЛГОРИТМА КАСКАДНОГО СБОРА ГЛОБАЛЬНОГО УРОВНЯ В МНОГОСЕТОЧНОМ РЕШАТЕЛЕ ПАКЕТА ПРОГРАММ ЛОГОС

А. В. Ялозо, А. С. Козелков, Д. П. Силаев, С. В. Лашкин

Российский федеральный ядерный центр –
Всероссийский НИИ экспериментальной физики, г. Саров

Неявная дискретизация основной системы вычислительной гидродинамики – системы уравнений Навье–Стокса, порождает систему разностных уравнений, которая в основном решается итерационными методами [1]. При этом классические итерационные методы либо перестают работать, либо дают очень медленную скорость сходимости [2]. Одним из методов решения данной проблемы является использование алгебраического многосеточного метода, основанного на использовании последовательности вложенных сеток и операторов перехода от одной сетки к другой [3, 4].

В пакете программ ЛОГОС, предназначенном для расчета задач гидродинамики, аэродинамики, тепломассопереноса и распространения тепла в твердотельных конструкциях, при решении СЛАУ используется собственная реализация многосеточного метода [5]. Разработанный ранее в рамках данного решателя последовательный алгоритм сбора матрицы глобального уровня позволил получить значительное ускорение решения СЛАУ, но в то же время данная реализация имеет существенные ограничения на максимально возможный размер матрицы СЛАУ: суммарный объем объединяемых уровней не должен превышать объема памяти одного узла вычислительного кластера, на котором производится расчет. Кроме того, ввиду роста размерности задач, и соответственно, роста числа узлов, требуемых для решения задачи, растет и время, затрачиваемое на построение, и последующее огрубление глобального уровня в скалярном режиме.

На рис. 1 схематично представлена скалярная реализация алгоритма формирования глобального уровня для задачи, выполняемой на n процессорах.

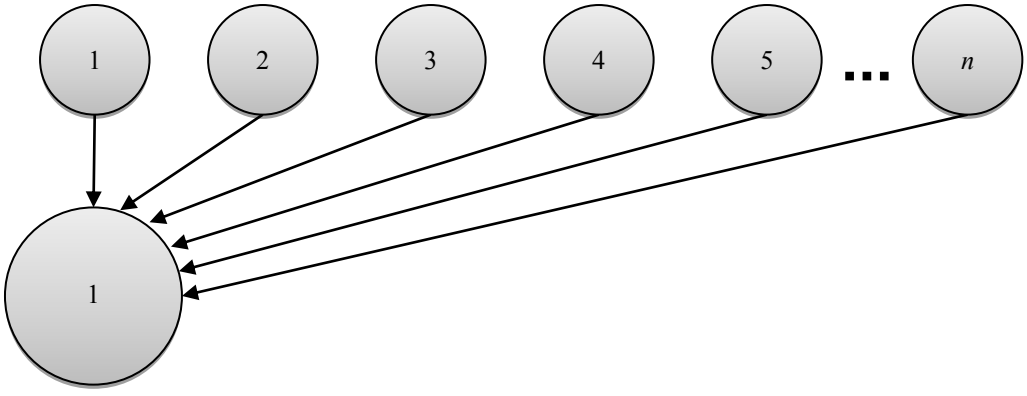

Рис. 1. Скалярная реализация построения глобального уровня

В случае расчета полномасштабных моделей задачи зачастую решаются на тысячах процессоров. Поэтому данный алгоритм становится самым узким местом в работе решателя, ввиду отсутствия масштабируемости: при определенном размере решаемой задачи памяти узла, на котором расположен основной процессор, может просто не хватить для построения глобального уровня.

Для устранения указанных недостатков проанализируем возможные методы ускорения решения задачи формирования глобального уровня. Рассмотрим более простую аналогию – заменим матрицы грубых уровней вещественными числами, а операцию объединения и огрубления двух матриц – операцией суммирования. Тогда формально процедуру построения глобального уровня можно представить в виде вычисления суммы произвольного набора вещественных чисел $S = \sum_{i=1}^{n} x_i$.

Традиционный алгоритм для решения данной задачи состоит в последовательном суммировании всех элементов заданного числового набора. Параллелизм алгоритма суммирования становится возможным только при ином способе построения процесса вычислений, основанном на использовании ассоциативности операции сложения. Получаемый новый вариант суммирования (известный в литературе как «*каскадная схема*») состоит в следующем [6] (см. рис. 2):

- на первой итерации каскадной схемы весь исходный числовой набор разбивается на пары и для каждой пары вычисляется сумма значений,
- далее все полученные суммы пар также разбиваются на пары, и снова выполняется суммирование значений новых пар,
- операции повторяются до тех пор, пока не будет найдена общая сумма.

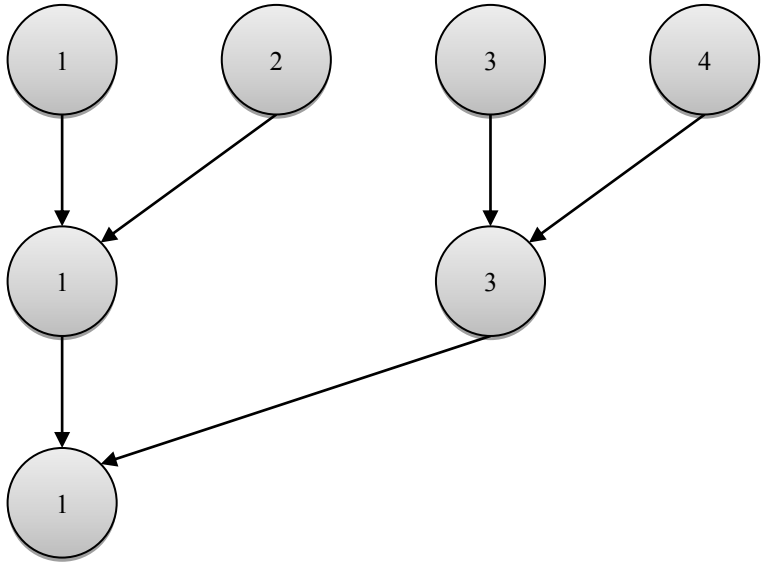

Рис. 2. Каскадная схема алгоритма суммирования

Данная вычислительная схема может быть определена посредством связного ациклического графа – дерева [6]. Для простоты оценок положим количество вершин дерева $n = 2^k$, где k – произвольное целое число.

Тогда общее количество итераций каскадной схемы можно оценить через высоту дерева, которая оказывается равной величине

$$k = \log_2 n. \tag{1}$$

Общее количество операций суммирования

$$L_{\text{посл}} = \frac{n}{2} + \frac{n}{4} + \ldots + 1 = n - 1 \tag{2}$$

совпадает с количеством операций последовательного варианта алгоритма суммирования. Однако при параллельном исполнении отдельных итераций каскадной схемы общее количество параллельных операций суммирования совпадает с высотой дерева и, как было указано в (1), будет равно

$$L_{\text{пар}} = k = \log_2 n. \tag{3}$$

Исходя из (2) и (3), можно оценить показатели ускорения S_p и эффективности E_p каскадной схемы алгоритма суммирования:

$$S_p = \frac{T_1}{T_p} = \frac{n-1}{\log_2 n}, \tag{4}$$

$$E_p = \frac{T_1}{pT_p} = \frac{n-1}{\left[\left(\mathbf{p} * \log_2\right]\right] n)} = \frac{n-1}{n \log_2 n}, \tag{5}$$

где $p = n$ есть необходимое для выполнения каскадной схемы количество процессоров.

Анализируя формулу (5), можно отметить, что эффективность использования процессоров уменьшается при увеличении количества суммируемых значений,

$$\lim_{n \to \infty} E_p \to 0. \tag{6}$$

Получить асимптотически ненулевую эффективность можно путем модификации стандартной каскадной схемы, например, складывая на каждом этапе не пару, а K значений. На рис. 3 схематично представлена схема каскадного суммирования с размером каскада, равным трем ($K = 3$).

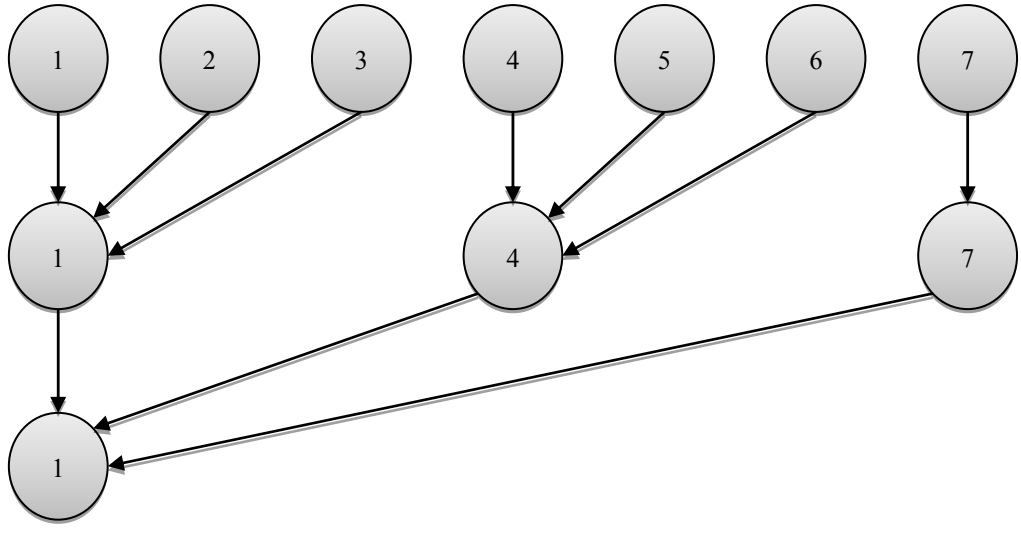

Рис. 3. Модифицированная каскадная схема алгоритма суммирования

Тогда показатели ускорения S_p и эффективности E_p модифицированной каскадной схемы будут иметь вид

$$S_p = \frac{(n-1)}{\log_K n}, \tag{7}$$

$$E_p = \frac{n-1}{n * \log_K n}. \tag{8}$$

Таким образом, варьируя значение параметра K, можно добиться приемлемого баланса эффективности и ускорения.

Основным преимуществом каскадной схемы суммирования является наличие масштабируемости алгоритма. При этом фактически снимается ограничение на максимальный размер решаемой задачи из-за возможной нехватки памяти узла, на котором производится формирование глобального уровня. Кроме того, увеличивается скорость построения глобального уровня, поскольку его составные части формируются и огрубляются независимо друг от друга.

Стоит особо отметить, что точная оценка ускорения и эффективности справедлива только для рассматриваемой нами упрощенной аналогии – суммирования ряда вещественных чисел, поскольку на практике под операцией суммирования будет подразумеваться процедура объединения нескольких грубых уровней в один, и его дальнейшее огрубление. Само по себе объединение уровней, помимо формирования информации о новом уровне, содержит в себе процедуру переопределения информации о межпроцессорных обменах, которая также требует затраты какой-то части процессорного времени. Кроме того, параллельная процедура огрубления каскадного уровня требует проведения межпроцессорных обменов, которые в случае со скалярной реализацией сбора глобального уровня отсутствуют. Поэтому о реальных показателях ускорения можно судить по результатам, приведенным далее.

Рассмотрим пример, описывающий стационарное турбулентное течение вязкой несжимаемой жидкости в прямолинейной круглой трубе.

СЛАУ давления в данной задаче решалась с относительной точностью 10^{-1}, максимальное число циклов многосеточного метода – 30. Расчеты осуществлялись до сходимости. В качестве результатов приводится сравнение среднего времени построения глобального уровня и среднего времени выполнения одного шага. Приводятся данные, полученные с использованием предыдущей скалярной версии алгоритма сбора глобального уровня, и каскадной версии, реализованной в рамках данной работы.

Результаты вычислительных экспериментов задачи турбулентного течения в трубе на сетке 481000 ячеек представлены в таблице.

Результаты вычислительных экспериментов задачи (481 000 ячеек)

Тип построения глобального уровня	**Число процессов**	**Среднее время построения глобального уровня**	**Среднее время решения СЛАУ**
Скалярный	16	0,000	0,417
Каскадный ($K = 2$)		0,000	0,423
Коэффициент ускорения, раз		***n/a***	**0,98**
Скалярный	128	0,003	0,096
Каскадный ($K = 2$)		0,002	0,100
Коэффициент ускорения, раз		**1,5**	**0,96**
Скалярный	256	0,016	0,174
Каскадный ($K = 2$)		0,002	0,091
Коэффициент ускорения, раз		**8**	**1,91**

На сравнительно небольшой сетке время построения глобального уровня составляет десятые доли процента от общего времени шага, и, несмотря на то, что каскадное построение во всех случаях работает быстрее, это ускорение нивелируется на фоне общего времени решения СЛАУ. В то же время, при искусственном измельчении процессорного поля (на 256 процессоров) видно, что каскадный сбор дает существенное ускорение общего решения СЛАУ, поскольку в данном случае скалярный алгоритм сгенерирует относительно большую матрицу глобального уровня, огрубление и решение которой проводится в последовательном режиме и требует значительных затрат времени.

Следующей тестовой задачей стал расчет турбулентного течения в канале за обратным уступом. Данная задача решалась на двух сетках в 14 и 30 млн. счетных ячеек. СЛАУ давления в данной задаче решалась с относительной точностью 10^{-1}, максимальное число циклов многосеточного метода – 30. Расчеты осуществлялись до сходимости. В качестве результатов приводится сравнение полного времени решения задачи и общего количества итераций метода SIMPLE при решении задачи на различном числе процессоров. Сравниваются данные, полученные с использованием предыдущей скалярной версии алгоритма сбора глобального уровня, и каскадной версии, реализованной в рамках данной работы.

На рис. 4 приведены результаты вычислительного эксперимента задачи расчета турбулентного течения в канале за обратным уступом на сетке 14 млн. ячеек.

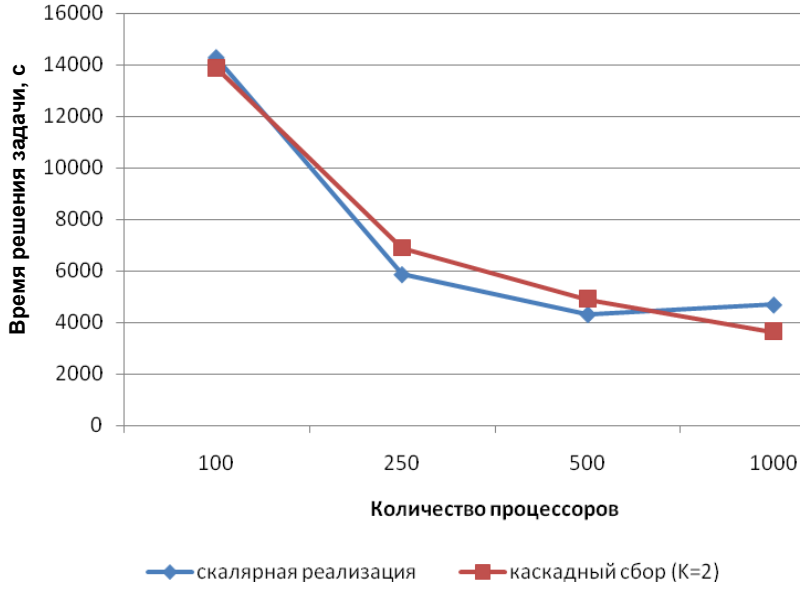

Рис. 4. Каскадная схема алгоритма суммирования

Из графика можно сделать вывод, что при расчете задачи на 250 и 500 процессорах алгоритм каскадного сбора проигрывает в производительности скалярной реализации алгоритма построения глобального уровня. Это обусловлено необходимостью проведения дополнительных межпроцессорных обменов процедурой огрубления каскадного уровня. Однако при более мелком разбиении задачи (на 1000 процессоров), алгоритм каскадного сбора сохраняет положительную тенденцию ускорения, в то время как скалярная реализация алгоритма показывает отрицательное ускорение.

Следующим набором тестов было решение данной задачи на разном количестве процессоров на более подробной сетке в 30 млн. ячеек. Результаты расчетов приведены на рис. 5.

По полученным результатам можно сделать вывод, что на небольшом количестве процессоров скалярный алгоритм построения глобального уровня дает лучшие результаты. Однако при измельчении задачи менее чем по ~60 тыс. расчетных ячеек на процессор имеем значительное и возрастающее преимущество по производительности алгоритма каскадного сбора глобального уровня.

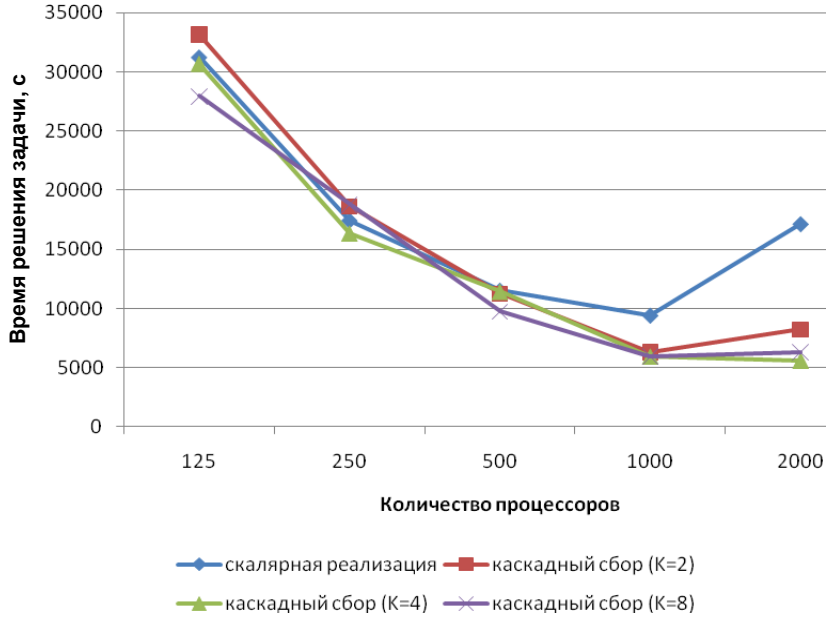

Рис. 5. Каскадная схема алгоритма суммирования

Для подтверждения данного предположения приведем график (рис. 6), на котором показано общее время решения задач различной размерности на разном числе процессоров при одинаковом количестве счетных ячеек на процессор, равном ~60 тыс. Данное число расчетных ячеек установлено экспериментально и является оптимальным по соотношению ускорения и эффективности для текущей реализации многосеточного метода на структуре памяти пакета программ ЛОГОС.

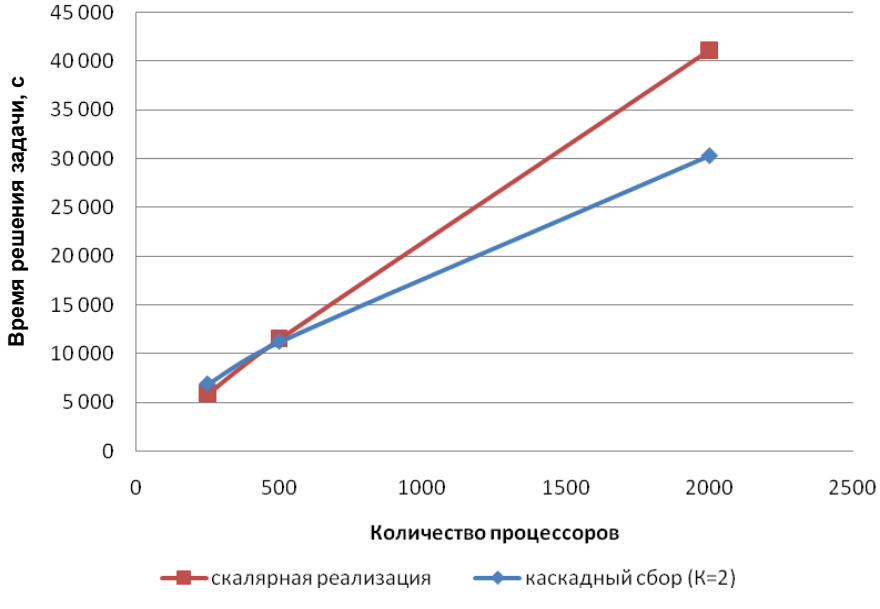

Рис. 6. Каскадная схема алгоритма суммирования

Как видно из графика, для задач до 30 млн. расчетных ячеек (при разбиении задачи по 60 тыс. расчетных ячеек на процессор), скалярная реализация алгоритма построения глобального уровня

оказывается более производительной. При увеличении размерности задачи, но сохранении одинакового числа расчетных ячеек на процессор, алгоритм каскадного сбора дает существенный выигрыш в производительности. Так, для задачи ~125 млн. счетных ячеек общее ускорение времени решения задачи составляет порядка 35 %.

Обобщая полученные результаты, можно сделать вывод, что использование алгоритма каскадного сбора оправдано для задач, общей размерностью от 30 млн. счетных ячеек (при оптимальном разбиении задачи на процессоры по ~60 тыс. расчетных ячеек). По нашим оценкам при дальнейшем увеличении размерности задачи положительная тенденция ускорения должна сохраниться. С увеличением количества процессоров тенденция к «проседанию» производительности, в отличие от предыдущей скалярной реализации алгоритма, проявляется значительно позже. Ввиду растущей необходимости расчетов полномасштабных моделей, решение которых зачастую требует использования тысяч процессоров, внедрение данного алгоритма становится особенно актуальным.

Литература

1. Saad Y. Iterative methods for sparse linear systems.

2. Brandt A. Guide to multigrid development // Lecture Notes in Mathematics. 1982. Vol. 960. P. 220–312.

3. Федоренко Р. П. Релаксационный метод решения разностных эллиптических уравнений // Журнал выч. мат.и мат. физ. 1961. Т. 1, № 5. С. 922–927.

4. Meurant G. Computer solution of large linear systems. Amsterdam, Lausanne, New York, Oxford, Shannon, Singapore, Tokyo: Elsevier, 1999.

5. Козелков А. С., Дерюгин Ю. Н., Зеленский Д. К. и др. Многофункциональный пакет программ ЛОГОС для расчета задач гидродинамики и тепломассопереноса на многопроцессорных ЭВМ: Базовые технологии и алгоритмы / Под ред. Р.М. Шагалиева // Труды. XII Международного семинара «Супервычисления и математическое моделирование». Саров: РФЯЦ-ВНИИЭФ, 2011. С. 215–230.

6. Емеличев В. А., Мельников О. И., Сарванов В. И., Тышкевич Р. И. Лекции по теории графов. М.: Наука, Физматлит, 1990.

7. Mikhaylyukov K. L., Ivanin I. A., Tatsenko M. V. et al. A method of experimental measurement of Point Spread Function (PSF) due to scintillatorfor high-energy protons // Workshop on High Energy Proton Microscopy НЕРМ-2010, г.Черноголовка, 2–4 июня 2010 г.

8. Михайлюков К. Л., Таценко М. В., Картанов С. А. Метод получения функции размытия точки в системе сцинтиллятор-регистратор при регистрации протонов высоких энергий // XI Международная конференция «Забабахинские научные чтения». Снежинск, 2012.

9. Eyges L. Multiple scattering with energy loss // Phys. Rev. 1948. Vol. 74, N 10. P. 1534–1535.

10. Agostinelli S., Allison J., Amako K. et al. Geant 4 – a simulation toolkit // Nuclear Instruments and Methods in Physics Research. Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. 2003. Vol. 506, N 3. P. 250–303.

11. Allison J., Amako K., Apostolakis J. et al. Geant 4 developments and applications // IEEE Transactions on Nuclear Science. 2006. Vol. 53, N 1. P. 270–278.