**BC/NW 2016 № 1 (28): 2.4**

**ПРИНЦИПЫ ОРГАНИЗАЦИИ ОБРАБОТКИ ДАННЫХ В ПРИКЛАДНЫХ ПРОГРАММАХ С ЦЕЛЬЮ ЭФФЕКТИВНОГО ИСПОЛЬЗОВАНИЯ СТРУКТУРНО-ФУНКЦИОНАЛЬНЫХ ОСОБЕННОСТЕЙ ПАМЯТИ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ**

Тивиков В.С., Филатов А.В.

Многие разработчики при создании алгоритмов своих прикладных программ заботятся в основном только о выполнении их логических и функциональных операций Меньше уделяют внимания эффективному использованию этими программами структурных и функциональных особенностей вычислительных систем. В частности, при разработке алгоритмов обработки данных в прикладных программах важно учитывать структуру и параметры иерархии памяти вычислительной системы, а также системные алгоритмы ее функционирования [1, 2]. В зависимости от того, какой вариант алгоритма будет выбран для реализации в программе, время выполнения этой программы может значительно отличаться.

Особо актуальна данная задача при осуществлении обработки больших объемов данных, располагаемых на разных уровнях иерархии памяти с разной емкостью и временем доступа. Удачно согласовав алгоритм обработки данных в программе с системным алгоритмом размещения данных в иерархии памяти, можно добиться существенного уменьшения времени выполнения программы.

Для написания эффективного кода программисты должны знать:

1) методы обхода кэша центрального процессора (ЦП);

2) методы оптимизации доступа к кэшу данных ЦП;

3) методы оптимизации доступа к кэшу инструкций ЦП;

4) способы предварительной загрузки данных в кэш ЦП;

5) способы выравнивания данных в памяти вычислительных систем;

6) порядок обхода данных в памяти вычислительных систем.

Понимая «узкие» места в памяти современных вычислительных систем

и возможности их обхода в алгоритмах обработки данных, разработчик

сможет писать эффективный код.

Литература

1. Ulrich Drepper. What Every Programmer Should Know About Memory, 2007.

2. Jonathan Rentzsch. Data alignment: Straighten up and fly right, 2005