П. В. Стасевич, Н. А. Жиляк
(БГТУ, Минск)

ИСПОЛЬЗОВАНИЕ ЯЗЫКА АССЕМБЛЕР И ЕГО ДОСТОИНСТВА

Использование языка ассемблер – это один из наиболее действенных методов оптимизации программ, и во многом методы, используемые для повышения производительности, схожи с теми, что используются в языках высокого уровня, выделим методы, свойственные только ассемблеру. Использование языка ассемблера во многом

решает проблему избыточности программного кода. Ассемблерный код более компактен, чем его аналог на языке высокого уровня. Чтобы убедиться в этом, достаточно сравнить дизассемблированные листинги одной и той же программы, написанной на ассемблере и на языке высокого уровня. Сгенерированный компилятором языка высокого уровня ассемблерный код с использованием опций оптимизации не устраняет избыточность кода приложения. В то же время язык ассемблера позволяет разрабатывать короткий и эффективный код.

Программный модуль на ассемблере обладает, как правило, более высоким быстродействием, чем написанный на языке высокого уровня. Это связано с меньшим числом команд, требуемых для реализации фрагмента кода. Это повышает производительность программы.

Можно разрабатывать отдельные модули полностью на ассемблере и присоединять их к программам на языке высокого уровня. Также можно использовать мощные встроенные средства языков высокого уровня для написания ассемблерных процедур непосредственно в теле основной программы. Такие возможности предусмотрены во всех языках высокого уровня. Эффективность использования встроенного ассемблера может быть очень высока. Встроенный ассемблер позволяет добиваться максимального эффекта при оптимизации математических выражений, программных циклов и обработки массивов данных в основной программе.

Производительность программ можно увеличить, используя:

– команды пересылки с нулевым или знаковым расширением;

– установки в байте значений «истина» или «ложь» в зависимости от содержимого флагов центрального процессора, что позволяет избавиться от команд условного перехода;

– команды проверки, установки, сброса и сканирования битов ;

– обобщенную индексную адресацию и режимы адресации с масштабированием индексов;

– быстрое умножение при помощи команды lea с использованием масштабированной индексной адресации;

– перемножение 32-разрядных чисел и деление 64-разрядного числа на 32-разрядное;

– операции для обработки многобайтных массивов данных и строк.