**14. Архитектура суперскалярных процессоров. Предварительная выборка и предсказание переходов.**

**Суперскалярная архитектура** – Способность выполнения нескольких машинных инструкций за один такт процессора. Появление этой технологии привело к существенному увеличению производительности. Основная идея, определяющая развитие суперскалярных микропроцес­соров, состоит в построении возможно большего количества парал­лельных структур при сохранении традиционных последовательных про­грамм. Это означает, что компиляторы и аппаратура микропроцессора сами, без вмешательства программиста, обеспечивают загрузку параллель­но работающих функциональных устройств микропроцессора. В соответствии с моделью последовательного программирования, про­граммы пишутся в предположении, что команды будут выполнены в том же порядке, в каком они представлены в программе. Однако с целью достиже­ния большей эффективности современные процессоры пытаются выполнять несколько команд одновременно и в некоторых случаях в порядке, отличном от их исходной последовательности в программе. Это переупорядочение мо­жет быть выполнено в трансляторе и (или) в аппаратных средствах во время выполнения. Суперскалярные и VLIW-процессоры принадлежат классу ар­хитектур, которые используют параллельность уровня команды (ILP). ILP-процессоры и компиляторы обычно преобразуют полностью упо­рядоченное множество команд исходной программы в частично упорядо­ченное множество, структурированное зависимостями по данным и управ­лению. Зависимости по управлению (которые проявляются как переходы по условию) представляют главное препятствие высокопараллельному выполнению потому, что эти зависимости должны быть установлены пре­жде, чем будут выполнены все последующие команды. Текст последовательной программы, представленной на языке высо­кого уровня, компилируется в машинный код, отражающий статическую структуру программы, т.е. упорядоченное множество команд (инструк­ций) в памяти компьютера. Процесс выполнения программы с конкрет­ными наборами входных данных может быть представлен динамической структурой программы, т.е. множеством последовательностей инструк­ций в порядке их исполнения. Повысить степень параллелизма программы можно изменяя соответст­вующим образом ее статическую или динамическую структуру. Поскольку статическая структура программы однозначно соответствует ее исходному тексту (в предположении неизменности компилятора), то изменение статиче­ской структуры сводится к изменению исходного кода, что, в общем случае, не всегда возможно.

Динамическая же структура программы может быть из­менена при неизменной статической структуре. И главной целью такого из­менения должно быть повышение степени параллельного исполнения команд. Допустимые границы преобразования динамической структуры про­граммы задают существующие на множестве инструкций отношения: за­висимость по управлению и зависимость по данным. При описании архи­тектур ССП часто используется модель окна ис­полнения. При исполнении программы микропроцессор как бы продви­гает по статической структуре программы окно исполнения. Команды в окне могут исполняться параллельно, если между ними нет зависимости. Для устранения зависимостей, вызванных командами переходов, исполь­зуется метод предсказания, позволяющий извлекать и условно исполнять команды предсказанного перехода. Если позднее обнаруживается, что пред­сказание было сделано, верно, то результаты условно исполненных команд принимаются. Если предсказание было ошибочным, состояние процессора восстанавливается на момент принятия решения о выполнении перехода. Все виды зависимостей по данным могут быть классифицированы по типу ассоциаций: RAR - "чтение после чтения", WAR - "запись после чте­ния" и WAW - "запись после записи", RAW - "чтение после записи". Действительной зави­симостью является только "чтение после записи" (RAW), так как необходи­мо прочитать предварительно записанные новые данные, а не старые.