Семантику памяти можно рассматривать как контракт между программным обеспечением и аппаратным обеспечением памяти. Если программное обеспечение соглашается следовать определенным правилам, то память соглашается выдавать определенные результаты.

Модели согласованности (правила):

- Строгая согласованность (при любом считывании из адреса X всегда возвращается самая последняя запись в X. Модель может быть реализована только следующим образом: должен быть один модуль памяти, который обслуживает все запросы по мере поступления, без кэш-памяти, без дублирования данных. А это очень сильно замедляет работу памяти)
- Согласованность по последовательности (при наличии нескольких запросов на чтение и запись порядок обработки запросов определяется аппаратно, но при этом все процессоры воспринимают один и тот же порядок. Может возникнуть ситуация, при которой к одной и той же переменной обратятся несколько процессоров для записи и считывания. Т.е. возможна неоднозначность толкования последовательности обращений и процессоры, обратившиеся к памяти практически одновременно (в течение одного цикла) могут считать различные результаты)
- Процессорная согласованность (все процессоры воспринимают записи любого процессора в том порядке, в котором они начинаются. Все процессоры видят записи в слово памяти в том порядке, в котором они происходят - если имело место несколько записей, то все процессоры должны воспринимать последнее значение. Не гарантируется, что каждый процессор видит одну и ту же последовательность)
- Слабая согласованность (время разделяется на последовательные периоды, разграниченные моментами синхронизации - все незаконченные записи завершаются, а новые не могут начаться пока не будут завершены все начатые и не будет проведена синхронизация. Синхронизация приводит память в стабильное состояние. Операции синхронизации согласованы по последовательности. Внутри периодов в последовательность может быть видна различными процессорами по-разному)
- Свободная согласованность (Чтобы считать или записать общую переменную, процессор (т.е. его ПО) сначала должно выполнить операцию *acquire* над переменной синхронизации, чтобы получить монопольный доступ к общим разделяемым данным. После использования процессор выполняет операцию *release* над переменной синхронизации. Операция *release* не требует завершения незаконченных записей, однако она сама не может быть завершена, пока не закончатся все начатые записи. Когда начинается новая операция *acquire,* производится проверка, все ли начатые операции *release* завершены. Если нет, то операции *acquire* будут задержаны)

Кэш-памятью с отслеживанием перехватывает запросы на шине от других процессоров и кэш-памятей и предпринимает определенные действия в необходимых случаях.

Протоколы согласования кэшей не допускают одновременного появления разных версий одной и той же строки в двух или более кэшах.

Сквозное кэширование. В результате всех операций записи записываемое слово обязательно проходит через основную память. Элемент, содержащий измененное слово помечается как недействительный. Все кэш-памяти постоянно отслеживают изменения на шине и каждый раз, когда записывается слово, его обновляют в кэш-памяти инициатора, в основной памяти и удаляют из остальных кэш-памятей.

MESI (протокол однократной записи). Каждый элемент кэш-памяти может находиться в одном из четырех состояний:

- Invalid – элемент кэш-памяти содержит недействительные данные.
- Shared – несколько кэшей могут содержать данную строку, основная память обновлена.
- Exclusive – никакой другой кэш не содержит эту строку, основная память обновлена.
- Modified – элемент действителен, основная память недействительна, копий элемента не существует