***35. Протокол MESI***

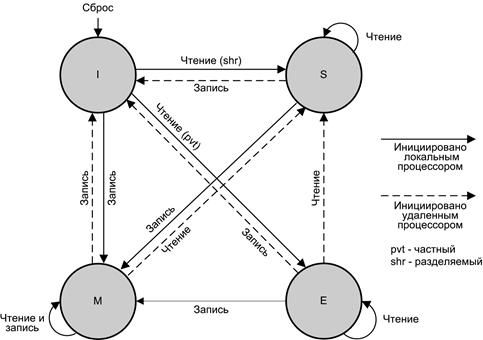
[MESI-wiki](https://en.wikipedia.org/wiki/MESI_protocol)

[mesi](http://studopedia.info/7-40186.html) – тут чётко

Протокол был разработан для кэш-памяти с обратной записью. Одной из основных задач протокола MESI является откладывание на максимально возможный срок операции обратной записи кэшированных данных в основную память системы. Это позволяет улучшить производительность системы за счет минимизации ненужных пересылок информации между кэшами и основной памятью.

Протокол MESI приписывает каждой кэш-строке одно из четырех состояний, которые контролируются двумя битами состояния MESI в теге данной строки. Состояние кэш-строки может быть изменено как процессором, для которого эта кэш-память является локальной, так и другими процессорами мультипроцессорной системы. Управление состоянием кэш-строк может быть возложено и на внешние логические устройства. Одна из версий протокола предусматривает использование ранее рассмотренной схемы однократной записи.

Каждая строка согласно протоколу MESI может быть в одном из четырех возможных состояний (в дальнейшем будем ссылаться на эти состояния с помощью букв M, E, S, I):

 Диаграмма состояний без учета однократной записи

* Модифицированная (M, Modified) — данные в кэш-строке, помеченной как M, были модифицированы (подверглись изменению), но измененная информация не была переписана в основную память. Это означает, что информация, содержащаяся в рассматриваемой строке, достоверна только в данном кэше, а в основной памяти и остальных кэшах — недостоверна;
* Эксклюзивная(E, Exclusive) — данная строка в кэше не подвергалась изменению посредством запроса на запись, совпадает с аналогичной строкой в основной памяти, но отсутствует в любом другом локальном кэше. Иными словами, данная строка достоверна в этом кэше и недостоверна в любом другом кэше;
* Разделяемая(S, Shared) — данная строка в кэше совпадает с аналогичной строкой в основной памяти (данные достоверны) и может присутствовать в одном или нескольких из прочих кэшей;
* Недействительная(I, Invalid) — кэш-строка, помеченная как недействительная, не содержит достоверных данных и становится логически недоступной.

Порядок перехода строки кэш-памяти из одного состояния в другое зависит от: текущего состояния строки, выполняемой операции (чтение или запись), результата обращения к строке (попадание или промах) и, наконец, от того является ли строка совместно используемой или нет.

Основное применение — спекулятивное использование критических секций. (!?!?!?!?)

**Критическая секция** — участок исполняемого кода программы, в котором производится доступ к общему ресурсу (данным или устройству), который не должен быть одновременно использован более чем одним потоком исполнения

**Speculative execution** - це метод оптимізації, коли комп'ютерна система виконує деякі завдання, які можуть бути насправді непотрібними. Основна ідея полягає в тому, щоб зробити роботу до того, як відомо, чи буде потрібна ця робота взагалі, з тим щоб запобігти затримці. Якщо з'ясовується, що робота була не потрібна в кінці кінців, будь-які зміни скасовуються і результати ігноруються. Мета полягає в тому, щоб забезпечити більше параллелизму[ru] якщо додаткові ресурси доступні. Такий підхід використовується в різних областях, у тому числі прогнозування розгалуженняв конвеєрних процесорах[ru] упереджувальна вибірка пам'яті та файлів, і оптимістичне управління паралелізмом[en] у системах баз даних

Перевод:

Спекулятивное выполнение - это метод оптимизации , когда компьютерная система выполняет некоторые задачи, которые могут быть на самом деле нужны. Основная идея заключается в том, чтобы сделать работу до того, как известно, будет ли нужна эта работа вообще, с тем чтобы предотвратить задержку. Если выясняется, что работа была не нужна в конце концов, любые изменения отменяются и результаты игнорируются. Цель состоит в том, чтобы обеспечить более параллелизму если дополнительные ресурсы доступны. Такой подход используется в различных областях, в том числе прогнозирования ветвления в конвейерных процессорах упреждающая выборка памяти и файлов , и оптимистическое управления параллелизмом в системах баз данных