Подводные камни при тестировании ОСРВ: опыт практического бенчмаркинга на Cortex-M4

Черкасов Михаил Сергеевич
АО «ЭРЕМЕКС», Москва
cherkasov@eremex.ru

Введение

Сравнительное тестирование операционных систем реального времени (ОСРВ) кажется задачей с очевидным решением: написать несколько тестов, замерить целевой параметр, сравнить цифры. На практике каждый из этих шагов предполагает принятие решений, которые способны либо обесценить результаты, либо привести к принципиально неверным выводам. Данный доклад обобщает практический опыт, полученный при проведении бенчмаркинга FreeRTOS, ThreadX и FXRTOS на микроконтроллере STM32F411 (Cortex-M4, 100 МГц). Изначально тесты проводились заказчиком, но были отредактированы, благодаря чему в некоторых тестах изменились полученные результаты и повысилась их стабильность.

Методология

Сбор данных осуществлялся через UART по прерыванию от аппаратного таймера. Время измерялось в процессорных тактах через счётчик DWT. Для каждого теста параллельно работали фоновые задачи, нагружающие планировщик, что приближает условия к реальному применению.

Тестировались следующие характеристики: накладные расходы мьютекса и семафора, прямые уведомления задач, группы событий, временное квантование при двух частотах системного таймера — 1000 и 5000 тиков/с.

Важность чёткого определения цели измерения

Первый и наиболее фундаментальный вопрос — что именно измеряется. Один и тот же тест, в зависимости от деталей реализации, может измерять принципиально разные вещи.

Рассмотрим тест на латентность семафора: задача ожидает семафор, который отдается из ISR аппаратного таймера, и измеряет время от срабатывания таймера до момента своего пробуждения. Если задаче-измерителю не назначить приоритет выше фоновых задач, то после выхода из ISR планировщик обойдёт все задачи с равным приоритетом прежде, чем передаст управление измерительной задаче. В результате тест измеряет не накладные расходы семафора, а время полного цикла планировщика с накладными расходами семафора.

Аналогичная ситуация возникает при выборе единиц измерения и метода сбора данных. Время в тактах, время в тиках ОСРВ, число вызовов функции за период — каждый подход отвечает на свой вопрос. Число вызовов удобно для сравнительной оценки различных параметров, абсолютное время в тактах — для оценки соответствия требованиям заказчика, а также детерминизма работы системы. Использование любого из этих подходов без привязки к цели тестирования ведёт к некорректным сравнениям.

Метод сбора данных меняет то, что измеряется

Классический подход к отладке встраиваемых систем — использование точки останова и чтение данных из отладчика. Для бенчмаркинга этот метод непригоден: результаты, полученные таким способом, описывают поведение системы в момент отладки, а не в рабочем режиме.

В описываемой работе реализовали сбор данных через UART по прерыванию от аппаратного таймера. Этот подход позволяет получать достоверные выборки в условиях, максимально приближённых к реальной эксплуатации.

Конфигурация ОСРВ как источник ошибок

ОСРВ предоставляют широкие возможности конфигурации, и многие параметры имеют значения по умолчанию, которые кажутся нейтральными, но способны существенно повлиять на результаты конкретного теста. Разработчик, сосредоточенный на логике самого теста, рискует не обратить внимания на параметры, которые «не трогал».

Практический пример: тест переключения задач измеряет время одного цикла переключения среди задач с одинаковым приоритетом. Если при этом включено временное квантование, системный таймер периодически вытесняет текущую задачу независимо от её вызовов переключения. В результате часть переключений происходит по инициативе планировщика, а не задачи, и измеренное время перестаёт отражать только накладные расходы самого переключения задач.

Общий принцип: конфигурация всех тестируемых систем должна быть явно задокументирована, а параметры, потенциально влияющие на результат конкретного теста, — осознанно выбраны, а не оставлены по умолчанию.

Понимание архитектуры примитивов синхронизации

Примитивы синхронизации с похожим интерфейсом могут иметь принципиально разную внутреннюю реализацию, и эта разница напрямую отражается в результатах измерений.

В FreeRTOS функции `vTaskNotifyGiveFromISR` и `xEventGroupSetBitsFromISR` решают схожую задачу — разбудить задачу из обработчика прерывания. Однако первая выполняет пробуждение непосредственно из ISR, тогда как вторая откладывает его, используя системный таймер. На практике это приводит к тому, что латентность Event Group в 4 раза выше латентности Task Notify при идентичных условиях теста. Разработчик, не знакомый с этой особенностью, может сделать вывод о низкой производительности ОСРВ в целом, тогда как речь идёт о характеристике конкретного примитива.

Выводы

Описанные проблемы не являются следствием незнания теории ОСРВ — они возникают в практической работе именно потому, что кажутся очевидными и не требующими специального внимания. Систематический подход к постановке задачи измерения позволяет избежать большинства из них. На основе описанного опыта можно сформулировать следующий список решений важных при тестировании ОСРВ.

1. До начала написания тестов: явно определить, какой параметр системы измеряется и для какого сценария применения он важен; выбрать метод сбора данных, не нарушающий работу системы.
2. При настройке тестового окружения: зафиксировать и задокументировать конфигурацию каждой тестируемой ОСРВ; проверить параметры, влияющие на результат конкретного теста, а не только на общую функциональность; убедиться в идентичности конфигураций всех сравниваемых систем.
3. При проектировании тестов: осознанно задать приоритеты задач — они определяют, что именно измеряется; изучить внутреннюю архитектуру тестируемых примитивов синхронизации, поскольку схожий интерфейс не означает схожего поведения; определить, требуется ли для поставленных целей

тестирования фоновая нагрузка и приблизить ее к реальным условиям эксплуатации.