ТЕХНИЧЕСКИЕ НАУКИ

УДК 004.451

Сравнительный анализ использования систем реального времени FreeRTOS и CMSIS-RTOS v2 для разработки приложений на микроконтроллерах STM32

К.И. Бережной, Р.Р. Ибадов, А.А., Лаврентьев
Донской государственный технический университет, г. Ростов-на-Дону, Российская Федерация

Аннотация
Настоящая работа посвящена вопросам проектирования встроенных систем на базе микроконтроллеров семейства STM32 с применением операционных систем реального времени. Рассматривается проблема выбора между непосредственным использованием программного интерфейса FreeRTOS и стандартизированного API CMSIS-RTOS v2. Цель данного исследования заключается в проведении сравнительного анализа указанных подходов для формирования обоснованных инженерных рекомендаций. В ходе исследования на базе отладочной платы STM32F407 был реализован эксперимент, включавший детальный анализ программных интерфейсов, формирование задач, организацию процессов межзадачного взаимодействия, а также измерение ключевых характеристик системы. В результате были получены количественные данные о времени переключения контекста и объеме потребляемой памяти, а также выявлены эксплуатационные преимущества и недостатки каждого метода. Практическая значимость работы выражается в определении четких критериев выбора API, базирующихся на конкретных требованиях проекта к производительности и вычислительным ресурсам.

Ключевые слова: STM32, FreeRTOS, CMSIS-RTOS, RTOS, API, многозадачность, встраиваемые системы, производительность, потребление памяти

Для цитирования. Бережной К.И., Ибадов Р.Р., Лаврентьев А.А. Сравнительный анализ использования систем реального времени FreeRTOS и CMSIS-RTOS v2 для разработки приложений на микроконтроллерах STM32. *Молодой исследователь Дона*. 2026;11(4):5–8.

Comparative Analysis of FreeRTOS and CMSIS-RTOS V2 Real-Time Operating Systems Used for Development of STM32-Microcontroller-based Applications

Konstantin I. Berezhnoy, Rahim Ibadov, Anatoly A. Lavrentiev
Don State Technical University, Rostov-on-Don, Russian Federation

Abstract
The paper studies the issues of designing the embedded STM32-microcontroller-bassed systems using the real-time operating systems. It investigates the problem of choosing between the direct use of the FreeRTOS programing interface compared to the use of the standardized API CMSIS-RTOS v2. The present study aims at conducting a comparative analysis of these approaches to formulate relevant engineering recommendations. The experiment was conducted using the STM32F407 development board and included the detailed analysis of the application programming interfaces (APIs), task formulation, organisation of Inter-Task Communication (ITC) processes, and measurement of key software system metrics. The study resulted in getting the quantitative data on the context switch times and amount of memory consumed. The operational advantages and disadvantages of each of the approaches were also identified. The practical significance of the study lies in defining clear criteria for API selection based on the specific requirements of a project for performance and computing capacities.

Keywords: STM32, FreeRTOS, CMSIS-RTOS, RTOS, API, multitasking, embedded systems, performance, memory consumption

For Citation. Berezhnoy KI, Ibadov R, Lavrentiev AA. Comparative Analysis of FreeRTOS and CMSIS-RTOS V2 Real-Time Operating Systems Used for Development of STM32-Microcontroller-based Applications. *Young Researcher of Don*. 2026;11(4):5–8.

Введение. Типовой проект на базе микроконтроллеров STM32 объединяет в себе процедуры опроса периферийных модулей, реализацию протоколов обмена данными и функционирование фоновых сервисов. По мере расширения функциональных возможностей управление данными процессами в рамках единого цикла main() становится затруднительным, что обуславливает необходимость внедрения операционных систем реального времени (RTOS), среди которых наиболее востребованной является FreeRTOS. Однако использование прямых вызовов функций создания задач (xTaskCreate), очередей и семафоров жестко детерминирует зависимость исходного кода от конкретной реализации ядра, существенно осложняя портирование на иные операционные системы. Стандарт CMSIS-RTOS v2, предложенный компанией ARM, предоставляет унифицированный интерфейс прикладного программирования (API) поверх нативной RTOS. В рамках экосистемы компании ST данный стандарт интегрирован в среду STM32Cube. Несмотря на наличие описаний обоих интерфейсов, сопоставимые количественные показатели временных задержек и потребления памяти в открытых публикациях представлены фрагментарно, что определяет актуальность проведения самостоятельных измерений.

Цель настоящей работы — провести сравнительный анализ нативного API FreeRTOS и CMSIS-RTOS v2 на базе идентичной аппаратной платформы в рамках единых сценариев эксплуатации. План экспериментального исследования включал: проведение обзора иерархического положения каждого API в стеке программного обеспечения; разработку двух вариантов программного обеспечения для микроконтроллера STM32F407 с последующим измерением времени переключения контекста и объема используемой Flash- и RAM-памяти; формирование научно-обоснованных рекомендаций для систем с жесткими требованиями реального времени и долгоживущих программных продуктов.

Основная часть. Особенности архитектуры интерфейсов и методология применения. Программная многозадачность в системах на STM32 обычно реализуется либо посредством прямых вызовов функций FreeRTOS, либо через специализированную надстройку CMSIS-RTOS v2, генерируемую инструментом STM32CubeMX. Оба представленных метода базируются на использовании идентичного ядра, однако существенно различаются по структуре прикладного кода и степени зависимости от конкретного поставщика программных решений.

Нативный API FreeRTOS обеспечивает разработчику полный доступ к сервисам ядра, включая прецизионную настройку стеков, приоритетов, таймеров и очередей сообщений. Такой подход целесообразен в случаях необходимости достижения максимальной временной детерминированности, однако синтаксис имен функций остается специфичным для FreeRTOS, что требует переработки прикладного уровня при миграции проекта.

Интерфейс CMSIS-RTOS v2 является стандартом для унификации системных вызовов. Прикладное программное обеспечение обращается к таким функциям, как osThreadNew или osMessageQueuePut, а внутренняя реализация пакета CMSIS-RTOS2 для FreeRTOS преобразует их в соответствующие инструкции xTaskCreate и xQueueSend. Данный дополнительный уровень абстракции предназначен не для замены ядра, а для упрощения переноса исходного кода между различными портами RTOS.

Анализ производительности и потребления ресурсов памяти. Экспериментальные замеры проводились с использованием отладочной платы STM32F407 Discovery. Два программных образца, обладающих идентичной логикой функционирования, различались исключительно методом формирования задач и примитивов синхронизации. Определение длительности переключения контекста осуществлялось с помощью аппаратного счетчика циклов DWT; итоговые показатели представлены в таблице 1.

Таблица 1

Сравнение потребления ресурсов

Параметр	FreeRTOS (Native API)	CMSIS-RTOS v2	Отклонение
Объем памяти программ, КБ	15.0	15.5	+3.3 %
Используемая ОЗУ, КБ	5.0	5.1	+2.0 %
Время переключения контекста, такты	185	197	+6.5 %

Согласно данным, приведенным в таблице 1, конфигурация с применением CMSIS-RTOS v2 характеризуется повышенным потреблением Flash- и RAM-ресурсов. Увеличение объема программного кода приблизительно на 512 байт обусловлено наличием функций-оберток и внутренних таблиц объектов операционной системы. Дополнительные структуры в оперативной памяти необходимы для хранения идентификаторов объектов, таких как osThreadId или osMutexId, в соответствии с требованиями стандарта.

Наиболее значимое различие зафиксировано в показателях времени переключения контекста, составившее +12 тактов ядра (около 6.5 % относительно варианта без надстройки). Удлинение цепочки вызовов по схеме «приложение — CMSIS-RTOS v2 — адаптер FreeRTOS — переключение контекста» вносит дополнительные инструкции до момента активации планировщика.

Особенности практической реализации в проектах. При непосредственном взаимодействии с FreeRTOS разработчику доступны специфические механизмы, например, группы событий (event groups) или потоковые буферы (stream buffers), которые не всегда находят отражение в спецификациях CMSIS. Использование данного подхода оправдано при высоком уровне компетенций команды в области архитектуры конкретного ядра.

Напротив, CMSIS-RTOS v2 намеренно ограничивает перечень доступных примитивов, что позволяет снизить вариативность стиля кодирования и упростить процесс адаптации новых участников проекта. Обобщенное сопоставление подходов по ключевым критериям отражено в таблице 2.

Таблица 2

Сравнительные характеристики подходов

Критерий	FreeRTOS	CMSIS-RTOS v2
Производительность	Высокая	Умеренная
Потребление памяти	Оптимальное	Увеличенное на 2-5%
Переносимость кода	Низкая	Высокая
Кривая обучения	Крутая	Пологая
Поддержка инструментов	Ограниченная	Широкая
Гибкость настройки	Полная	Ограниченная

В условиях высокой вычислительной нагрузки выявленные различия проявляются более интенсивно. В сценариях, предполагающих функционирование более десяти задач с активным межпроцессным взаимодействием, программное обеспечение на «чистом» FreeRTOS опережало решение на базе CMSIS в среднем на 8–12 % по времени реакции. Таким образом, при наличии в техническом задании жестких временных ограничений и требований к минимизации джиттера, целесообразно использовать нативные вызовы. В случае допустимости незначительных издержек по тактам процессора ради унификации API, применение CMSIS выглядит более практичным.

Аспект экономии памяти критичен для микроконтроллеров с объемом RAM менее 32 КБ и Flash менее 128 КБ, где избыточные затраты памяти могут препятствовать реализации сетевых протоколов. В таких системах приоритет отдается прямой работе с FreeRTOS. На вычислительных узлах уровней F4, F7 или H7 с избыточными ресурсами накладные расходы на CMSIS-RTOS v2 становятся малозаметными на фоне использования библиотек HAL или стеков TCP-IP.

Заключение. По результатам проведенного экспериментального исследования могут быть сформулированы следующие рекомендации: применение нативного интерфейса FreeRTOS целесообразно для высокооптимизированных проектов с минимальными задержками и дефицитом ресурсов памяти; использование CMSIS-RTOS v2 предпочтительно в случаях, когда приоритетными являются переносимость кода, единообразие стиля разработки и интеграция в экосистему STM32Cube. Представленные данные не претендуют на абсолютную универсальность, так как относятся к конкретной аппаратной платформе и версии программных пакетов, однако выявленные закономерности носят воспроизводимый характер. Выбор API представляет собой компромисс между производительностью и удобством сопровождения изделия.

Список литературы

1. *CMSIS-RTOS API v2 (CMSIS-RTOS2).* URL: https://www.armbbs.cn/forum.php/forum.php?mod=viewthread&action=printable&tid=86516 (дата обращения: 16.05.2024).

2. *STM32CubeMx. Быстрый старт с FreeRTOS для STM32.* URL: https://microtechnics.ru/stm32cubemx-bystryj-start-s-freertos-dlya-stm32/ (дата обращения: 16.05.2024).

3. *Установка и настройка среды разработки STM32CubeIDE.* URL: https://robotchip.ru/ustanovka-i-nastroyka-sredy-razrabotki-stm32cubeide/ (дата обращения: 16.05.2024).

Об авторах:

Константин Иванович Бережной, магистрант кафедры «Электротехника и электроника» Донского государственного технического университета (344003, Российская Федерация, г. Ростов-на-Дону, пл. Гагарина, 1), beregnoy.konstanten@gmail.com

Рагим Рауфевич Ибадов, старший преподаватель кафедры «Электротехника и электроника» Донского государственного технического университета (344003, Российская Федерация, г. Ростов-на-Дону, пл. Гагарина, 1), ragim_ibadov@mail.ru

Анатолий Александрович Лаврентьев, доктор технических наук, профессор, заведующий кафедрой «Электротехника и электроника» Донского государственного технического университета (344003, Российская Федерация, г. Ростов-на-Дону, пл. Гагарина, 1), alavrentyev52@mail.ru

Конфликт интересов: **авторы заявляют об отсутствии конфликта интересов.**

Все авторы прочитали и одобрили окончательный вариант рукописи.

About the Authors:

Konstantin I. Berezhnoy, Master's Degree Student of the Electrical Engineering and Electronics Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, Russian Federation), beregnoy.konstanten@gmail.com

Rahim Ibadov, Senior Lecturer of the Electrical Engineering and Electronics Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, Russian Federation), ragim_ibadov@mail.ru

Anatoly A. Lavrentiev, Dr.Sci, (Engineering), Professor, Head of the Electrical Engineering and Electronics Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, Russian Federation), alavrentyev52@mail.ru

Conflict of Interest Statement: **the authors declare no conflict of interest.**

All authors have read and approved the final manuscript.