Методика технического анализа возможностей систем реального времени на различных аппаратных платформах

Таран Евгений Вячеславович

Синельников Евгений Александрович

1 Введение

Отправной точкой данного исследования стала работа над продуктом "Альт Контроллер который представляет собой адаптацию операционной системы Альт под работу на программируемых логических контроллерах (ПЛК).

ПЛК - специализированные вычислительные системы, используемые в промышленности для управления технологическими процессами, станками, конвейерами, системами оповещения, любыми автоматическими механизмами в режиме реального времени. Такие системы имеют широкое применение в сфере автоматизации и могут быть построены на базе различных аппаратных платформ с широким спектром использования коммуникационных интерфейсов.

Специфика использования и внутренней логики работы ПЛК показывает, что одной из ключевых характеристик каждой такой системы является величина средней скорости и детерминированность отклика контроллера на некоторые события внутри системы. В зависимости от решаемой контроллером задачи таким событием может быть как очередные данные

для обработки от одного из устройств системы, так и важное сообщение о чрезвычайном происшествии в системе, требующей немедленных действий. Такие события принято называть критическими.

На этом шаге возникает противоречие: классический ПЛК — это детерминированная система реального времени, а операционная система на базе ядра Linux — это система общего назначения, где процесс печати или обновления драйвера может «отобрать» время у управления важным аппаратным модулем. Тем не менее, существует большое количество решений на рынке ПЛК, которые по различным причинам используют именно Linux в качестве бортовой системы. Среди таких причин можно выделить возможность тонкой настройки ядра с применением патчей реального времени, что позволяет свести задержки к минимуму и гарантировать детерминированное поведение критически важных потоков управления. Не менее значимы широкая аппаратная поддержка и развитый сетевой стек, упрощающие интеграцию контроллера в сложные IIoT-инфраструктуры. Именно на решение задач по сочетанию мягкого реального времени с этой функциональной гибкостью и нацелен "Альт Контроллер".

Указанное противоречие вызвало необходимость выработки ключевых метрик и способа оценки соответствия требованиям детерминированности операционной системы, размещённой на конкретной аппаратной базе.

Целью данного исследования стала разработка методики технического анализа систем реального времени на различных аппаратных платформах, которую в дальнейшем можно будет использовать для анализа возможности применения продукта “Альт Контроллер” для решения конкретных задач заказчика.

В соответствии с целью были установлены задачи исследования:

- выработка метрик оценки исследуемой системы с точки зрения задач реального времени;

- построение адаптируемого тестового стенда, применимого для большинства технических решений;
- агрегация полученных в ходе тестирования данных в удобной для последующего анализа форме.

Отдельной стороной этого исследования стало включение элементов исследования в практический курс дисциплины “Системы реального времени” на факультете КНиИТ.

2 Разработка тестового стенда

2.1 Метрики оценки

При оценке пригодности программно-аппаратной платформы для задач реального времени ключевыми являются метрики, характеризующие временные задержки и их вариативность [1]. Можно выделить две основные метрики, применяемые для сравнительного анализа RT-решений: латентность (задержка, время отклика) и RMS-джиттер.

Латентность определяется как интервал времени между моментом возникновения внешнего события (поступлением аппаратного прерывания) и моментом начала обработки этого события целевой системой [2]. Латентность характеризует абсолютную величину задержки реакции системы.

Для систем управления технологическими процессами (ПЛК) латентность определяет время от обнаружения важного события до начала управляющего воздействия. Чем меньше значение латентности, тем быстрее система реагирует на внешние события.

Математически латентность вычисляется по формуле:

$$L_i = t_{\text{response},i} - t_{\text{event},i}$$

где L_i — значение латентности для i-го измерения, $t_{\text{event},i}$ — временная метка возникновения внешнего события (момент переключения входного сигнала), $t_{\text{response},i}$ — временная метка начала реакции системы (момент изменения выходного сигнала).

Джиттер характеризует вариативность (нестабильность) временны́х интервалов между последовательными событиями относительно среднего значения латентности [3]. Высокий джиттер означает значительный разброс значений латентности, что критично для задач управления, требующих равномерной дискретизации (например, ШИМ-управление).

В контексте периодических задач реального времени джиттер определяется как отклонение фактического межсобытийного интервала от среднего значения латентности:

$$J_i = |L_i - \bar{L}|$$

где J_i — значение джиттера для i-го измерения, L_i — значение латентности для i-го измерения, $\bar{L} = \frac{1}{N}\sum_{i=1}^{N} L_i$ — среднее арифметическое значение латентности по N измерениям.

Для интегральной оценки стабильности системы используется среднеквадратичное отклонение (RMS-джиттер):

$$\sigma_J = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(L_i - \bar{L})^2}$$

2.2 Построение стенда и установка методики тестирования

Общая методология измерения латентности и джиттера была заимствована из работы [2], авторы которой предложили использовать внешнюю измерительную систему на базе микроконтроллера для объективной оценки временных характеристик без присущих самой системе способов измерений. Данный подход согласуется с рекомендациями по оценке производительности систем реального времени [1]. В данном исследовании аналогич-

ный подход был адаптирован и реализован с использованием платформы Arduino Mega в качестве внешнего измерителя.

Кроме того, для исследования латентности исследуемых устройств по аналогии с работой [2] было принято решение сфокусироваться на 5% наихудших показателей задержки, поскольку именно максимально длительные периоды от генерации внешнего события до его обработки представляют главную ценность в оценке скорости отклика системы.

Разработанный измерительный стенд [4] включает три компонента: внешний измеритель на базе микроконтроллера Arduino Mega, тестируемую вычислительную платформу и управляющий компьютер для координации измерений и сбора данных.

Внешний измеритель реализован на платформе без операционной системы, что обеспечивает отсутствие дополнительных временных задержек, вносимых планировщиком или иными программными компонентами.
Arduino Mega выполняет генерацию периодических импульсов на выходном пине, фиксацию момента получения ответа от тестируемой системы на входном пине, вычисление временных интервалов между событиями и накопление статистики по группам измерений.

Управляющий компьютер обеспечивает координацию процесса измерений, сбор данных от измерителя через последовательный интерфейс, их сохранение в базу данных и последующий анализ с визуализацией результатов.

Тестируемая вычислительная платформа представляет собой одноплатный компьютер с установленной операционной системой на базе ядра Linux. На данной платформе функционирует программа, реализующая ответную логику на импульсы от Arduino: при поступлении сигнала на входной GPIO-вывод платформа формирует ответный сигнал на выходном GPIO-выводе. Программа реализует два основных потока исполнения: поток обработки GPIO-событий с приоритетом реального времени и поток сбора метрик за-

грузки системы.

Отдельное внимание было уделено подбору приоритетов планировщика задач для процессов, участвующих в тестировании. Приняв во внимание рекомендации из книги [1] и проведя подбор в процессе исследования были зафиксированы следующие приоритеты процессов: в режиме FIFO максимальный приоритет (99) был установлен для потока прерывания, по которому программа на измерителе считывает изменения фронта на линии GPIO, повышенный приоритет был назначен основному потоку программы (80), отвечающему за генерацию ответного импульса и потоку фиксирующему текущую нагрузку на процессор и память (79).

При такой конфигурации приоритетов планировщика наблюдалась наименьшая латентность системы и при этом поток, ответственный за мониторинг нагрузки получал достаточно ресурсов, чтобы каждой группе измерений были сопоставлены средние и максимальные значения параметров нагруженности системы.

Измерение латентности выполняется по следующему алгоритму:

1. Arduino генерирует периодический сигнал (меандр) с заданным интервалом T_{gen} на выходном пине 7;
2. Тестируемая платформа принимает сигнал через входной GPIO и формирует ответный сигнал на выходном GPIO;
3. Arduino фиксирует момент поступления ответа на входном пине 20;
4. Латентность вычисляется как разность между моментом получения ответа и моментом генерации импульса: $L = t_{response} - t_{pulse}$.

По результатам первых тестовых измерений было решено зафиксировать основные параметры измерительной сессии:

Таблица 1: Параметры измерительной сессии

Параметр	Значение по умолчанию	Описание
Количество групп	2500 измерений	Количество групп в одной сессии
Размер группы	1500 измерений	Количество измерений в одной группе
Длительность импульса	300 мкс	Полупериод генерации меандра
Количество худших	75 (5% от размера)	N наибольших задержек в группе
Порог латентности	250 мкс	Временной порог для классификации превышений

Под порогом латентности (задержки) в данном случае понимается установленная эмпирически временная величина, превышение которой означает приближение к длительности интервала между подаваемыми измерителем импульсами(по умолчанию 300 мкс).

3 Применение стенда для тестирования различных платформ

В рамках исследования были протестированы три современные одноплатные вычислительные платформы, построенные на различных аппаратных архитектурах:

Lichee RV Dock — бюджетная платформа на базе RISC-V процессора XuanTie C906 C906 (1 ГГц) и модулем оперативной памяти на 512 Мбайт. Плата оснащена разъёмом GPIO для подключения внешних устройств и предназначена для применения во встраиваемых системах и интернет вещах.

Radxa ROCK 5B — высокопроизводительная платформа на базе ARM процессора Rockchip RK3588 5B с восемью процессорными ядрами (4×Cortex-A76 + 4×Cortex-A55) и модуля оперативной памяти на 8 Гбайт. Данная платформа позиционируется как решение для edge-вычислений и промышленных применений.

StarFive VisionFive 2 — довольно производительный одноплатный ком-

пьютер на базе восьмиядерного RISC-V процессора JH7110 и модуля оперативной памяти на 8 Гбайт. Плата разработана компанией StarFive и предназначена для различных задач: систем умного дома, промышленности и тд.

Тестирование проводилось на перечисленных платформах с двумя конфигурациями ядра Linux: стандартное ядро и ядро с патчем PREEMPT_RT.

Патч PREEMPT_RT представляет собой набор модификаций ядра Linux, направленных на повышение прерываемости kernel space [1, 3]. Основные механизмы работы патча включают: вынесение обработчиков прерываний в отдельные потоки ядра, замену spinlock-ов на mutex там, где это допустимо, и реализацию механизмов управления высокоточными таймерами. Применение патча позволяет существенно сократить максимальную латентность системы, однако сопряжено с рядом издержек: увеличением накладных расходов на планирование (что может повышать среднюю латентность), некоторым снижением общей пропускной способности системы и необходимостью поддержки отдельной ветки ядра.

На рисунке 1 представлен пример графика, иллюстрирующего результаты одной из измерительных сессий. График демонстрирует динамику латентности и джиттера во времени с наложением информации о загрузке процессора и памяти платы Lichee RV Dock.

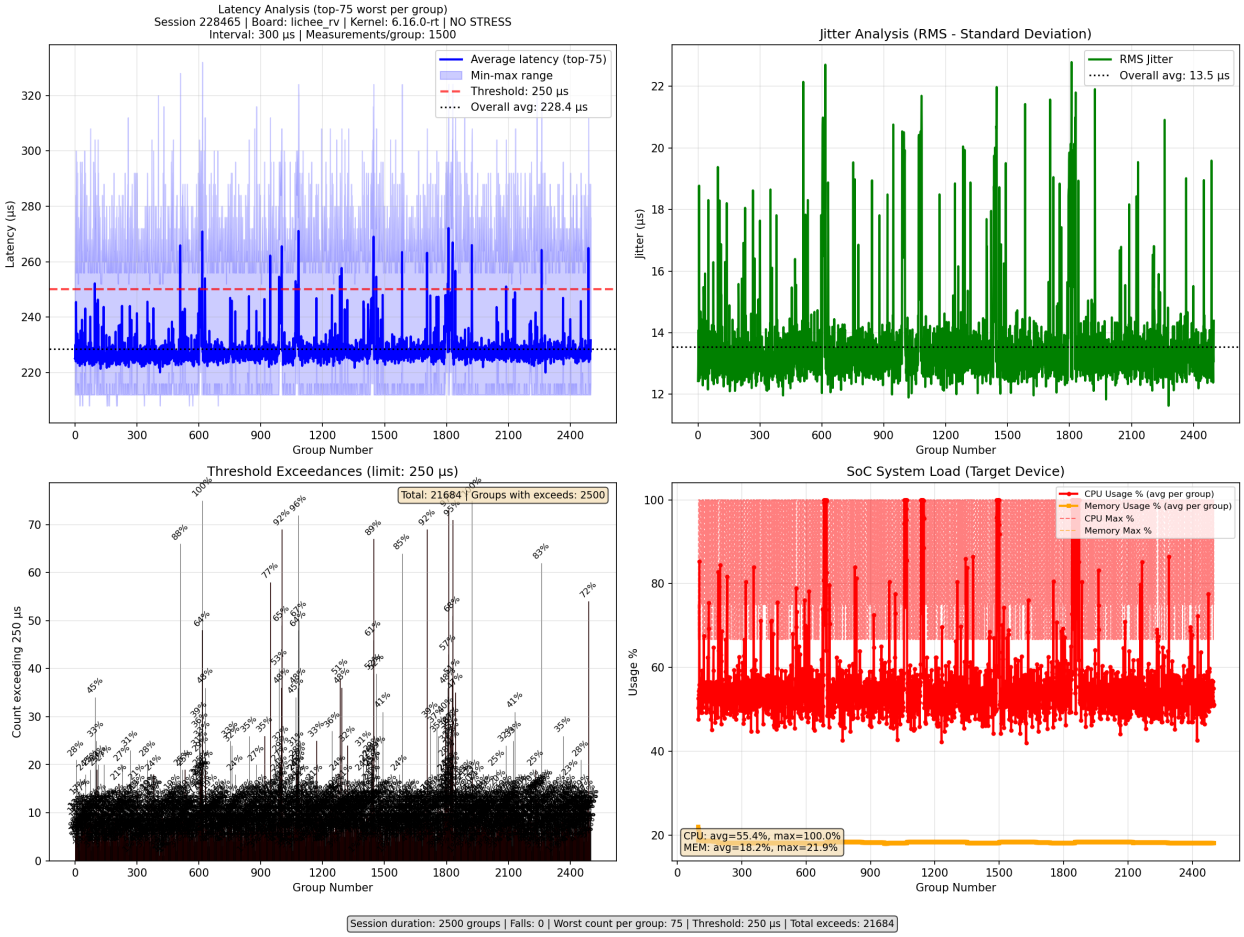

Рис. 1: Динамика латентности и джиттера во времени с отображением нагрузки

Для каждой конфигурации указаны: наличие падений системы (ситуация, при которой измерительная программа на тестируемой платформе не формировала новую группу измерений в течение 2 секунд, что свидетельствует о полном зависании ответа на прерывания), количество превышений временного порога, максимальная зарегистрированная задержка, средняя задержка среди 5% наихудших значений, а также средний RMS-джиттер.

Испытания проводились как в условиях отсутствия внешней нагрузки, так и при искусственной нагрузке системы утилитой stress-ng.

Сводные результаты тестирования представлены в таблице 1.

Таблица 2: Результаты тестирования одноплатных компьютеров в пользовательском пространстве

SoC	Конфигурация	Падения	Превышения	Макс. задержка	Ср. задержка	RMS-джиттер
Lichee	6.12	2	45 812	335	249,8	17,1
Lichee	6.12+stress	0	95 198	575	252,8	18,8
Lichee	6.12-rt	0	21 684	330	228,4	13,5
Lichee	6.12-rt+stress	0	93 643	325	251,1	16,9
Radxa	6.12.76	0	0	150	29,5	2,5
Radxa	6.12.76+stress	2	3 647	350	66,3	11,2
Radxa	6.12.76-rt	0	0	125	36,5	3,6
Radxa	6.12.76-rt+stress	0	99	360	39,2	4,0
VisionFive	6.18	0	5 033	350	164,7	25,8
VisionFive	6.18+stress	0	71 240	600	185,4	36,6
VisionFive	6.18-rt	0	3 469	310	172,2	27,4
VisionFive	6.18-rt+stress	0	42 934	520	149,0	26,1

Анализ данных, представленных в таблице 1, позволяет сделать ряд выводов о характеристиках исследуемых платформ.

Наиболее высокие показатели демонстрирует одноплатный компьютер Radxa ROCK 5B, которая в конфигурации с патчем PREEMPT_RT показывает минимальные значения средней задержки среди 5% наихудших случаев (36,5–39,2 мкс) и наименьший RMS-джиттер (3,6–4,0 мкс). При этом данная платформа не испытывает падений системы в принципе, а количество превышений временного порога остаётся крайне низким (99 случаев с патчем против 3647 без патча).

Платформа StarFive VisionFive v2 занимает промежуточное положение с показателями средней задержки 149,0–185,4 мкс и RMS-джиттером 25,8–36,6 мкс. Можно отметить, что применение патча PREEMPT_RT на данной платформе под нагрузкой приводит к снижению средней задержки с

185,4 до 149,0 мкс, что свидетельствует о положительном влиянии патча на стабильность временных характеристик в условиях высокой загрузки системы.

Платформа Lichee RV Dock демонстрирует наихудшие результаты с высокой базовой задержкой (249,8–255,6 мкс), что приближается к установленному порогу в 250 мкс. Положительное влияние патча PREEMPT_RT также можно заметить по результатам тестирования этого бюджетного одноплатного компьютера: значения максимальной и средней длительности отклика значительно снижены в тестах с аналогичной нагрузкой.

Сравнительный анализ влияния патча PREEMPT_RT показывает его различную эффективность на разных платформах. На Radxa патч незначительно увеличивает среднюю задержку (с 29,5 до 36,5 мкс), но радикально снижает количество превышений порога под нагрузкой. На VisionFive патч улучшает показатели под нагрузкой, снижая как задержку, так и количество превышений.

Устойчивость платформ к внешней нагрузке также варьируется: Radxa демонстрирует наивысшую устойчивость с минимальным ухудшением показателей, VisionFive показывает значительное увеличение количества превышений порога под нагрузкой (в 14 раз), а Lichee, имея изначально высокие задержки, демонстрирует относительно стабильные показатели при нагрузке.

Результаты тестирования демонстрируют возможности и ограничения предложенной методики.

Предложенная методика позволяет оценить и сравнить:

- абсолютные значения латентности и джиттера для различных платформ;
- эффективность использования различных вариаций операционной системы и ПО на конкретном аппаратном обеспечении;

- устойчивость работы исследуемой системы к внешней нагрузке;

Вместе с тем, методика имеет ограничения. Тестовая программа реализует лишь базовую логику ответа на прерывания без дополнительной обработки данных. Испытания проводились на ограниченном наборе платформ.

Для улучшения методики целесообразно реализовать kernelspace-вариант тестовой программы для получения более объективной оценки [5], добавить моделирование типичной нагрузки реальных приложений [5], расширить перечень тестируемых платформ и ввести длительные испытания (несколько часов) для выявления редких событий с большой латентностью.

Дополнительно, для комплексного анализа производительности систем реального времени можно использовать специализированные инструменты оценки, такие как cyclictest для измерения задержек планирования и Ftrace для анализа источников задержек.

4 Включение элементов исследования в практический курс дисциплины "Системы реального времени"

Представленная в исследовании методология оценки систем реального времени, привела к идее разработки отдельного цикла лабораторных в рамках практического курса по дисциплине "Системы реального времени". Цикл построен по принципу поэтапного освоения: от анализа временных характеристик на уровне микроконтроллеров до исследования детерминизма в операционной системе Linux.

Так например, первая и третья лабораторные работы посвящены реализации генераторов импульсных сигналов на платформе Arduino. В первой работе используется подход прямого управления аппаратными таймерами

и регистрами портов микроконтроллера, что демонстрирует принципы минимизации накладных расходов при работе с микросекундными интервалами. Третья работа представляет альтернативную реализацию на основе кооперативного планировщика задач, раскрывая влияние алгоритмов планирования на временные характеристики. В обеих работах для отладки и верификации работы прошивок применяется осциллограф, что формирует у студентов навыки инструментальной проверки временных параметров.

Вторая лабораторная работа реализует концепцию объективных внешних измерений. На отдельной плате Arduino по аналогии с исследовательским стендом создаётся измерительная система, использующая внешние прерывания для фиксации фронтов импульсов и вычисления статистических показателей: минимальной, максимальной и средней длительности импульса, а также среднеквадратичного отклонения. Этот подход исключает смещение от самоизмерения тестируемой системы и соответствует методологии, описанной в исследовании [2]. Формализация результатов в JSON-отчётах вводит студентов в практику структурированного представления экспериментальных данных и позволяет в дальнейшем сравнить между собой реализации генераторов и измерительной системы с точки зрения точности генерируемых сигналов и их фиксации.

Четвёртая работа интегрирует предыдущие этапы, реализуя сравнительный анализ генераторов из первой и третьей работ с использованием измерителя из второй работы. Результатом ее выполнения является обобщенная статистика работы генераторов в табличном формате, что развивает методику сравнительной оценки разных подходов к реализации систем реального времени.

Пятая лабораторная работа расширяет цикл переходом к системам на основе Linux в качестве исследуемой системы. На одноплатном компьютере Lichee RV Dock (платформе, используемой в исследовании) реализуется генератор меандра с управлением через библиотеку libgpiod. Работа иссле-

дует влияние патча PREEMPT_RT на детерминизм временных характеристик при разных приоритетах планирования, что соответствует экспериментальной части настоящего исследования.

Концепция разрабатываемого цикла заключается в последовательном освоении методологии оценки систем реального времени: от понимания важности внешних измерений и владения метриками латентности и джиттера до навыков статистического анализа и сравнительной оценки реализаций. Практическая работа с инструментами и платформами, используемыми в исследовании, позволяет расширить данный цикл лабораторных, за счёт более глубокого изучения системы Linux в контексте систем реального времени, использования разных аппаратных решений для оценки их соответствия требованиям систем реального времени.

5 Заключение

Проведённое исследование привело к разработке методики технического анализа возможностей систем реального времени на различных аппаратных платформах. Методология основана на измерении латентности и джиттера с использованием внешней измерительной системы, что исключает смещение от самоизмерения тестируемой системы.

Экспериментальная часть исследования продемонстрировала применение методики на трёх одноплатных компьютерах разной мощности и назначения: Lichee RV Dock, Radxa ROCK 5 Model B и StarFive VisionFive 2. Сравнительный анализ позволил оценить влияние патча PREEMPT_RT на временные характеристики систем, подтвердив его эффективность для снижения максимальной латентности, а также оценить с точки зрения задач реального времени каким требованиям могут соответствовать исследуемые устройства с операционной системой Альт на борту.

Разработанная методика интегрирована в учебный процесс через раз-

работку цикла лабораторных работ, который реализует поэтапное освоение принципов оценки систем реального времени. Цикл охватывает поэтапное освоение: от анализа временных характеристик на уровне микроконтроллеров до исследования детерминизма в операционных системах реального времени. Дальнейшее расширение цикла практических работ позволит установить преемственность между учебной практикой и профессиональной деятельностью.

Перспективы развития методики включают реализацию kernelspace-варианта тестовой программы для более объективной оценки, добавление моделирования типичной нагрузки реальных приложений, расширение перечня тестируемых платформ и введение длительных испытаний для выявления редких событий с большой латентностью.

Представленная методика предлагает основу для систематической оценки пригодности программно-аппаратных платформ для задач реального времени, что имеет практическое значение для разработки промышленных систем управления, встраиваемых устройств и других приложений, требующих детерминированного поведения.

Инструкция к использованию исследовательского стенда, результаты измерения и исходный код всех его компонентов публикуются в открытом репозитории на ресурсе [4].

Список литературы

[1] Крис Симмондс. Встраиваемые системы на основе Linux. ДМК Пресс, М., 2017.

[2] J. H. Brown and B. Martin. How fast is fast enough? choosing between xenomai and linux for real-time applications. Technical report, Rep Invariant Systems, Inc., 2012.

[3] Дауд Абдо and В. М. Квашнин. Linux в системах реального времени: подходы, возможности и анализ производительности. Вестник науки, 4(12):1580–1591, 2024.

[4] Alt Linux Space. Репозиторий с исходным кодом тестового стенда и результатами измерений. https://altlinux.space/besogon1238/rt-tester, 2026. Дата обращения: 23.04.2026.

[5] Microchip Technology Inc. Real time solutions on at91sam soc. https://developerhelp.microchip.com/xwiki/bin/view/applications/linux4sam/faq/realtime/, 2025. Дата обращения: 23.04.2026.