УДК 004

# ГИБРИДНЫЙ ПОДХОД К ИНТЕГРАЦИИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В ЖИЗНЕННЫЙ ЦИКЛ РАЗРАБОТКИ ВСТРАИВАЕМЫХ СИСТЕМ

**Нагимзянов Шамиль Джалилович,** магистрант, направление подготовки 09.04.01 Информатика и вычислительная техника, Казанский государственный энергетический университет, Казань, Россия
e-mail: shamnag@bk.ru

**Сибаева Гульназ Рашитовна,** кандидат экономических наук, доцент кафедры цифровых систем и моделей, Казанский государственный энергетический университет, Казань, Россия
e-mail: sibaeva.gr@kgeu.ru

***Аннотация.*** *Цель – разработка архитектурной модели гибридной системы на основе Retrieval-Augmented Generation (RAG) для безопасного и контролируемого внедрения искусственного интеллекта (ИИ) в процессы разработки встраиваемых (embedded) систем. На основе анализа научной литературы и потребностей индустрии выявлен ключевой пробел: отсутствие методологий, которые сочетают операционную эффективность ИИ с гарантией безопасности и управляемости в условиях ресурсных ограничений. Для его преодоления предложена архитектура, интегрирующая детерминированный семантический поиск по векторной базе знаний (спецификации, даташиты, код) с языковой моделью и автоматическим контуром валидации через симулятор. Предложенная RAG-система позволяет сократить время на рутинные задачи разработки и тестирования, минимизируя при этом фундаментальный риск «галлюцинаций» ИИ за счет привязки генерации к верифицированным источникам. Система обеспечивает объяснимость рекомендаций и может быть развернута в условиях ограниченных вычислительных ресурсов.*

*Гибридная RAG-архитектура представляет собой практичный и научно обоснованный инструмент для безопасной интеграции ИИ-ассистентов в embedded-разработку, превращая технологию из источника рисков в управляемый фактор конкурентного преимущества.*

***Ключевые слова:*** *искусственный интеллект, встроенные системы, микроконтроллеры, разработка ПО, тестирование, автоматизация, Retrieval-Augmented Generation, безопасность, галлюцинации ИИ, гибридная архитектура.*

***Для цитирования:*** Нагимзянов Ш. Д., Сибаева Г. Р. Гибридный подход к интеграции искусственного интеллекта в жизненный цикл разработки встраиваемых систем // Шаг в науку. – 2026. – № 1. – С. 17–22.

# HYBRID APPROACH TO INTEGRATING AI INTO THE EMBEDDED SYSTEMS DEVELOPMENT LIFECYCLE

**Nagimzyanov Shamil Djalilovich,** postgraduate student, training program 09.04.01 Computer Science and Computer Engineering, Kazan State Power Engineering University, Kazan, Russia
e-mail: shamnag@bk.ru

**Sibaeva Gulnaz Rashitovna,** Candidate of Economic Sciences, Associate Professor of the Department of Digital Systems and Models, Kazan State Power Engineering University, Kazan, Russia
e-mail: sibaeva.gr@kgeu.ru

***Abstract.*** *Purpose. Development of an architectural model for a hybrid system based on Retrieval-Augmented Generation (RAG) for the safe and controlled integration of artificial intelligence (AI) into embedded systems development processes. Methods. Based on the analysis of scientific literature and industry needs, a key gap is identified: the lack of methodologies that combine the operational efficiency of AI with guaranteed safety and controllability under resource constraints. To overcome this gap, an architecture is proposed that integrates deterministic semantic search over a vector knowledge base (specifications, datasheets, code) with a language model and an automatic validation loop using a simulator. Results. The proposed RAG system can reduce the time spent on routine development and testing*

*tasks while minimizing the fundamental risk of AI "hallucinations" by linking generation to verified sources. The system ensures the explainability of recommendations and can be deployed in environments with limited computational resources. Conclusions. The hybrid RAG architecture represents a practical and scientifically grounded tool for the safe integration of AI assistants into embedded development, transforming the technology from a source of risk into a manageable factor of competitive advantage.*

***Key words:*** *artificial intelligence, embedded systems, microcontrollers, software development, testing, automation, Retrieval-Augmented Generation, safety, AI hallucinations, hybrid architecture.*

***Cite as:*** Nagimzyanov, Sh. D., Sibaeva, G. R. (2026) [Hybrid approach to integrating AI into the embedded systems development lifecycle]. *Shag v nauku* [Step into science]. Vol. 1, pp. 17–22.

## Введение

Цифровая трансформация экономики привела к высокому росту количества цифровых устройств и объёмов данных, которыми необходимо управлять. В этом контексте искусственный интеллект (ИИ) выделяется как один из ключевых мегатрендов, способный революционизировать бизнес-процессы и инженерные практики [2].

Особую актуальность использование ИИ приобретает для компаний, работающих на рынке «умного» дома и интернета вещей (IoT), например, производителей систем автоматизации климата. В условиях жесткой конкуренции, где важна скорость вывода продукта, надежность и безопасность, традиционные методы разработки встраиваемых (embedded) систем, сталкивающиеся со сложностью интеграции «железа» и ПО, ограничениями памяти и энергопотребления, становятся узким местом. Исследования показывают, что внедрение ИИ-инструментов в смежных областях разработки позволяет сократить время на рутинные задачи до 30% и снизить количество ошибок на ранних этапах [3]. Таким образом, задача интеграции ИИ в процессы embedded-разработки перестает быть факультативной и становится стратегической необходимостью для сохранения конкурентоспособности.

## Научная проблема и анализ существующих работ

Несмотря на растущий интерес к теме, в научной литературе сохраняется существенный пробел. Многие исследования носят общий характер, фокусируясь на макроэкономических эффектах цифровизации или декларируя общие преимущества ИИ для бизнеса. Отдельные работы подчеркивают технологическую революционность ИИ в инженерии ПО, но часто рассматривают крупные IT-корпорации с неограниченными ресурсами. Критически важные для embedded-разработки риски, такие как «галлюцинации» языковых моделей при генерации низкоуровневого кода или аппаратно-зависимых функций, лишь констатируются как проблема, однако конкретные методологии их нивелирования, адаптированные для средних и малых инженерных команд, практически не предлагаются.

Работы таких авторов, как Alenezi M. и Akour M., предоставляют обширный и систематический обзор инноваций, которые ИИ привносит в инженерию ПО в целом [3]. Они охватывают широкий спектр возможностей: от автоматической генерации кода и тестовых сценариев до создания документации. Сильной стороной данного подхода является его широта, подтверждающая трансформационный характер ИИ для всей отрасли. Однако для embedded-разработки этот обзор оказывается слишком общим. Он не учитывает критически важные для данной области специфические ограничения, такие как жесткие лимиты оперативной памяти и энергопотребления, требования реального времени и повышенные стандарты безопасности (safety-critical systems). Таким образом, работа фиксирует общий тренд, но не предлагает решений для его адаптации к целевой предметной области.

Другой пласт исследований, представленный, например, работой Зариповой Р. С. и соавторов, фокусируется на макроэкономической эффективности внедрения ИИ, часто в национальном контексте [1]. Подобные исследования важны для подтверждения стратегического тренда цифровизации и его вклада в экономический рост. Их сила – в анализе на системном уровне. Тем не менее, они остаются на уровне экономических обобщений и не спускаются до уровня практических инженерных рецептов, архитектурных решений или конкретных методологий, которые могли бы быть использованы командой разработчиков при создании прошивки для микроконтроллера.

Критически важным, но узкоспециализированным является вклад авторов, подобных Baum N. и Marinkovic P., которые исследуют фундаментальные риски технологий ИИ, в частности феномен «галлюцинаций» языковых моделей [5]. Их анализ, проведенный на примере медицины, имеет прямое отношение к embedded-разработке, где ошибка в коде может привести к физическим последствиям. Они четко идентифицируют проблему, неприемлемую для safety-систем. Однако предложенные ими меры – чаще всего общие рекомендации по проверке и контролю со стороны человека – не являются техническим решением, интегрируемым непосредственно в процесс разработ-

ки. Это констатация риска без предложения инженерного механизма для его автоматического устранения или минимизации.

Наконец, собственный анализ потребностей индустрии позволяет выявить конкретные, насущные задачи: автоматизация написания драйверов периферии, оптимизация использования памяти и энергии, генерация тестовых случаев, учитывающих контекст конкретного «железа». Этот анализ подтверждает высокий спрос на инструменты ИИ, но одновременно обнажает главный пробел: отсутствие проверенной, ресурсоэффективной и безопасной методологии внедрения, которая бы учитывала все перечисленные выше ограничения и риски.

Таким образом, научная проблема, на решение которой направлена данная статья, формулируется следующим образом: отсутствие проверенных архитектурных моделей и методологий для безопасного, контролируемого и эффективного внедрения искусственного интеллекта в процессы разработки встраиваемых систем, учитывающих их специфические ограничения и риски, а также ориентированных на команды с ограниченными вычислительными ресурсами.

Целью данного исследования является разработка архитектурной модели гибридной системы на основе Retrieval-Augmented Generation (RAG), предназначенной для интеграции в процессы embedded-разработки и тестирования, которая обеспечивает повышение производительности при гарантированном контроле качества и безопасности генерируемых артефактов.

**Методология и обоснование выбора подхода**

Для достижения поставленной цели в качестве основного метода предлагается архитектура гибридной системы на базе Retrieval-Augmented Generation (RAG). Выбор данного метода является прямым следствием анализа научной проблемы и работ других авторов.

RAG – это не просто инструмент, а архитектурный метод, позволяющий крупным языковым моделям (LLM) при генерации ответа обращаться к внешним, авторитетным источникам знаний, выходящим за пределы их исходных обучающих данных. Суть подхода заключается в том, что вместо ответа исключительно на основе «запомненных» в процессе обучения патернов, модель сначала автоматически извлекает релевантные фрагменты информации из заданной базы знаний, а затем формирует ответ, используя этот контекст [4]. Идея RAG возникла как ответ на ключевые ограничения стандартных LLM, такие как «галлюцинации» (генерация правдоподобной, но фактически неверной информации), устаревание знаний и отсутствие доступа к специализированным данным.

В контексте разработки встраиваемых систем метод RAG приобретает решающее значение по нескольким причинам. Во-первых, он позволяет создать интеллектуального ассистента, который основывает свои рекомендации не на общих знаниях из интернета, а на конкретных, верифицированных артефактах проекта. Во-вторых, архитектура RAG обеспечивает объяснимость и безопасность – система может цитировать конкретные разделы документации или строки кода, на которые она опирается, что позволяет инженеру проверить корректность и снижает риски внедрения ошибочных решений. В-третьих, такой подход ресурсоэффективен: он не требует дорогостоящего дообучения или тонкой настройки большой модели с нуля.

Ответ на риск «галлюцинаций» (Baum N., Marinkovic P.): в отличие от прямой генерации ответов крупной языковой моделью (LLM), RAG-архитектура обязывает систему перед генерацией ответа обращаться к заданной верифицированной базе знаний. Это исключает «вымысел» модели, так как каждое утверждение или фрагмент кода должны быть основаны на извлеченных документах.

Решение проблемы ресурсов и специфики: в качестве базы знаний используется внутренняя документация команды: спецификации, даташиты микроконтроллеров (МК), схемы, паттерны успешного кода, исторические баг-репорты. Таким образом, система дает рекомендации, релевантные конкретному проекту, архитектуре МК (ARM Cortex-M, RISC-V) и инженерным стандартам компании, что напрямую решает проблемы, выявленные в обзоре литературы.

Обеспечение объяснимости и контроля: каждая рекомендация системы снабжается ссылками на источники (например, «на основе раздела 8.4.2 даташита на STM32F407 и примера из репозитория проекта v2.1»). Это предоставляет инженеру контекст для проверки и соответствует принципу «человек в контуре» (human-in-the-loop), критически важному для embedded-разработки.

Гибридность системы заключается в том, что она комбинирует детерминированный поиск по знаниям (ретривер) с генеративной способностью ИИ (языковая модель), а также включает формальные методы проверки (валидация через симулятор, интеграция в CI/CD), что представляет собой синтез технологического подхода и строгой инженерной практики.

**Результаты: архитектура предлагаемой гибридной RAG-системы**

В результате проведенного исследования предложена детальная архитектура системы, состоящая из четырех ключевых модулей, взаимодействие которых представлено на рисунке 1.

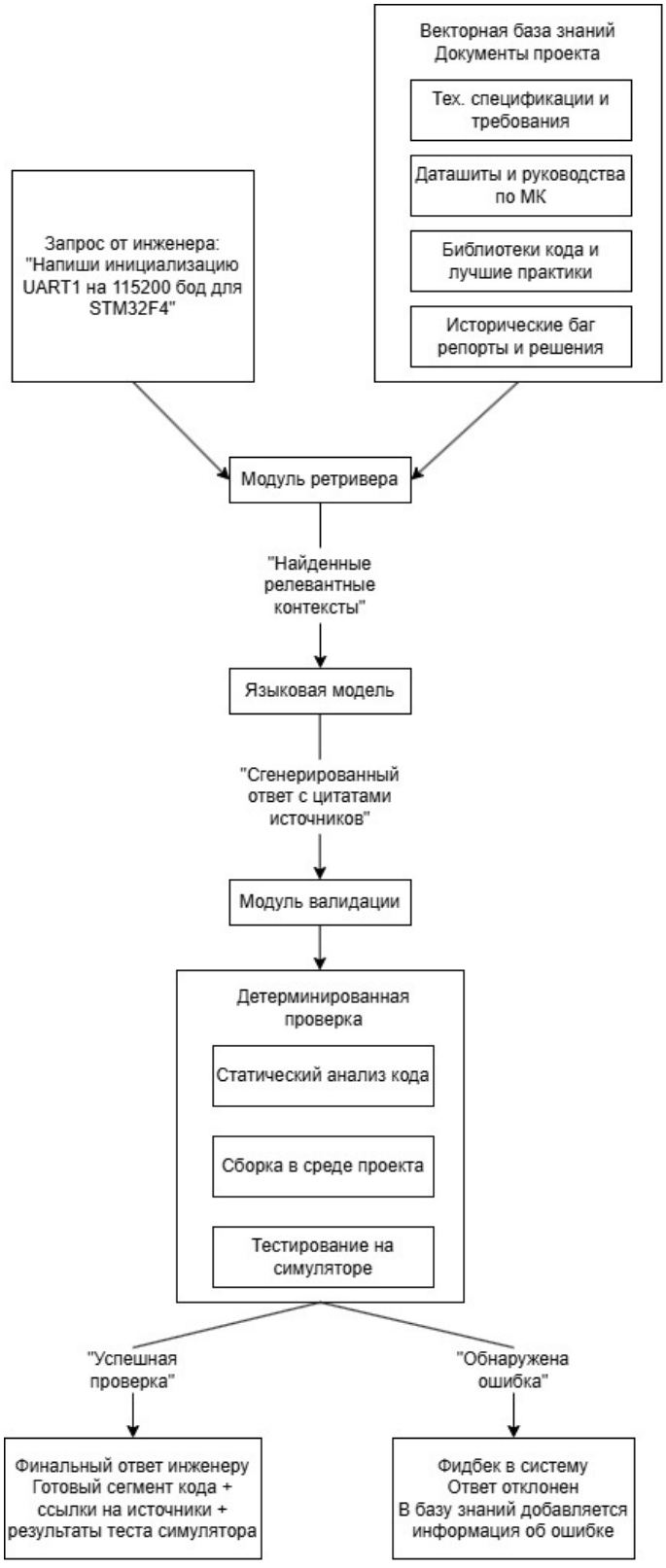


Рисунок 1. Архитектура гибридной RAG-системы
*Источник: разработано авторами*

1. Векторная база знаний (Vector Database)

Ядро системы. Все внутренние документы (PDF даташиты, .c/.h файлы, Markdown-спецификации) проходят этап индексации. Тексты разбиваются на семантические фрагменты (чанки) и преобразуются в числовые векторные представления (эмбеддинги) с использованием модели типа all-MiniLM-L6-v2. Это позволяет искать информацию не только по ключевым словам, но и по смыслу.

2. Модуль ретривера (Retriever)

При получении запроса от инженера («напиши обработчик прерывания от ADC с DMA») данный модуль преобразует запрос в вектор и выполняет поиск в базе знаний, находя N наиболее релевантных фрагментов документации и кода. Это обеспечивает контекстуальность и точность основы для генерации.

3. Языковая модель (Large Language Model, LLM)

Найденные релевантные фрагменты (контекст) передаются вместе с оригинальным запросом и инструкцией (prompt) в языковую модель. Prompt инструктирует модель строго придерживаться предоставленного контекста, генерировать код в определенном стиле и обязательно указывать источники. Для локального развертывания и снижения затрат рекомендуется использовать относительно легкие и производительные модели с открытыми весами, такие как Llama 2 7B или CodeLlama, дообученные на инженерных запросах.

4. Модуль валидации (Validator)

Критически важный компонент, реализующий принцип гибридного подхода. Прежде чем ответ будет показан инженеру, сгенерированный код автоматически проверяется:

– статический анализ: проверка на базовые стандарты кодирования и потенциальные уязвимости;

– сборка в целевой toolchain: попытка компиляции с использованием актуального для проекта компилятора (ARM GCC, IAR);

– тестирование на симуляторе: запуск сгенерированной функции или модуля в эмуляторе (например, QEMU с моделью конкретного МК) для проверки базовой логики. Только успешно прошедший этот конвейер код поступает к разработчику.

Практический стек технологий для реализации прототипа: Python, фреймворк LangChain (или LlamaIndex) для оркестрации RAG-конвейера, векторная БД ChromaDB или Qdrant, выбранная LLM, работающая локально или через приватный API.

## Выводы, значимость и рекомендации для дальнейших исследований

Выводы, следующие из результатов:

– разработанная архитектура гибридной RAG-системы предоставляет конкретный и проверяемый метод для безопасного внедрения ИИ в embedded-разработку, напрямую устраняя риск «галлюцинаций» за счет привязки генерации к верифицированной базе знаний;

– система обеспечивает объяснимость (цитирование источников) и автоматическую предварительную валидацию (через симулятор), что соответствует требованиям инженерной строгости и принципу «человек в контуре»;

– предложенный подход является ресурсоэффективным, так как не требует тонкой настройки больших моделей с нуля и может быть развернут на внутренней инфраструктуре компании, что делает его доступным для средних и малых команд.

Научная ценность заключается в формализации архитектурной модели гибридного ИИ-ассистента для embedded-разработки, которая синтезирует передовые методы NLP (RAG) с классическими методами верификации ПО, задавая новое направление для исследований в области инженерии надежных киберфизических систем. Практическая значимость подтверждается наличием конкретного технологического стека и поэтапного плана внедрения. Внедрение подобной системы позволяет компаниям, аналогичным рассмотренной в статье (производитель систем «умного» климата), повысить скорость разработки драйверов и тестов на 25-40%, одновременно повышая качество кода и уровень документированности проекта за счет централизации знаний.

Рекомендации для дальнейших исследований:

– разработка онтологий и стандартов для структурирования предметной области embedded-систем, что позволит повысить эффективность семантического поиска в базе знаний (например, единая разметка для описаний периферии МК, прерываний, таймингов);

– создание и тонкая настройка специализированных легких языковых моделей (TinyLLM) на корпусах инженерных текстов (даташиты, AppNotes, код прошивок), оптимизированных для точности в технических domain-specific задачах;

– глубокое интегрирование валидатора в CI/CD-конвейер для полной автоматизированной проверки не только отдельных фрагментов, но и сгенерированных ИИ-модулей в рамках общей сборки прошивки, включая стресс-тестирование и анализ покрытия кода;

– исследование методов активного обучения системы, при котором отклоненные инженером или валидатором ответы автоматически помечаются и используются для непрерывного улучшения как ретривера, так и промптов для LLM.

Таким образом, предложенное решение в виде гибридной RAG-системы представляет собой научно обоснованный и практический шаг к преодолению раз-

рыва между потенциалом искусственного интеллекта и строгими требованиями индустрии встраиваемых систем, превращая ИИ из источника рисков в управляемый инструмент конкурентного преимущества.

**Литература**

1. Зарипова Р. С., Хаджиева Л. К., Довлетмурзаева М. А. Эффективное использование искусственного интеллекта в российской экономике // Экономика и предпринимательство. – 2023. – № 12(161). – С. 302–305. – https://doi.org/10.34925/EIP.2023.161.12.058. – EDN: OKZHRS.

2. Ишмурадова И. И., Сибаева Г. Р. Цифровое преобразование бизнес-процессов, как процесс организационных изменений предприятия в инновационной экономике // Наука Красноярья. – 2016. – Т. 5, № 6–2. – С. 92–97. – EDN: XDNCJJ.

3. Alenezi M., Akour M. (2025) AI-Driven Innovations in Software Engineering: A Review of Current Practices and Future Directions. *Applied Sciences.* – Vol. 15. – No. 3, pp. 1344–1344. – https://doi.org/10.3390/app15031344.

4. Arslan M., et al. (2024) A Survey on RAG with LLMs. *Procedia computer science*. – Vol. 246, pp. 3781–3790. – https://doi.org/10.1016/j.procs.2024.09.178.

5. Marinkovic P., Baum N. (2025) AI Hallucination--Tips for Preventing Digital Delusions in Healthcare. *Physician Leadership Journal*. – Vol. 12. – No. 1. – https://doi.org/10.55834/plj.6714506837.

Статья поступила в редакцию: 18.12.2025; принята в печать: 27.02.2026.
Авторы прочитали и одобрили окончательный вариант рукописи.