УДК: 49.33.31
DOI: https://doi.org/10.66941/3106-9037/2026/iss3pp51-67

М. Кунелбаев, А. Рахыш*
Казахский национальный университет им. аль Фараби, Алматы, Казахстан
E-mail: rakhysh_aigerim3@live.kaznu.kz

РАСЧЁТНАЯ ОЦЕНКА КОММУНИКАЦИОННЫХ ОГРАНИЧЕНИЙ СОБЫТИЙНОЙ СИСТЕМЫ ВИДЕОМОНИТОРИНГА НА БАЗЕ ESP32-CAM И FPGA

Мурат Кунелбаев, профессор, старший научный сотрудник Института информационных и вычислительных технологий (ИИВТ) и преподаватель кафедры искусственного интеллекта и Big Data КазНУ им. аль-Фараби;

Айгерім Рахыш, PhD докторант по специальности «Системы информационной безопасности» кафедры «Кибербезопасность и Криптология».

Аннотация. В данном исследовании с помощью математического моделирования и анализа ранее полученных экспериментальных данных изучены ограничения канала связи в системе событийного видеоконтроля, построенной на элементах ESP32-CAM и FPGA. Предложена вычислительная модель передачи данных по шине UART, которая дает возможность определить степень загруженности линии, наименьшую длительность отправки сообщений, запас пропускной способности, а также оценить воздействие служебных байтов на процесс обмена. Проанализированы три варианта отправки данных: целого сжатого кадра, фрагмента (ROI) и итоговых данных детекции. На скорости 921 600 бод доля загрузки канала составила 32,55 %, 7,81 % и 2,78 % соответственно. Выявлено, что замена полного кадра на ROI приводит к падению полезной нагрузки на 76 %. Рассчитан порог выделения области интереса — 11,07 мс; превышение этого значения делает целесообразным переход на передачу ROI для экономии времени. Анализ опубликованных данных выявил уменьшение медианной задержки системы ESP32-CAM с FPGA на 22,58 % по сравнению с конфигурацией без FPGA. Предложены расчетные критерии для выбора формата сообщения и оценки последовательной передачи пакетов. Полученные результаты позволяют определить условия рационального использования пропускной способности UART.
Ключевые слова: событийный видеомониторинг; ESP32-CAM; FPGA; UART; задержка; нагрузка канала.

М. Кунелбаев, А. Рахыш*
әл-Фараби атындағы Қазақ ұлттық университеті,Алматы, Қазақстан
E-mail: rakhysh_aigerim3@live.kaznu.kz

ESP32-CAM ЖӘНЕ FPGA НЕГІЗІНДЕГІ ОҚИҒАЛЫҚ БЕЙНЕ МОНИТОРИНГ ЖҮЙЕСІНІҢ КОММУНИКАЦИЯЛЫҚ ШЕКТЕУЛЕРІН ЕСЕПТІК БАҒАЛАУ

Мұрат Кунелбаев, профессор, Ақпараттық және есептеуіш технологиялар институтының (АЕТИ) аға ғылыми қызметкері және әл-Фараби атындағы ҚазҰУ-дың Жасанды интеллект және Big Data кафедрасының оқытушысы;
Айгерім Рахыш, Киберқауіпсіздік және криптология кафедрасының «Ақпараттық қауіпсіздік жүйелері» мамандығы бойынша PhD докторант.

Аннотация. Бұл зерттеуде математикалық модельдеу және бұрын алынған эксперименттік деректерді талдау арқылы ESP32-CAM және FPGA негізінде құрылған оқиғаларға негізделген бейнемониторинг жүйесіндегі байланыс арнасының шектеулері зерттелді. UART шинасы арқылы деректерді берудің есептеу моделі ұсынылды. Модель байланыс желісінің жүктелу деңгейін, хабарламаларды берудің ең аз ұзақтығын және өткізу қабілетінің қорын анықтауға,

сондай-ақ қызметтік байттардың дерек алмасуға әсерін бағалауға мүмкіндік береді. Деректерді берудің үш нұсқасы талданды: толық сығылған кадрды, қызығушылық аймағын (ROI) және детекцияның қорытынды деректерін беру. 921 600 бод жылдамдықта арнаның жүктелу деңгейі тиісінше 32,55 %, 7,81 % және 2,78 % болды. Толық кадрдың орнына ROI беру пайдалы жүктеме көлемін 76 %-ға азайтатыны анықталды. Қызығушылық аймағын бөліп алудың шекті уақыты 11,07 мс деп есептелді; осы мәннен асқан жағдайда уақытты үнемдеу үшін ROI беруге көшу орынды болады. Жарияланған деректерді талдау FPGA қолданылатын ESP32-CAM жүйесінің медианалық кідірісі FPGA қолданылмайтын конфигурациямен салыстырғанда 22,58 %-ға төмендегенін көрсетті. Хабарлама пішімін таңдауға және пакеттерді тізбектей беруді бағалауға арналған есептік критерийлер ұсынылды. Алынған нәтижелер UART өткізу қабілетін ұтымды пайдалану шарттарын анықтауға мүмкіндік береді.
Түйін сөздер: оқиғалық бейнемониторинг; ESP32-CAM; FPGA; UART; кідіріс; арна жүктемесі.

Kunelbayev M., Rakhysh A.*
al-Farabi Kazakh National University, Almaty, Kazakhstan
E-mail: rakhysh_aigerim3@live.kaznu.kz

CALCULATION-BASED ASSESSMENT OF COMMUNICATION CONSTRAINTS IN AN EVENT-DRIVEN VIDEO MONITORING SYSTEM BASED ON ESP32-CAM AND FPGA

Murat Kunelbayev, Professor, Senior Researcher at the Institute of Information and Computational Technologies (IICT), and Lecturer at the Department of Artificial Intelligence and Big Data, Al-Farabi Kazakh National University;
Aigerim Rakhysh, PhD student in Information Security Systems at the Department of Cybersecurity and Cryptology.
Abstract. This study investigates communication channel constraints in an event-driven video monitoring system based on ESP32-CAM and FPGA through mathematical modeling and the analysis of previously obtained experimental data. A computational model of data transmission over the UART interface is proposed to determine channel utilization, the minimum message transmission time, and bandwidth headroom, as well as to assess the impact of overhead bytes on data exchange. Three data transmission options are analyzed: a complete compressed frame, a region of interest (ROI), and detection results. At a baud rate of 921,600, channel utilization was 32.55%, 7.81%, and 2.78%, respectively. Replacing a complete frame with an ROI was found to reduce the payload by 76%. A threshold of 11.07 ms was calculated for region-of-interest extraction; exceeding this value makes switching to ROI transmission advantageous for saving time. An analysis of published data revealed a 22.58% reduction in the median latency of the ESP32-CAM system incorporating an FPGA compared with the configuration without an FPGA. Calculation-based criteria are proposed for selecting the message format and evaluating sequential packet transmission. The results identify the conditions for efficient utilization of UART bandwidth.
Keywords: event-driven video monitoring; ESP32-CAM; FPGA; UART; latency; channel utilization.

Введение

Передача каждого видеопотока на аппаратный ускоритель в системах распределенного наблюдения приводит к повышенной нагрузке на канал связи между камерой и вычислительным блоком. Хотя запуск анализа лишь при фиксации движения снижает объем обрабатываемых кадров, продолжительность отдельного цикла остается ограниченной временем, необходимым для передачи данных, их обработки и публикации итогов. Следовательно, решение о том, что именно отправлять — сжатое изображение, выделенную область или краткий результат, — должно базироваться на объективных характеристиках пропускной способности канала. В источнике [1] описывается система видеонаблюдения, реагирующая на события, которая включает в себя ПИР-сенсор, микроконтроллер ESP32-CAM и программируемую логическую интегральную схему (ПЛИС). Данная конструкция выступает базовой моделью для текущего исследования. Чтобы проанализировать функциональные

возможности камеры и контроллера, целесообразно опираться на данные тестирования модуля ESP32-CAM, опубликованные в работе [2]. Авторы работ [3] и [4] проанализировали системы мониторинга и детекции перемещений, построенные на микроконтроллере ESP32, тогда как в источнике [5] исследовался вопрос передачи видеопотока через сеть Wi-Fi. Стратегия обработки: локальная или отправка кадров на сервер — напрямую определяет трафик сети и задержку при получении итогового результата. В исследовании [6] проанализированы подходы к распределению задач при периферийном выявлении объектов на платформе ESP32. Характеристики ESP32-CAM как программируемого модуля для исследовательских задач изложены в источнике [7]. Использование энергоэффективных камерных систем в дистанционных сетях видеонаблюдения обсуждается в работе [8], тогда как аспекты выбора протоколов беспроводной передачи изображений освещены в публикации [9]. Проблематика адаптивного мониторинга в условиях нехватки вычислительных мощностей рассматривалась в работе [10]. Для повышения скорости анализа видеопотока активно используются программируемые логические матрицы (FPGA). В частности, Саркар и соавторы [11] разработали систему интерфейса между камерой и FPGA, тогда как исследование [12] описывает FPGA-ускоритель для задач обнаружения объектов в промышленной сфере. Тематике аппаратных ускорителей детектирования объектов посвящен обзор [13]. В работе [14] анализируются специфические требования, предъявляемые к функционированию систем в режиме реального времени. Обобщающий обзор методов оптимизации FPGA-интерпретаций семейства алгоритмов YOLO представлен в источнике [15]. Что касается практической реализации, то примеры использования FPGA-технологий в задачах многокамерного видеонаблюдения и в составе автономных комплексов детально описаны в публикациях [16]–[18]. Особое внимание уделяется анализу информации, поступающей от событийных камер. В работе [19] рассматривается задача выявления перемещающихся объектов с применением таких сенсоров и программируемых логических интегральных схем (ПЛИС). Обзор существующих решений для событийного зрения на базе ПЛИС содержится в источнике [20], а фундаментальные основы данной технологии систематизированы в статье [21]. В источнике [22] изложена аппаратно-программная структура для мониторинга множества объектов. Работа [23] посвящена созданию камеры, измеряющей оптический поток на основе событий, тогда как в [24] предложена система выявления и сопровождения объектов, ориентированная на периферийные устройства.

Задачей работы выступает повторный анализ материалов [1]: производится вычисление утилизации UART-интерфейса при передаче сообщений различных категорий, определяется воздействие служебных байтов на эффективность канала, а также сопоставляются медианные значения задержек для конфигураций ESP32-CAM и комбинированной системы ESP32-CAM с FPGA. Детальное исследование временных затрат на каждом этапе обработки исключено из-за отсутствия опубликованных временных меток для этих операций.

Методология. Когда срабатывает инфракрасный датчик движения, модуль ESP32-CAM фиксирует снимок с помощью камеры OV2640. Полученные данные передаются на FPGA через интерфейс UART, после чего сигнал о фиксации объекта отправляется обратно на ESP32 и транслируется в сеть по протоколу Wi-Fi. Структура системы и внешний вид прототипа, описанные в источнике [1], представлены на рисунках 1 и 2. Функции обеспечения безопасности данных, выделенные в исходной схеме, трактуются здесь как часть архитектурного решения; при этом в работе [1] отсутствуют сведения об оценке их стоимости и проверке корректности реализации.

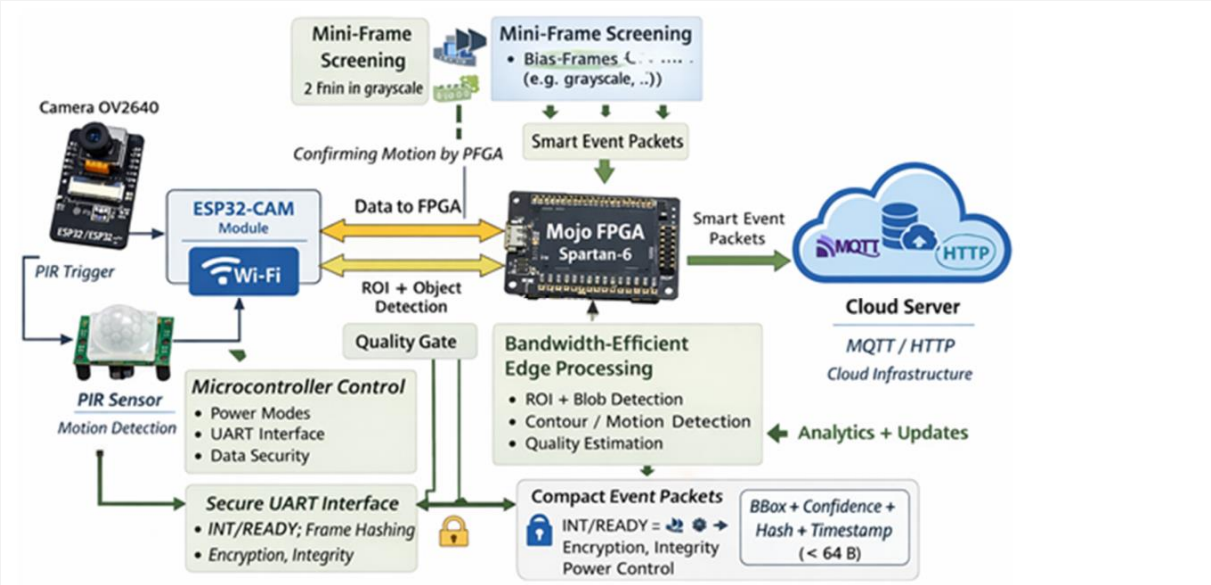


Рисунок 1 – Архитектура событийного видеомониторинга [1]

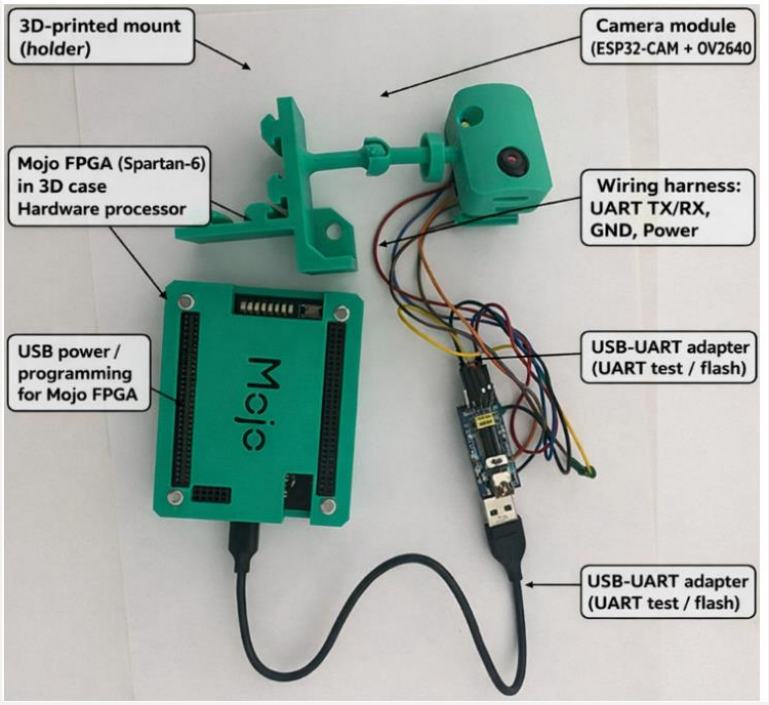


Рисунок 2 – Экспериментальный образец [1].

В таблице 1 агрегированы первоначальные параметры [1]. Частоты 25 и 40 пакетов в секунду отражают различные режимы работы и не предполагают одновременную передачу всех трех потоков в рамках одного теста. Показатели времени реакции фиксируют интервал между активацией PIR-датчика и подтверждением публикации, при этом они не изолируют время, затраченное исключительно на вычисления в FPGA.

Таблица 1. Опубликованные данные, использованные в расчётах [1]

Поток	Средний размер, Б	P95 размера, Б	Частота, пак./с
Сжатый кадр	1200	1600	25
Область интереса / признаки	180	240	40

Результат обнаружения	64	96	40

Математическая модель обмена ESP32 CAM и FPGA по UART. Предложенная модель применяется для вторичной обработки сводных данных, размещенных в источнике [1]. Ее задача –определить минимальную продолжительность сеанса связи, оценить занятость последовательного порта и обосновать выбор структуры пакета. Важно отметить, что рассчитанные показатели не отражают фактическое время выполнения этапов работы прототипа.

В рамках каждого конкретного сценария i берутся следующие параметры: средняя длина сообщения S_i (в байтах), 95-й перцентиль длины $S_{95,i}$ (в байтах), частота отправки f_i (сообщений в секунду) и пропускная способность канала $B = 921,600$ бод. При использовании протокола UART с конфигурацией 8N1 на передачу одного байта уходит 10 бит. Анализ каналов связи ESP32→FPGA и FPGA→ESP32 проводится независимо друг от друга, поэтому частоты передачи для различных сценариев не суммируются.

$$R_i + S_i, C = \frac{B}{10} = 92160\, {}^{B}/_{C}\,, U_i = {}^{R_i}/_{C} = 10\, S_i f_i / B \qquad (1)$$

R_i – полезный поток (Б/с); C – идеальная верхняя граница пропускной способности для 8N1; U_i – безразмерная расчётная загрузка. В процентах U_i умножается на 100. Уравнение (1) не учитывает паузы, служебные байты и повторные передачи.

$$\Delta R_i = 1 - {}^{R_i}\!/_{R_{\text{кадр}}} \qquad (2)$$

В уравнении (2) снижение потока оценивается относительно сценария передачи сжатого кадра при указанных для каждого сценария частотах; оно не доказывает одинаковую полезность передаваемых данных.

$$t_{i,min} = 10\, S_i f_i / B,\ t_{95,\text{разм},i} = 10\, S_{95,i} / B \qquad (3)$$

Первая величина рассчитана для среднего размера, вторая – для P_{95} размера. t_{95}, размер не является квантилем P_{95} фактически измеренной задержки: ожидание интерфейса, очереди и обработка FPGA в уравнение (3) не входят.

$$R_i(h_i) = (S_i + h_i) f_i, U_i(h_i) = \frac{10(S_i + h_i) f_i}{B} \qquad (4)$$

$$t_{i,min}(h_i) = \frac{10(S_i + h_i) f_i}{B}, \frac{\partial U_i}{\partial h_i} = 10 f_i / B \qquad (5)$$

h_i – добавленные байты на одно сообщение. Значения h_i = 0, 32 и 64 Б используются только как сценарии чувствительности; размер служебных полей прототипа в [1] не опубликован. Уравнения (4)–(5) не моделируют повторы и интервалы простоя.

$$f_{i,max}(\alpha, h_i) = \frac{\alpha B}{[10(S_i + h_i) f_i]}, 0 < \alpha < 1 \qquad (6)$$

α – заданная проектная доля допустимой загрузки. Это свободный параметр сценария, а не измеренная характеристика установки. Если f_i превышает $f_{i,max}$, выбранный резерв интерфейса не обеспечивается даже в идеализированной модели.

Кадр и область интереса передаются в направлении ESP32→FPGA. Пусть t_{ROI} – дополнительное время выделения области интереса на ESP32. Для одного и того же события переход на ROI сокращает сумму времени выделения и минимального времени передачи только при следующем условии:

$$t_{ROI} + 10(S_{ROI} + h_{ROI})/10(S_{cadr} + h_{cadr})/B \qquad (7)$$

$$t_{ROI} < t_{ROI,max} = 10[S_{cadr} + h_{cadr}) - (S_{ROI} + h_{ROI})/B \qquad (8)$$

При S_{cadr} = 1200 Б, S_{ROI} = 180 Б и $h_{cadr} = h_{ROI} = 0$ расчётный порог $t_{ROI,max}$ = 11,07 мс. Время фактического выделения ROI не опубликовано, поэтому уравнение (8) задаёт проверяемое условие, а не установленное преимущество. Оно сопоставляет только указанные стадии и не охватывает полную задержку события.

$$\Delta T_p = T_{p,}ESP32 - T_{p,}ESP32 + FPGA$$

$$\delta_p = \frac{\Delta T_p}{T_p}\text{v} \qquad (9)$$

$$p = 50{,}95{,}99$$

T_p обозначает опубликованный квантиль времени от срабатывания PIR-датчика до подтверждения публикации. Вычитание квантилей даёт разницу опубликованных характеристик режимов, а не квантиль попарных разностей для одних и тех же событий.

Таблица 2. Расчёт по опубликованным данным

Сценарий	S, Б	f, 1/с	U, %	t_min, мс
Сжатый кадр	1200	25	32,55	13,02
Область интереса	180	40	7,81	1,95
Результат обнаружения	64	40	2,78	0,69

Согласно данным таблицы 2, полные задержки для конфигураций ESP32 и ESP32 с FPGA составляют: P50 – 620 и 480 мс, P95 – 980 и 780 мс, P99 – 1300 и 1050 мс [1]. Из формулы (9) следует, что абсолютная разница равна 140, 200 и 250 мс, а относительное снижение – 22,58 %, 20,41 % и 19,23 %. Исследование базировалось на 80 событиях для каждого режима, однако детали исходных выборок и временные метки отдельных этапов не публиковались. В связи с этим невозможно рассчитать доверительные интервалы или установить точный вклад UART и FPGA в формирование указанных показателей.

Методика вторичного анализа. Полезная нагрузка потока, характеризующегося средней длиной сообщения S и частотой f, вычисляется по формуле R = S·f (в байтах в секунду). При использовании асинхронного формата передачи данных 8N1 один байт кодируется десяти битами (включая стартовый, восемь информационных и стоповый биты). Следовательно, теоретический предел пропускной способности для скорости 921 600 бод равен C = B/10 = 92 160 Б/с. Коэффициент загрузки канала определяется как U = R/C. Данный расчет базируется на предположении о непрерывном потоке без перерывов и игнорирует служебные данные, такие как заголовки, подтверждения, контрольные суммы и механизмы повторной передачи.

Минимальное время отправки пакета выражается формулой $t_s = 10S/B$. Для 95-го процентиля размера пакета аналогичным образом выводится значение $t_{95,разм} = 10S_{95}/B$. Важно отметить, что эта величина не отражает реальный 95-й процентиль задержки, так как квантиль размера пакета не учитывает время в очереди, задержки на инициализацию FPGA и задержки, вызванные повторными отправлениями. Относительное снижение полезной нагрузки по сравнению со сжатым кадром рассчитывается через $\Delta R = 1 - R_i/R_{кадр}$.

Каждому типу сообщения в расчетах отводится собственный сценарий. Направление ESP32→FPGA охватывает сжатый кадр и область интереса, тогда как короткое сообщение с результатом направлено от FPGA к ESP32. Следовательно, полученные показатели нагрузки сопоставляются по сценариям, а не суммируются как три параллельно измеряемых потока. Для UART, использующего общий физический канал, требуется отдельно выяснить, позволяет ли конкретная реализация одновременный обмен; в доступных источниках режим полнодуплексной работы не детализирован.

Чтобы оценить влияние служебных данных, вводится переменная h, обозначающая количество дополнительных байтов на сообщение. Расчетная нагрузка определяется как R(h) = (S + h)·f, загрузка канала – U(h) = 10(S + h)f/B, а минимальное время передачи – t(h) = 10(S + h)/B. При нулевом значении h получаются показатели, представленные в таблице 2. Поскольку в

источнике [1] не указаны размеры заголовков, подтверждений и повторных передач, численные значения для h > 0 в данной работе не рассчитываются.

Алгоритм обработки исходных данных следующий: для каждого типа сообщения фиксируются опубликованные значения среднего размера (S), 95-го процентиля ((P95)) и частоты (f). На основе общей скорости (B) вычисляются (R), (C), (U), (t_s) и (t_{95}); затем при единой базе сравнения определяется относительное снижение потока. Округление производится после всех вычислений: загрузка и относительные изменения приводятся с точностью до 0,01 процентного пункта, а время — до 0,01 мс. Исходные данные в таблице 1 остаются без изменений.

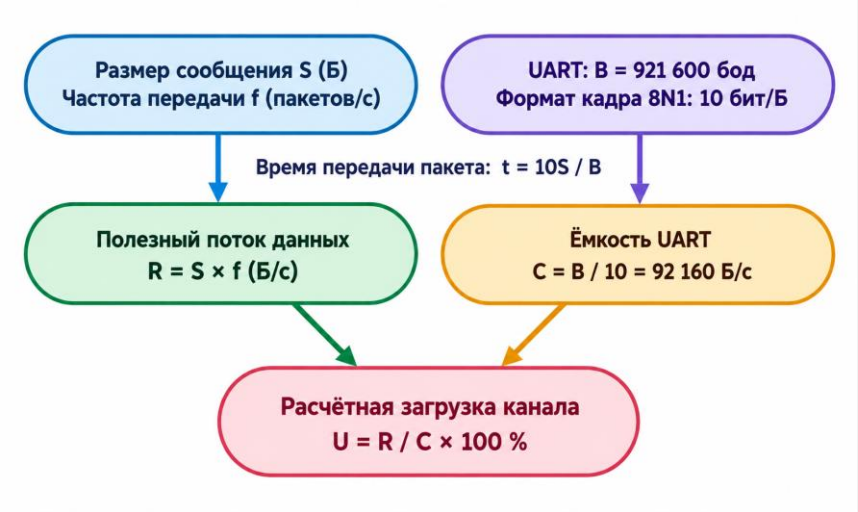


Рисунок 3. Методика расчёта загрузки UART

На рисунке 3 представлена методика вычисления загрузки линии UART. Поток данных RR вычисляется на основе среднего размера сообщения SS и частоты передачи ff, тогда как пропускная способность канала CC определяется скоростью UART и форматом 8N1. Оценка загрузки осуществляется через отношение R/C, а минимальное время отправки одного пакета рассчитывается по формуле 10S/B. Полученные показатели являются теоретическими и не принимают во внимание служебную информацию, интервалы пауз и случаи повторной отправки.

Для сопоставления опубликованных задержек применяется разность квантилей ΔT_p = T, ESP32 – T, FPGA и относительное изменение $\delta_p = \Delta T_p/T_p$,ESP32. В источнике [1] приведено по 80 событий для каждого режима, однако отсутствуют парные временные метки этапов или исходные данные. В связи с этим оценка доверительных интервалов и статистической значимости различий в данном случае не проводится.



Рисунок 4. Этапы обработки события и предлагаемые точки регистрации времени

Представленная на рисунке 4 блок-схема определяет последовательность фиксации временных меток, необходимую для пошаговой оценки задержек. Показанные интервалы t_0–t_6 служат ориентиром для измерений; при этом в исходной публикации [1] доступны лишь квантили общей задержки, а детальные журналы отсутствуют. Вычисление разностей между смежными отметками позволит проанализировать вклад каждого этапа: захвата, передачи данных на FPGA, их обработки, возврата результата и сетевой публикации.



Рисунок 5. Блок-схема методики вторичного анализа опубликованных данных

На рисунке 5 представлена блок-схема метода вторичного анализа опубликованных данных. Схема разделяет расчетные показатели передачи по UART, сценарный анализ служебных байтов и сравнение опубликованных квантилей задержки. Времена отдельных этапов обработки в источнике [1] не измерялись.

Резерв пропускной способности. При установлении проектного значения α на уровне 0,70 пропускная способность для передачи полезной информации в конфигурации 8N1 достигает 64 512 бит/с. В таблице 4 приведены данные о допустимой частоте, обеспечивающей

указанный резерв. Полученные расчетные границы характеризуют каждый отдельный сценарий и не должны восприниматься как фактические измеренные показатели скорости передачи данных устройства.

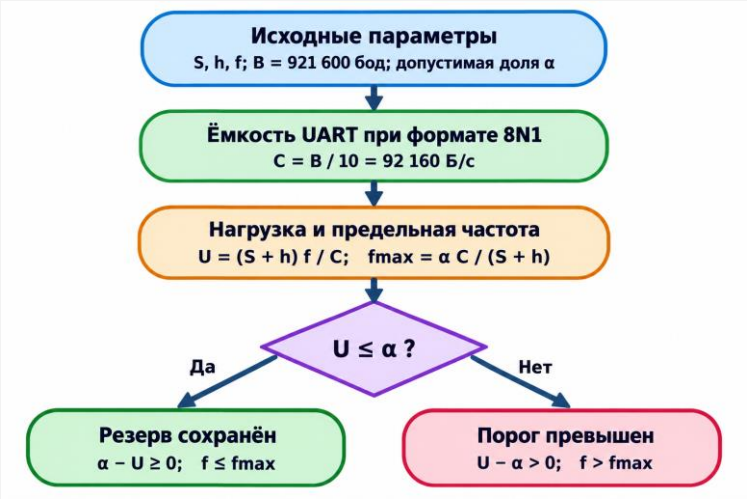


Рисунок 6. Блок-схема оценки резерва пропускной способности UART

На рисунке 6 изображена блок-схема, предназначенная для оценки запаса пропускной способности интерфейса UART. Расчет для каждого вида сообщения производится независимо. Параметр α определяет допустимую долю от теоретической емкости канала; так, значение α = 0,70 отражает проектную нагрузку на уровне 70 %. Символ h указывает на объем дополнительных байт, добавляемых к сообщению, однако данное значение не было измерено в первоисточнике.

Таблица 3. Расчётный предел частоты сообщений при α = 0,70 и B = 921 600 бод

Сценарий	S, Б	fmax при h = 0, 1/с	fmax при h = 64 Б, 1/с
Сжатый кадр	1200	53,76	51,04
Область интереса	180	358,40	264,39
Результат обнаружения	64	1008,00	504,00

Таблица 3 демонстрирует вычисленный порог частоты обмена данными при параметрах α = 0,70 и B = 921 600 бод. В базовых условиях отправка сжатого кадра с частотой 25 сообщений в секунду использует 32,55 % пропускной способности UART, тогда как при 40 сообщениях/с показатель ROI составляет 7,81 %. Если добавить к каждому сообщению 64 байта служебной информации, доля занятой ёмкости возрастет до 34,29 % и 10,59 % соответственно. Поскольку размер заголовка h, использованный в работе [1], не был верифицирован, полученные данные отражают чувствительность модели к выбранному значению этого параметра.

Пороговое время выделения ROI. При опубликованных размерах кадра и области интереса (ROI), равных 1200 и 180 байт соответственно, а также нулевых значениях hкадр и hROI, пороговое значение tROI,max составляет 11,07 мс (расчет: 10(1200 – 180)/921600). В случае, если процесс выделения ROI на микроконтроллере ESP32 занимает времени меньше указанного порога, происходит сокращение совокупного времени, включающего время передачи и минимальную задержку. Когда же hROI увеличивается до 64 байт при hкадр = 0, порог снижается до 10,37 мс. Следует отметить, что полученные числовые значения не отражают реальное

ускорение системы, так как в источнике [1] отсутствуют данные о времени выделения ROI и других этапах формирования общей задержки.

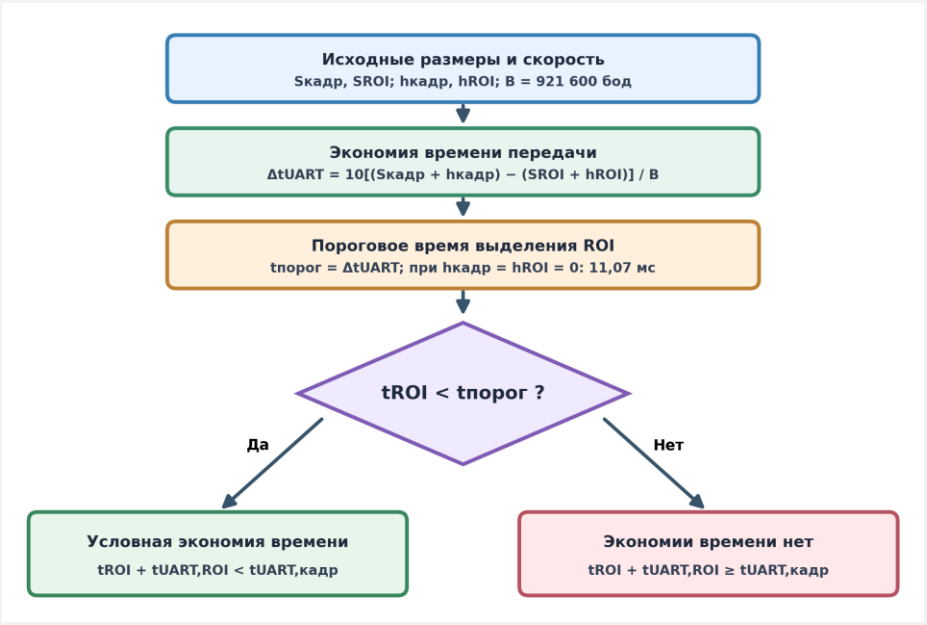


Рисунок 7. Блок-схема оценки порогового времени выделения области интереса

Рисунок 7 демонстрирует алгоритм вычисления порогового значения времени для выделения целевой области. Величина порога 11,07 мс определена для отдельного пакета данных при размере кадра 1200 байт и области интереса (ROI) 180 байт, без учета служебных заголовков. Для подтверждения практической целесообразности требуется экспериментальное измерение времени сегментации ROI на микроконтроллере ESP32-CAM, поскольку в источнике [1] такие данные не приводятся. Предложенная модель учитывает исключительно длительность процесса выделения и теоретическое время передачи, не охватывая общую задержку обработки события.

Последовательная отправка серии сообщений. На рисунке 8 представлена минимальная продолжительность последовательной передачи пакета из N сообщений равного объема. Для четырех сообщений (N=4) время составляет 52,08 мс для сжатых кадров и 7,81 мс для области интереса (ROI) без служебных байтов; в обоих случаях последний пакет ожидает завершения передачи трех предыдущих. При увеличении количества до десяти (N=10) эти значения возрастают до 130,21 и 19,53 мс соответственно. Данный сценарий применяется для оценки

воздействия одновременного поступления сообщений и не отражает фактическую временную последовательность событий, описанную в опубликованных материалах.

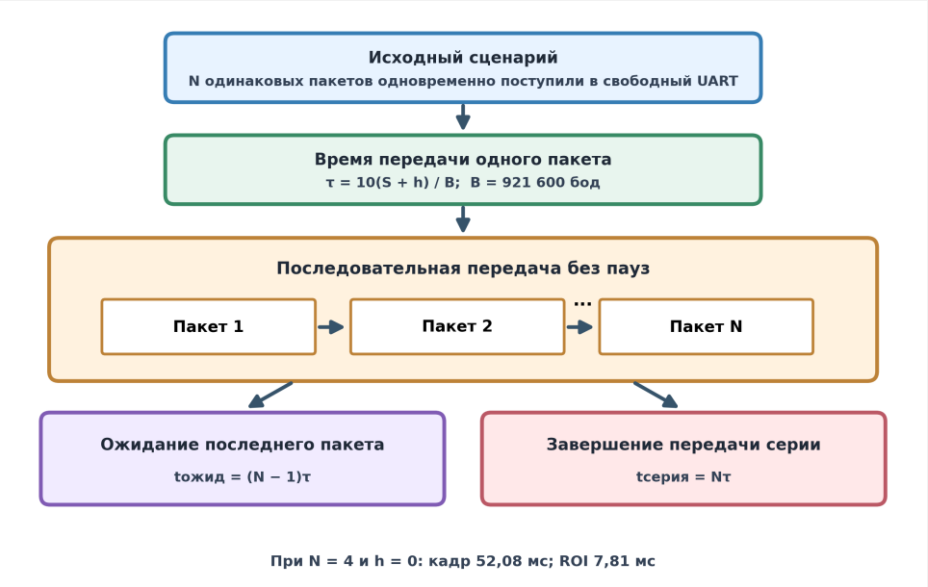


Рисунок 8. Блок-схема расчёта времени последовательной передачи серии сообщений

Проект предусматривает параллельный прием N идентичных пакетов через свободный UART-интерфейс, осуществляемый непрерывно, без временных задержек, при использовании стандарта 8N1. В качестве примера взяты размеры кадра: S = 1200 байт для сжатого формата и S = 180 байт для ROI. Данные о фактических интервалах поступления сообщений, очередях и повторениях, содержащихся в опубликованных источниках [1], отсутствуют.

Результаты. В представленном материале изложены итоги вычислительного исследования параметров связи в системе видеонаблюдения событийного типа, реализованной на базе микросхем ESP32-CAM и FPGA. Оперируя данными из источника [1], авторы оценили степень задействованности порта UART, длительность минимальной отправки сообщений и запас емкости канала при использовании разных форматов данных. Проанализированы три независимые ситуации: отправка сжатого изображения, выделенной зоны интереса (ROI) и итогов обнаружения. Кроме того, изучено воздействие служебных байтов и последовательной передачи пакетов, а также сопоставлены опубликованные значения задержек для систем с ESP32-CAM и с добавлением FPGA. Расчетные данные помогают выявить ограничения коммуникации и обосновать выбор формата сообщения. Важно отметить, что полученные оценки времени передачи не включают задержки, связанные с обработкой, очередями и публикацией в сети.

Таблица 4. Расчётные показатели UART при 921 600 бод, 8N1

Передаваемые данные	Нагрузка, Б/с	Загрузка, %	t_s, мс	t_{95},разм, мс
Сжатый кадр	30 000	32,55	13,02	17,36
Область интереса / признаки	7 200	7,81	1,95	2,60
Результат обнаружения	2 560	2,78	0,69	1,04

Как показывает анализ данных таблицы 4, при использовании потока сжатых видеоданных доля полезной информации, передаваемой в зонах повышенного интереса,

снижается на 76,0 %, тогда как для сообщений с итоговыми результатами этот показатель достигает 91,5 %. В результате вычислительная задержка при отправке одного усредненного сообщения сокращается на 11,07 и 12,33 миллисекунды соответственно. Полученные цифры отражают характеристики транспортировки данных в теоретически идеальном интерфейсе и не учитывают фактическое местоположение процесса извлечения признаков; размещение данного алгоритма на микроконтроллере ESP32 неизбежно приведет к увеличению локальных вычислительных затрат.

Таблица 5. Повторное представление опубликованных квантилей задержки [1]

Квантиль	ESP32, мс	ESP32 + FPGA, мс	Разница, мс	Снижение, %
P50	620	480	140	22,58
P95	980	780	200	20,41
P99	1300	1050	250	19,23

В таблице 5 разница медианных значений составляет 140 мс, тогда как выигрыш при передаче среднего кадра и сообщения с результатом оценивается всего в 12,33 мс. Нельзя однозначно связывать всю эту разницу исключительно с обменом по интерфейсу UART, поскольку итоговая задержка формируется под воздействием множества факторов: получения кадра, состояния очереди интерфейса, процессов обработки, работы Wi-Fi-модуля и времени ответа сервера. Подобным образом, сокращение показателя P99 на 250 мс свидетельствует об изменении хвостовых задержек в масштабах всей системы, однако без детальной временной разметки каждого этапа невозможно точно определить вклад FPGA.

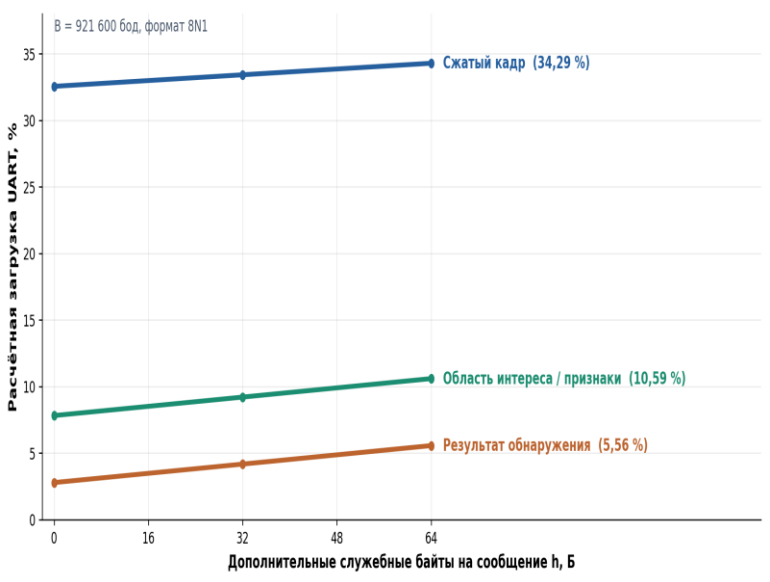


Рисунок 9. Чувствительность расчётной загрузки UART к дополнительным байтам на сообщение (сценарный расчёт по данным [1])

На рисунке 9 продемонстрировано, что при неизменной частоте отправки пакетов увеличение размера служебной заголовочной части приводит к линейному росту нагрузки. Крутизна этой зависимости варьируется в зависимости от скорости передачи: для режима 40 пакетов в секунду она равна примерно 0,043 процентного пункта на каждый добавленный байт, тогда как для 25 пакетов/с – около 0,027. Важно отметить, что представленный график отражает лишь теоретические данные, не учитывая фактический объем переданного трафика, а также

игнорирует такие факторы, как наличие очередей и необходимость повторной передачи данных.

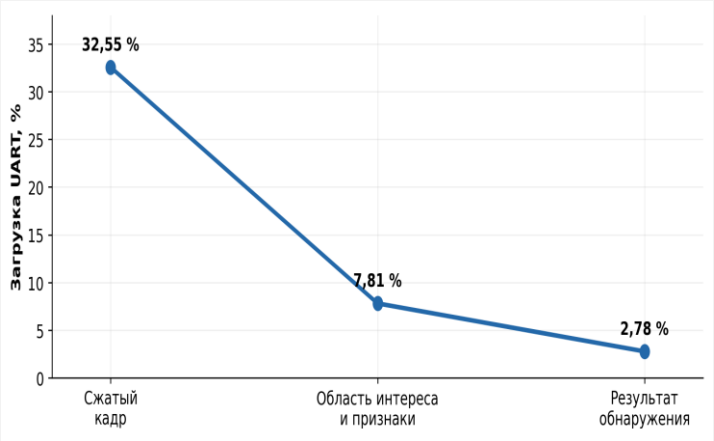


Рисунок 10. Расчётная загрузка UART для трёх типов сообщений

На рисунке 10 представлена расчетная загрузка интерфейса UART: 32,55 % для сжатых кадров, 7,81 % для областей интереса и 2,78 % для данных обнаружения. На рисунке 6 показано минимальное время передачи, рассчитанное при среднем размере пакета, а также значения P95. При этом следует отметить, что показатели P95 по размеру не отражают квантили фактической задержки. Оба графика сформированы на основе исходных данных, приведенных в источнике [1].

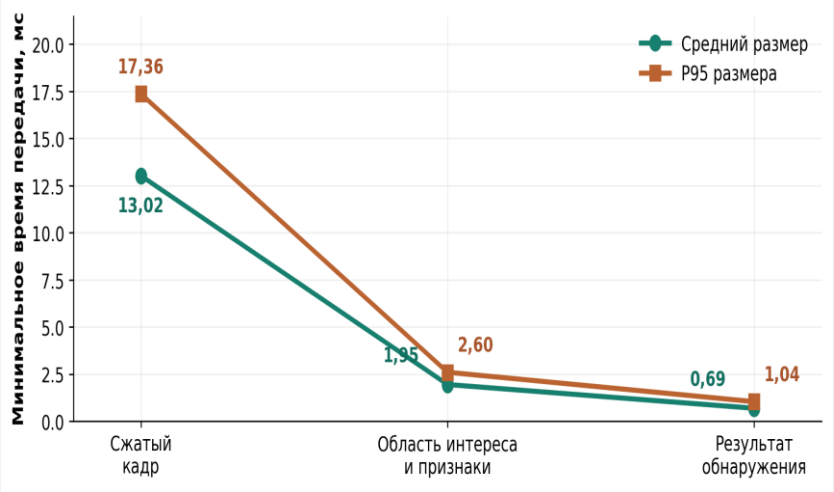


Рисунок 11. Минимальное время передачи сообщений среднего размера и размера P95

На рисунке 11 представлена расчетная минимальная продолжительность отправки сообщений по интерфейсу UART, рассчитанная для среднего размера пакета и значения P95. Для сжатого кадра эти показатели равны 13,02 и 17,36 миллисекунды соответственно; для области интереса –1,95 и 2,60 мс, а для результата обнаружения – 0,69 и 1,04 мс. Важно отметить, что в данном контексте параметр P95 характеризует объем передаваемых данных (размер пакета), а не фактическое время задержки при передаче.

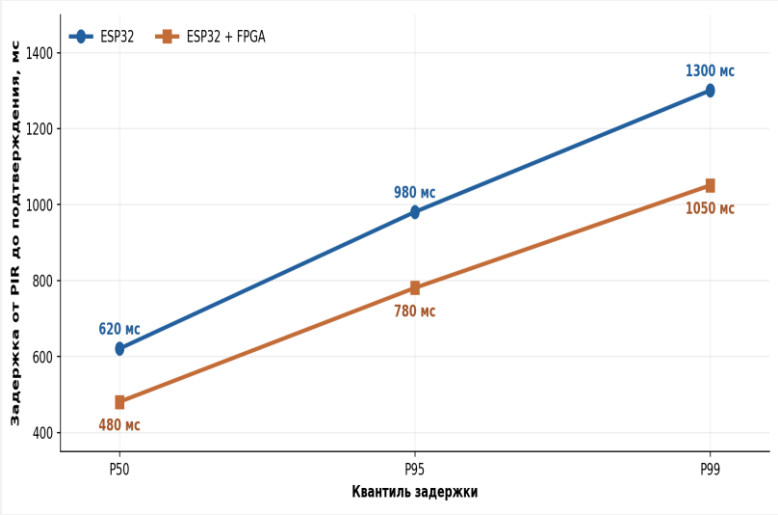


Рисунок 12. Опубликованные квантили полной задержки для режимов ESP32 и ESP32 + FPGA [1]

Как показано на рисунке 12, конфигурация ESP32 совместно с FPGA демонстрирует более низкие значения квантилей P50, P95 и P99, уменьшение которых составляет 140, 200 и 250 мс соответственно. Поскольку эти метрики отражают общую длительность цикла от срабатывания датчика PIR до успешной публикации данных, они не позволяют изолированно оценить время вычислений на стороне FPGA.

Разница медианных задержек достигает 140 мс, в то время как экономия времени исключительно на передаче среднего кадра и сообщения с результатом оценивается всего в 12,33 мс. Приписывать всю выявленную разницу исключительно обмену данными по интерфейсу UART некорректно, так как итоговая задержка формируется под влиянием множества факторов: захвата видеопотока, состояния очереди интерфейса, процессов обработки, работы модуля Wi-Fi и времени ответа сервера. Подобным образом, снижение показателя P99 на 250 мс свидетельствует об улучшении показателей «хвостовых» задержек всей системы в целом, однако без детальной маркировки каждого этапа процесса невозможно точно определить долю вклада FPGA в это улучшение.

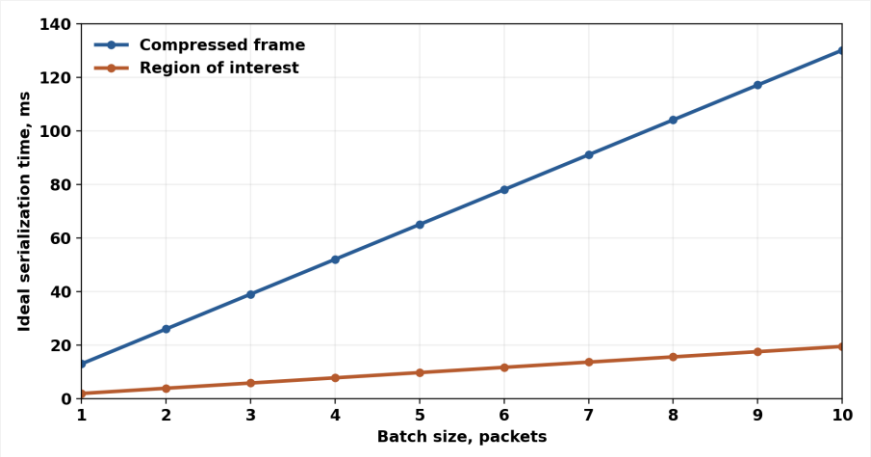


Рисунок 13. Минимальное время передачи серии сообщений через UART при h = 0

На рисунке 13 показано наименьшее время, необходимое для отправки пакета данных по интерфейсу UART при условии h = 0. Заданные значения резерва частоты, порога определения области интереса (ROI) и параметров серии сообщений формируют набор проверяемых ограничений для предстоящих тестов. Для проведения проверки необходимо наличие временных меток момента создания ROI, а также временных показателей начала и завершения передачи каждого отдельного пакета. Кроме того, требуются данные о служебных полях и состоянии очереди UART. Отсутствие этой информации не позволяет достоверно определить реальное снижение задержки или корректно соотнести наблюдаемую разницу квантилей с конкретной стадией процесса.

Заключение. Исследования показали, что передача только области интереса (ROI) вместо целого кадра уменьшает вычислительную нагрузку на интерфейс UART на 76%. Установлено пороговое значение времени формирования ROI, составляющее 11,07 мс, что позволяет определить целесообразность применения предварительной обработки изображения. Анализ существующих данных свидетельствует о сокращении медианной задержки системы ESP32-CAM с FPGA на 22,58 %, однако степень влияния отдельных этапов обработки остается неясной. В перспективе требуется экспериментальная верификация предложенных расчетных зависимостей, измерение задержек на каждом этапе и оценка стабильности работы системы при высокой частоте событий.

Литература

1. Adilzhanova S. et al. A Hardware Platform for Smart Video Monitoring Based on ESP32-CAM and Mojo FPGA (Spartan-6) with Event Activation Triggered by a PIR Sensor**.** *ETASR*, 2026. DOI: 10.48084/etasr.18359.
2. Nowroz S. T. et al. A Benchmark Reference for ESP32-CAM Module**.** 2025. DOI: 10.48550/arXiv.2505.24081.
3. Abordo P. R. C. et al. **S**mart surveillance system using ESP32 and camera-based motion detection with IM technolog**y.** *International Journal of Research Studies in Educational Technology*, 2024. DOI: 10.5861/ijrset.2024.8012.
4. Okokpujie K. et al. Development of an Affordable Real-Time IoT-Based Surveillance System Using ESP32 and TWILIO API**.** *Journal of Safety and Security Engineering*, 2023. DOI: 10.18280/ijsse.130609.
5. Zhaxalikov A. et al. Surveillance Camera Using Wi-Fi Connection**.** *Procedia Computer Science*, 2024. DOI: 10.1016/j.procs.2023.12.147.
6. Design and Implementation of ESP32-Based Edge Computing for Object Detection**.** *Sensors*, 2025. DOI: 10.3390/s25061656..
7. ESP32-CAM as a Programmable Camera Research Platform**.** *Electronic Imaging*, 2022. Lech P. et al. A Low-Cost Energy-Efficient IoT Camera Trap Network for Remote Forest Surveillance**.** *Electronics*, 2025. DOI: 10.3390/electronics14214266.
8. Hahn F. et al. Mango Fruit Fly Trap Detection Using Different Wireless Communications**.** *Agronomy*, 2023. DOI: 10.3390/agronomy13071736.
9. Kok C. L. et al. Energy-, Cost-, and Resource-Efficient IoT Hazard Detection System with Adaptive Monitoring**.** *Sensors*, 2025. DOI: 10.3390/s25061761.
10. Sarkar et al. FPGACam: A FPGA based efficient camera interfacing architecture for real time video processing**.** *IET Circuits, Devices & Systems*, 2021. DOI: 10.1049/cds2.12074.
11. Yan Z. et al. An FPGA-Based YOLOv5 Accelerator for Real-Time Industrial Vision Applications**.** *Micromachines*, 2024. DOI: 10.3390/mi15091164.
12. Zeng K. et al. FPGA-based accelerator for object detection: a comprehensive survey. *The Journal of Supercomputing*, 2022. DOI: 10.1007/s11227-022-04415-5.
13. Hozhabr S. H., Giorgi R. A Survey on Real-Time Object Detection on FPGAs**.** *IEEE Access*, 2025. DOI: 10.1109/ACCESS.2025.3544515.

14. Heng C. W. et al. Optimizing FPGA-based YOLO series accelerators: A survey of techniques. *Neurocomputing*, 2025. DOI: 10.1016/j.neucom.2025.130874.
15. Al-Shamma O., Fadhel M. A. Trusted outdoor multi-camera tracking system powered by FPGA. *Journal of Engineering Research*, 2025. DOI: 10.1016/j.jer.2024.10.010.
16. Chowdhury M. R. Z. et al. Towards Next-Generation FPGA-Accelerated Vision-Based Autonomous Driving: A Comprehensive Review. *Signals*, 2025. DOI: 10.3390/signals6040053.
17. Al Amin R., Obermaisser R. Real-Time Object Detection and Classification using YOLO for Edge FPGAs. *ELMAR*, 2025. DOI: 10.1109/ELMAR66948.2025.11193980.
18. Shimoda M., Sato S., Nakahara H. Power Efficient Object Detector with an Event-Driven Camera for Moving Object Surveillance on an FPGA // *IEICE Transactions on Information and Systems*. 2019. Vol. E102-D, No. 5. P. 1020–1028. DOI: 10.1587/transinf.2018RCP0005
19. Kryjak T. Event-Based Vision on FPGAs — a Survey. *Euromicro DSD*, 2024. DOI: 10.1109/DSD64264.2024.00078.
20. Gallego G. et al. Event-Based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. DOI: 10.1109/TPAMI.2020.3008413.
21. Gao Y. et al. REMOT: A Hardware-Software Architecture for Attention-Guided Multi-Object Tracking with Dynamic Vision Sensors on FPGAs. *ACM/SIGDA FPGA*, 2022. DOI: 10.1145/3490422.3502365.
22. Liu M., Delbruck T. EDFLOW: Event Driven Optical Flow Camera With Keypoint Detection and Adaptive Block Matching // *IEEE Transactions on Circuits and Systems for Video Technology*. 2022. Vol. 32, No. 9. P. 5776–5789. DOI: 10.1109/TCSVT.2022.3156653
23. Lu Y., Shi Y., Li Z., Li J., Kim T. T.-H. A Real-Time Event Vision Sensor Based Object Detection and Tracking System for Edge Applications // *2024 IEEE Asian Solid-State Circuits Conference (A-SSCC)*. P. 1–3. DOI: 10.1109/A-SSCC60305.2024.10849082
24. Honarparvar S., Saeedi S., Liang S., Squires J. Design and Development of an Internet of Smart Cameras Solution for Complex Event Detection in COVID-19 Risk Behaviour Recognition // *ISPRS International Journal of Geo-Information*. 2021. Vol. 10, No. 2. Article 81. DOI: 10.3390/ijgi10020081

References

1. Adilzhanova S. et al. A Hardware Platform for Smart Video Monitoring Based on ESP32-CAM and Mojo FPGA (Spartan-6) with Event Activation Triggered by a PIR Sensor. ETASR, 2026. DOI: 10.48084/etasr.18359.
2. Nowroz S. T. et al. A Benchmark Reference for ESP32-CAM Module. 2025. DOI: 10.48550/arXiv.2505.24081.
3. Abordo P. R. C. et al. Smart surveillance system using ESP32 and camera-based motion detection with IM technology. International Journal of Research Studies in Educational Technology, 2024. DOI: 10.5861/ijrset.2024.8012.
4. Okokpujie K. et al. Development of an Affordable Real-Time IoT-Based Surveillance System Using ESP32 and TWILIO API. Journal of Safety and Security Engineering, 2023. DOI: 10.18280/ijsse.130609.
5. Zhaxalikov A. et al. Surveillance Camera Using Wi-Fi Connection. Procedia Computer Science, 2024. DOI: 10.1016/j.procs.2023.12.147.
6. Design and Implementation of ESP32-Based Edge Computing for Object Detection. Sensors, 2025. DOI: 10.3390/s25061656..
7. ESP32-CAM as a Programmable Camera Research Platform. Electronic Imaging, 2022. Lech P. et al. A Low-Cost Energy-Efficient IoT Camera Trap Network for Remote Forest Surveillance. Electronics, 2025. DOI: 10.3390/electronics14214266.
8. Hahn F. et al. Mango Fruit Fly Trap Detection Using Different Wireless Communications. Agronomy, 2023. DOI: 10.3390/agronomy13071736.
9. Kok C. L. et al. Energy-, Cost-, and Resource-Efficient IoT Hazard Detection System with Adaptive Monitoring. Sensors, 2025. DOI: 10.3390/s25061761.

10. Sarkar et al. FPGACam: A FPGA based efficient camera interfacing architecture for real time video processing. IET Circuits, Devices & Systems, 2021. DOI: 10.1049/cds2.12074.

11. Yan Z. et al. An FPGA-Based YOLOv5 Accelerator for Real-Time Industrial Vision Applications. Micromachines, 2024. DOI: 10.3390/mi15091164.

12. Zeng K. et al. FPGA-based accelerator for object detection: a comprehensive survey. The Journal of Supercomputing, 2022. DOI: 10.1007/s11227-022-04415-5.

13. Hozhabr S. H., Giorgi R. A Survey on Real-Time Object Detection on FPGAs. IEEE Access, 2025. DOI: 10.1109/ACCESS.2025.3544515.

14. Heng C. W. et al. Optimizing FPGA-based YOLO series accelerators: A survey of techniques. Neurocomputing, 2025. DOI: 10.1016/j.neucom.2025.130874.

15. Al-Shamma O., Fadhel M. A. Trusted outdoor multi-camera tracking system powered by FPGA. Journal of Engineering Research, 2025. DOI: 10.1016/j.jer.2024.10.010.

16. Chowdhury M. R. Z. et al. Towards Next-Generation FPGA-Accelerated Vision-Based Autonomous Driving: A Comprehensive Review. Signals, 2025. DOI: 10.3390/signals6040053.

17. Al Amin R., Obermaisser R. Real-Time Object Detection and Classification using YOLO for Edge FPGAs. ELMAR, 2025. DOI: 10.1109/ELMAR66948.2025.11193980.

18. Shimoda M., Sato S., Nakahara H. Power Efficient Object Detector with an Event-Driven Camera for Moving Object Surveillance on an FPGA // IEICE Transactions on Information and Systems. 2019. Vol. E102-D, No. 5. P. 1020–1028. DOI: 10.1587/transinf.2018RCP0005

19. Kryjak T. Event-Based Vision on FPGAs — a Survey. Euromicro DSD, 2024. DOI: 10.1109/DSD64264.2024.00078.

20. Gallego G. et al. Event-Based Vision: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022. DOI: 10.1109/TPAMI.2020.3008413.

21. Gao Y. et al. REMOT: A Hardware-Software Architecture for Attention-Guided Multi-Object Tracking with Dynamic Vision Sensors on FPGAs. ACM/SIGDA FPGA, 2022. DOI: 10.1145/3490422.3502365.

22. Liu M., Delbruck T. EDFLOW: Event Driven Optical Flow Camera With Keypoint Detection and Adaptive Block Matching // IEEE Transactions on Circuits and Systems for Video Technology. 2022. Vol. 32, No. 9. P. 5776–5789. DOI: 10.1109/TCSVT.2022.3156653

23. Lu Y., Shi Y., Li Z., Li J., Kim T. T.-H. A Real-Time Event Vision Sensor Based Object Detection and Tracking System for Edge Applications // 2024 IEEE Asian Solid-State Circuits Conference (A-SSCC). P. 1–3. DOI: 10.1109/A-SSCC60305.2024.10849082

24. Honarparvar S., Saeedi S., Liang S., Squires J. Design and Development of an Internet of Smart Cameras Solution for Complex Event Detection in COVID-19 Risk Behaviour Recognition // ISPRS International Journal of Geo-Information. 2021. Vol. 10, No. 2. Article 81. DOI: 10.3390/ijgi10020081