Разработка автономной системы мониторинга производственной зоны с бесконтактной идентификацией на базе микроконтроллера Arduino и одноплатного компьютера Raspberry Pi Zero W

А.Е. Самарина, В. В. Казанцев, А.В. Леонов

Аннотация – Разработана и реализована автономная система мониторинга производственных зон, использующая технологию радиочастотной идентификации и ультразвуковую дистанционную диагностику. Выполнен краткий анализ существующих подходов к организации систем контроля доступа и мониторинга на промышленных предприятиях, определены их ограничения в контексте требований к автономности, масштабируемости и удалённому управлению. Представлена архитектура распределённой системы на основе микроконтроллера Arduino Uno и одноплатного компьютера Raspberry Pi Zero W, обоснован выбор аппаратных компонентов и программных решений. Описана логика обработки данных RFID-модуля MFRC522, ультразвукового датчика HC-SR04 и многоуровневой световой индикации, на основе которой построены алгоритмы управления микроконтроллером. Разработан протокол взаимодействия между микроконтроллером и одноплатным компьютером через интерфейс UART, обеспечивающий передачу телеметрии и команд управления. Практическая значимость работы заключается в возможности применения разработанной системы на производственных предприятиях для контроля доступа к критическим зонам, мониторинга перемещения персонала и регистрации инцидентов безопасности с минимальными затратами на внедрение.

Ключевые слова **– Arduino, Raspberry Pi, RFID, C++, система контроля доступа, ультразвуковой датчик, автономная система, интернет вещей, микроконтроллер, производственный мониторинг, python.**

I. ВВЕДЕНИЕ

Обеспечение безопасности на производственных предприятиях является одной из приоритетных задач современной промышленности. Согласно статистике Ростехнадзора, значительная доля производственных инцидентов связана с несанкционированным доступом персонала в опасные зоны или нарушением регламентов перемещения по территории предприятия. Традиционные системы контроля доступа на основе турникетов и электромагнитных замков не всегда применимы в условиях производственных цехов, где требуется оперативный контроль за перемещением сотрудников без создания физических барьеров.

В последние годы активно развиваются технологии интернета вещей (IoT), позволяющие создавать распределённые системы мониторинга с минимальными затратами на аппаратное обеспечение. [1,2] Использование микроконтроллеров семейства Arduino и одноплатных компьютеров Raspberry Pi открывает возможности для разработки гибких, масштабируемых и экономически эффективных решений в области промышленной автоматизации. Особый интерес представляет применение технологии радиочастотной идентификации (RFID) для бесконтактной идентификации персонала, что позволяет исключить необходимость физического контакта с устройствами считывания и повысить гигиеничность системы. [3,4,5] На наш взгляд, стоит заострить внимание на автономных системах мониторинга, не требующих постоянного подключения к корпоративной сети предприятия. Такие системы особенно актуальны для временных производственных участков, мобильных бригад технического обслуживания, а также для зон с ограниченной инфраструктурой связи. Существующие коммерческие решения в области контроля доступа представляют собой дорогостоящие комплексы, требующие профессионального монтажа и интеграции с инфраструктурой предприятия. [6,7,8] Их применение

затруднительно для малых и средних производственных предприятий, а также для временных объектов. Кроме того, подобные системы не всегда предоставляют гибкие возможности для адаптации под специфические требования конкретного производства. Научная новизна состоит в предложенной трёхуровневой архитектуре, обеспечивающей изолированную работу системы без подключения к корпоративной сети, а также в реализации текстового протокола обмена данными.

II. АРХИТЕКТУРА РАЗРАБОТАННОЙ СИСТЕМЫ

При разработке автономной системы мониторинга производственной зоны нашей целью было обеспечить надёжное функционирование всех подсистем в условиях ограниченных энергетических ресурсов, минимизировать время отклика на события идентификации и обнаружения объектов, а также реализовать удобный интерфейс для удалённого управления и анализа накопленных данных. Такой комплексный подход к проектированию системы требует грамотного выбора аппаратных компонентов и программных технологий, применяемых на каждом уровне архитектуры.

Разработанная система имеет трёхуровневую архитектуру. На нижнем уровне располагается микроконтроллер Arduino Uno, выполняющий функции управляющего контроллера и осуществляющий непосредственное взаимодействие с периферийными устройствами: RFID-модулем MFRC522, ультразвуковым датчиком HC-SR04 и световыми индикаторами. Средний уровень представлен одноплатным компьютером Raspberry Pi Zero W, который выполняет функции сервера логирования, обработки данных и организации точки доступа WiFi. Верхний уровень составляет пользовательский интерфейс, доступный через веб-браузер на любом устройстве, подключённом к беспроводной сети системы. Взаимодействие между Arduino Uno и Raspberry Pi Zero W осуществляется через последовательный интерфейс UART со скоростью 9600 бод. Выбор относительно низкой скорости обмена обусловлен необходимостью обеспечения стабильной передачи данных при возможных электромагнитных помехах в производственной среде. Микроконтроллер передаёт на одноплатный компьютер пакеты телеметрии, содержащие информацию о событиях идентификации, измеренных расстояниях и текущем состоянии системы. В обратном направлении передаются команды управления, позволяющие удалённо изменять режимы работы системы.

В качестве управляющего контроллера был выбран Arduino Uno на базе микроконтроллера ATmega328P. Данное решение обусловлено несколькими факторами. Во-первых, Arduino Uno обладает достаточным количеством цифровых и аналоговых портов ввода-вывода для подключения всех необходимых периферийных устройств: четырёх светодиодных индикаторов, RFID-модуля через интерфейс SPI, ультразвукового датчика и индикатора состояния системы. Во-вторых, встроенный преобразователь USB-UART упрощает процесс программирования и отладки, а также обеспечивает стабильное взаимодействие с Raspberry Pi через последовательный порт. В-третьих, энергопотребление Arduino Uno в активном режиме составляет порядка 50 миллиампер при напряжении 5 вольт, что позволяет обеспечить длительную автономную работу от внешнего аккумулятора. Важным преимуществом платформы Arduino является наличие обширных программных библиотек для работы с различными периферийными устройствами, что существенно ускоряет процесс разработки. [9,10,11] Для реализации функции бесконтактной идентификации был выбран RFID-модуль MFRC522. Данный модуль обеспечивает дальность считывания до 5 сантиметров, что достаточно для надёжной идентификации при поднесении карты оператором. В качестве датчика расстояния применён ультразвуковой модуль HC-SR04, работающий на частоте 40 кГц. Данный датчик обеспечивает измерение расстояния до объектов в диапазоне от 2 сантиметров до 4 метров с точностью около 3 миллиметров. Принцип работы датчика основан на измерении времени распространения ультразвукового импульса от излучателя до объекта и обратно.

Для световой индикации состояния системы применены четыре светодиода различных цветов: зелёный, оранжевый и два красных. Выбор цветовой кодировки обусловлен стандартами промышленной безопасности, где зелёный цвет традиционно означает нормальное состояние, оранжевый означает предупреждение, красный означает опасность. В качестве вычислительного модуля второго уровня был выбран одноплатный компьютер Raspberry Pi Zero W. Выбор именно модели Zero W обусловлен её компактными размерами, невероятно низким энергопотреблением относительно других устройств и наличием встроенного WiFi-адаптера. Последнее позволяет организовать точку доступа для подключения клиентских устройств без использования внешних адаптеров. Для хранения данных используется microSD-карта, на которой размещены операционная система,

Python-скрипты обработки данных, веб-сервер и база данных событий в формате SQLite. Raspberry Pi Zero W получает данные от микроконтроллера Arduino через интерфейс UART, доступный на контактах GPIO 14 (TXD) и GPIO 15 (RXD). Для обработки входящих данных разработан Python-скрипт, работающий в фоновом режиме как системная служба. Этот скрипт выполняет парсинг пакетов телеметрии, сохранение событий в базу данных, автоматическое архивирование устаревших логов и предоставление данных веб-серверу через REST API.

Вся система питается от внешнего аккумулятора. Общее потребление системы составляет приблизительно 250 миллиампер, что обеспечивает теоретическое время автономной работы до 80 часов. На практике реальное время работы составляет около 12 часов непрерывной эксплуатации, что вполне достаточно для одной производственной смены. Опционально система может быть дополнена модулем ESP32, работающим в режиме автономной точки доступа WiFi. В этом случае Raspberry Pi Zero W подключается к сети ESP32 как клиент, что позволяет разгрузить его WiFi-адаптер и снизить энергопотребление. ESP32 питается от отдельного источника, что повышает надёжность беспроводной сети даже при полном разряде основного аккумулятора системы. Нельзя не отметить тот факт, что наша архитектура допускает адаптацию для различных производственных задач, что обеспечивает её явную практическую значимость для целого класса подобных систем. Например, при необходимости контроля нескольких зон одновременно можно подключить несколько микроконтроллеров Arduino к одному Raspberry Pi через USB-хаб или коммутатор последовательных интерфейсов. Аналогично, при наличии корпоративной сети предприятия Raspberry Pi может быть интегрирован в неё через Ethernet-адаптер, что позволит передавать данные в центральную систему мониторинга. [12,13,14] Модульность и масштабируемость составляют достаточно важное архитектурное преимущество разработанного решения перед закрытыми коммерческими аналогами.

III. РЕАЛИЗАЦИЯ ЛОГИКИ УПРАВЛЕНИЯ МИКРОКОНТРОЛЛЕРОМ

Одной из главных задач, которую мы решили на уровне программного обеспечения Arduino Uno, стало обеспечение детерминированного и безотказного обслуживания нескольких асинхронных подсистем в условиях жёстко ограниченных вычислительных ресурсов микроконтроллера ATmega328P. К этим подсистемам относятся ультразвуковой датчик расстояния, модуль радиочастотной идентификации, блок световой индикации и канал связи с Raspberry Pi. Каждая из них требует регулярного опроса или немедленной реакции на события, что исключает применение классических блокирующих задержек. В качестве основы программной логики был выбран неблокирующий цикл событий, синхронизированный по функции millis(). Мы смогли задать строго определённые временные интервалы для всех периодических операций и гарантировать, что время выполнения одной задачи не повлияет на частоту вызова других. [15, 16, 17] Повышение надёжности и отказоустойчивости было достигнуто за счёт трёх архитектурных приёмов, реализованных в ключевых функциях. Первый из них касается обработки ошибок измерения расстояния. Функция measureDistance() построена на принципе эхо-локации. Она формирует на выводе TRIG_PIN импульс длительностью 10 мкс с предварительным сбросом линии в низкий уровень на 5 мкс. Датчик HC-SR04 излучает ультразвуковую волну на частоте 40 кГц, после чего время её распространения до объекта и обратно фиксируется функцией pulseIn на выводе ECHO_PIN с таймаутом 30000 мкс. Таймаут предотвращает бесконечное ожидание при отсутствии цели. Если за указанное время эхо-сигнал не возвращается, функция возвращает не нулевое или приблизительное значение, а код ошибки -1. Обработка этого значения в вышестоящей логике приводит к принудительному гашению всех светодиодов. Тем самым реализуется принцип безопасного отказа, обязательный для современных систем. [18] Второй связан с защитой от ложных срабатываний радиочастотной идентификации. Обработка RFID-событий реализована не как линейная процедура, а как конечный автомат с флагом cardPresent. Метод processRFID, вызываемый на каждой итерации цикла, считывает уникальный идентификатор карты модулем MFRC522, форматирует его в шестнадцатеричную строку и сравнивает с авторизованным значением. Флаг cardPresent исключает многократную обработку одной и той же карты, пока она удерживается в поле считывателя, что защищает от дребезга и случайных переключений. [19, 20, 21] Третьим приёмом является модульная организация кода. Логика световой индикации размещена в функции updateIndicators(). Изоляция этой логики позволяет при необходимости изменить пороги или добавить новые зоны, не затрагивая основной цикл опроса датчиков. Фрагмент этой функции с сокращённым ветвлением выглядит следующим образом:

```
«if (distance > 210) { digitalWrite(LED_GREEN, HIGH); // остальные выключены Serial.println("STATUS:ZONE_SAFE"); } else if (distance > 130) { digitalWrite(LED_ORANGE, HIGH); Serial.println("STATUS:ZONE_CAUTION"); }».
```

Взаимодействие между микроконтроллером и серверным уровнем системы построено на разработанном текстовом протоколе обмена данными. Его ключевая архитектурная роль заключается в обеспечении независимости разработки двух узлов. Каждое сообщение протокола имеет вид KEYWORD:DATA, где разделителем служит первое двоеточие. Такой выбор разделителя позволяет передавать в поле DATA строки, которые сами по себе содержат двоеточия. Это делает протокол расширяемым без модификации транспортного уровня. Микроконтроллер периодически отправляет пакет телеметрии, включающий временную метку millis(), измеренное расстояние и текущее состояние системы. Входящие команды обрабатываются в processIncomingCommands. Команда CMD:STATUS возвращает текущее состояние системы, CMD:RESET принудительно её деактивирует, а CMD:SET_UID: позволяет сменить авторизованный идентификатор без перепрошивки микроконтроллера. Неизвестная команда вызывает ответ ERROR:UNKNOWN_COMMAND. По итогу прошивка микроконтроллера оказывается изолированной от изменений в веб-интерфейсе на Raspberry Pi. Серверная логика может развиваться независимо, пока соблюдается спецификация протокола. [22, 23, 24]

Основной цикл «loop» выстроен на двух временных интервалах, полученных с помощью функции millis(). Измерение расстояния выполняется каждые 250 мс (частота 4 Гц). Этого достаточно для плавного обновления индикации, поскольку инерционность человеческого восприятия делает более частые обновления избыточными. Телеметрия отправляется каждые 500 мс (частота 2 Гц), что поддерживает актуальную картину на сервере и не перегружает последовательный канал. Вызовы processRFID и processIncomingCommands происходят на каждой итерации, благодаря чему реакция на карты и команды остаётся мгновенной. В неактивном состоянии все светодиоды принудительно гасятся. Структура цикла в сокращённом виде, специально для статьи, чтобы показать суть, такова:

```
«if (currentTime - lastMeasurement >= 250) { long distance = measureDistance(); if (systemActive) updateIndicators(distance); else { /* погасить все светодиоды */ } if (currentTime - lastTelemetry >= 500) { sendTelemetry(distance); lastTelemetry = currentTime; } lastMeasurement = currentTime; } processRFID(); processIncomingCommands();».
```

Программное обеспечение микроконтроллера реализует детерминированное обслуживание датчиков, идентификации и канала связи в условиях ограниченных ресурсов. Модульная структура с изолированными функциями упрощает отладку и развитие системы, а текстовый протокол делает взаимодействие с серверным узлом прозрачным и устойчивым к расширению.

IV. ПОДСИСТЕМА ЛОГИРОВАНИЯ И УДАЛЁННОГО ДОСТУПА НА RASPBERRY PI

Одноплатный компьютер Raspberry Pi Zero W выполняет функции серверного узла, замыкающего трёхуровневую архитектуру системы. На него возложены приём телеметрии от Arduino, логирование событий в базу данных, автоматическое архивирование устаревших записей и предоставление веб-интерфейса для удалённого мониторинга. Программная часть реализована на Python 3 с применением микрофреймворка Flask и встроенной библиотеки sqlite3. Значимым архитектурным решением стало разделение программного обеспечения на три независимых компонента, которые взаимодействуют друг с другом только через общую базу данных SQLite. Демон приёма данных оформлен как системная служба systemd и автоматически запускается при загрузке, а в случае аварийного завершения перезапускается. Веб-сервер на Flask обслуживает HTML-интерфейс для оператора и REST API для внешних приложений. Скрипт архивирования запускается планировщиком cron, сжимает устаревшие логи и высвобождает место на карте памяти. Декомпозиция такого рода повышает отказоустойчивость всей подсистемы, сбой веб-сервера не прерывает сбор данных, а ошибка архивирования не затрагивает работу демона и пользовательского интерфейса. Связь через базу данных упрощает модификацию каждого компонента по отдельности и позволяет заменять их без переписывания остальных.

Демон приёма данных реализован в виде класса ArduinoReceiver, инкапсулирующего работу с последовательным портом и базой. При инициализации создаются две таблицы. Таблица events хранит дискретные события (идентификацию карты, активацию и

деактивацию системы) и содержит поля с типом события, временной меткой ISO 8601, дополнительными данными в формате JSON и флагом processed. Таблица telemetry накапливает периодические измерения: расстояние, состояние системы и временную метку Arduino в миллисекундах. Хранение дополнительных сведений в JSON обеспечивает гибкость при развитии системы, поскольку для введения новых полей не требуется изменять схему базы. При создании подключения к SQLite демон использует флаг check_same_thread=False, так как по умолчанию SQLite запрещает обращаться к одному подключению из разных потоков, а в данной архитектуре демон и веб-сервер работают параллельно. Флаг снимает это ограничение и сохраняет все преимущества встроенной СУБД.

Соединение с последовательным портом потребовало учёта аппаратной особенности Arduino. При открытии USB-соединения плата автоматически перезагружается, поэтому сразу после создания объекта Serial вводится двухсекундная пауза. Без неё первые байты телеметрии были бы потеряны. Параллельно задаётся таймаут чтения в одну секунду. Если связь с микроконтроллером временно пропадает, демон не зависает, а возвращает управление в основной цикл и предпринимает следующую попытку чтения. Метод parse_message разбирает входящие строки по первому двоеточию, отделяя ключевое слово от полезной нагрузки. В сумме методика позволяет передавать внутри поля data значения, содержащие двоеточия, и делает протокол устойчивым к расширению формата. В основном цикле демона каждая строка, полученная от Arduino, декодируется и передаётся в parse_message. Если ключевое слово равно TELEMETRY, вызывается обработчик телеметрии; в противном случае сообщение интерпретируется как событие. Телеметрия после извлечения числовых компонентов записывается в таблицу telemetry с текущей временной меткой ISO 8601. События RFID и изменения статуса сохраняются в таблицу events, причём дополнительные сведения сериализуются в JSON, что позволяет единообразно хранить как идентификаторы карт, так и названия зон безопасности.

Ключевой фрагмент, иллюстрирующий установку связи и парсинг:

```
«def connect_serial(self):
    try:
        self.serial_connection = serial.Serial(
            self.port, self.baudrate, timeout=1
        )
        time.sleep(2)
        return True
    except serial.SerialException as e:
        print(f"Failed to connect: {e}")
        return False
def parse_message(self, message):
    message = message.strip()
    if not message:
        return None
    parts = message.split(':', 1)
    if len(parts) < 2:
        return None
    return {'keyword': parts[0], 'data': parts[1]}».
```

Веб-сервер предоставляет два способа взаимодействия. Пользователь открывает HTML-страницу и видит текущее состояние системы, а внешние приложения могут обращаться к REST API. Маршрут /api/status возвращает последнюю запись телеметрии и последнее событие в формате JSON. Если Arduino ещё не передал ни одного пакета, сервер отвечает кодом 503, чтобы клиент мог отличить работающий, но пустой сервер от аварийной ситуации. Маршрут /api/telemetry позволяет запросить исторические данные за произвольный интервал, что используется для построения графиков изменения расстояния во времени. Все SQL-запросы в этих маршрутах параметризованы, что исключает риск SQL-инъекций. Удалённое управление организовано через POST-запросы к маршруту /api/command. Команда в виде строки передаётся в последовательный порт, после чего демон в течение полусекунды ожидает ответа от Arduino. Так можно сменить авторизованный идентификатор карты или принудительно деактивировать систему, не имея физического доступа к оборудованию. Веб-сервер запущен на всех сетевых интерфейсах, поэтому интерфейс доступен с любого устройства внутри WiFi-сети, развёрнутой точкой доступа.

Длительная эксплуатация неизбежно ведёт к накоплению данных на microSD-карте. Для предотвращения переполнения разработан скрипт архивирования, запускаемый ежедневно через cron. Он извлекает из базы записи старше семи дней, сериализует их в JSON и сжимает с помощью gzip. Степень сжатия составляет от 5 до 10 раз в зависимости от характера данных. Операции архивирования и последующего удаления старых записей выполняются атомарно: данные сначала записываются в сжатый файл, и только после этого очищаются в базе. Такой подход гарантирует, что даже при внезапном отключении питания архив не будет утерян. Фрагмент, иллюстрирующий атомарность процесса:

```
«with gzip.open(archive_path, 'wt', encoding='utf-8') as f:
    json.dump(archive_data, f, indent=2)
cursor.execute('DELETE FROM telemetry WHERE timestamp < ?', (cutoff_date,))
cursor.execute('DELETE FROM events WHERE timestamp < ?', (cutoff_date,))
conn.commit()».
```

Предложенная архитектура серверной части опирается на три независимых компонента, связанных через общую базу данных. Использование systemd и cron для управления процессами, SQLite для хранения, Flask для веб-доступа и gzip для сжатия логов обеспечивает надёжную и полностью автономную работу системы без необходимости постоянного администрирования.

V. АНАЛИЗ ЭФФЕКТИВНОСТИ РАЗРАБОТАННОГО РЕШЕНИЯ

После завершения разработки программного обеспечения и сборки аппаратной части было проведено комплексное тестирование системы. Его целью являлась проверка соответствия эксплуатационных характеристик заявленным требованиям, а также оценка того, насколько корректно предложенные архитектурные решения работают в реальных условиях.

Тестирование включало точность и быстродействие датчиков, энергопотребление и время автономной работы, а также надёжность передачи данных. Калибровка ультразвукового датчика HC-SR04 проводилась с эталонным объектом, который последовательно размещался на различных расстояниях в диапазоне от 20 до 300 сантиметров. В каждой точке выполнялась серия из ста замеров. Анализ накопленных данных показал, что в рабочем диапазоне от 50 до 210 сантиметров средняя абсолютная ошибка не превышает 0.5 сантиметра при стандартном отклонении от 0.4 до 0.8 сантиметра. Максимальная зафиксированная ошибка составила 2.1 сантиметра на границе диапазона в 210 сантиметров. Это значение более чем в два раза меньше ширины самой узкой зоны индикации, которая составляет 20 сантиметров. Следовательно, погрешность датчика не способна вызвать ложное переключение между зонами, и четырёхуровневая индикация, заложенная в архитектуру системы, функционирует с достаточным запасом по точности. Выбор HC-SR04 и установленных пороговых значений подтверждается экспериментально. Тестирование RFID-идентификации показало среднее время отклика 187 миллисекунд при стандартном отклонении 23 миллисекунды. Минимальное зафиксированное время составило 152 миллисекунды, максимальное 241 миллисекунду, по сути, мгновенная реакция. Измерение энергопотребления подтвердило расчётные значения, полученные на этапе архитектурного проектирования. Реальное время непрерывной автономной работы от внешнего аккумулятора составило около 12 часов, что полностью покрывает одну производственную смену. Результаты тестирования в совокупности подтверждают, что предложенные архитектурные решения обеспечивают требуемые эксплуатационные характеристики. Система стабильно работает в автономном режиме, точно определяет расстояние до объектов, мгновенно реагирует на предъявление RFID-карт и сохраняет данные без потерь даже при интенсивной эксплуатации. Низкая стоимость компонентов в сочетании с характеристиками, сопоставимыми с коммерческими аналогами, делает разработанное решение экономически эффективным для малых и средних производственных предприятий и ,как мы писали ранее, модульная архитектура позволяет адаптировать его под специфические требования конкретного производства.

ЗАКЛЮЧЕНИЕ

В результате работы предложена и реализована архитектура автономной системы мониторинга производственной зоны на базе Arduino Uno и Raspberry Pi Zero W. Архитектура обеспечивает полностью автономную работу без подключения к корпоративной сети предприятия. Разработан текстовый протокол обмена данными между микроконтроллером и серверным узлом. Протокол устойчив к электромагнитным помехам и допускает расширение формата сообщений без изменения транспортного уровня. Реализована четырёхуровневая световая индикация приближения к опасной зоне. Также спроектирована серверная подсистема из трёх независимых компонентов, которые взаимодействуют через общую базу SQLite. Такая декомпозиция обеспечивает отказоустойчивость и параллельный сбор и выдачу данных в условиях ограниченных вычислительных ресурсов одноплатного компьютера.

БИБЛИОГРАФИЯ

[1] Курмаев Т.И. Сравнение протоколов передачи данных в интернете вещей // Международный научно-исследовательский журнал. - 2022. - №1-1 (115). - С. 45-47.

[2] Бужинская Н.В., Васева Е.С., Искандаров Р.Н., Шубина Н.В. Система контроля и управления доступом на базе микроконтроллеров Arduino // Вестник Дагестанского государственного технического университета. Технические науки. - 2019. - №1. - С. 103-112.
[3] Лоднева О.Н., Ромасевич Е. П. Анализ трафика устройств Интернета вещей // Современные информационные технологии и ИТ-образование. - 2018. - №1. - С. 149-169.
[4] Довгаль В.А., Довгаль Д.В. Построение IoT-системы безопасности на базе Arduino // Вестник Адыгейского государственного университета. - 2018. - № №3 (226). - С. 142-148.
[5] Калхиташвили Д.Ш. Операционные системы интернета вещей: возможности, проблемы и решения // Международный научно-исследовательский журнал. - 2023. - № 5 (131) . - С. 8. doi:10.23670/IRJ.2023.131.18.
[6] Омельченко Е.Я., Танич В.О., Маклаков А.С., Карякина Е.А. Краткий обзор и перспективы применения микропроцессорной платформы Arduino // ЭС и К. - 2013. - №№21. - С. 28-33.
[7] Вострухин А.В., Мастепаненко М.А. , Вахтина Е.А. Энергосберегающий асинхронный интерфейс для беспроводных датчиков // Вестник Томского государственного университета. Управление, вычислительная техника и информатика . - 2023. - №63. - С. 92-102.
[8] Имамов Р.Р., Нурушев А.М. Контроллер удалённого управления трубчатой печью на платформе Arduino // Вестник Югорского государственного университета . - 2013. - №2 (29). - С. 74-80.
[9] Пономаренко В.И., Караваев А.С. Использование платформы Arduino в измерениях и физическом эксперименте // Известия высших учебных заведений. Прикладная нелинейная динамика. - 2014 . - №4. - С. 77-90.
[10] Балабаев С.А., Лупин С.А., and Шакиров Р.Н. Вычислительный кластер на основе смартфонов android и микрокомпьютеров raspberry pi // International Journal of Open Information Technologies . - 2022. - №7. - С. 86-93.
[11] Estrella D.R.Ñ., Estrella A.M.Ñ., Tapia J.L.J., Cazco C.A.O. DESIGN OF PROGRAMMABLE ELECTRONIC CIRCUITS WITH MID-RANGE PIC MICROCONTROLLERS FOR ROBOTICS LABORATORIES // Russian Law Journal. - 2023. - №7S. - С. 540-547.
[12] Кондратьев А.А., Беззубцев А.Ю., Смирнов А.В. Применение распределенной системы обработки данных в задаче построения автоматизированной системы видеонаблюдения // Программные системы: теория и приложения. - 2017. - №1 (28). - С. 135-149.
[13] Зеленский А.А., Харьков М.А., Ивановский С.П., Абдуллин Т.Х. Высокопроизводительная система числового программного управления на базе программируемых логических интегральных схем // Вестник Воронежского государственного технического университета. - 2018. - №5. - С. 8-12.
[14] Petrov A.A., Smirnov A.S., Epishev V.V., Shevtsov A.V. “movement-mirroring” arm exoskeleton in rehabilitation" // Человек. Спорт. Медицина . - 2018. - №3. - С. 113-119
[15] И. В. Богомолов, А. В. Алексиянц, А. В. Шер [и др.] Метод тестирования производительности и стресс-тестирования центральных сервисов идентификации облачных систем на примере Openstack Keystone // Труды Института системного программирования РАН. – 2015. – Т. 27, № 5. – С. 49-58. https://doi.org/10.15514/ISPRAS-2015-27(5)-4
[16] Добрынин С.Л., Бурковский В.Л. Проблематика управления аддитивным производством на основе технологий промышленного интернета вещей // Вестник Воронежского государственного технического университета. - 2021. - №2. - С. 7-13.
[17] Крылов И. Д., Тимкин А. К., Кича И. В., Селищев В. А. АНАЛИЗ СОСТОЯНИЯ ЗАЩИЩЕННОСТИ ИНФОРМАЦИИ IoT-СЕТЕЙ // Известия Тульского государственного университета. Технические науки . - 2023. - № 8. - С. 309-316. doi:10.24412/2071-6168-2023-8-309-310.
[18] Battana, H., Gupta, S., Roopa, M., Sushitha, K., & Reddy, S. Design and Development of a Journal Publishing Platform Using the MERN Stack // International Research Journal on Advanced Engineering and Management (IRJAEM), . - 2025. - №3. - С. 449-456.. https://doi.org/10.47392/IRJAEM.2025.0071
[19] Дикий Д.И. Анализ протокола MQTT на атаки «Отказ в обслуживании» // Научно-технический вестник информационных технологий, механики и оптики. - 2020. - №2. - С. 223-232.
[20] Ivanusa A., Tkachuk, R., Brych T., Balatska V., Tkachenko, A. METHODS AND MODELS FOR THE DESIGN OF AUTOMATED VULNERABILITY DETECTION SYSTEMS IN WEB APPLICATIONS // Bulletin of Lviv State University of Life Safety. - 2024. - №30. - С. 110-122.
[21] Довгаль В.А., Довгаль Д.В. Интернет Вещей: концепция, приложения и задачи // Вестник Адыгейского государственного университета. Серия 4: Естественно-математические и технические науки. - 2018. - №1 (212). - С. 129-135.
[22] Куприяновский В.П., Намиот Д.Е., Дрожжинов В.И., Куприяновская Ю.В., Иванов М.О. Интернет Вещей на промышленных предприятиях // International Journal of Open Information Technologies . - 2016. - №12. - С. 69-78.
[23] Барахнин В. Б., Федотов А. М. Информационная система: взгляд на понятие // Вестник Новосибирского государственного университета. Информационные технологии. - 2007. - №2. - С. 12-19.
[24] Осипов Д.С., Малахов С.В., Якупов Д.О., Храмова А.Е., Крючкова А.С., Джакимова А.Т. Обнаружение уязвимостей интернета вещей и способы защиты от нарушения информационной безопасности IOT // Международный научно-исследовательский журнал. - 2024. - №6 (144). - С. 44. doi:10.60797/IRJ.2024.144.110.

Статья получена: 10.01.2026

Самарина А.Е. - Смоленский государственный университет, email: a.e.samarina@gmail.com
Казанцев В.В. – Смоленский государственный университет, email: valera.kaz2017@yandex.ru
Леонов А.В. - Смоленский государственный университет, email: alexsandr.leo@yandex.ru

Development of an autonomous monitoring system for a production area with contactless identification based on an Arduino microcontroller and a Raspberry Pi Zero W single-board computer

A.E. Samarina, V. V. Kazantsev, A.V. Leonov

Abstract – An autonomous monitoring system for production areas using radio frequency identification technology and ultrasound remote diagnostics has been developed and implemented. A brief analysis of existing approaches to the organization of access control and monitoring systems in industrial enterprises has been performed, and their limitations have been identified in the context of requirements for autonomy, scalability, and remote management. The architecture of a distributed system based on an Arduino Uno microcontroller and a Raspberry Pi Zero W single-board computer is presented, and the choice of hardware components and software solutions is justified. The logic of data processing of the MFRC522 RFID module, HC-SR04 ultrasonic sensor and multi-level light indication based on which microcontroller control algorithms are built is described. A protocol has been developed for the interaction between a microcontroller and a single-board computer via a UART interface, which provides the transmission of telemetry and control commands. The practical significance of the work lies in the possibility of using the developed system in production facilities to control access to critical areas, monitor personnel movement and register security incidents with minimal implementation costs.

Keywords – Arduino, Raspberry Pi, RFID, access control system, ultrasonic sensor, autonomous system, industrial safety, Internet of Things, microcontroller, event logging, production monitoring

REFERENCES

[1] Kurmaev T.I. Sravnenie protokolov peredachi dannyh v internete veshchej // Mezhdunarodnyj nauchno-issledovatel'skij zhurnal. - 2022. - №1-1 (115). - S. 45-47.

[2] Buzhinskaya N.V., Vaseva E.S., Iskandarov R.N., SHubina N.V. Sistema kontrolya i upravleniya dostupom na baze mikrokontrollerov Arduino // Vestnik Dagestanskogo gosudarstvennogo tekhnicheskogo universiteta. Tekhnicheskie nauki. - 2019. - №1. - S. 103-112.

[3] Lodneva O.N., Romasevich E. P. Analiz trafika ustrojstv Interneta veshchej // Sovremennye informacionnye tekhnologii i IT-obrazovanie. - 2018. - №1. - S. 149-169.

[4] Dovgal' V.A., Dovgal' D.V. Postroenie IoT-sistemy bezopasnosti na baze Arduino // Vestnik Adygejskogo gosudarstvennogo universiteta. - 2018. - № №3 (226). - S. 142-148.

[5] Kalhitashvili D.SH. Operacionnye sistemy interneta veshchej: vozmozhnosti, problemy i resheniya // Mezhdunarodnyj nauchno-issledovatel'skij zhurnal. - 2023. - № 5 (131) . - S. 8. doi:10.23670/IRJ.2023.131.18.

[6] Omel'chenko E.YA., Tanich V.O., Maklakov A.S., Karyakina E.A. Kratkij obzor i perspektivy primeneniya mikroprocessornoj platformy Arduino // ES i K. - 2013. - №№21. - S. 28-33.

[7] Vostruhin A.V., Mastepanenko M.A. , Vahtina E.A. Energosberegayushchij asinhronnyj interfejs dlya besprovodnyh datchikov // Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitel'naya tekhnika i informatika . - 2023. - №63. - S. 92-102.

[8] Imamov R.R., Nurushev A.M. Kontroller udalyonnogo upravleniya trubchatoj pech'yu na platforme Arduino // Vestnik YUgorskogo gosudarstvennogo universiteta . - 2013. - №2 (29). - S. 74-80.

[9] Ponomarenko V.I., Karavaev A.S. Ispol'zovanie platformy Arduino v izmereniyah i fizicheskom eksperimente // Izvestiya vysshih uchebnyh zavedenij. Prikladnaya nelinejnaya dinamika. - 2014 . - №4. - S. 77-90.

[10] Balabaev S.A., Lupin S.A., and SHakirov R.N. Vychislitel'nyj klaster na osnove smartfonov android i mikrokomp'yuterov raspberry pi // International Journal of Open Information Technologies . - 2022. - №7. - S. 86-93.

[11] Estrella D.R.Ñ., Estrella A.M.Ñ., Tapia J.L.J., Cazco C.A.O. DESIGN OF PROGRAMMABLE ELECTRONIC CIRCUITS WITH MID-RANGE PIC MICROCONTROLLERS FOR ROBOTICS LABORATORIES // Russian Law Journal. - 2023. - №7S. - S. 540-547.

[12] Kondrat'ev A.A., Bezzubcev A.YU., Smirnov A.V. Primenenie raspredelennoj sistemy obrabotki dannyh v zadache postroeniya avtomatizirovannoj sistemy videonablyudeniya // Programmnye sistemy: teoriya i prilozheniya. - 2017. - №1 (28). - S. 135-149.

[13] Zelenskij A.A., Har'kov M.A., Ivanovskij S.P., Abdullin T.H. Vysokoproizvoditel'naya sistema chislovogo programmnogo upravleniya na baze programmiruemyh logicheskih integral'nyh skhem // Vestnik Voronezhskogo gosudarstvennogo tekhnicheskogo universiteta. - 2018. - №5. - S. 8-12.

[14] Petrov A.A., Smirnov A.S., Epishev V.V., Shevtsov A.V. “movement-mirroring” arm exoskeleton in rehabilitation" // CHelovek. Sport. Medicina . - 2018. - №3. - S. 113-119

[15] I. V. Bogomolov, A. V. Aleksiyanc, A. V. SHer [i dr.] Metod testirovaniya proizvoditel'nosti i stress-testirovaniya

central'nyh servisov identifikacii oblachnyh sistem na primere Openstack Keystone // Trudy Instituta sistemnogo programmirovaniya RAN. – 2015. – T. 27, № 5. – S. 49-58. https://doi.org/10.15514/ISPRAS-2015-27(5)-4

[16] Dobrynin S.L., Burkovskij V.L. Problematika upravleniya additivnym proizvodstvom na osnove tekhnologij promyshlennogo interneta veshchej // Vestnik Voronezhskogo gosudarstvennogo tekhnicheskogo universiteta. - 2021. - №2. - S. 7-13.

[17] Krylov I. D., Timkin A. K., Kicha I. V., Selishchev V. A. ANALIZ SOSTOYANIYA ZASHCHISHCHENNOSTI INFORMACII IoT-SETEJ // Izvestiya Tul'skogo gosudarstvennogo universiteta. Tekhnicheskie nauki . - 2023. - № 8. - S. 309-316. doi:10.24412/2071-6168-2023-8-309-310.

[18] Battana, H., Gupta, S., Roopa, M., Sushitha, K., & Reddy, S. Design and Development of a Journal Publishing Platform Using the MERN Stack // International Research Journal on Advanced Engineering and Management (IRJAEM), . - 2025. - №3. - S. 449-456.. https://doi.org/10.47392/IRJAEM.2025.0071

[19] Dikij D.I. Analiz protokola MQTT na ataki «Otkaz v obsluzhivanii» // Nauchno-tekhnicheskij vestnik informacionnyh tekhnologij, mekhaniki i optiki. - 2020. - №2. - S. 223-232.

[20] Ivanusa A., Tkachuk, R., Brych T., Balatska V., Tkachenko, A. METHODS AND MODELS FOR THE DESIGN OF AUTOMATED VULNERABILITY DETECTION SYSTEMS IN WEB APPLICATIONS // Bulletin of Lviv State University of Life Safety. - 2024. - №30. - S. 110-122.

[21] Dovgal' V.A., Dovgal' D.V. Internet Veshchej: koncepciya, prilozheniya i zadachi // Vestnik Adygejskogo gosudarstvennogo universiteta. Seriya 4: Estestvenno-matematicheskie i tekhnicheskie nauki. - 2018. - №1 (212). - S. 129-135.

[22] Kupriyanovskij V.P., Namiot D.E., Drozhzhinov V.I., Kupriyanovskaya YU.V., Ivanov M.O. Internet Veshchej na promyshlennyh predpriyatiyah // International Journal of Open Information Technologies . - 2016. - №12. - S. 69-78.

[23] Barahnin V. B., Fedotov A. M. Informacionnaya sistema: vzglyad na ponyatie // Vestnik Novosibirskogo gosudarstvennogo universiteta. Informacionnye tekhnologii. - 2007. - №2. - S. 12-19.

[24] Osipov D.S., Malahov S.V., YAkupov D.O., Hramova A.E., Kryuchkova A.S., Dzhakimova A.T. Obnaruzhenie uyazvimostej interneta veshchej i sposoby zashchity ot narusheniya informacionnoj bezopasnosti IOT // Mezhdunarodnyj nauchno-issledovatel'skij zhurnal. - 2024. - №6 (144). - S. 44. doi:10.60797/IRJ.2024.144.110.