СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день микроэлектронику трудно представить без такой важной составляющей, как микроконтроллеры. На помощь человеку приходит всё больше и больше электронных помощников. Многие привыкли к ним и часто даже не подозревают, что во многих подобных устройствах присутствует микроконтроллер.

Микроконтроллерные технологии очень эффективны в решении некоторых задач. Одно и то же устройство, которое раньше собиралось на традиционных схемотехнических элементах, будучи собрано с применением микроконтроллеров, становится проще в проектировании и эксплуатации. Оно не требует регулировки и зачастую меньше в размере. Кроме того, с применением микроконтроллеров появляются практически безграничные возможности по добавлению новых потребительских функций и возможностей к уже существующим устройствам.

Телевизор, мобильный телефон, игровая приставка, стиральная машина, калькулятор – это те самые устройства, которые имеют как минимум один микроконтроллер. В случае данного дипломного проекта на микроконтроллере выполняется коммуникационный контроллер с шифрованием данных для системы «Умный дом».

Устройства систем типа «Умный дом» используются в быту для облегчения осуществления ежедневных рутинных действий в домашнем обиходе. Обычно это регулирование параметров микроклимата, управление «умной» бытовой техники дистанционно, охранные системы и системы санкционированного доступа в помещение. Для реализации подобных систем необходимо подключение множества различных устройств друг к другу и/или к общему командному серверу.

Одной из важнейших проблем таких сетей на данный момент является их незащищённость от несанкционированного доступа. Эту проблему и должно решать разрабатываемое в данном проекте устройство, выполняя шифрование передаваемых данных в сети устройств «Умного дома». Контроллер будет позволять осуществлять шифрование и передачу данных между устройствами в системе «Умный дом» по различным интерфейсам, включая беспроводную связь.

Задачи, которые были решены в рамках дипломного проектирования:

- анализ существующих систем и готовых решений;
- разработка базового прототипа;
- проектирование и разработка устройства.

1 ОБЗОР АНАЛОГИЧНЫХ РАЗРАБОТОК

Большинство разработчиков систем умного дома не раскрывает подробностей про то, имеется ли в их продукте встроенная защита передачи данных. Делается это по нескольким причинам. Во-первых, основная масса потребителей не компетентна в достаточной мере для проведения сравнительного анализа, а так же не слишком заинтересована в мерах обеспечения повышенной безопасности сети. Во-вторых, производители не стремятся публиковать информацию об этих характеристиках собственных систем, чтобы избежать интереса злоумышленников к их разработкам в сфере информационной безопасности.

Рассмотрим некоторые стандарты сетевых протоколов, а также готовые решения на рынке и способы защиты информации в них.

ZigBee является одной из самых распространённых в использовании на данный момент спецификацией сетевых протоколов для интернета вещей (IoT) [1]. Основная особенность технологии ZigBee заключается в том, что она при малом энергопотреблении поддерживает не только простые топологии сети («точка-точка», «дерево» и «звезда»), но и самоорганизующуюся и самовосстанавливающуюся ячеистую (mesh) топологию с ретрансляцией и маршрутизацией сообщений. Кроме того, спецификация ZigBee содержит возможность выбора алгоритма маршрутизации в зависимости от требований приложения и состояния сети, механизм стандартизации приложений — профили приложений, библиотека стандартных кластеров, конечные точки, привязки, гибкий механизм безопасности (рисунок 1.1), а также обеспечивает простоту развертывания, обслуживания и модернизации.

Шифрование данных в ZigBee основано на протоколе 802.15.4 (рисунок 1.2). Алгоритм шифрования, используемый в ZigBee, — это AES (Advanced Encryption Standard) [2] с длиной ключа 128 бит (16 байт). Алгоритм AES используется не только для шифрования информации, но и для проверки отправляемых данных. Эта концепция называется проверкой целостности данных и достигается при помощи кода целостности сообщения (MIC), также называемого кодом аутентификации сообщения (MAC), который добавляется к сообщению. Этот код обеспечивает целостность заголовка MAC и прикрепленных данных полезной нагрузки.

Он создается путем шифрования частей кадра MAC-адреса IEEE с использованием ключа сети, поэтому, если мы получим сообщение от недоверенного узла, мы увидим, что MAC-адрес, сгенерированный для

отправленного сообщения, не соответствует тому, который был бы сгенерирован с использованием текущего секретного ключа, поэтому мы можем отбросить это сообщение. MAC может иметь разный размер: 32, 64, 128 бит, однако всегда создается с использованием 128-битного алгоритма AES. Его размер - это просто длина битов, которая прикреплена к каждому кадру. Чем больше размер, тем безопаснее (хотя сообщение может принять меньшую полезную нагрузку). Защита данных осуществляется путем шифрования поля полезных данных с помощью 128-битного ключа.

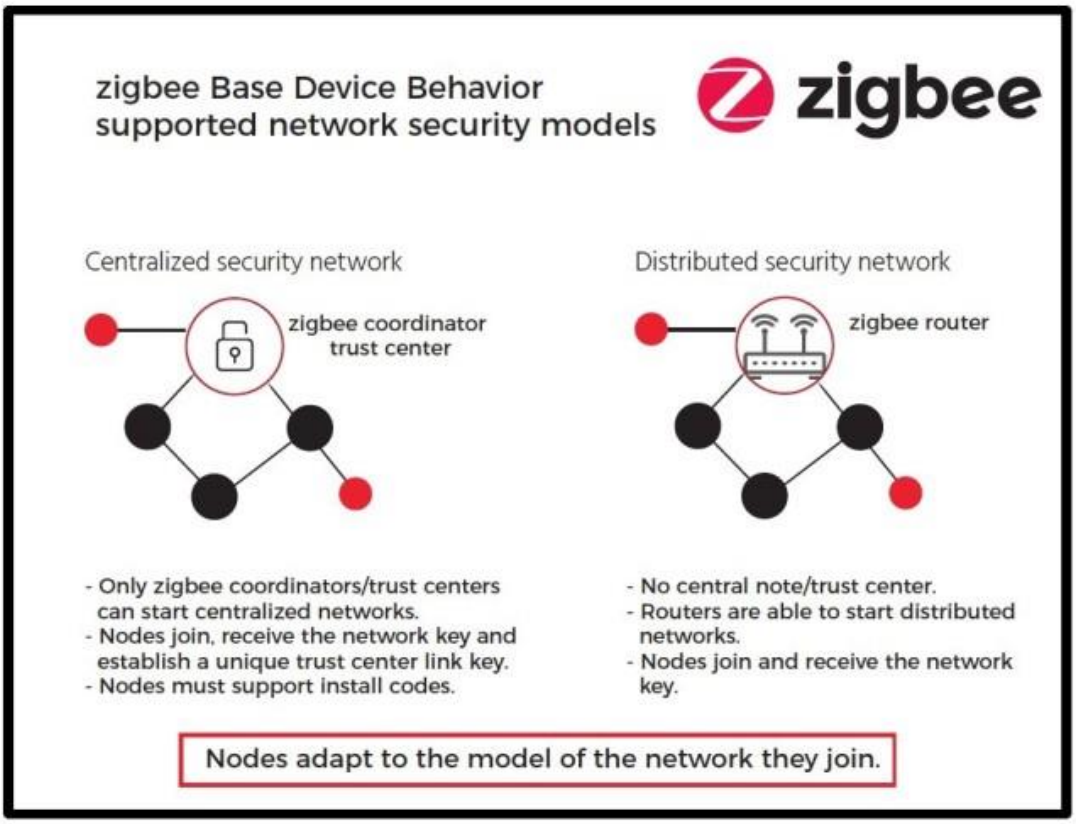


Рисунок 1.1 – Модели безопасности ZigBee

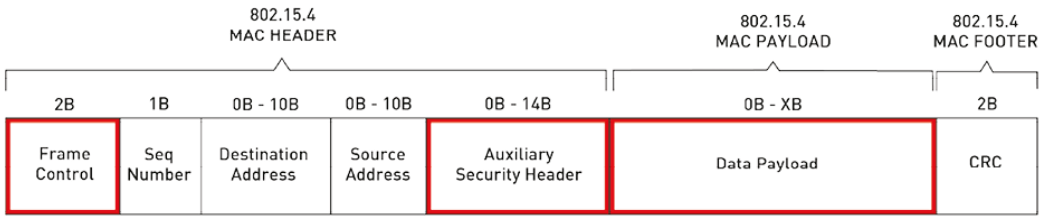


Рисунок 1.2 – Структура посылки в протоколе IEEE 802.15.4

Системы умного дома разработчика MiMiSmart (рисунок 1.3) [3] также используют передовой алгоритм шифрования Advanced Encryption Standard (AES), который широко используется в банковской сфере в том числе. Так что можно прийти к выводу, что данное шифрование довольно популярна на рынке IoT.

Рисунок 1.3 – Схема устройства систем умного дома MiMiSmart

KNX — коммуникационная шина, широко используемая для автоматизации зданий [4]. Стандарт шины KNX стал развитием более ранней разработки EIB (European Installation Bus). EIB — устаревшее обозначение, но оно продолжает использоваться, особенно в Европе. Иногда используется обозначение EIB/KNX. Продукция KNX (рисунок 1.4) распространялась под несколькими торговыми марками. Наиболее известны Instabus, ABB i-Bus, Tebis, Theben.

Рисунок 1.4 – KNX-Transceiver-Board фирмы Elmos

Автоматизация дома и зданий с KNX безопасна. Данная технология соответствует всем необходимым правилам безопасности. Технология KNX Secure стандартизирована в соответствии с EN 50090-3-4 [5], что означает, что KNX успешно блокирует хакерские атаки на цифровую инфраструктуру сетевых зданий. Таким образом, сводится к минимуму риск цифровых взломов.

Кроме того, KNX Secure соответствует самым высоким стандартам шифрования (согласно ISO 18033-3, таким как шифрование AES 128 CCM), чтобы эффективно предотвращать атаки на цифровую инфраструктуру зданий и достигать высочайшего уровня защиты данных.

KNX Secure гарантирует максимальную защиту. KNX IP Secure расширяет протокол IP таким образом, что все передаваемые телеграммы и данные полностью зашифрованы. KNX Data Secure эффективно защищает данные пользователя от несанкционированного доступа и манипуляций с помощью шифрования и аутентификации.

Производителем и разработчиком систем KNX является компания Xiot (рисунок 1.5).

Рисунок 1.5 – Xiot – разработчик KNX Secure

Control4 — поставщик систем автоматизации и сетевых систем для дома и бизнеса, предлагающий персонализированную и унифицированную систему умного дома для автоматизации и управления подключенными устройствами, включая освещение, аудио, видео, климат-контроль, внутреннюю связь и безопасность [6]. Control4 (рисунок 1.6) использует AES для шифрования данных, а также иные средства защиты персональной информации, которые не раскрываются компанией.

Рисунок 1.6 – Компания Control4

Z-Wave — это распространённый радио протокол передачи данных, предназначенный для домашней автоматизации [7]. Характерной особенностью Z-Wave является стандартизация от физического уровня, до уровня приложения. Т.е. протокол покрывает все уровни OSI классификации, что позволяет обеспечивать совместимость устройств разных производителей при создании гетерогенных сетей.

Что позволяет делать технология Z-Wave:

- управление освещением (реле/диммеры), шторами, рольставнями и воротами;
- управление жалюзи и другими моторами (10-230 В);
- включение/выключение любых нагрузок до 3.5 кВт (модуль в розетку или встраиваемое реле);
- дистанционное управление с ПДУ;

- управление обогревом (электрические тёплые полы с защитой от перегрева, электро котлы и радиаторы, термостаты для водяных клапанов радиаторов);
- управление кондиционерами (через ИК интерфейс имитируя пульт);
- детектирование тревожных событий (датчики движения, открытия двери/окна, протечки, сухие контакты);
- мониторинг состояния (датчики температуры, влажности, освещённости);
- управление A/V аппаратурой (по протоколу Z-Wave или через ИК интерфейс имитируя пульт);
- связь с любым программным обеспечением через ПК контроллер;
- сбор данных со счётчиков.

В различных готовых решениях на основе данной технологии для шифрования обычно используется AES128. На пример в модуле ZM2102, изображенном на рисунке 1.7, на одном кристалле SD3402.

Рисунок 1.7 – Модуль ZM2102

В итоге можно прийти к следующим выводам:

- большинство производителей используют собственные реализации одного и того же международного стандарта, что повышает риски обнаружения злоумышленника уязвимости и использования ее сразу для всех устройств;
- полное или частичное сокрытие способов реализации криптозащиты является ошибкой, так как исследователи не смогут обнаружить из общедоступных источников уязвимости в алгоритмах или конкретной реализации раньше злоумышленников;
- не было обнаружено использование отечественных методов шифрования данных, что в том числе и повлияло на выбор

алгоритма шифрования для разрабатываемого в дипломном проекте контроллера;

- практически полное отсутствие на рынке реализаций на отдельных микроконтроллерах, что могло бы повысить универсальность и переиспользуемость на различных системах «Умного дома» и не только, а также повысить быстродействие системы.

Из вышеизложенных тезисов и были сделаны выводы о необходимости разработки коммуникационного контроллера с шифрованием данных для системы «Умный дом».

2 АНАЛИЗ ТЕХНИЧЕСКОГО ЗАДАНИЯ

В данном дипломном проекте необходимо разработать и реализовать коммуникационный контроллер с шифрованием данных для системы «Умный дом».

Системы «Умный дом» представляют собой различные электронные устройства прежде всего бытового назначения, связанные в сети посредством микроконтроллеров и линий связи.

Разработке подлежит аппаратная и программная части контроллера. Контроллер должен быть построен на базе микроконтроллера. Система должна иметь достаточное быстродействие для обработки команд с передачей данных по сети. Система должна иметь удобное управление для пользователя.

В современных системах в основном используется беспроводная связь на основе сети Wi-Fi, о чем можно судить по анализу аналогов из предыдущего раздела данного дипломного проекта. Именно посредством микроконтроллера ESP8266EX и будет осуществлен данный канал связи.

Шифрованные данные с устройства системы «Умный дом» будут отправляться по беспроводной связи на удаленный сервер, а также будут приниматься входящие шифрованные данные с этого сервера для управления устройством.

В качестве аппаратной платформы выбран микроконтроллер ATmega328P благодаря своей простоте в программировании и встраивании в различные системы на кристалле и универсальности. Именно на его вычислительных мощностях и будет реализовано шифрование данных и их транспортировка по всем имеющимся информационным каналам.

В качестве метода шифрования выбран государственный стандарт Республики Беларусь СТБ.34.101.31-2011 «Криптографические алгоритмы шифрования и контроля целостности» [8]. Выбор был сделан на основе соответствия проектируемого изделия отечественным стандартам, что может сильно упростить выход на внутренний рынок.

Для получения данных с устройства системы «Умный дом» и передачи на него будет использован интерфейс SPI, обеспечивающий высокую скорость передачи данных.

Загрузка исходных данных (ключа шифрования, синхропосылки, данных сервера для подключения по Wi-Fi) на коммуникационный контроллер будет осуществляться по последовательному интерфейсу передачи данных UART.

Требования к конструкции разрабатываемого устройства вытекают из его функционального назначения и условий его эксплуатации. Конструкция прибора должна обеспечивать ремонтопригодность, удобство в эксплуатации, иметь, по возможности малые габариты и вес, и высокую надежность в работе. Конструкция прибора должна отвечать требованиям к технологичности по ГОСТ 14.205-83 [9].

Технические характеристики:

- напряжение питания: 5 В;
- ток потребления: 500 мА;
- количество каналов управления: 1;
- разрядность АЦП: 10 бит;
- напряжение на входе АЦП: до 1 В.

Общие технические требования:

- класс точности 1,0 по ГОСТ 31819.21-2012 [10];
- температура окружающей среды от +10 до +35°С;
- относительная влажность окружающей среды до 85% при температуре 25°С;
- время непрерывной работы микропроцессорного блока без технического обслуживания – 720 часов;
- потребляемая мощность не более 10 Вт;
- масса не более 1 кг;
- средняя наработка на отказ прибора с учетом технического обслуживания – 30000 часов;
- установленная безотказная наработка 3000 часов при уровне доверия 0,8;
- полный средний срок службы устройства – 5 лет;
- средний ресурс до среднего ремонта – 10000 часов;
- установленный срок сохраняемости 1 год на период до ввода устройства в эксплуатацию.

Изделия должны сохранять свои параметры в пределах норм, установленных техническими заданиями, стандартами или техническими условиями в течение сроков службы и сроков сохраняемости, указанных в техническом задании после или в процессе воздействия климатических факторов, значения которых установлены ГОСТ 15150-69 [11].

Изделия предназначают для эксплуатации в одном или нескольких макроклиматических районах и изготавливают в различных климатических исполнениях.

Разрабатываемое устройство предназначено для эксплуатации в районах с умеренным и холодным климатами.

К макроклиматическому району с умеренным климатом относятся районы, где средняя из абсолютных максимумов температура воздуха равна или ниже + 40 °С.

К макроклиматическому району с холодным климатом относятся районы, в которых средняя из ежегодных абсолютных минимумов температура воздуха ниже - 45 °С.

Исходя из вышесказанного, устройство будет изготавливаться в климатическом исполнении УХЛ.

Следует отметить, что изделия в исполнении УХЛ могут эксплуатироваться в теплом влажном, жарком сухом и очень жарком сухом климатических районах по ГОСТ 16350-80 [12], в которых средняя из ежегодных абсолютных максимумов температура воздуха выше 35 °С, и сочетание температуры, равной или выше -10 °С, и относительной влажности, равной или выше 90%, наблюдается более 1 часов в сутки за непрерывный период более двух месяцев в году.

Изделия в различных климатических исполнениях в зависимости от места размещения при эксплуатации в воздушной среде на высотах до 4300 м изготавливают по категориям размещения изделий.

При разработке должны использоваться только такие объекты интеллектуальной собственности, права на которые приобретены (получены) и используются без нарушений прав на интеллектуальную собственность третьих лиц. Это требование должно обеспечивать соблюдение авторских, смежных, патентных и иных прав.

3 РАЗРАБОТКА СТРУКТУРЫ УСТРОЙСТВА

3.1 Основание для разработки

Принцип действия устройства сводится к следующему. По интерфейсу UART производится взаимодействие контроллера с внешним компьютером для программной установки данных сервера Wi-Fi, ключа шифрования и синхропосылки. SPI интерфейс необходим для связи контроллера с устройством системы «Умный дом». По этому интерфейсу осуществляется нешифрованный канал связи. Wi-Fi контроллер отвечает за организацию связи по беспроводной сети с сервером системы. Данный контроллер подключается к основному микроконтроллеру устройства по программному последовательному порту. На управляющем микроконтроллере осуществляется шифрование канала беспроводной связи в обоих направлениях. Модуль ГТИ осуществляет генерацию тактовых импульсов на необходимой для корректной работы микроконтроллера частоте. Питание устройства обеспечивается внешним подключением напряжения +5 В по разъему питания. Тактовые кнопки используются для управления состоянием контроллером, а именно для сброса устройства и для начала загрузки исходных данных. Светодиодная индикация необходима для индикации осуществления передачи данных по SPI и для отображения подключения к серверу по сети Wi-Fi. Таким образом, последовательная цифровая передача и обработка информации приводит к системе, обладающей такими свойствами, как хорошее качество, большая скорость передачи-приема сообщений, высокая степень автоматизации, надежность, гибкость и т.д.

Структурная схема устройства изображена на рисунке 3.1

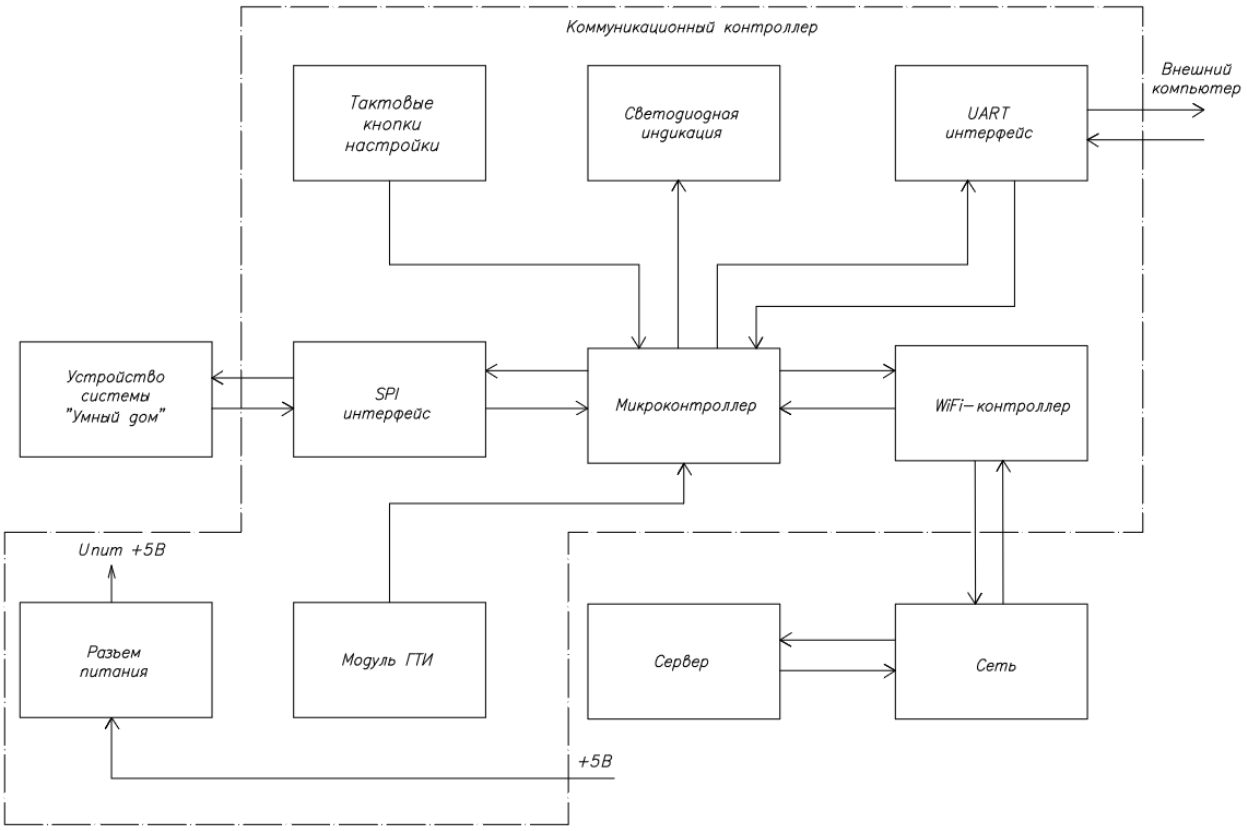


Рисунок 3.1 – Схема электрическая структурная устройства

3.2 Источники разработки

При разработке использовались следующие источники:

1. ГОСТ 34.003-90. Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения [13].
2. ГОСТ 34.201-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначение документов при создании автоматизированных систем [14].
3. ГОСТ 34.601-90. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания [15].
4. ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы [16].
5. ГОСТ Р МЭК 62657-2-2016. Сети промышленной коммуникации. Беспроволочные коммуникационные сети [17].

4 АППАРАТНО-ПРОГРАММНАЯ РЕАЛИЗАЦИЯ УСТРОЙСТВА

4.1 Составление функциональной схемы

На основе схемы структурной электрической разработана схема функциональная электрическая.

Блок управления согласует все блоки, входящие в структурную схему. На блоке управления выполняется управляющая программа. Блок управления считывает информацию с цифровых датчиков, обрабатывает её, производит необходимые вычисления, генерирует сигнал управления на исполнительные устройства. С учетом функций блока управления для его реализации лучше использовать микроконтроллер.

Основной функциональный узел схемы – микроконтроллер DD1. С ним связаны все остальные функциональные узлы разрабатываемого устройства. Микроконтроллер – микросхема, предназначенная для управления электронными устройствами.

Типичный микроконтроллер сочетает на одном кристалле функции процессора и периферийных устройств, содержит ОЗУ и (или) ПЗУ. По сути, это однокристальный компьютер, способный выполнять относительно простые задачи.

Кроме ОЗУ, микроконтроллер может иметь встроенную энергонезависимую память для хранения программы и данных. Многие модели контроллеров вообще не имеют шин для подключения внешней памяти.

Наиболее дешёвые типы памяти допускают лишь однократную запись, либо хранимая программа записывается в кристалл на этапе изготовления (конфигурацией набора технологических масок). Такие устройства подходят для массового производства в тех случаях, когда программа контроллера не будет обновляться. Другие модификации контроллеров обладают возможностью многократной перезаписи программы в энергонезависимой памяти.

Неполный список периферийных устройств, которые могут использоваться в микроконтроллерах, включает в себя:

- универсальные цифровые порты, которые можно настраивать как на ввод, так и на вывод;
- различные интерфейсы ввода-вывода, такие, как UART, I²C, SPI, CAN, USB, IEEE 1394, Ethernet;
- аналого-цифровые и цифро-аналоговые преобразователи;

- компараторы;
- широтно-импульсные модуляторы (ШИМ-контроллер);
- таймеры;
- контроллеры бесколлекторных двигателей, в том числе шаговых;
- контроллеры дисплеев и клавиатур;
- радиочастотные приемники и передатчики;
- массивы встроенной флеш-памяти;
- встроенный тактовый генератор и сторожевой таймер.

Микроконтроллер отличается от микропроцессора интегрированными в микросхему устройствами ввода-вывода, таймерами и другими периферийными устройствами.

Факторы цены и энергопотребления ограничивают тактовую частоту контроллеров. Хотя производители стремятся обеспечить работу своих изделий на высоких частотах, они, в то же время, предоставляют заказчикам выбор, выпуская модификации, рассчитанные на разные частоты и напряжения питания. Во многих моделях микроконтроллеров используется статическая память для ОЗУ и внутренних регистров. Это даёт контроллеру возможность работать на меньших частотах и даже не терять данные при полной остановке тактового генератора. Часто предусмотрены различные режимы энергосбережения, в которых отключается часть периферийных устройств и вычислительный модуль.

Использование в современном микроконтроллере достаточного мощного вычислительного устройства с широкими возможностями, построенного на одной микросхеме вместо целого набора, значительно снижает размеры, энергопотребление и стоимость построенных на его базе устройств.

В то время как 8-разрядные микропроцессоры общего назначения полностью вытеснены более производительными моделями, 8-разрядные микроконтроллеры продолжают широко использоваться. Это объясняется тем, что существует большое количество применений, в которых не требуется высокая производительность, но важна низкая стоимость. В то же время, есть микроконтроллеры, обладающие большими вычислительными возможностями, например, цифровые сигнальные процессоры, применяющиеся для обработки большого потока данных в реальном времени (например, аудио-, видеопотоков).

Для индикации информации служат светодиоды HL1,2. HL1,2 представляют собой два зеленых светодиода.

В качестве Wi-Fi-контроллера будет использоваться микроконтроллер DD2, представляющий собой кристалл ESP8266EX, подключаемый по Programmed Serial (программный последовательный порт).

Для питания DD2 служит регулятор напряжения на 3,3 В DA1.

WA1 является антенной, служащей для передачи данных с контроллера DD2 беспроводным методом.

SW1,2 представляют собой тактовые кнопки, подключаемые к микроконтроллеру DD1.

XS2 и XS3 являются интерфейсами SPI и UART соответственно. Они служат для внешних подключений к разрабатываемому устройству.

ZQ1 и ZQ2 – это кварцевые резонаторы, являющиеся основной частью генераторов тактовых импульсов, соответствующих требуемым для микроконтроллеров тактовым частотам.

Схема электрическая функциональная коммуникационного контроллера с шифрованием данных для системы «Умный дом» составляется согласно разработанной структурной схеме и имеет вид, приведённый на рисунке 4.1.

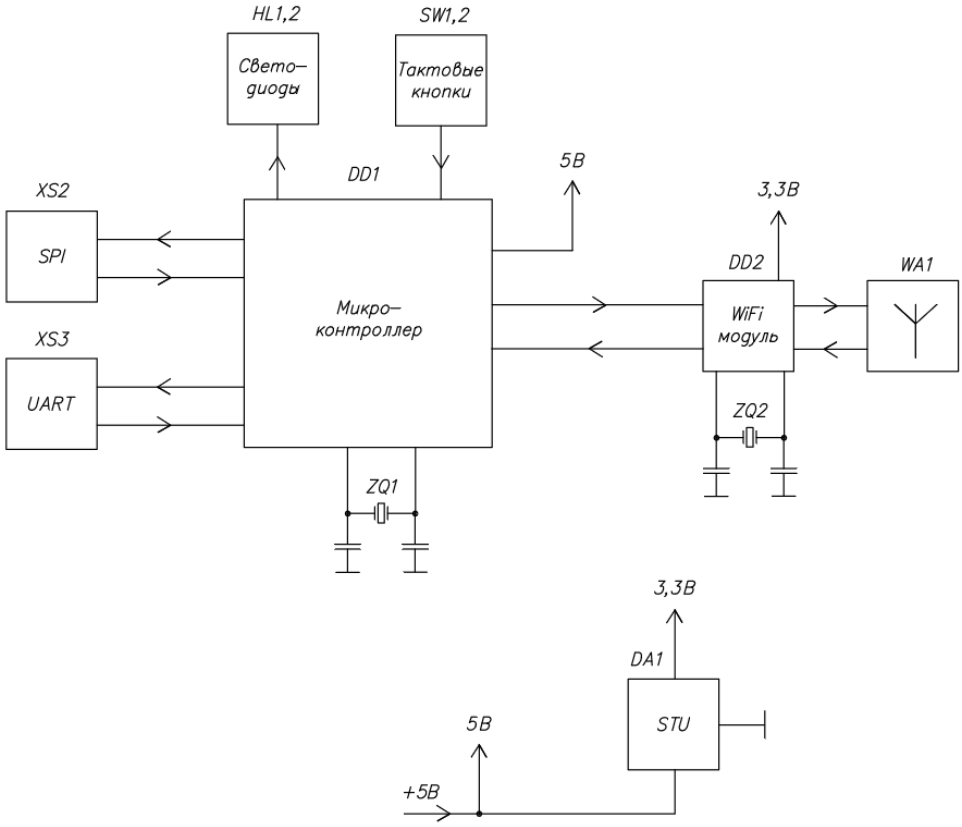


Рисунок 4.1 – Схема электрическая функциональная устройства

4.2 Описание работы составных частей принципиальной схемы

После подключения питания происходит инициализация портов микроконтроллера и его периферии. При нажатии кнопки SW1 он входит в режим ожидания ввода входных данных с внешнего компьютера посредством подключения к разъему XS3. После ввода пользователем ключа шифрования, имени сети Wi-Fi и пароля для подключения, контроллер подключается к серверу посредством микроконтроллера DD2 и инициализирует переменные для шифрования данных. В случае успешного подключения к серверу происходит зажигание светодиода HL1.

Далее, согласно программе, реализованной на управляющем микроконтроллере DD1, устройство осуществляет постоянный опрос данных по шине SPI и сети Wi-Fi. Как только хотя бы по одной из них начинает осуществляться передача данных, контроллер осуществляет шифрование или расшифровку (в зависимости от направления передачи данных) данных с использованием загруженной на микроконтроллер библиотеки шифрования. Происходит индикация передачи данных по шине SPI путем зажигания светодиода HL2.

При помощи нажатия кнопки SW2 происходит сброс микроконтроллера к функции инициализации. Изменять ключ шифрования и данные сервера Wi-Fi можно и без сброса контроллера, что было описано выше.

Синхропосылка генерируется на стороне сервера и отправляется по беспроводной передаче данных на коммуникационный контроллер.

Регулятор напряжения DA1 обеспечивает питание микросхемы DD2.

4.3 Светодиоды

В качестве светодиодной индикации HL1,2 используется два светодиода зеленых [18]. На рисунке 4.2 изображен внешний их вид.

Рисунок 4.2 – Внешний вид светодиодов HL1,2

Технические параметры светодиодов:

- цвет свечения – зеленый;
- длина волны – 567 нм;
- минимальная сила света – 7 мКд;
- при токе – 20 мА;
- видимый телесный угол - 120°;
- максимальное прямое напряжение – 2 В;
- типоразмер – 2013;
- посадочное место – SMD0805;
- вес – 0,1 г.

4.4 Wi-Fi модуль

Чип ESP8266EX компании Espressif [19] — высокоинтегрированное Wi-Fi SoC решение, удовлетворяющее запросы индустрии Интернета вещей в низком энергопотреблении, компактности и надежности.

Имея полноценный Wi-Fi и сетевой стэк, чип ESP8266EX может как выполнять приложения самостоятельно, так и работать под управлением внешнего микроконтроллера. Работая самостоятельно, ESP8266EX выполняет приложение, загружая его из внешней флэш-памяти. Встроенный высокоскоростной кэш повышает производительность системы и позволяет эффективно использовать оперативную память. Работая под управлением внешнего микроконтроллера, ESP8266EX может выступать в роли Wi-Fi адаптера, передавая данные через SPI, SDIO, I2C или UART интерфейсы.

ESP8266EX содержит антенный переключатель, согласующий трансформатор (балун), усилитель мощности, малошумящий усилитель,

фильтры, модули управления питанием. Компактная конструкция и высокая степень интеграции позволяют минимизировать размер печатной платы и число внешних компонентов.

На рисунке 4.3 изображен микроконтроллер ESP8266EX.

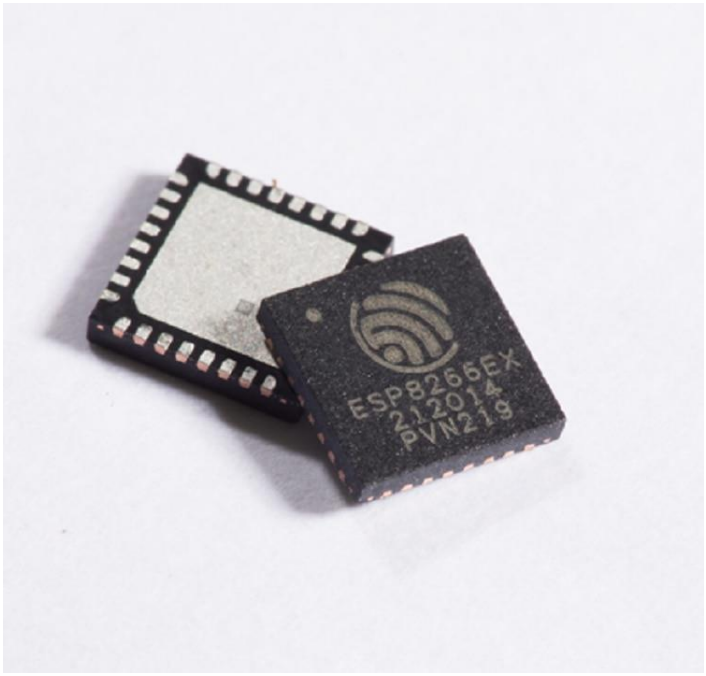


Рисунок 4.3 – Внешний вид микроконтроллера ESP8266EX

Основные технические параметры:

- сертификаты – FCC/CE/TELEC/SRRC;
- протоколы Wi-Fi – 802.11 b/g/n;
- диапазон частот – 2,4-2,5 ГГц (2400-2483,5 М);
- мощность передачи сигнала - +20 дБм для 802.11 b, +17 дБм для 802.11 g, +14 дБм для 802.11 n;
- чувствительность приемника – -91 дБм (11 Мбит/с) для 802.11 b, -75 дБм (54 Мбит/с) для 802.11 g, -72 дБм (MCS7) для 802.11 n;
- тип антенны – антенна на печатной плате, внешняя антенна, керамическая чип-антенна;
- центральный процессор – 32-битный микропроцессор Tensilica Lx106;
- периферийные интерфейсы – UART/SDIO/SPI/I2C/I2S/IR Remote Control, GPIO/ADC/PWM;
- рабочее напряжение – 3,0-3,6 В;
- рабочий ток – 80 мА;
- диапазон рабочих температур – -40..125°C;

- рабочая температура окружающей среды – нормальная температура;
- размер корпуса – 5x5 мм;
- режимы Wi-Fi – клиент (station), точка доступа (SoftAP), SoftAP+station;
- безопасность – WPA/WPA2;
- шифрование – WEP/TKIP/AES;
- сетевые протоколы – IPv4, TCP/UDP/HTTP/FTP;
- конфигурация пользователя – управление с помощью AT команд, облачный сервер, приложения для Android/iOS.

4.5 Выбор микроконтроллера

В качестве микроконтроллера DD1 выбран ATmega328P фирмы ATMEL. Он представляет собой 8-разрядный высокопроизводительный микроконтроллер с малым потреблением и имеет прогрессивную RISC архитектуру. Ниже перечислены его технические параметры [20]:

- 130 высокопроизводительных команд, большинство команд выполняется за один тактовый цикл;
- 32 8-разрядных рабочих регистра общего назначения;
- полностью статическая работа;
- производительность 16 MIPS при тактовой частоте 16 МГц;
- 8 Кбайт внутрисистемно программируемой Flash памяти (In-System Self-Programmable Flash);
- 512 байт EEPROM;
- 1 Кбайт встроенной SRAM;
- два 8-разрядных таймера/счетчика с отдельным предварительным делителем;
- один 16-разрядный таймер/счетчик с отдельным предварительным делителем и режимами захвата и сравнения;
- три канала PWM;
- 8-канальный аналого-цифровой преобразователь;
- программируемый последовательный USART;
- последовательный интерфейс SPI (ведущий/ведомый).

Выводы I/O и корпуса:

- 23 программируемые линии ввода/вывода;
- 28-выводной корпус PDIP, 32-выводной корпус TQFP и 32-выводной корпус MLF.

Рабочие напряжения:

- 2,7 - 5,5 В (ATmega328L);
- 4,5 - 5,5 В (ATmega328);
- рабочая частота: 0-8 МГц (ATmega328L), 0-16 МГц (ATmega328).

Расположение выводов ATMega328 в корпусе DIP28 и TQFP показан на рисунках 4.4, 4.5.

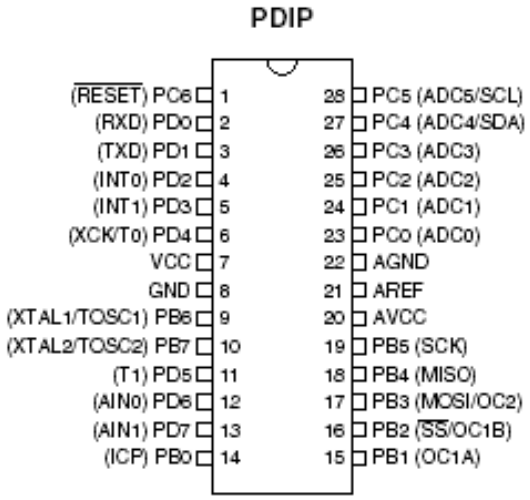


Рисунок 4.4 – Расположение выводов ATmega328 (корпус DIP28)

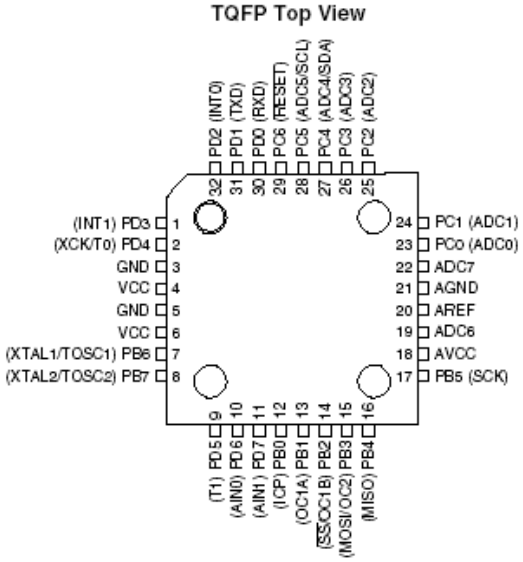


Рисунок 4.5 – Расположение выводов ATmega328 (корпус TQFP)

В таблице 4.1 приведено описание выводов микроконтроллера.

Таблица 4.1 – Описание выводов микроконтроллера ATmega328

№	Название	Тип	Описание
1	2	3	4
Питание:			
7	VCC	Вход	напряжение питания от +4.5 до +5.5 В
8,22	GND	Вход	общий (земля)
20	AVcc	Вход	напряжение питания + 5 В для модуля АЦП
21	ARef	Вход	вход опорного напряжения для АЦП
Порт B:			
14	PB0	Вход/Выход	цифровой порт PB0
14	ICP1	Вход	захват входа 1
15	PB1	Вход/Выход	цифровой порт PB1
15	OC1A	Выход	выход сравнения/ШИМ 1A
16	PB2	Вход/Выход	цифровой порт PB2
16	OC1B	Выход	выход сравнения/ШИМ 1B
16	SS	Вход	вход Slave для SPI
17	PB3	Вход/Выход	цифровой порт PB3
17	OC2	Выход	выход сравнения/ШИМ 2
17	MOSI	Вход/Выход	вход данных в режиме Slave для SPI и ISP / выход данных в режиме Master для SPI и ISP
18	PB4	Вход/Выход	цифровой порт PB4
18	MISO	Вход/Выход	вход данных в режиме Master для SPI и ISP / выход данных в режиме Slave для SPI и ISP
19	PB5	Вход/Выход	цифровой порт PB5
19	SCK	Вход/Выход	тактовый вход в режиме Slave для SPI и ISP / тактовый выход в режиме Master для SPI и ISP
9	PB6	Вход/Выход	цифровой порт PB6 при работе от встроенного генератора
9	XTAL1	Вход	тактовый вход, кварцевый или керамический резонатор
9	TOSC1	Вход	не используется при работе от внешнего генератора
10	PB7	Вход/Выход	цифровой порт PB7 при работе от встроенного генератора
10	XTAL2	Вход	для подключения кварцевого или керамического резонатора
10	TOSC2	Выход	тактовый выход при работе от встроенного генератора
Порт C:			
23	PC0	Вход/Выход	цифровой порт PC0
23	ADC0	Вход	аналоговый вход канал 0
24	PC1	Вход/Выход	цифровой порт PC1
24	ADC1	Вход	аналоговый вход канал 1
25	PC2	Вход/Выход	цифровой порт PC2
25	ADC2	Вход	аналоговый вход канал 2
26	PC3	Вход/Выход	цифровой порт PC3
26	ADC3	Вход	аналоговый вход канал 3
27	PC4	Вход/Выход	цифровой порт PC4
27	ADC4	Вход	аналоговый вход канал 4
27	SDA	Вход/Выход	канал данных для 2-проводного последовательного интрефеиса

Продолжение таблицы 4.1

1	2	3	4
28	PC5	Вход/Выход	цифровой порт PC5
28	ADC5	Вход	аналоговый вход канал 5
28	SCL	Выход	тактовый выход для 2-проводного последовательного интерфейса
1	PC6	Вход/Выход	цифровой порт PC6
1	RESET	Вход	внешний сброс
Порт D:			
2	PD0	Вход/Выход	цифровой порт PD0
2	RxD	Вход	вход приемника USART
3	PD1	Вход/Выход	цифровой порт PD1
3	TxD	Выход	выход передатчика USART
4	PD2	Вход/Выход	цифровой порт PD2
4	INT0	Вход	внешнее прерывание канал 0
5	PD3	Вход/Выход	цифровой порт PD3
5	INT1	Вход	внешнее прерывание канал 1
6	PD4	Вход/Выход	цифровой порт PD4
6	XCK	Вход/Выход	внешний такт для USART
6	T0	Вход	внешний вход Timer 0
11	PD5	Вход/Выход	цифровой порт PD5
11	T1	Вход	внешний вход Timer 1
12	PD6	Вход/Выход	цифровой порт PD6
12	AIN0	Вход	вход аналогового компаратора канал 0
13	PD7	Вход/Выход	цифровой порт PD7
13	AIN1	Вход	вход аналогового компаратора канал 1

4.6 Выбор элементов обвязки

Для питания Wi-Fi-модуля необходимо стабилизированное напряжение 3,3 В. Для этих целей служит регулятор напряжения на DA1 (LF33CV). LF33CV – LDO-регулятор напряжения 3,3 В [21].

Основные технические характеристики:

- корпус – TO220AB;
- тип регулятора – LDO;
- минимальное входное напряжение – 4,3 В;
- максимальное входное напряжение – 16 В;
- максимальный выходной ток – 500 мА;
- полярность – положительная;
- тип выхода – фиксированный;
- количество выходов – 1;
- выходное напряжение – 3,3 В;
- падение напряжения при выходном токе 500 мА – 0,7 В;
- рабочая температура – -40..125°C;

- вес – 2,5 г.

На рисунке 4.6 изображен корпус регулятора напряжения.

Рисунок 4.6 – Внешний вид регулятора напряжения LF33CV

Кварцевые резонаторы для генераторов таковых импульсов использованы HC-49S 16 МГц (ZQ1) и 24 МГц (ZQ2) [22].

Кварцевые резонаторы предназначены для использования в аналогово-цифровых цепях для стабилизации и выделения электрических колебаний определённой частоты или полосы частот.

Принцип работы: в широкой полосе частот сопротивление прибора имеет ёмкостной характер и только на некоторых (рабочих) частотах имеет широко выраженный резонанс (уменьшение сопротивления).

Кварцевый резонатор имеет лучшие характеристики, чем другие приборы для стабилизации частоты (колебательные контуры, пьезокерамические резонаторы): такие как стабильность по частоте (уход частоты) и температуре (изменение частоты резонанса в зависимости от температуры окружающей среды).

Избирательный, ярко выраженный резонансный характер сопротивления этих компонентов определяет основные области применения кварцевых резонаторов - высокостабильные генераторы тактовых сигналов и опорных частот, цепи частотной селекции, синтезаторы частоты и т.д

Технические параметры:

- резонансная частота – 16 или 24 МГц;
- точность настройки – 50 dF/Fx10-6;
- температурный коэффициент – 50 Ктх10-6;
- нагрузочная емкость – 32 пФ;
- рабочая температура – -20..70°C;
- корпус – HC-49S;

- длина корпуса – 11,05 мм;
- ширина корпуса – 4,65 мм;
- вес – 1 г.

Внешний вид кварцевого резонатора изображен на рисунке 4.7.

Рисунок 4.6 – Внешний вид кварцевого резонатора HC-49S

Тактовые кнопки SW1,2 в данном дипломном проекте использованы следующие: KLS7-TS6601-4.3-180 [23].

Технические параметры:

- тип – прямая;
- способ монтажа – в отверстия на плату;
- рабочее напряжение – 12 В;
- рабочий ток – 50 мА;
- высота – 4,3 мм;
- типоразмер – 6x6;
- вес – 0,24 г.

Тактовая кнопка изображена на рисунке 4.7.

Рисунок 4.6 – Внешний вид тактовой кнопки KLS7-TS6601-4.3-180

Разъемы XS1,3 представлены клеммниками разъемными угловыми 3,81 мм на 4 контакта 15EDGRC-3.81-04 [24].

Внешний вид данных разъемов изображен на рисунке 4.7.

Рисунок 4.7 – Внешний вид клеммника разъемного углового 3,81 мм на 4 контакта 15EDGRC-3.81-04

Разъем XS2, соответственно, - клеммником разъемным угловым 3,5 мм на 2 контакта 15EDGRC-3.5-02 [25].

Внешний вид данного разъема изображен на рисунке 4.8.

Рисунок 4.8 – Внешний вид клеммника разъемного углового 3,5 мм на 2 контакта 15EDGRC-3.5-02

Технические параметры:

- совместимость – 15edgk;
- серия – 15edgrc;
- функциональное назначение – клеммник разъемный;
- шаг контактов – 3,81 или 3,5 мм соответственно;
- рабочий ток – 8 А;
- количество контактов – 4 или 2 соответсвенно;

- тип исполнения – угловой;
- рабочее напряжение – 300 В;
- способ монтажа – пайка в отверстия платы;
- вес – 0,88 или 0,49 г соответсвенно.

Резисторы и конденсаторы подобраны по требованиям технической документации к используемым микросхемам, а также с учетом соблюдения технического задания к дипломному проекту.

Компания Murata [26] специализируется на выпуске электронных компонентов из высококачественных керамических материалов (оксид титана, титанат бария и др.). За 60-летнюю историю компания заняла лидирующую позицию по выпуску керамических компонентов.

Серия GRM - безвыводные керамические неполярные конденсаторы общего применения. Имеют превоходные импульсные харатеристки и малый уровень собственных шумов благодаря низкому импедансу на высоких частотах. Конденсаторы серии GRM выпускаются с различными типами диэлектриков - тип диэлектрика определяет ТКЕ данного конденсатора. Рабочее напряжение: 6.3, 10, 16, 25, 50, 100, 200 и 630 В. Диапазон возможных емкостей: 0.3 пФ – 100 мкФ. Размерный ряд: от 0201 до 2220.

Пример конденсаторов данной компании изображен на рисунке 4.9.

Рисунок 4.9 – Внешний вид конденсатора керамического smd 0.1 мкФ X7R 50В 10% 0805, GRM21BR71H104K**

Конденсаторы С1, С4, С7 реализованы конденсаторами керамическими Murata GRM21BR71H104K.

Конденсаторы C2, C3, C9, C10 являются конденсаторами керамическими Murata GRM1555C1ER22BA01D.

Конденсатор C5 является конденсатором керамическим Murata GRM1555C1H5R6C.

Конденсатор C6 является конденсатором керамическим Murata GRM21BR61A106K.

Конденсатор C8 является конденсатором керамическим Murata GRM155R61C105KA12D.

Технические характеристики:

- корпус – SMD 0805;
- тип – grm21;
- рабочее напряжение – 50 В;
- допуск номинала – 10 %;
- температурный коэффициент емкости – x7r;
- рабочая температура – -55..125°C;
- вес – 0,04 г.

Резисторы компании Yageo [27] сконструированы в высококачественном керамическом корпусе. Внутренние металлические электроды добавлены на каждом конце, чтобы обеспечить контакт с толстопленочным резистивным элементом. В состав резистивного элемента входит благородный металл, залитый стеклом и закрытый вторым слоем стекла для защиты от воздействия окружающей среды. Резистор подгоняется лазером до номинального значения сопротивления. Резистор покрыт защитным эпоксидным покрытием, и, наконец, добавлены две внешние клеммы (матовое олово с никелевым барьером).

Пример резисторов данной компании изображен на рисунке 4.10.

Рисунок 4.10 – Внешний вид резистора 0.062Вт 0402 220 Ом, 1%

Резисторы R1, R2, R6, R7, R8 являются резисторами Yageo SMD0402 220 Ом 0,062 Вт.

Резисторы R3, R4 являются резисторами Yageo SMD0805 4,7 кОм 0,125 Вт.

Резистор R5 является резистором Yageo SMD0402 330 Ом 0,062 Вт.

Резистор R8 является резистором Yageo SMD0805 12 кОм 0,125 Вт.

Резистор R10 является резистором Yageo SMD0402 2,2 кОм 0,062 Вт.

Резистор R11 является резистором Yageo SMD0805 1 кОм 0,125 Вт.

Технические характеристики:

- монтаж – SMD 0402, 0805;
- номинальное сопротивление – 220, 330, 1000, 2200, 4700, 12000 Ом;
- точность – 1 %;
- номинальная мощность 0,062, 0,125 Вт;
- максимальное рабочее напряжение – 50 В;
- рабочая температура – -55..155°C;
- вес – 0,01 г.

5 РАЗРАБОТКА АЛГОРИТМА РАБОТЫ УСТРОЙСТВА

5.1 Проектирование алгоритма работы системы

Процесс разработки программного обеспечения для микроконтроллера – это наиболее сложная часть конструирования, именно это вызывает наибольшие сложности. Совпадение целей и интересов у автора-разработчика и человека, решившего повторить конструкцию, происходит редко, и это порождает множество вопросов.

При написании программ для микроконтроллеров все большую популярность приобретает язык Си. При его использовании сокращается время на разработку, что особенно заметно при написании больших программ, обеспечивается их переносимость на другие платформы. Недостатком языка Си по сравнению с языком Ассемблер является больший объем кода и, как следствие, более низкая скорость работы. Однако благодаря тому, что в архитектуре AVR изначально заложено эффективное декодирование и исполнение инструкций, генерируемых компиляторами, после компиляции Си-программ получается высокопроизводительный код.

Разработка программного обеспечения для микроконтроллеров AVR мало отличается от разработки любых иных программ, разве что от программиста требуется чуть более глубокие знания электроники – хотя бы на уровне понимания действия логических элементов и триггеров. Разумеется, чем лучше программист владеет электроникой, тем более качественные программы он сможет создать. Тесная связь «софта и железа» по сути требует, чтобы программист был электронщиком или же электронщик был программистом. В настоящее время обе ситуации имеют место. Существует три более-менее устоявшихся подхода к разработке микроконтроллерных устройств:

- от схемы к программе;
- от программы к схеме;
- смешанный.

Далее описан процесс разработки и отладки программы на языке Си для дипломного проекта. В качестве среды программирования использована программа AVR Studio.

Алгоритм – точный набор инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за определенное время.

Вообще, написать программу можно в любом текстовом редакторе, так же как вы бы написали письмо другу, например. После этого, текст надо скомпилировать (иногда говорят - ассемблировать) т.е. перевести в форму, понятную процессору. Суть работы компилятора в переводе письменных символов понятных для человека в машинный код (в код нулей и единиц) и создание нового файла с расширением .hex

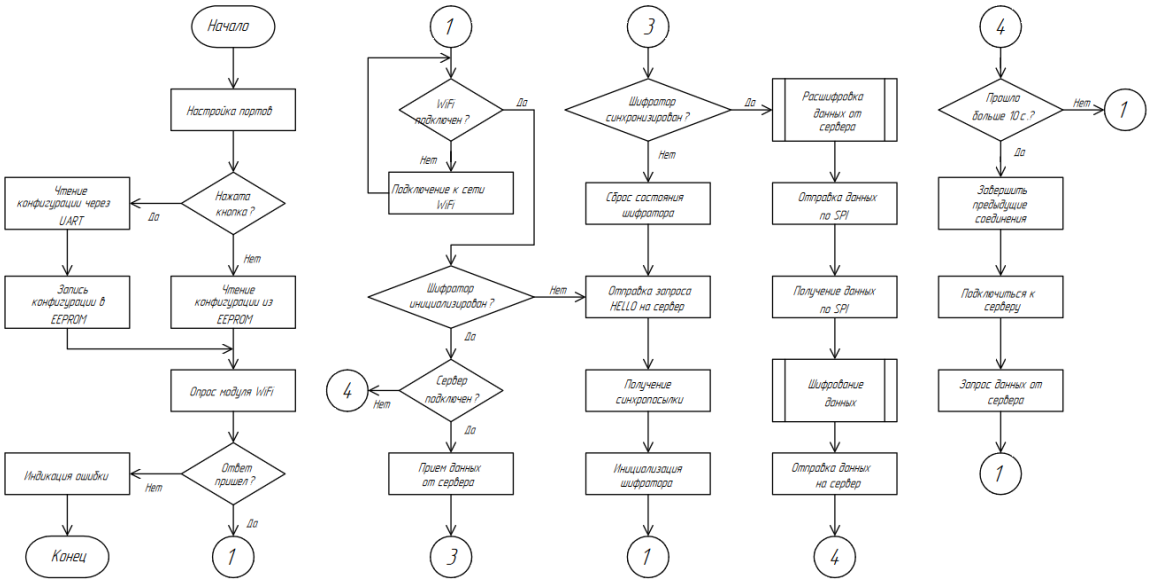


Рисунок 5.1 – Алгоритм работы устройства

Алгоритм функционирования устройства определяется управляющей программой МК.

Устройство функционирует следующим образом:

- включение;
- инициализация микроконтроллера;
- инициализация переменных;
- подключение к серверу Wi-Fi;
- инициализация шифратора по ключу и синхропосылке;
- расшифровка данных от сервера;
- отправка и полчение данных по SPI;
- шифрование данных;
- отправка данных на сервер;
- индикация;
- условие выхода из цикла – отключение или перенастройка системы.

После инициализации устройство переходит в режим запроса о дальнейшем состоянии. Алгоритм главного цикла работы представлен на рисунке 5.1.

Инициализация – создание, активация, подготовка к работе, определение параметров. Приведение программы или устройства в состояние готовности к использованию.

Представленные уровневые структуры алгоритма легли в основу разработанной программы системы команд микроконтроллера ATmega328.

Назначение языка программирования – предоставить программисту средства для изложения алгоритма решения какой-либо задачи в форме, воспринимаемой компилятором (специальной программой, служащей для «перевода» с одного языка на другой, понятный конкретной аппаратной платформе, т.е. процессору или микроконтроллеру).

5.2 Разработка программного обеспечения

AVR Studio — основанная на Visual Studio бесплатная проприетарная интегрированная среда разработки (IDE) для разработки приложений для 8- и 32-битных микроконтроллеров семейства AVR и 32-битных микроконтроллеров семейства ARM от компании Atmel, работающая в операционных системах Windows. AVR Studio содержит компилятор GNU C/C++ и эмулятор, позволяющий отладить выполнение программы без загрузки в микроконтроллер.

Ранее среда разработки носила название AVR Studio, но начиная с версии 4.0, вышедшей в 2012 году, в неё была добавлена поддержка разработки для микроконтроллеров архитектуры ARM, также выпускаемых фирмой Atmel, и среда разработки получила новое название Atmel Studio.

Atmel Studio содержит в себе менеджер проектов, редактор исходного кода, инструменты виртуальной симуляции и внутрисхемной отладки, позволяет писать программы на ассемблере или на C/C++.

Внешний вид окна программы (IDE) AVR Studio 5 представлен на рисунке 5.2.

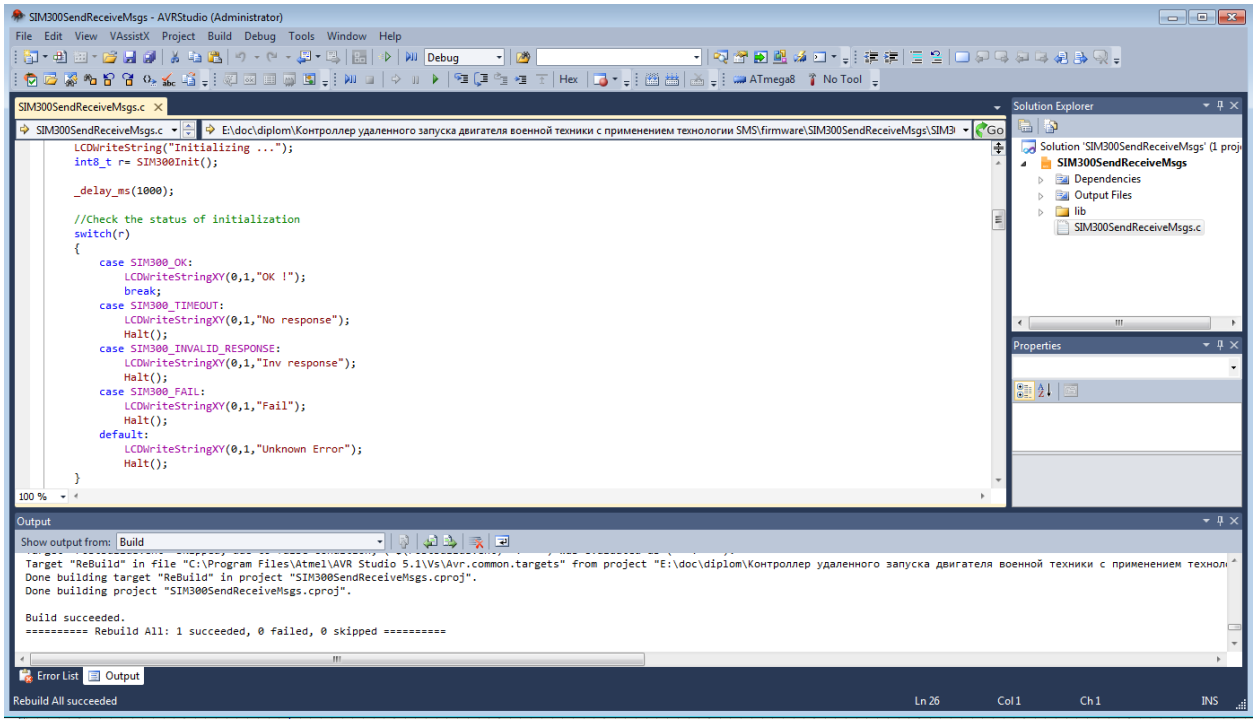


Рисунок 5.2 – Интерфейс ИСР AVR Studio 5

Листинг программы приведён в приложении А.

5.3 Протокол передачи данных

Для обмена данными между разрабатываемым коммуникационным контроллером и сервером был реализован протокол передачи данных. Протокол включает в себя следующие этапы:

- приветствие;
- установка параметров шифрования;
- передача полезных данных в шифрованном виде.

Обмен данными осуществляется при помощи информационных посылок, имеющих формат, описанный на рисунке 5.3:

16 bit **cid**	8 bit **cmd**	8 bit **len**	16 bit **counter**	**data**

Рисунок 5.3 – Формат информационной посылки

Описание полей:

1 cid (2 байта): уникальный идентификатор клиента, назначется сервером в ответ на приветствие от клиента, в котором cid=0 и cmd=0.

2 cmd (1 байт): идентификатор команды. Может принимать следующие значения:

- CMD_HELLO (0) – приветствие, отправляется клиентом на сервер при первом подключении для получения cid и параметров шифрования. Параметры cid, len, counter при этом равны 0, поле data отсутсвует;
- CMD_SETCID (1) – установить cid, отправляется сервером клиенту и для устновки уникального идентификатора клиента и синхропосылки для шифратора в режиме счетчика. При этом поле len = 16, counter = 0, data содержит синхропосылку (16 байт). При получении данной команды клиент обнуляет счетчики принятых и переданных данных, утсанавливает новый cid, и переинициализирует шифратор и дешифратор с использованием полученной синхропосылки;
- CMD_DATA (2) – передача полезных данных в шифрованом виде.

3 len (1 байт): длина полезных данных.

4 counter (2 байта): счетчик переданых полезных данных. Необходим для синхронизации шифраторов клиента и сервера. Клиент и сервер ведут независимый учет принятых и переданных данных. Если количество принятых клиентом либо сервером данных отличается от значения counter в принятой посылке, то счетчики и состояние шифратора сбрасывается, клиент заново отправляет серверу команду CMD_HELLO, а сервер клиенту CMD_SETCID.

5 data (произвольная длина, до 256 байт): зашифрованные полезные данные.

5.4 Шифрование данных

В некоторых случаях данные, собираемые датчиками системы умный дом могут содержать чувтсвительную информацию. Например, по ним можно определить, находятся ли дома люди. Поэтому передаваемая информация должна быть надежно защищена.

Для защиты данных, которыми обменивается устройство был выбран стандарт BelT. BelT — государственный стандарт симметричного шифрования и контроля целостности Республики Беларусь. Полное название стандарта — СТБ 34.101.31-2007 «Информационные технологии и безопасность. Криптографические алгоритмы шифрования и контроля целостности» [8]. Принят в качестве предварительного стандарта в 2007 году. Введен в действие в качестве окончательного стандарта в 2011 году.

BelT — блочный шифр с 256-битным ключом и 8 циклами криптопреобразований, оперирующий с 128-битными словами. Криптографические алгоритмы стандарта построены на основе базовых режимов шифрования блоков данных. Все алгоритмы стандарта делятся на 8 групп:

- алгоритмы шифрования в режиме простой замены;
- алгоритмы шифрования в режиме сцепления блоков;
- алгоритмы шифрования в режиме гаммирования с обратной связью;
- алгоритмы шифрования в режиме счётчика;
- алгоритм выработки имитовставки;
- алгоритмы одновременного шифрования и имитозащиты данных;
- алгоритмы одновременного шифрования и имитозащиты ключей;
- алгоритм хэширования.

В данной работе нас интересуют первые четыре группы, которые предназначены для обеспечения безопасного обмена сообщениями. Каждая группа включает алгоритм зашифрования и алгоритм расшифрования на секретном ключе. Стороны, располагающие общим ключом, могут организовать обмен сообщениями путём их зашифрования перед отправкой и расшифрования после получения. В режимах простой замены и сцепления блоков шифруются сообщения, которые содержат хотя бы один блок, а в режимах гаммирования с обратной связью и счётчика — сообщения произвольной длины.

Поскольку наше устройтво обменивается небольшими посылками данных, в качестве алгоритма шифрования был выбран BelT в режиме счетчика. Этот режим позволяет сократить объем передаваемых данных, т. к. работает с посылками любой длины, а также сократить объем кода, т. к. для расшифрования может быть использована та же функция, что и для шифрования данных.

При шифровании в режимах сцепления блоков, гаммирования с обратной связью и счетчика используется синхропосылка $S \in \{0, 1\}_{128}$,

которая обеспечивает уникальность криптографических преобразований на фиксированном ключе.

Список обозначений:

- $\{0,1\}^n$ – множество всех слов длины n в алфавите $\{0,1\}$;
- $\{0,1\}^*$ – множество всех слов конечной длины в алфавите $\{0,1\}$ (включая пустое слово длины 0);
- $|u|$ – длина слова $u \in \{0,1\}^*$;
- $\{0,1\}^{n*}$ – множество всех слов из $\{0,1\}^*$, длина которых кратна n;
- α^n – слово длины n из одинаковых символов $\alpha \in \{0,1\}$;
- $L_m(u)$ – слово из первых m символов слова u, $m \leq |u|$;
- $u \parallel v$ – конкатенация $u_1u_2 \dots u_nv_1v_2 \dots v_m$ слов $u = u_1u_2 \dots u_n$ $v = v_1v_2 \dots v_m$;
- $01234 \dots_{16}$ – представление $u \in \{0,1\}^{4*}$ шестнадцатеричным словом, при котором последовательным четырем символам u соответсвует один шестнадцатеричный символ (например, $10100010 = A2_{16}$);
- $U \bmod m$ – для целого U и натурального m остаток от деления U на m;
- $u \oplus v$ – для $u = u_1u_2 \dots u_n \in \{0,1\}^n$ и $v = v_1v_2 \dots v_n \in \{0,1\}^n$ слово $w = w_1w_2 \dots w_n \in \{0,1\}^n$ из символов $w_i = (u_i + v_i) \bmod 2$;
- $\bar{u}$ – для $u = u_1u_2 \dots u_n \in \{0,1\}^8$ число $2^7u_1 + 2^6u_2 + \cdots + u_8$, а для $u = u_1 \parallel u_2 \parallel \cdots \parallel u_n, u_i \in \{0,1\}^8$, число $\bar{u}_1 + 2^8\bar{u}_2 + \cdots + 2^{8(n-1)}\bar{u}_n$;
- $\langle U \rangle_{8n}$ – для целого U слово $u \in \{0,1\}^{8n}$ такое, что $\bar{u} = U \bmod 2^{8n}$;
- $u \boxplus v$ – для $u, v \in \{0,1\}^{8n}$ слово $\langle \bar{u} + \bar{v} \rangle_{8n}$;
- $u \boxminus v$ – для $u, v \in \{0,1\}^{8n}$ слово $w \in \{0,1\}^{8n}$ такое, что $u = v \boxplus w$;
- $a \leftarrow u$ – присвоение переменной a значения u;
- $F_\theta(X)$ – результат зашифрования блока $X \in \{0,1\}^{128}$ на ключе $\theta \in \{0,1\}^{256}$.

Входными данными алгоритмов зашифрования и расшифрования являются сообщение $X \in \{0,1\}^*$, ключ $\theta \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$.

Выходными данными являютсмя слово $Y \in \{0,1\}^{|X|}$ – результат зашифрования либо расшифрования X на ключе θ при использовании синхропосылки S.

Входное сообщение X записывается в виде

$$X = X_1 \parallel X_2 \parallel \cdots \parallel X_n, \quad |X_1| = |X_2| = \cdots = |X_{n-1}| = 128, \quad |X_1| \leq 128.$$

При шифровании словам X_i ставятся в соответствии слова $Y_i \in \{0,1\}^{|X_i|}$, из которых затем составляется Y.

Используется переменная s со значениями из $\{0,1\}^{128}$.

Зашифрование сообщения X на ключе θ при использовании синхропосылки S состоит в выполнении следующих шагов:

1 Установить $s \leftarrow F_\theta(S)$.

2 Для $i = 1, 2, \ldots, n$ выполнить $s \leftarrow s \boxplus \langle 1 \rangle_{128}$, $Y_i \leftarrow X_i \oplus L_{|X_i|}(F_\theta(S))$.

3 Установить $Y \leftarrow Y_i \| Y_2 \| \cdots \| Y_n$.

4 Возвратить Y.

Расшифрование сообщения X на ключе θ при использовании синхропосылки S состоит из выполнения тех же шагов, что и при зашифровании.

6 РАЗРАБОТКА ПЕЧАТНО УЗЛА УСТРОЙСТВА

Рассмотрим технологию разработки печатной платы в САПР Altium Designer. Начальным этапом разработки любого радиоэлектронного устройства является описание его работы на некотором уровне абстракции, в данном случае – схемы электрической принципиальной.

Формирование новой электрической схемы в Altium Designer начинается с создания нового файла проекта и листа схемы командами File>New>Project>PCB Project и File>New>Schematic. Для сохранения проекта выполняется File>Save Project As, для схемы – File>Save.

Настройки во всех редакторах Altium Designer можно разделить на глобальные – относящиеся ко всем документам и локальные – относящиеся только к текущему документу. Настройки текущего документа устанавливаются на вкладке Design>Document Options. Она содержит три вкладки: Sheet Options, на которой задаются настройки листа (размер листа, шаги сеток, ориентация листа и т.д.), Parameters и Units, в которой задаются единицы измерения. Глобальные настройки находятся в меню: DXP>Preferences>Schematic.

Перед формированием схемы необходимо подключить библиотеки, в которых находятся компоненты схемы. Для поиска компонентов и подключения библиотек служит панель управления библиотеками Libraries, которая вызывается выбором вкладки в правой части окна Design Explorer. Если вкладка отсутствует, то панель можно вызвать через меню View> Workspace Panels>System>Library или через меню вызова панелей System>Libraries, расположенном в правом нижнем углу рабочего окна.

В верхней части окна панели имеются 3 вкладки: Project – библиотеки проекта, Installed – установленные библиотеки, Search Path – поисковая система.

Для поиска компонентов ко всем доступным библиотекам независимо от того, подключены они или нет, служит кнопка Search панели Libraries.

Инструменты формирования электрической схемы сгруппированы в панели инструментов Wiring. Величину шага сетки можно изменить командой View>Grid>Set Snap Grid. После выбора необходимого компонента в библиотеке необходимо воспользоваться кнопкой Place. На рисунке 6.1 изображен редактор схемы электрической принципиальной для наглядности.

Прежде чем зафиксировать компонент на плате, нажатием клавиши Tab следует открыть окно Component Properties, в котором информация о компоненте разбита на группы: Properties – основные свойства компонента, Library Link – ссылка на соответствующий библиотечный элемент, Graphical

– параметры графического изображения, Parameters – атрибуты компонента, Models – модели компонента, Edit Pins – таблица выводов компонента.

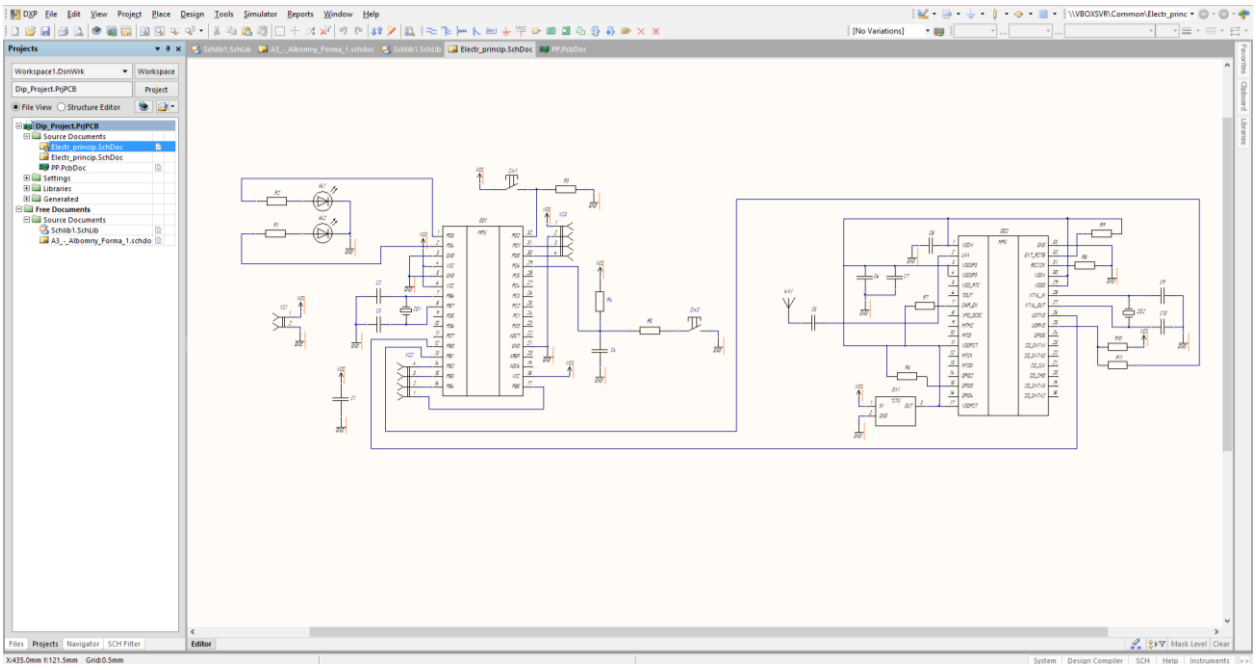


Рисунок 6.1 – Редактор схемы электрической принципиальной в Altium Designer

Соединение элементов электрическими цепями осуществляется командой Place>Wire. В процессе рисования цепей можно клавишами Shift + Space выбирать один из 4 режимов рисования: 90º, 45º, произвольный угол и режим Auto Wire (соединение двух выбранных точек по оптимальному маршруту).

Имя (метка) электрической цепи присваивается командой Place>Net Label.

Расстановка позиционных обозначений компонентов схемы выполняется командой Tools>Annotate Schematic. Выполнение команды приводит к появлению окна Annotate, в котором можно выбрать требуемые параметры расстановки.

В процессе компиляции обнаруживаются нарушения, ошибки в проекте (верификация проекта), создается отчет о корректности проекта, найденные нарушения помечаются на схеме и сопровождаются комментариями об их природе. Результат компиляции – отлаженный файл проекта, готовый к проектированию печатной платы. Процесс компиляции состоит из следующих этапов:

– настройка параметров проекта (схемы), которая заключается в задании правил проверки схемы (ERC – Electrical Rule Check). Командой Project >Project Options создается окно Options for Project, во вкладках которого и задаются правила проверки;

– выполнение компиляцию проекта командой Project>Compile PCB Project. Результаты компиляции будут показаны на панели Compiled, где будут описаны компоненты, цепи, выводы и др. Обнаруженные нарушения будут указаны на панели Messages. Двойным щелчком мыши на строке с ошибкой компиляции можно вызвать панель Compile Errors с подробным описанием ошибки. Двойной щелчок на значке приведет к появлению наглядного изображения этого элемента на схеме, остальная часть схемы будет маскирована;

– отладка схемы. Необходимо добиться, чтобы в списке нарушений в окне Messages не осталось ни одной ошибки (Fatal Error и Error). Рекомендуется отладку выполнять постепенно: исправить ошибку и снова провести компиляцию.

Редактор печатных плат предназначен для создания, редактирования и тестирования печатных плат, генерации файлов для изготовления фотошаблонов.

Для перемещения по чертежу можно использовать браузер Mini Viewer (прямоугольник из пунктирных линий Zoom Box показывает поле просмотра) путем его перетаскивания с помощью левой клавиши мыши (ЛКМ). Щелчком ЛКМ по кнопке Magnifier возможно осуществить перемещение курсора в вид лупы с изменением степени увеличения клавишей Space Bar (три возможных значения).

Панорамирование чертежа возможно выполнять с помощью четырех стрелок клавиатуры. Другой способ выполнять панорамирование – применить инструмент Slider Hand, который активизируется, если нажать и не отпускать правую кнопку мыши (ПКМ), после чего изображение в окне можно передвинуть. Просмотреть чертеж можно с помощью команд меню View.

Объекты, которые может обрабатывать редактор плат, делятся на примитивы и составные объекты. Под обработкой понимаются такие операции, как размещение объектов, выделение, копирование, перемещение, изменение, удаление и др. Разновидности объектов редактора плат: примитивы-графические объекты (линии, дуги, текстовые строки), примитивы-электрические объекты (проводники, контактные площадки, переходные отверстия, области металлизации), составные объекты,

создаваемые пользователем (компоненты, полигоны), составные объекты, создаваемые системой (размеры, координатные метки).

Для идентификации объекта в окне редактора плат достаточно навести на него указатель мыши. Информация появится в виде текстовой строки, а также в строке состояния.

Для выделения объектов применяют два подхода: выделение фокусом и комплексное выделение. Выделение фокусом осуществляется щелчком ЛКМ по объекту, в результате объект становиться активным. После щелчка ЛКМ по прямоугольнику в его углах появятся 5 меток манипуляторов: в углах – метки для изменения размеров объекта, в центре – метка вращения. Выделенный объект можно удалить, нажав на клавиатуре клавишу Del. Удаление можно выполнить и с помощью меню командой Edit>Delete, затем указать удаляемые объекты. Если объекты накладываются друг на друга или расположены близко, то следует выполнить двойной щелчок ЛКМ, после чего на экране появится табличка со списком объектов. Щелчком ЛКМ выбирается нужный объект.

В редакторе плат системы Altium Designer проектируемая конструкция представляется в виде совокупности слоев. Все слои разбиты на следующие группы: Signal Layers, Internal, Mechanical, Mask, Silkscreen и Other Layers.

Signal Layers – сигнальные слои. Их может быть до 32 в многослойной печатной плате. Из них: Top – верхний слой, Mid – внутренние сигнальные слои, Bottom – нижний слой.

Internal Layers – экранные слои. Это могут быть внутренние слои питания и земли, металлизированные полигоны. Отображение форм на экранных слоях инверсное.

Mechanical Layers – механические слои, предназначенные для прорисовки вспомогательных элементов чертежа печатной платы, которые не должны быть на самой плате.

Mask Layers – слои паяных паст и защитных масок. Слои Top Solder и Bottom Solder предназначены для прорисовки масок, используемых при нанесении припоя на верхнюю и нижнюю стороны печатной платы. Слои Top Paste и Bottom Paste предназначены для прорисовки масок, используемых при нанесении паяльной пасты на верхнюю и нижнюю стороны печатной платы.

Silkscreen Layers – слои шелкографии. Слои Top Overlay и Bottom Overlay предназначены для нанесения рисунков и надписей, выполненных методом шелкографии, на верхнюю и нижнюю стороны печатной платы.

Other Layers – дополнительные слои, к которым относятся: Drill Guide – слой сверления. Здесь создается чертеж расположения центров всех отверстий на печатной плате, Multi-Layers – слой для размещения контактных площадок и переходных отверстий многослойных печатных плат, Keep Out – слой для задания областей, где разрешено размещение компонентов и проводников. Области Keep Out могут быть заданы для отдельных слоев. Для этого нужный контур необходимо разместить командой Place>Keep Out.

Активизация слоев осуществляется командой Design>Options> Document Options>Layers>включить переключатели отображения нужных слоев. В результате вкладка с именем слоя появится в нижней части окна редактора плат.

Первый шаг конструкторского проектирования печатной платы – создание заготовки чертежа платы, в которой заданы её границы и набор слоёв. Возможны следующие варианты получения заготовки чертежа

платы: вручную с использованием инструментов редактора плат; с помощью специализированного мастера PCB Wizard; с помощью мастера PCB Wizard можно в качестве заготовки использовать один из готовых шаблонов плат промышленных стандартов или ранее подготовленных разработчиком шаблонов; импорт из другой САПР.

Предварительно следует установить начало системы координат командой Edit>Origin>Set, по которой начало координат устанавливается в текущую позицию курсора.

Шаг сетки можно изменить в любой момент проектирования командой

Design>Options>Document Options или комбинацией горячих клавиш Ctrl+G. Задать величину шага сетки Electrical Grid можно командой Design>Options Document Options>Options. Включить или выключить электрическую сетку в процессе проектирования можно комбинацией горячих клавиш Shift+E. Сетка Component Grid помогает разработчику ориентироваться при размещении компонентов и ее действие аналогично сетке SnapGrid. Сетки Visible Grid облегчают ориентацию в чертеже.

Контур платы можно задать командой меню Design>Board Shape>Redefine Board, Define Board Cutout.

Крепежные отверстия устанавливаются как обычные контактные площадки командой Place>Pad, затем в свойствах указываются нулевые значения в параметрах формы контактной площадки (Size and Shape) и отключается металлизацию внутри отверстия (Plated). Проектные данные, которые передаются из редактора схем в редактор плат – это список

электрических цепей. При этом извлекается информация о каждом компоненте и параметрах связанности из схемы электрической принципиальной, отыскивается соответствующее компоненту топологическое посадочное место (Footprint) в библиотеке элементов. Посадочное место размещается на чертеже платы с добавлением линий соединений.

Правила проектирования должны быть согласованы с ограничениями используемой технологии производства печатных плат и конструктивными ограничениями компонентов, применяемых в проекте. Настройка и редактирование правил проектирования может производиться вручную (команды меню Design>Rules) или с помощью мастера Rule Wizard. Каждое задаваемое правило имеет область действия (Scope) от всей платы до отдельного объекта.

Чтобы использовать разработанные правила в других последующих проектах, их можно записать в отдельный файл. Для этого в окне PCB Rules and Constraints нажать ПКМ в списке правил и выбрать Export Rules. Далее выбрать нужное правило и нажать кнопку OK.

САПР Altium Designer содержит две программы автоматического размещения компонентов. Для интерактивного размещения используются в основном инструменты меню Tools>Component Placement.

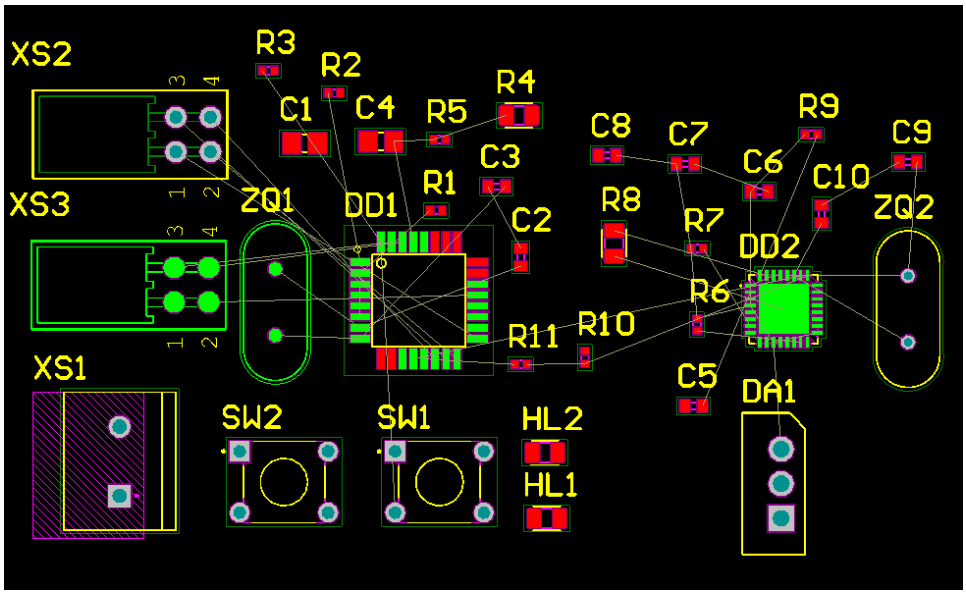


Рисунок 6.2 – Компоненты и связи в Altium Designer

Предварительно следует настроить основные опции размещения (Tools > Preference > Options).

Некоторые компоненты, бывает необходимо зафиксировать на нужном месте. Для этого следует выполнить двойной щелчок мыши по компоненту и в появившемся диалоговом окне Component на вкладке Properties установить флажок Locked.

Основные параметры настройки интерактивной трассировки находятся в окне DXP > Preferences > PCB Editor > Interactive Routing и соответствуют установленным ранее правилам проектирования. Эти установки могут быть изменены в процессе прокладки трассы при нажатии клавиши Tab.

Рисунок 6.3 – Проектирование металлизации отверстий в Altium Designer

В процессе прокладки трасс редактор печатных плат непрерывно контролирует выполнение правил проектирования (on-line DRC) и препятствует их нарушению.

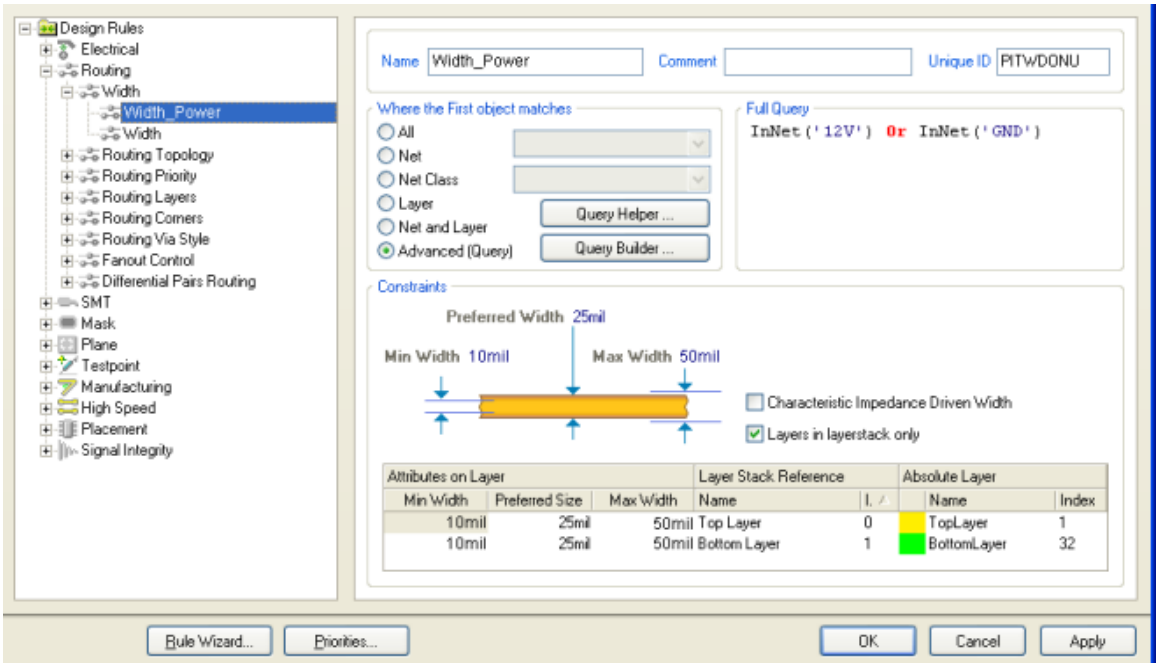


Рисунок 6.4 – Настройка параметров печатных проводников в Altium Designer

Для увеличения числа доступных каналов трассировки количество контактных площадок компонентов, попадающих в узлы сетки Snap Grid, должно быть как можно большим. Проверить, находятся ли компоненты в сетке, можно с помощью команды меню Edit >Select>Off Grid Pads. Привязку всех компонентов к узлам сетки размещения можно выполнить с помощью команды меню Tools>Interactive Placement>Move to Grid, после чего на экране появится диалоговое окно, позволяющее пользователю установить параметры сетки.

Все неразведенные проводники представляются в слое Connection в виде тонких виртуальных линий связи (маршрутов From-To). Режим интерактивной трассировки включается командой Place>Interactive Routine либо соответствующей пиктограммой на панели инструментов. В результате указатель мыши примет вид креста. После выбора начального контакта трассы указатель примет вид восьмиугольника. Это признак того, что сработала электрическая сетка – произошел захват электрического объекта. После выбора ЛКМ очередной точки трассы система пунктиром прорисовывает предполагаемый следующий сегмент трассы. Прокладка трассы завершается нажатием ПКМ.

Полигонами называют области металлизации неправильной формы, которые могут состоять из одной или нескольких частей, соединенных с заданной цепью.

Область металлизации может быть создана на любом сигнальном слое. Сначала прямыми линиями или дугами задаются границы полигона, который впоследствии автоматически будет залит медью в соответствии с заранее определенными правилами проектирования. Границы заливки можно редактировать в любой момент работы над проектом. После изменения положения отдельных компонентов и проводников на плате можно выполнить повторную заливку полигона. Размещение полигонов выполняется с помощью команды меню Place>Polygon Pour, после чего появится диалоговое окно Polygon Pour, которое позволяет установить нужные параметры полигона. Изменения границ полигона выполняется командой Edit>Change>Polygon Vertices и последующим перемещением его вершин. В диалоговом окне Place > Polygon Pour выбирается вариант заливки полигона, характер соединение полигона с электрической цепью и прочие опции.

Верификация печатной платы заключается в проверке выполнения правил проектирования. Для верификации предназначен программный модуль Design Rule Checker (DRC). Запуск верификации печатной платы

осуществляется нажатием кнопки Run Design Rule Clock в окне Design Rule Checker. В результате объекты, которые содержат обнаруженную ошибку, будут подсвечены соответствующим цветом.

После завершения разработки печатной платы ее чертеж можно экспортировать для оформления по ЕСКД в САПР AutoCAD командой меню File>Export (формат dxf или dwg).

Интерфейс САПР AutoCAD во время разработки сборочного чертежа устройства изображен на рисунке 6.5.

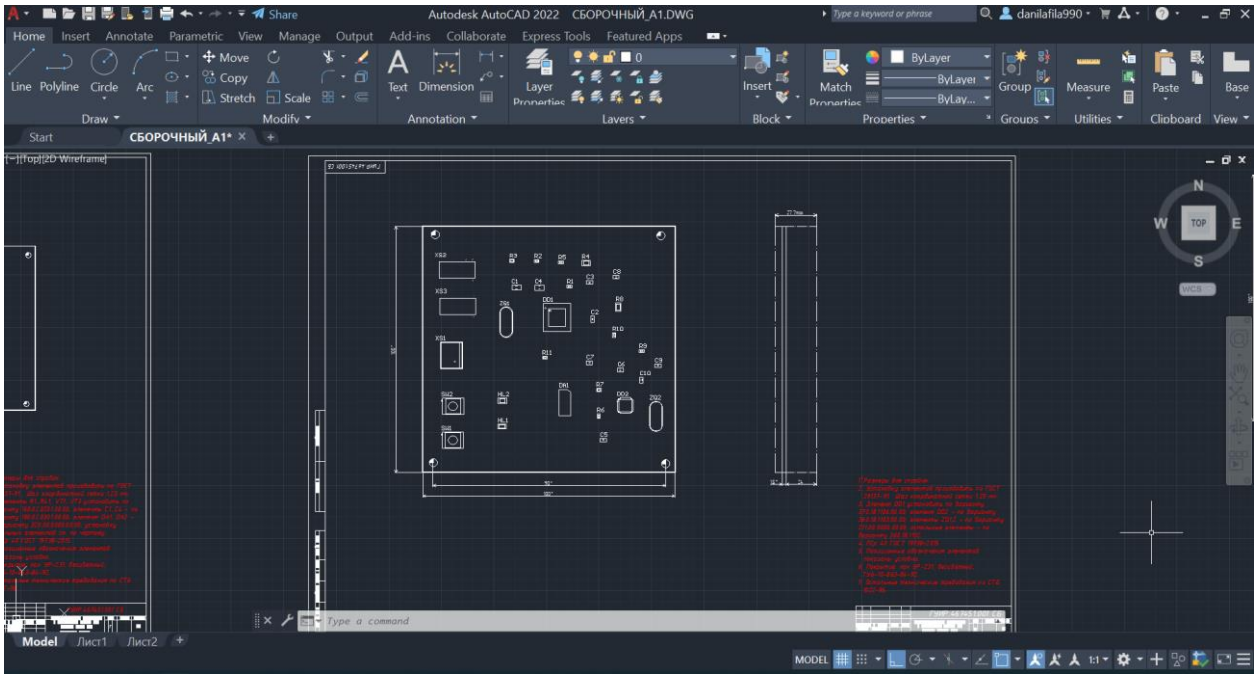


Рисунок 6.5 – Интерфейс САПР AutoCAD

7 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ И ПРОИЗВОДСТВА КОММУНИКАЦИОННОГО КОНТРОЛЛЕРА С ШИФРОВАНИЕМ ДАННЫХ ДЛЯ СИСТЕМЫ «УМНЫЙ ДОМ»

7.1 Характеристика изделия

Разрабатываемый в дипломном проекте коммуникационный контроллер с шифрованием данных для системы «Умный дом» — это устройство управления, которое предназначено для приема данных с устройства системы, их шифрования и передачи на серверную часть системы. Внедрение нового контроллера позволит сократить расходы на производство.

Разрабатываемое устройство должно иметь меньшую стоимость чем аналогичные устройства и не уступать им по функционалу.

В главе ТЭО рассчитывается прогнозируемый экономический эффект от разработки и внедрения в производство устройства за 4 года.

7.2 Расчет затрат на производство изделия

Расчет затрат по статье «Основные и вспомогательные материалы», в которую включается стоимость необходимых для изготовления изделия основных и вспомогательных материалов, осуществляется по формуле

$$Р_м = К_{тр} \cdot \sum_{i=1}^{n} Н_{рi} \cdot Ц_{отпi}, \quad (7.1)$$

где $К_{тр}$ – коэффициент транспортных расходов (1,2); n – номенклатура применяемых материалов; $Н_{рi}$ – норма расхода материала i-го вида на единицу изделия, нат.ед./шт.; $Ц_{отпi}$ – цена за единицу материала i-го вида, р. (в соответствии с действующими на момент проведения расчетов ценами).

Расчет затрат на материалы произведён в таблице 7.1.

Таблица 7.1 – Расчет затрат на материалы

Наименование материала	Единица измерения	Норма расхода	Цена за единицу, р.	Сумма, р.
Стеклотекстолит	кг	0,10	3,00	0,3
Спирт	л	0,02	4,40	0,09
Припой	кг	0,05	70,00	3,5
Лак	л	0,02	40,00	0,8
Клей	кг	0,06	12,00	0,72
Флюс	л	0,06	20,00	1,2
Итого				6,61
Всего затрат с учетом транспортных расходов (по формуле (7.1))				7,94
Примечание – В таблице приведены условные наименования материалов				

Расчет затрат по статье «Покупные комплектующие изделия, полуфабрикаты», в которую включается стоимость необходимых для изготовления изделия комплектующих изделий, осуществляется по формуле

$$\text{Р}_\text{к} = \text{К}_\text{тр} \cdot \sum_{i=1}^{m} N_i \cdot \text{Ц}_{\text{отп}i}, \qquad (7.2)$$

где $\text{К}_\text{тр}$ – коэффициент транспортных расходов (1,2); m – номенклатура применяемых комплектующих; N_i – количество комплектующих i-го вида на единицу изделия, нат.ед./шт.; $\text{Ц}_{\text{отп}i}$ – цена за единицу комплектующего i-го вида, р. (в соответствии с действующими на момент проведения расчетов ценами по данным магазина https://www.chipdip.by/).

Расчет затрат на комплектующие изделия и полуфабрикаты произведен в таблице 7.2.

Таблица 7.2 – Расчет затрат на комплектующие изделия и полуфабрикаты

Наименование комплектующего изделия или полуфабриката	Количество на единицу, шт.	Цена за единицу, р.	Сумма, р.
1	2	3	4
Микроконтроллер ATmega328P-AU	1	10,61	10,61
Микроконтроллер ESP8266EX	1	3,65	3,65
Микросхема LF33CV	1	4,10	4,10
Резонатор кварцевый 16 МГц HC-49S	1	0,83	0,83
Резонатор кварцевый 24 МГц HC-49C	1	0,66	0,66
Конденсатор 0,22 пФ GRM1555C1ER22BA01D	4	0,01	0,04
Конденсатор 0,1 мкФ GRM21BR71H104K**	3	0,15	0,45
Конденсатор 5,6 пФ GRM1555C1H5R6C	1	0,09	0,09
Конденсатор 1 мкФ GRM155R61C105KA12D	1	0,08	0,08
Конденсатор 10 мкФ GRM21BR61A106K	1	0,34	0,34
Разъём 2pin 15EDGRC-3.5-02	1	0,94	0,94
Разъём 4pin 15EDGRC-3.81-04	2	1,15	2,3
Кнопка тактовая KLS7-TS6601-4.3-180	2	0,26	0,52
Резистор 330 Ом 0,062 Вт SMD0402 1%	1	0,03	0,03
Резистор 4,7 кОм 0,125 Вт SMD0805 5%	2	0,04	0,08
Резистор 220 Ом 0,062 Вт SMD0402 1%	5	0,02	0,1

Продолжение таблицы 7.2

1	2	3	4
Резистор 12 кОм 0,125 Вт SMD0805 1%	1	0,03	0,03
Резистор 2,2 кОм 0,062 Вт SMD0402 1%	1	0,02	0,02
Резистор 1 кОм 0,125 Вт SMD0805 5%	1	0,03	0,03
Светодиод зеленый SMD0805 567 нм	2	0,42	0,84
Итого			25,74
Всего с учётом транспортных расходов (по формуле (7.2))			30,89

Формирование отпускной цены нового изделия проведено в соответствии с методикой, представленной в таблице 7.3.

Таблица 7.3 – Методика формирования отпускной цены нового изделия на основе полной себестоимости

Показатель	Формула/таблица для расчета
Материалы	Формула (7.1), таблица 7.1
Покупные комплектующие изделия	Формула (7.2), таблица 7.2
Накладные расходы	$$Р_{накл} = \frac{(Р_м + Р_к) \cdot Н_{накл}}{100}, \quad (7.3)$$ где $Р_м$, $Р_к$ – расходы на материалы и комплектующие изделия, р.; m – номенклатура применяемых комплектующих; $Н_{накл}$ – норматив накладных расходов, % (54 %)
Полная себестоимость	$$С_п = Р_м + Р_к + Р_{накл} \quad (7.4)$$
Плановая прибыль	$$П_{ед} = \frac{С_п \cdot Р_{пр}}{100}, \quad (7.5)$$ где $Р_{пр}$ – рентабельность продукции (30%)
Отпускная цена изделия	$$Ц_{отп} = С_п + П_{ед} \quad (7.6)$$

Примечание – Прямые расходы на оплату труда не выделяются в отдельные статьи, так как в данном случае предполагается автоматизированное производство нового изделия

Результат формирования отпускной цены нового изделия представлен в таблице 7.4.

Таблица 7.4 – Формирование отпускной цены нового изделия на основе полной себестоимости

Показатель	Формула/таблица для расчета	Сумма, р.
Материалы	Таблица 7.1	7,94
Покупные комплектующие изделия	Таблица 7.2	30,89
Накладные расходы	По формуле (7.3): $\mathrm{Р}_{\text{накл}} = \frac{(7{,}94 + 30{,}89) \cdot 54}{100}$	20,97
Полная себестоимость	По формуле (7.4): $\mathrm{С}_{\text{п}} = 7{,}94 + 30{,}89 + 20{,}97$	59,8
Плановая прибыль	По формуле (7.5): $\mathrm{П}_{\text{ед}} = \frac{59{,}8 \cdot 30}{100}$	17,94
Отпускная цена изделия	По формуле (7.6): $\mathrm{Ц}_{\text{отп}} = 59{,}8 + 17{,}94$	77,74

Отпускная цена изделия составит 77,74 рубля, что меньше, чем цена ближайшего по функционалу модуля шифрования ZM2102, интегрированного в систему Z-Wave, которая составляет 80,56 рублей, учитывая, что данный модуль не предусматривает многого функционала, реализованного в разработанном в данном проекте устройстве. Из этого можем сделать выводы о достаточной конкурентоспособности проектируемого устройства.

7.3 Расчет экономического эффекта от производства и реализации изделия

Экономическим эффектом от производства и реализации нового изделия является прирост чистой прибыли, полученной от его реализации.

Расчет прироста чистой прибыли от реализации нового изделия осуществляется по формуле (7.7)

$$\Delta П_{ч} = N_{п} \cdot П_{ед}\left(1 - \frac{Н_{п}}{100}\right), \quad (7.7)$$

где $N_{п}$ – прогнозируемый годовой объем производства и реализации изделий, шт.; $П_{ед}$ – плановая прибыль, приходящаяся на единицу изделия, р.; $Н_{п}$ – ставка налога на прибыль согласно действующему законодательству, % (18 %).

В год по данным предприятия-производителя запланировано производство $N_{п} = 1000$ изделий. В первый неполный год запланировано производство 50% ($N_{п} = 500$ изделий) от нормального объема. Тогда, имея эти входные данные, чистая прибыль за первый год составит:

$$\Delta П_{ч\,1} = 500 \cdot 17{,}95\left(1 - \frac{18}{100}\right) = 7539{,}5 \text{ р.}$$

В последующие годы запланирован выход на производство 1000 изделий в год. Прирост чистой прибыли для этих данных составит:

$$\Delta П_{ч\,посл} = 1000 \cdot 17{,}95\left(1 - \frac{18}{100}\right) = 14719 \text{ р.}$$

7.4 Расчет инвестиций в производство изделия

Инвестиции в производство данного изделия включают:

- инвестиции в разработку изделия;
- инвестиции в прирост основного капитала;
- инвестиции в прирост собственного оборотного капитала.

Инвестиции в разработку нового изделия, согласно смете стороннего разработчика, составят $И_{р} = 20000$ р.

Инвестиции в прирост основного капитала не требуются, так как производство нового изделия планируется осуществлять на действующем оборудовании в связи с наличием на предприятии-производителе свободных производственных мощностей.

Инвестиций в прирост собственного оборотного капитала вычисляются в процентах от годовой потребности в материалах и комплектующих

изделиях – β (20 %, исходя из среднего уровня по экономике, $\beta = 0{,}2$) – по формуле (7.8).

$$И_{с.о.к.} = \beta \cdot (П_м + П_к), \quad (7.8)$$

где $П_м$ – годовая потребность в материалах, р.; $П_к$ – годовая потребность в комплектующих изделиях, р.

Годовая потребность в материалах определяется по формуле (7.9).

$$П_м = Р_м \cdot N_п, \quad (7.9)$$

где $Р_м$ – затраты на материалы на единицу изделия, р. (таблица 7.1).

$$П_м = 7{,}94 \cdot 1000 = 7940 \text{ р.}$$

Годовая потребность в комплектующих изделиях определяется по формуле (7.10).

$$П_к = Р_к \cdot N_п, \quad (7.10)$$

где $Р_к$ – затраты на комплектующие изделия на единицу продукции, р. (таблица 7.2).

$$П_к = 30{,}89 \cdot 1000 = 30890 \text{ р.}$$

Инвестиций в прирост собственного оборотного капитала, рассчитанные по формуле (7.8), составят:

$$И_{с.о.к.} = 0{,}2 \cdot (7940 + 30890) = 7766 \text{ р.}$$

7.5 Расчет показателей экономической эффективности инвестиций в производство нового изделия

При оценке эффективности инвестиционных проектов необходимо осуществить приведение затрат и результатов, полученных в разные периоды времени, к расчетному году, путем умножения затрат и результатов на коэффициент дисконтирования α_t, который определяется по формуле (7.11).

$$\alpha_t = \frac{1}{(1+d)^{t-t_p}}, \quad (7.11)$$

где d – требуемая норма дисконта ($d = 0{,}16$); t – порядковый номер года, доходы и затраты которого приводятся к расчетному году; t_p – расчетный год, к которому приводятся доходы и инвестиционные затраты ($t_p = 1$).

Таким образом, коэффициенты дисконтирования по годам составят:

$$\alpha_1 = \frac{1}{(1+0{,}16)^{1-1}} = 1,$$

$$\alpha_2 = \frac{1}{(1+0{,}16)^{2-1}} = 0{,}86,$$

$$\alpha_3 = \frac{1}{(1+0{,}16)^{3-1}} = 0{,}74,$$

$$\alpha_4 = \frac{1}{(1+0{,}16)^{4-1}} = 0{,}64,$$

Расчет экономической эффективности инвестиций осуществлен согласно методике, описанной в таблице 7.5.

Таблица 7.5 – Методика расчета основных показателей эффективности инвестиций

Показатель	Методика расчета
1	2
Простой срок окупаемости инвестиций ($T_{\text{ок}}$, PP)	$T_{\text{ок}}\,(PP) = \frac{\sum_{t=1}^{n} З_t}{\frac{1}{n} \cdot \sum_{t=1}^{n} \Delta П_{\text{ч}t}}$, (7.12) где n – расчетный период, лет; $З_t$ – затраты (инвестиции) в году t, р.; $\Delta П_{\text{ч}t}$ – прирост чистой прибыли в году t в результате реализации проекта, р.

Продолжение таблицы 7.5

1	2
Средняя норма прибыли/ рентабельности инвестиций ($Р_и$, ARR)	$$Р_и\ (ARR) = \frac{\frac{1}{n} \cdot \sum_{t=1}^{n} \Delta П_{чt}}{\sum_{t=1}^{n} З_t} \cdot 100\ \%, \quad (7.13)$$
Чистый дисконтированный доход (ЧДД, NPV)	$$ЧДД\ (NPV) = \sum_{t=1}^{n} \Delta П_{чt} \cdot \alpha_t - \sum_{t=1}^{n} З_t \cdot \alpha_t, \quad (7.14)$$
Динамический (дисконтированный) срок окупаемости инвестиций (DPP)	$DPP = n$, при котором $$\sum_{t=1}^{n} \Delta П_{чt} \cdot \alpha_t \geq \sum_{t=1}^{n} З_t \cdot \alpha_t, \quad (7.15)$$
Индекс доходности инвестиций (ИД, PI)	$$ИД\ (PI) = \frac{\sum_{t=1}^{n} \Delta П_{чt} \cdot \alpha_t}{\sum_{t=1}^{n} З_t \cdot \alpha_t}, \quad (7.16)$$

Расчет чистого дисконтированного дохода за расчетный период по формуле (7.14), динамического срока окупаемости инвестиций по формуле (7.15) и индекса доходности по формуле (7.16) представлен в таблице 7.6.

Таблица 7.6 – Расчет эффективности инвестиций в реализацию проектного решения

Показатель	Значение по годам расчетного периода			
	2022	2023	2024	2025
1	2	3	4	5
Результат				
Прирост чистой прибыли, р.	7539,5	14719	14719	14719
Дисконтированный результат, р.	7539,5	12658,34	10829,06	9420,16
Затраты				
Инвестиции в реализацию проектного решения, р.	27766	0	0	0
Дисконтированные инвестиции, р.	27766	0	0	0
Чистый дисконтированный доход по годам, р.	-20226,5	12658,34	10829,06	9420,16

Продолжение таблицы 7.6

1	2	3	4	5
Чистый дисконтированный доход нарастающим итогом (формула (7.14)), р.	-20226,5	-7568,16	3260,9	12681,06
Коэффициент дисконтирования, доли единицы	1	0,86	0,74	0,64

Из расчетных значений таблицы следует, что $DPP = 3$, так как именно на третий год расчетного периода выполняется условие, описанное в формуле (7.15).

Индекс доходности инвестиций, рассчитанный по формуле (7.16), составит:

$$\text{ИД}\ (PI) = \frac{(7529{,}5 + 12658{,}34 + 10829{,}06 + 9420{,}16)}{27766} = 1{,}46,$$

что составляет значение, большее целевого значения (1), и, соответственно, является хорошим показателем эффективности инвестиций.

Простой срок окупаемости инвестиций, согласно формуле (7.12), составит:

$$T_{\text{ок}}\ (PP) = \frac{27776}{\frac{1}{4} \cdot (7539{,}5 + 14719 + 14719 + 14719)} = 2{,}15 \text{ лет},$$

что отображает высокую эффективность инвестиций в реализацию проектного решения, так как значение меньше целевого значения в три-четыре года.

Средняя норма прибыли/рентабельности инвестиций рассчитан по формуле (7.13) следующим образом:

$$\text{Р}_{\text{и}}\ (ARR) = \frac{\frac{1}{4} \cdot (7539{,}5 + 14719 + 14719 + 14719)}{27776} \cdot 100\ \% = 46{,}53\ \%,$$

что существенно больше требуемой нормы дисконта (16 %), что также демонстрирует высокую эффективность инвестиций.

В процессе технико-экономического обоснования эффективности внедрения коммуникационного контроллера с шифрованием данных для системы «Умный дом» получены следующие результаты:

- экономический эффект от внедрения в производство изделия за четыре года составит 51696,5 р.;
- инвестиции окупятся на третий год производства;
- рентабельность проекта составит 46,53 %.

Разработка и внедрение нового изделия будут эффективны для предприятия-производителя, устройство будет конкурентоспособно на рынке, что означает, что все поставленные при разработке устройства экономические задачи были полностью реализованы.

8 АНАЛИЗ РЕЗУЛЬТАТОВ ПРОЕКТИРОВАНИЯ

8.1 Тестирование пропускной способности

Алгоритмы шифрования как правильно требуют значительных вычислительных ресурсов. В связи с этим, важным вопросом является оценка скорости работы алгоритма шифрования BelT в режиме счетчика на микронтроллере ATmega328, а также пропускной способоности разрабатываемого коммуникационного контроллера.

Для оценки скорости работы шифрования и пропускной способности контроллера была разработана тестовая программа для микроконтроллера Atmega328, осуществляющая шифрование и расшифрование блока данных размером 100 000 байт, а также производящая замер интервала времени. Результаты выводятся в терминал подключенного компьютера через порт UART с использованием библиотеки Serial.

```
belt_ctr_st stateEncr = {0};
 long time_counter = 0;
 unsigned long interval = 0;
 // инициализация состояния шифратора
 beltCTRStart(&stateEncr, key, len, iv);

 unsigned long start_time = millis();
 for(unsigned long k=0;k<100000;k++) {
   // шифрование данных
   beltCTRStepE(&outData, 1, &stateDecr);
 }
 unsigned long end_time = millis();
 unsigned long elapsed = end_time - start_time;
 Serial.print("Elapsed ms: ");
 Serial.println(elapsed);
```

В результате запуска данной программы на микроконтроллере ATmega328 была получена оценка скорости шифрования при помощи алгоритма BelT в режиме счетчика, которая составила 176 килобит в секунду, что является достаточным для устройств системы умный дом. На рисунке 8.1 изображены результаты тестирования пропускной способности.

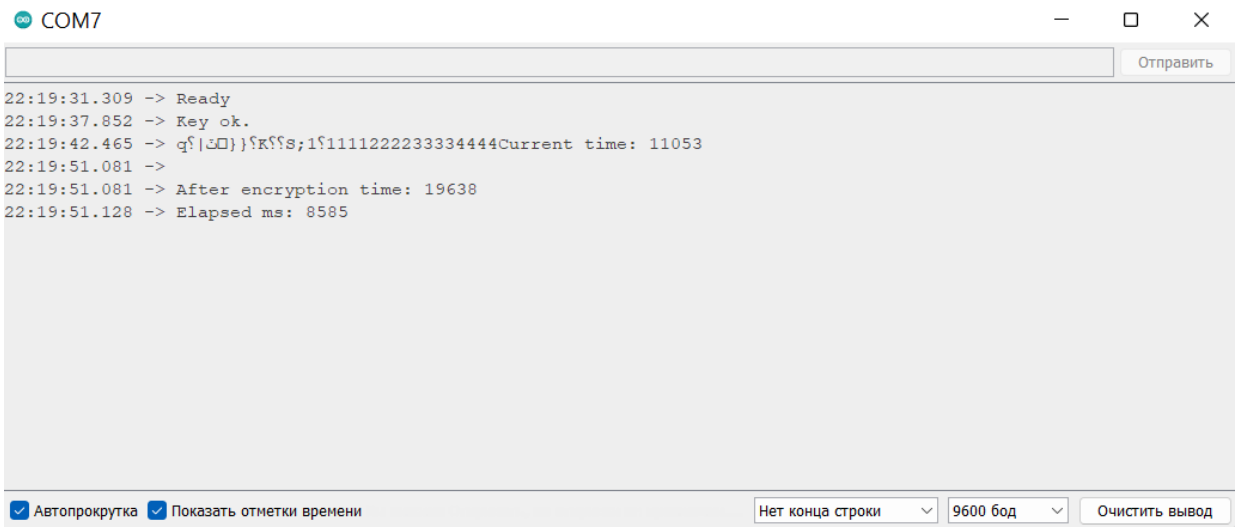


Рисунок 8.1 – Результаты тестирования пропускной способности

8.2 Результаты проектирования и программирования

В дипломном проекте разработана аппаратная и программная части коммуникационного контроллера с шифрованием данных для системы «Умный дом». Предлагаемое устройство представляет собой контроллер, предназначенный для обеспечения безопасной передачи данных в системе «Умный дом». Спроектированное устройство позволяет легко наращивать функционал аппаратной части, а также быстро модифицировать программную составляющую, что является, несомненно, положительными аспектами, при построении устройств такого рода.

Кроме того, к плюсам устройства относится его относительно малая стоимость по сравнению с профессиональными коммерческими аналогами. Также в будущем планируется реализовать возможность подключения по другим интерфейсам, реализовать возможность выхода в глобальную сеть «Интернет», удаленное управление устройством по сети, удаленную загрузку обновлений, оптимизацию вычислительных процессов. Также с аппаратной точки зрения можно было бы добавить драйверы на все имеющиеся интерфейсы в устройстве для возможности подключения по длинным линиям связи.

В ходе проектирования устройства была реализована различная инженерная чертежная документация, что позволяет инженерам или технологам проще вникнуть в устройство контроллера и осуществлять производство или доработку имеющегося устройства.

ЗАКЛЮЧЕНИЕ

В ходе проектирования коммуникационного контроллера с шифрованием данных для системы «Умный дом», учитывая последние разработки аналогичных приборов, были разработаны структурная схема устройства, функциональная схема и схема электрическая принципиальная.

При выполнении данного дипломного проекта широко использованы возможности вычислительной техники и пакеты систем автоматизированного проектирования современного уровня. К таким пакетам относятся Altium Designer, AutoCAD, Proteus Design и пакет Microsoft Office.

Спроектированное микропроцессорное устройство удовлетворяет техническим требованиям задания на дипломное проектирование, пояснительная записка содержит все основные разделы.

Была успешно разработана аппаратная часть устройства на заявленном в техническом задании микроконтроллере, а также иных радиоэлектронных составляющих.

Программная часть также была успешно реализована. Она включила в себя основную часть – шифрование по государственному стандарту РБ.

К достоинствам устройства можно отнести простоту и компактность, которая обусловлена использованием в устройстве минимального количества электронных компонентов. Конструкция устройства может быть в дальнейшем доработана расширением возможных интерфейсов, реализацией иных режимов шифрования данных и добавлением протокола загрузки исходных данных на устройство сетевым образом, не требуя непосредственного подключения внешнего компьютера.

В ходе выполнения дипломного проекта были произведены сравнение с аналогами на рынке и оценка экономической рентабельности разработки устройства. Результатом данных исследований стали доказательства конкурентоспособности устройства и технико-экономическое обоснование его разработки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] IEEE 802.15.4 [Электронный ресурс]. – Режим доступа : https://standards.ieee.org/ieee/802.15.4/7029/.

[2] Advanced Encription Standard [Электронный ресурс]. – Режим доступа : https://csrc.nist.gov/csrc/media/publications/fips/197/final/documents/fips-197.pdf.

[3] MiMiSmart [Электронный ресурс]. – Режим доступа : http://mimismart.ru/.

[4] KNX [Электронный ресурс]. – Режим доступа : https://www.knx.org/knx-en/for-professionals/What-is-KNX/A-brief-introduction/index.php.

[5] Home and Building Electronic Systems (HBES) - Part 3-4: Secure Application Layer, Secure Service, Secure configuration and security Resources / Deutsches Institut fur Normung e. V. – Berlin, Germany, 2018.

[6] Control4 [Электронный ресурс]. – Режим доступа : https://www.control4.com/.

[7] Z-Wave [Электронный ресурс]. – Режим доступа : https://z-wave.by/.

[8] СТБ 34.101.31 Информационные технологии и безопасность. Защита информации. Криптографические алгоритмы шифрования и контроля целостности; Введен 31.01.2011.

[9] ГОСТ 14.205-83 Технологичность конструкции изделий. Термины и определения; Введен 09.02.1983.

[10] ГОСТ 31819.21-2012 Аппаратура для измерения электрической энергии переменного тока. Частные требования. Часть 21. Статические счетчик активной энергии классов точности 1 и 2; Введен 01.01.2014.

[11] ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды; Введен 01.01.1971.

[12] ГОСТ 16350-80 Климат СССР. Районирование и статистические параметры климатических факторов для технических целней; Введен 01.07.1981.

[13] ГОСТ 34.003-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения; Введен 01.01.1992.

[14] ГОСТ 34.201-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначение документов при создании автоматизированных систем; Введен 01.01.1990.

[15] ГОСТ 34.601-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания; Введен 01.01.1992.

[16] ГОСТ 34.602-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы; Введен 01.01.1990.

[17] ГОСТ 62657-2-2016 Сети промышленной коммуникации. Беспроволочные коммуникационные сети. Часть 2. Обеспечение совместимости; Введен 01.04.2017.

[18] Светодиод GNL-0805GC [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/gnl-0805gc.

[19] ESP8266EX [Электронный ресурс]. – Режим доступа : https://www.euromobile.ru/upload/iblock/38e/38edea9ed541014c941ac8a47619db65.pdf.

[20] ATmega328P-AU [Электронный ресурс]. – Режим доступа : https://static.chipdip.ru/lib/303/DOC000303014.pdf.

[21] LF33CV [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/lf33cv.

[22] HC-49S [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/24mhz-hc-49s.

[23] KLS7-TS6601-4.3-180 [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/kls7-ts6601-4.3-180.

[24] 15EDGRC-3.81-04 [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/15edgrc-3.81-04.

[25] 15EDGRC-3.5-02 [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/15edgrc-3.5-02.

[26] Конденсаторы Murata [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/grm21br71h104k.

[27] Резисторы Yageo [Электронный ресурс]. – Режим доступа : https://www.chipdip.by/product/0.062w-0402-220-om-1.

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг программы микроконтроллера на языке С

```
      //
      //file belt.h
      //

      /*
      ************************************************************************
*******
      Шифрование в режиме счетчика
      ************************************************************************
*******
      */

      #define B_PER_W 16
      #define O_PER_W (B_PER_W / 8)
      typedef unsigned short WORD;
      typedef unsigned long u32;
      typedef signed long i32;
      typedef unsigned char u8;
      typedef signed char i8;
      typedef u8 octet;
      typedef unsigned int size_t;

      typedef struct
      {
       u32 key[8];  //форматированный ключ
       u32 ctr[4];  //счетчик
       octet block[16];//блок гаммы
       size_t reserved;//резерв октетов гаммы
      } belt_ctr_st;

      /*
      ************************************************************************
*******
      Ускорители

      Реализованы быстрые операции над блоками и полублоками belt. Блок
      представляется либо как [16]octet, либо как [4]u32,
      либо как [W_OF_B(128)]word.

      Суффикс U32 в именах макросов и функций означает, что данные
интерпретируются
      как массив u32. Суффикс W означает, что данные интерпретируются как
      массив word.
      ************************************************************************
*******
      */

      #define beltBlockIncU32(block)\
       if ((((u32*)(block))[0] += 1) == 0 &&\
        (((u32*)(block))[1] += 1) == 0 &&\
        (((u32*)(block))[2] += 1) == 0)\
        ((u32*)(block))[3] += 1\

      #define beltBlockCopy(dest, src)\
       ((WORD*)(dest))[0] = ((const WORD*)(src))[0],\
       ((WORD*)(dest))[1] = ((const WORD*)(src))[1],\
```

```
 ((WORD*)(dest))[2] = ((const WORD*)(src))[2],\
 ((WORD*)(dest))[3] = ((const WORD*)(src))[3]\

#define beltBlockXor2(dest, src)\
 ((WORD*)(dest))[0] ^= ((const WORD*)(src))[0],\
 ((WORD*)(dest))[1] ^= ((const WORD*)(src))[1],\
 ((WORD*)(dest))[2] ^= ((const WORD*)(src))[2],\
 ((WORD*)(dest))[3] ^= ((const WORD*)(src))[3]\

/*!
*******************************************************************
*******
Блоб -- объект в памяти определенного размера. В функциях работы с
блобами
используются их дескрипторы -- "умные" указатели. С дескрипторами можно
работать как с обычными указателями, т.е. использовать их в функциях
типа
memcpy, memset. Дополнительно по указателю можно определить размер
блоба.

Реализация работы с блобами может быть платформенно-зависимой.

Реализация должна гарантировать защиту содержимого блобов от утечек,
например, через файл подкачки. Поэтому в блобах рекомендуется размещать
ключи и другие критические объекты.

В функциях работы с блобами дескрипторы входных блобов корректны.
*******************************************************************
*******
*/

// память для блобов выделяется страницами
#define BLOB_PAGE_SIZE 1024

// требуется страниц
#define blobPageCount(size)\
 (((size) + sizeof(size_t) + BLOB_PAGE_SIZE - 1) / BLOB_PAGE_SIZE)

// требуется памяти на страницах
#define blobActualSize(size)\
 (blobPageCount(size) * BLOB_PAGE_SIZE)

// heap-указатель для блоба
#define blobPtrOf(blob) ((size_t*)blob - 1)

// размер блоба
#define blobSizeOf(blob) (*blobPtrOf(blob))

// страничный размер блоба
#define blobActualSizeOf(blob) (blobActualSize(blobSizeOf(blob)))

// блоб для heap-указателя
#define blobValueOf(ptr) ((blob_t)((size_t*)ptr + 1))

// дескриптор блоба
typedef void* blob_t;

/* Инициализация шифрования в режиме CTR

 По ключу [len]key и синхропосылке iv в state формируются
 структуры данных, необходимые для шифрования в режиме CTR.
 len == 16 || len == 24 || len == 32.
 По адресу state зарезервировано beltCTR_keep() октетов.
```

```
       Буферы key и state могут пересекаться.
      */
      void beltCTRStart(
       void* state,  //[out] состояние
       const octet key[],  //[in] ключ
       size_t len,   //[in] длина ключа в октетах
       const octet iv[16]  //[in] синхропосылка
      );

      /* Зашифрование фрагмента в режиме CTR

       Буфер [count]buf зашифровывается в режиме CTR на ключе, размещенном
       в state.
       beltCTRStart() < beltCTRStepE()*.
      */
      void beltCTRStepE(
       void* buf,  //[in/out] открытый текст / шифртекст
       size_t count, //[in] число октетов текста
       void* state  //[in/out] состояние
      );

      /* Расшифрование фрагмента в режиме CTR
       Зашифрование в режиме CTR не отличается от расшифрования.
      */
      #define beltCTRStepD beltCTRStepE

      /* Шифрование в режиме CTR

       Буфер [count]src зашифровывается или расшифровывается на ключе
       [len]key с использованием синхропосылки iv. Результат шифрования
       размещается в буфере [count]dest.
       {ERR_BAD_INPUT} len == 16 || len == 24 || len == 32.
       ERR_OK, если шифрование завершено успешно, и код ошибки
       в противном случае.
       Буферы могут пересекаться.
      */

      //
      //file belt.cpp
      //

      /*
      ********************************************************************
*******
      STB 34.101.31 (belt): CTR encryption
      ********************************************************************
*******
      */

      #include "belt.h"
      #include <string.h>
      #include <avr/pgmspace.h>

      /*
      ********************************************************************
*******
      Загрузка
      ********************************************************************
*******
      */

      void u32From(u32 dest[], const void* src, size_t count)
      {
```

```
	memmove(dest, src, count);
	if (count % 4)
	 memset((octet*)dest + count, 0, 4 - count % 4);
	}

	/*
	*******************************************************************************
	Расширение ключа
	*******************************************************************************
	*/

	void beltKeyExpand(u32 key_[8], const octet key[], size_t len)
	{
	u32From(key_, key, len);
	if (len == 16)
	{
	 key_[4] = key_[0];
	 key_[5] = key_[1];
	 key_[6] = key_[2];
	 key_[7] = key_[3];
	}
	else if (len == 24)
	{
	 key_[6] = key_[0] ^ key_[1] ^ key_[2];
	 key_[7] = key_[3] ^ key_[4] ^ key_[5];
	}
	}

	/*
	*******************************************************************************
	H-блок
	*******************************************************************************
	*/
	/*
	static const octet H[256] = {
	 0xB1,0x94,0xBA,0xC8,0x0A,0x08,0xF5,0x3B,0x36,0x6D,0x00,0x8E,0x58,0x4A,0
x5D,0xE4,
	 0x85,0x04,0xFA,0x9D,0x1B,0xB6,0xC7,0xAC,0x25,0x2E,0x72,0xC2,0x02,0xFD,0
xCE,0x0D,
	 0x5B,0xE3,0xD6,0x12,0x17,0xB9,0x61,0x81,0xFE,0x67,0x86,0xAD,0x71,0x6B,0
x89,0x0B,
	 0x5C,0xB0,0xC0,0xFF,0x33,0xC3,0x56,0xB8,0x35,0xC4,0x05,0xAE,0xD8,0xE0,0
x7F,0x99,
	 0xE1,0x2B,0xDC,0x1A,0xE2,0x82,0x57,0xEC,0x70,0x3F,0xCC,0xF0,0x95,0xEE,0
x8D,0xF1,
	 0xC1,0xAB,0x76,0x38,0x9F,0xE6,0x78,0xCA,0xF7,0xC6,0xF8,0x60,0xD5,0xBB,0
x9C,0x4F,
	 0xF3,0x3C,0x65,0x7B,0x63,0x7C,0x30,0x6A,0xDD,0x4E,0xA7,0x79,0x9E,0xB2,0
x3D,0x31,
	 0x3E,0x98,0xB5,0x6E,0x27,0xD3,0xBC,0xCF,0x59,0x1E,0x18,0x1F,0x4C,0x5A,0
xB7,0x93,
	 0xE9,0xDE,0xE7,0x2C,0x8F,0x0C,0x0F,0xA6,0x2D,0xDB,0x49,0xF4,0x6F,0x73,0
x96,0x47,
	 0x06,0x07,0x53,0x16,0xED,0x24,0x7A,0x37,0x39,0xCB,0xA3,0x83,0x03,0xA9,0
x8B,0xF6,
	 0x92,0xBD,0x9B,0x1C,0xE5,0xD1,0x41,0x01,0x54,0x45,0xFB,0xC9,0x5E,0x4D,0
x0E,0xF2,
	 0x68,0x20,0x80,0xAA,0x22,0x7D,0x64,0x2F,0x26,0x87,0xF9,0x34,0x90,0x40,0
x55,0x11,
```

```
       0xBE,0x32,0x97,0x13,0x43,0xFC,0x9A,0x48,0xA0,0x2A,0x88,0x5F,0x19,0x4B,0
x09,0xA1,
       0x7E,0xCD,0xA4,0xD0,0x15,0x44,0xAF,0x8C,0xA5,0x84,0x50,0xBF,0x66,0xD2,0
xE8,0x8A,
       0xA2,0xD7,0x46,0x52,0x42,0xA8,0xDF,0xB3,0x69,0x74,0xC5,0x51,0xEB,0x23,0
x29,0x21,
       0xD4,0xEF,0xD9,0xB4,0x3A,0x62,0x28,0x75,0x91,0x14,0x10,0xEA,0x77,0x6C,0
xDA,0x1D,
      };

      const octet* beltH()
      {
       return H;
      }
      */

      /*
      ****************************************************************************
*******
      Расширенные H-блоки

       Описание построено с помощью функции:

       void beltExtendBoxes()
       {
        unsigned r, x;
        u32 y;
        for (r = 5; r < 32; r += 8)
        {
         printf("static const u32 H%u[256] = {", r);
         for (x = 0; x < 256; x++)
          y = H[x],
          y = y << r | y >> (32 - r),
          printf(x % 8 ? "0x%08X," : "\n\t0x%08X,", y);
         printf("\n};\n");
        }
       }
      ****************************************************************************
*******
      */

      static const u32 H5[256] PROGMEM = {
       0x00001620,0x00001280,0x00001740,0x00001900,0x00000140,0x00000100,0x000
01EA0,0x00000760,
       0x000006C0,0x00000DA0,0x00000000,0x000011C0,0x00000B00,0x00000940,0x000
00BA0,0x00001C80,
       0x000010A0,0x00000080,0x00001F40,0x000013A0,0x00000360,0x000016C0,0x000
018E0,0x00001580,
       0x000004A0,0x000005C0,0x00000E40,0x00001840,0x00000040,0x00001FA0,0x000
019C0,0x000001A0,
       0x00000B60,0x00001C60,0x00001AC0,0x00000240,0x000002E0,0x00001720,0x000
00C20,0x00001020,
       0x00001FC0,0x00000CE0,0x000010C0,0x000015A0,0x00000E20,0x00000D60,0x000
01120,0x00000160,
       0x00000B80,0x00001600,0x00001800,0x00001FE0,0x00000660,0x00001860,0x000
00AC0,0x00001700,
       0x000006A0,0x00001880,0x000000A0,0x000015C0,0x00001B00,0x00001C00,0x000
00FE0,0x00001320,
       0x00001C20,0x00000560,0x00001B80,0x00000340,0x00001C40,0x00001040,0x000
00AE0,0x00001D80,
       0x00000E00,0x000007E0,0x00001980,0x00001E00,0x000012A0,0x00001DC0,0x000
011A0,0x00001E20,
```

```
       0x00001820,0x00001560,0x00000EC0,0x00000700,0x000013E0,0x00001CC0,0x000
00F00,0x00001940,
       0x00001EE0,0x000018C0,0x00001F00,0x00000C00,0x00001AA0,0x00001760,0x000
01380,0x000009E0,
       0x00001E60,0x00000780,0x00000CA0,0x00000F60,0x00000C60,0x00000F80,0x000
00600,0x00000D40,
       0x00001BA0,0x000009C0,0x000014E0,0x00000F20,0x000013C0,0x00001640,0x000
007A0,0x00000620,
       0x000007C0,0x00001300,0x000016A0,0x00000DC0,0x000004E0,0x00001A60,0x000
01780,0x000019E0,
       0x00000B20,0x000003C0,0x00000300,0x000003E0,0x00000980,0x00000B40,0x000
016E0,0x00001260,
       0x00001D20,0x00001BC0,0x00001CE0,0x00000580,0x000011E0,0x00000180,0x000
001E0,0x000014C0,
       0x000005A0,0x00001B60,0x00000920,0x00001E80,0x00000DE0,0x00000E60,0x000
012C0,0x000008E0,
       0x000000C0,0x000000E0,0x00000A60,0x000002C0,0x00001DA0,0x00000480,0x000
00F40,0x000006E0,
       0x00000720,0x00001960,0x00001460,0x00001060,0x00000060,0x00001520,0x000
01160,0x00001EC0,
       0x00001240,0x000017A0,0x00001360,0x00000380,0x00001CA0,0x00001A20,0x000
00820,0x00000020,
       0x00000A80,0x000008A0,0x00001F60,0x00001920,0x00000BC0,0x000009A0,0x000
001C0,0x00001E40,
       0x00000D00,0x00000400,0x00001000,0x00001540,0x00000440,0x00000FA0,0x000
00C80,0x000005E0,
       0x000004C0,0x000010E0,0x00001F20,0x00000680,0x00001200,0x00000800,0x000
00AA0,0x00000220,
       0x000017C0,0x00000640,0x000012E0,0x00000260,0x00000860,0x00001F80,0x000
01340,0x00000900,
       0x00001400,0x00000540,0x00001100,0x00000BE0,0x00000320,0x00000960,0x000
00120,0x00001420,
       0x00000FC0,0x000019A0,0x00001480,0x00001A00,0x000002A0,0x00000880,0x000
015E0,0x00001180,
       0x000014A0,0x00001080,0x00000A00,0x000017E0,0x00000CC0,0x00001A40,0x000
01D00,0x00001140,
       0x00001440,0x00001AE0,0x000008C0,0x00000A40,0x00000840,0x00001500,0x000
01BE0,0x00001660,
       0x00000D20,0x00000E80,0x000018A0,0x00000A20,0x00001D60,0x00000460,0x000
00520,0x00000420,
       0x00001A80,0x00001DE0,0x00001B20,0x00001680,0x00000740,0x00000C40,0x000
00500,0x00000EA0,
       0x00001220,0x00000280,0x00000200,0x00001D40,0x00000EE0,0x00000D80,0x000
01B40,0x000003A0,
      };
      static const u32 H13[256] PROGMEM = {
       0x00162000,0x00128000,0x00174000,0x00190000,0x00014000,0x00010000,0x001
EA000,0x00076000,
       0x0006C000,0x000DA000,0x00000000,0x0011C000,0x000B0000,0x00094000,0x000
BA000,0x001C8000,
       0x0010A000,0x00008000,0x001F4000,0x0013A000,0x00036000,0x0016C000,0x001
8E000,0x00158000,
       0x0004A000,0x0005C000,0x000E4000,0x00184000,0x00004000,0x001FA000,0x001
9C000,0x0001A000,
       0x000B6000,0x001C6000,0x001AC000,0x00024000,0x0002E000,0x00172000,0x000
C2000,0x00102000,
       0x001FC000,0x000CE000,0x0010C000,0x0015A000,0x000E2000,0x000D6000,0x001
12000,0x00016000,
       0x000B8000,0x00160000,0x00180000,0x001FE000,0x00066000,0x00186000,0x000
AC000,0x00170000,
       0x0006A000,0x00188000,0x0000A000,0x0015C000,0x001B0000,0x001C0000,0x000
FE000,0x00132000,
```

```
       0x001C2000,0x00056000,0x001B8000,0x00034000,0x001C4000,0x00104000,0x000
AE000,0x001D8000,
       0x000E0000,0x0007E000,0x00198000,0x001E0000,0x0012A000,0x001DC000,0x001
1A000,0x001E2000,
       0x00182000,0x00156000,0x000EC000,0x00070000,0x0013E000,0x001CC000,0x000
F0000,0x00194000,
       0x001EE000,0x0018C000,0x001F0000,0x000C0000,0x001AA000,0x00176000,0x001
38000,0x0009E000,
       0x001E6000,0x00078000,0x000CA000,0x000F6000,0x000C6000,0x000F8000,0x000
60000,0x000D4000,
       0x001BA000,0x0009C000,0x0014E000,0x000F2000,0x0013C000,0x00164000,0x000
7A000,0x00062000,
       0x0007C000,0x00130000,0x0016A000,0x000DC000,0x0004E000,0x001A6000,0x001
78000,0x0019E000,
       0x000B2000,0x0003C000,0x00030000,0x0003E000,0x00098000,0x000B4000,0x001
6E000,0x00126000,
       0x001D2000,0x001BC000,0x001CE000,0x00058000,0x0011E000,0x00018000,0x000
1E000,0x0014C000,
       0x0005A000,0x001B6000,0x00092000,0x001E8000,0x000DE000,0x000E6000,0x001
2C000,0x0008E000,
       0x0000C000,0x0000E000,0x000A6000,0x0002C000,0x001DA000,0x00048000,0x000
F4000,0x0006E000,
       0x00072000,0x00196000,0x00146000,0x00106000,0x00006000,0x00152000,0x001
16000,0x001EC000,
       0x00124000,0x0017A000,0x00136000,0x00038000,0x001CA000,0x001A2000,0x000
82000,0x00002000,
       0x000A8000,0x0008A000,0x001F6000,0x00192000,0x000BC000,0x0009A000,0x000
1C000,0x001E4000,
       0x000D0000,0x00040000,0x00100000,0x00154000,0x00044000,0x000FA000,0x000
C8000,0x0005E000,
       0x0004C000,0x0010E000,0x001F2000,0x00068000,0x00120000,0x00080000,0x000
AA000,0x00022000,
       0x0017C000,0x00064000,0x0012E000,0x00026000,0x00086000,0x001F8000,0x001
34000,0x00090000,
       0x00140000,0x00054000,0x00110000,0x000BE000,0x00032000,0x00096000,0x000
12000,0x00142000,
       0x000FC000,0x0019A000,0x00148000,0x001A0000,0x0002A000,0x00088000,0x001
5E000,0x00118000,
       0x0014A000,0x00108000,0x000A0000,0x0017E000,0x000CC000,0x001A4000,0x001
D0000,0x00114000,
       0x00144000,0x001AE000,0x0008C000,0x000A4000,0x00084000,0x00150000,0x001
BE000,0x00166000,
       0x000D2000,0x000E8000,0x0018A000,0x000A2000,0x001D6000,0x00046000,0x000
52000,0x00042000,
       0x001A8000,0x001DE000,0x001B2000,0x00168000,0x00074000,0x000C4000,0x000
50000,0x000EA000,
       0x00122000,0x00028000,0x00020000,0x001D4000,0x000EE000,0x000D8000,0x001
B4000,0x0003A000,
      };
      static const u32 H21[256] PROGMEM = {
       0x16200000,0x12800000,0x17400000,0x19000000,0x01400000,0x01000000,0x1EA
00000,0x07600000,
       0x06C00000,0x0DA00000,0x00000000,0x11C00000,0x0B000000,0x09400000,0x0BA
00000,0x1C800000,
       0x10A00000,0x00800000,0x1F400000,0x13A00000,0x03600000,0x16C00000,0x18E
00000,0x15800000,
       0x04A00000,0x05C00000,0x0E400000,0x18400000,0x00400000,0x1FA00000,0x19C
00000,0x01A00000,
       0x0B600000,0x1C600000,0x1AC00000,0x02400000,0x02E00000,0x17200000,0x0C2
00000,0x10200000,
       0x1FC00000,0x0CE00000,0x10C00000,0x15A00000,0x0E200000,0x0D600000,0x112
00000,0x01600000,
```

```
       0x0B800000,0x16000000,0x18000000,0x1FE00000,0x06600000,0x18600000,0x0AC
00000,0x17000000,
       0x06A00000,0x18800000,0x00A00000,0x15C00000,0x1B000000,0x1C000000,0x0FE
00000,0x13200000,
       0x1C200000,0x05600000,0x1B800000,0x03400000,0x1C400000,0x10400000,0x0AE
00000,0x1D800000,
       0x0E000000,0x07E00000,0x19800000,0x1E000000,0x12A00000,0x1DC00000,0x11A
00000,0x1E200000,
       0x18200000,0x15600000,0x0EC00000,0x07000000,0x13E00000,0x1CC00000,0x0F0
00000,0x19400000,
       0x1EE00000,0x18C00000,0x1F000000,0x0C000000,0x1AA00000,0x17600000,0x138
00000,0x09E00000,
       0x1E600000,0x07800000,0x0CA00000,0x0F600000,0x0C600000,0x0F800000,0x060
00000,0x0D400000,
       0x1BA00000,0x09C00000,0x14E00000,0x0F200000,0x13C00000,0x16400000,0x07A
00000,0x06200000,
       0x07C00000,0x13000000,0x16A00000,0x0DC00000,0x04E00000,0x1A600000,0x178
00000,0x19E00000,
       0x0B200000,0x03C00000,0x03000000,0x03E00000,0x09800000,0x0B400000,0x16E
00000,0x12600000,
       0x1D200000,0x1BC00000,0x1CE00000,0x05800000,0x11E00000,0x01800000,0x01E
00000,0x14C00000,
       0x05A00000,0x1B600000,0x09200000,0x1E800000,0x0DE00000,0x0E600000,0x12C
00000,0x08E00000,
       0x00C00000,0x00E00000,0x0A600000,0x02C00000,0x1DA00000,0x04800000,0x0F4
00000,0x06E00000,
       0x07200000,0x19600000,0x14600000,0x10600000,0x00600000,0x15200000,0x116
00000,0x1EC00000,
       0x12400000,0x17A00000,0x13600000,0x03800000,0x1CA00000,0x1A200000,0x082
00000,0x00200000,
       0x0A800000,0x08A00000,0x1F600000,0x19200000,0x0BC00000,0x09A00000,0x01C
00000,0x1E400000,
       0x0D000000,0x04000000,0x10000000,0x15400000,0x04400000,0x0FA00000,0x0C8
00000,0x05E00000,
       0x04C00000,0x10E00000,0x1F200000,0x06800000,0x12000000,0x08000000,0x0AA
00000,0x02200000,
       0x17C00000,0x06400000,0x12E00000,0x02600000,0x08600000,0x1F800000,0x134
00000,0x09000000,
       0x14000000,0x05400000,0x11000000,0x0BE00000,0x03200000,0x09600000,0x012
00000,0x14200000,
       0x0FC00000,0x19A00000,0x14800000,0x1A000000,0x02A00000,0x08800000,0x15E
00000,0x11800000,
       0x14A00000,0x10800000,0x0A000000,0x17E00000,0x0CC00000,0x1A400000,0x1D0
00000,0x11400000,
       0x14400000,0x1AE00000,0x08C00000,0x0A400000,0x08400000,0x15000000,0x1BE
00000,0x16600000,
       0x0D200000,0x0E800000,0x18A00000,0x0A200000,0x1D600000,0x04600000,0x052
00000,0x04200000,
       0x1A800000,0x1DE00000,0x1B200000,0x16800000,0x07400000,0x0C400000,0x050
00000,0x0EA00000,
       0x12200000,0x02800000,0x02000000,0x1D400000,0x0EE00000,0x0D800000,0x1B4
00000,0x03A00000,
      };
      static const u32 H29[256] PROGMEM = {
       0x20000016,0x80000012,0x40000017,0x00000019,0x40000001,0x00000001,0xA00
0001E,0x60000007,
       0xC0000006,0xA000000D,0x00000000,0xC0000011,0x0000000B,0x40000009,0xA00
0000B,0x8000001C,
       0xA0000010,0x80000000,0x4000001F,0xA0000013,0x60000003,0xC0000016,0xE00
00018,0x80000015,
       0xA0000004,0xC0000005,0x4000000E,0x40000018,0x40000000,0xA000001F,0xC00
00019,0xA0000001,
```

```
      0x6000000B,0x6000001C,0xC000001A,0x40000002,0xE0000002,0x20000017,0x200
0000C,0x20000010,
      0xC000001F,0xE000000C,0xC0000010,0xA0000015,0x2000000E,0x6000000D,0x200
00011,0x60000001,
      0x8000000B,0x00000016,0x00000018,0xE000001F,0x60000006,0x60000018,0xC00
0000A,0x00000017,
      0xA0000006,0x80000018,0xA0000000,0xC0000015,0x0000001B,0x0000001C,0xE00
0000F,0x20000013,
      0x2000001C,0x60000005,0x8000001B,0x40000003,0x4000001C,0x40000010,0xE00
0000A,0x8000001D,
      0x0000000E,0xE0000007,0x80000019,0x0000001E,0xA0000012,0xC000001D,0xA00
00011,0x2000001E,
      0x20000018,0x60000015,0xC000000E,0x00000007,0xE0000013,0xC000001C,0x000
0000F,0x40000019,
      0xE000001E,0xC0000018,0x0000001F,0x0000000C,0xA000001A,0x60000017,0x800
00013,0xE0000009,
      0x6000001E,0x80000007,0xA000000C,0x6000000F,0x6000000C,0x8000000F,0x000
00006,0x4000000D,
      0xA000001B,0xC0000009,0xE0000014,0x2000000F,0xC0000013,0x40000016,0xA00
00007,0x20000006,
      0xC0000007,0x00000013,0xA0000016,0xC000000D,0xE0000004,0x6000001A,0x800
00017,0xE0000019,
      0x2000000B,0xC0000003,0x00000003,0xE0000003,0x80000009,0x4000000B,0xE00
00016,0x60000012,
      0x2000001D,0xC000001B,0xE000001C,0x80000005,0xE0000011,0x80000001,0xE00
00001,0xC0000014,
      0xA0000005,0x6000001B,0x20000009,0x8000001E,0xE000000D,0x6000000E,0xC00
00012,0xE0000008,
      0xC0000000,0xE0000000,0x6000000A,0xC0000002,0xA000001D,0x80000004,0x400
0000F,0xE0000006,
      0x20000007,0x60000019,0x60000014,0x60000010,0x60000000,0x20000015,0x600
00011,0xC000001E,
      0x40000012,0xA0000017,0x60000013,0x80000003,0xA000001C,0x2000001A,0x200
00008,0x20000000,
      0x8000000A,0xA0000008,0x6000001F,0x20000019,0xC000000B,0xA0000009,0xC00
00001,0x4000001E,
      0x0000000D,0x00000004,0x00000010,0x40000015,0x40000004,0xA000000F,0x800
0000C,0xE0000005,
      0xC0000004,0xE0000010,0x2000001F,0x80000006,0x00000012,0x00000008,0xA00
0000A,0x20000002,
      0xC0000017,0x40000006,0xE0000012,0x60000002,0x60000008,0x8000001F,0x400
00013,0x00000009,
      0x00000014,0x40000005,0x00000011,0xE000000B,0x20000003,0x60000009,0x200
00001,0x20000014,
      0xC000000F,0xA0000019,0x80000014,0x0000001A,0xA0000002,0x80000008,0xE00
00015,0x80000011,
      0xA0000014,0x80000010,0x0000000A,0xE0000017,0xC000000C,0x4000001A,0x000
0001D,0x40000011,
      0x40000014,0xE000001A,0xC0000008,0x4000000A,0x40000008,0x00000015,0xE00
0001B,0x60000016,
      0x2000000D,0x8000000E,0xA0000018,0x2000000A,0x6000001D,0x60000004,0x200
00005,0x20000004,
      0x8000001A,0xE000001D,0x2000001B,0x80000016,0x40000007,0x4000000C,0x000
00005,0xA000000E,
      0x20000012,0x80000002,0x00000002,0x4000001D,0xE000000E,0x8000000D,0x400
0001B,0xA0000003,
      };

      /*
      ***************************************************************************
*******
      G-блоки
```

```
      ********************************************************************
*******
      */
      #define G5(x)\
       pgm_read_dword_near(H5 + ((x) & 255)) ^ pgm_read_dword_near(H13 + ((x)
>>  8  &  255))  ^  pgm_read_dword_near(H21  +  ((x)  >>  16  &  255))  ^
pgm_read_dword_near(H29 + ((x) >> 24))
      #define G13(x)\
       pgm_read_dword_near(H13 + ((x) & 255)) ^ pgm_read_dword_near(H21 + ((x)
>>  8  &  255))  ^  pgm_read_dword_near(H29  +  ((x)  >>  16  &  255))  ^
pgm_read_dword_near(H5 + ((x) >> 24))
      #define G21(x)\
       pgm_read_dword_near(H21 + ((x) & 255)) ^ pgm_read_dword_near(H29 + ((x)
>>  8  &  255))  ^  pgm_read_dword_near(H5  +  ((x)  >>  16  &  255))  ^
pgm_read_dword_near(H13 + ((x) >> 24))

      /*
      ********************************************************************
*******
      Тактовая подстановка

      Макрос R реализует шаги 2.1-2.9 алгоритмов зашифрования и расшифрования.

      На шагах 2.4-2.6 дополнительный регистр e не используется.
      Нужные данные сохраняются в регистрах b и c.

      Параметр-макрос subkey задает порядок использования тактовых ключей:
      порядок subkey = subkey_e используется при зашифровании и расшифровании
      ********************************************************************
*******
      */
      #define R(a, b, c, d, K, i, subkey)\
       *b ^= G5(*a + subkey(K, i, 0));\
       *c ^= G21(*d + subkey(K, i, 1));\
       *a -= G13(*b + subkey(K, i, 2));\
       *c += *b;\
       *b += G21(*c + subkey(K, i, 3)) ^ i;\
       *c -= *b;\
       *d += G13(*c + subkey(K, i, 4));\
       *b ^= G21(*a + subkey(K, i, 5));\
       *c ^= G5(*d + subkey(K, i, 6));\

      #define subkey_e(K, i, j) K[(7 * i - 7 + j) % 8]

      /*
      ********************************************************************
*******
      Такты зашифрования

      Перестановка содержимого регистров a, b, c, d реализуется перестановкой
      параметров макроса R. После выполнения последнего макроса R и шагов
2.10-2.12
      алгоритма зашифрования в регистрах a, b, c, d будут находиться значения,
      соответствующие спецификации belt.

      Окончательная перестановка abcd -> bdac реализуется инверсиями:
      a <-> b, c <-> d, b <-> c.
      ********************************************************************
*******
      */
      #define E(a, b, c, d, K)\
       R(a, b, c, d, K, 1, subkey_e);\
       R(b, d, a, c, K, 2, subkey_e);\
```

```
       R(d, c, b, a, K, 3, subkey_e);\
       R(c, a, d, b, K, 4, subkey_e);\
       R(a, b, c, d, K, 5, subkey_e);\
       R(b, d, a, c, K, 6, subkey_e);\
       R(d, c, b, a, K, 7, subkey_e);\
       R(c, a, d, b, K, 8, subkey_e);\
       *a ^= *b, *b ^= *a, *a ^= *b;\
       *c ^= *d, *d ^= *c, *c ^= *d;\
       *b ^= *c, *c ^= *b, *b ^= *c;\

      /*
      ********************************************************************
*******
      Зашифрование блока
      ********************************************************************
*******
      */

      void beltBlockEncr(u32 block[4], const u32 key[8])
      {
       E((block + 0), (block + 1), (block + 2), (block + 3), key);
      }

      /*
      ********************************************************************
*******
      Расшифрование блока
      ********************************************************************
*******
      */

      void memXor2(void* dest, const void* src, size_t count)
      {
       for (; count >= O_PER_W; count -= O_PER_W)
       {
        *(WORD*)dest ^= *(const WORD*)src;
        src = (const WORD*)src + 1;
        dest = (WORD*)dest + 1;
       }
       while (count--)
       {
        *(octet*)dest ^= *(const octet*)src;
        src = (const octet*)src + 1;
        dest = (octet*)dest + 1;
       }
      }

      /*
      ********************************************************************
*******
      Шифрование в режиме CTR

      Для ускорения работы счетчик ctr хранится в виде [4]u32. Это позволяет
      зашифровывать счетчик с помощью функции beltBlockEncr(), в которой
      не используется реверс октетов  даже на платформах BIG_ENDIAN.
      Реверс применяется только перед использованием зашифрованного счетчика
      в качестве гаммы.
      ********************************************************************
*******
      */

      void beltCTRStart(void* state, const octet key[], size_t len,
       const octet iv[16])
```

```
{
 belt_ctr_st* st = (belt_ctr_st*)state;
 beltKeyExpand(st->key, key, len);
 u32From(st->ctr, iv, 16);
 beltBlockEncr(st->ctr, st->key);
 st->reserved = 0;
}

void beltCTRStepE(void* buf, size_t count, void* state)
{
 belt_ctr_st* st = (belt_ctr_st*)state;
 // есть резерв гаммы?
 if (st->reserved)
 {
  if (st->reserved >= count)
  {
   memXor2(buf, st->block + 16 - st->reserved, count);
   st->reserved -= count;
   return;
  }
  memXor2(buf, st->block + 16 - st->reserved, st->reserved);
  count -= st->reserved;
  buf = (octet*)buf + st->reserved;
  st->reserved = 0;
 }
 // цикл по полным блокам
 while (count >= 16)
 {
  beltBlockIncU32(st->ctr);
  beltBlockCopy(st->block, st->ctr);
  beltBlockEncr((u32*)st->block, st->key);
  beltBlockXor2(buf, st->block);
  buf = (octet*)buf + 16;
  count -= 16;
 }
 // неполный блок?
 if (count)
 {
  beltBlockIncU32(st->ctr);
  beltBlockCopy(st->block, st->ctr);
  beltBlockEncr((u32*)st->block, st->key);
  memXor2(buf, st->block, count);
  st->reserved = 16 - count;
 }
}

//
//file dip_proj.ino
//

#include <OneWire.h>
#include <DallasTemperature.h>
#include <SPI.h>
#include <Ethernet.h>
#include <EEPROM.h>
#include <belt.h>

#define ONE_WIRE_BUS 6

OneWire  oneWire(ONE_WIRE_BUS);  //Настройка  1wire  для  работы  с  6-м
выводом Ардуино
DallasTemperature sensors(&oneWire); //Подключаем датчик температуры
```

```
      bool SPI_is_receiving = false; //индикация получения данных по SPI

      size_t len = 16; //длина ключа
      octet key[16] = {0}; //объявление глобального массива для хранения ключа
      const octet iv[16] = { 'z', 'j', 'l', 'b', 'y', ':', 'b', 'd', '0', 'q',
'f', 'l', 'h', 'e', 'u', '7' }; //синхропосылка

      char outData[256] = {0}; //выходные данные, нуждающиеся в шифровании
      char inData[256] = {0}; //входные данные, нуждающиеся в расшифровке

      belt_ctr_st stateEncr = {0};
      belt_ctr_st stateDecr = {0};

      int ii=1;
      int ij=0;
      byte r=0,g=0,b=0;
      char buf[100];
      char bb[40];

      // задание MAC-адреса устройства
      byte mac[] = {0x0E, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};

      // Параметры сети по умолчанию при отсуствии DHCP-свервера
      // Настройка IP-адреса, шлюза, DNS-сервера, и маски сети
      IPAddress ip(192,168,1,30);
      IPAddress myDns(8,8,8,8);
      IPAddress gateway(192,168,1,1);
      IPAddress subnet(255,255,255,0);

      EthernetClient client;
      int addr_len=0;
      int addr = 0;
      char server[100];
      int buttonState = 1;

      unsigned long lastConnectionTime = 0; // время последнего подключения к
серверу (в миллисекундах)
      boolean lastConnected = false; // последнее состояние подключения
      const unsigned long postingInterval = 10*1000; // задержка между
обновлениями данных (в миллисекундах)

      void setup() {
          int i;

          pinMode(3, OUTPUT); // LED 1
          pinMode(4, INPUT); // Кнопка PD4
          pinMode(5, OUTPUT); // LED 2
          pinMode(7, OUTPUT); // LED 3

          // инициализация SPI:
          SPI.begin();
          SPI.setBitOrder(MSBFIRST); // MSBFIRST - приоритет старшего бита,
LSBFIRST - приоритет младшего бита
          SPI.setClockDivider(SPI_CLOCK_DIV4); //установка делителя частоты
SPI (16Mhz/4 = 4Mhz)
          digitalWrite(SS, HIGH);

          // инициализация UART:
          Serial.begin(9600); //установка скорости передачи данных UART

          // Ожидание загрузки Ethernet-модуля
          delay(1000);
          buttonState = digitalRead(4);
```

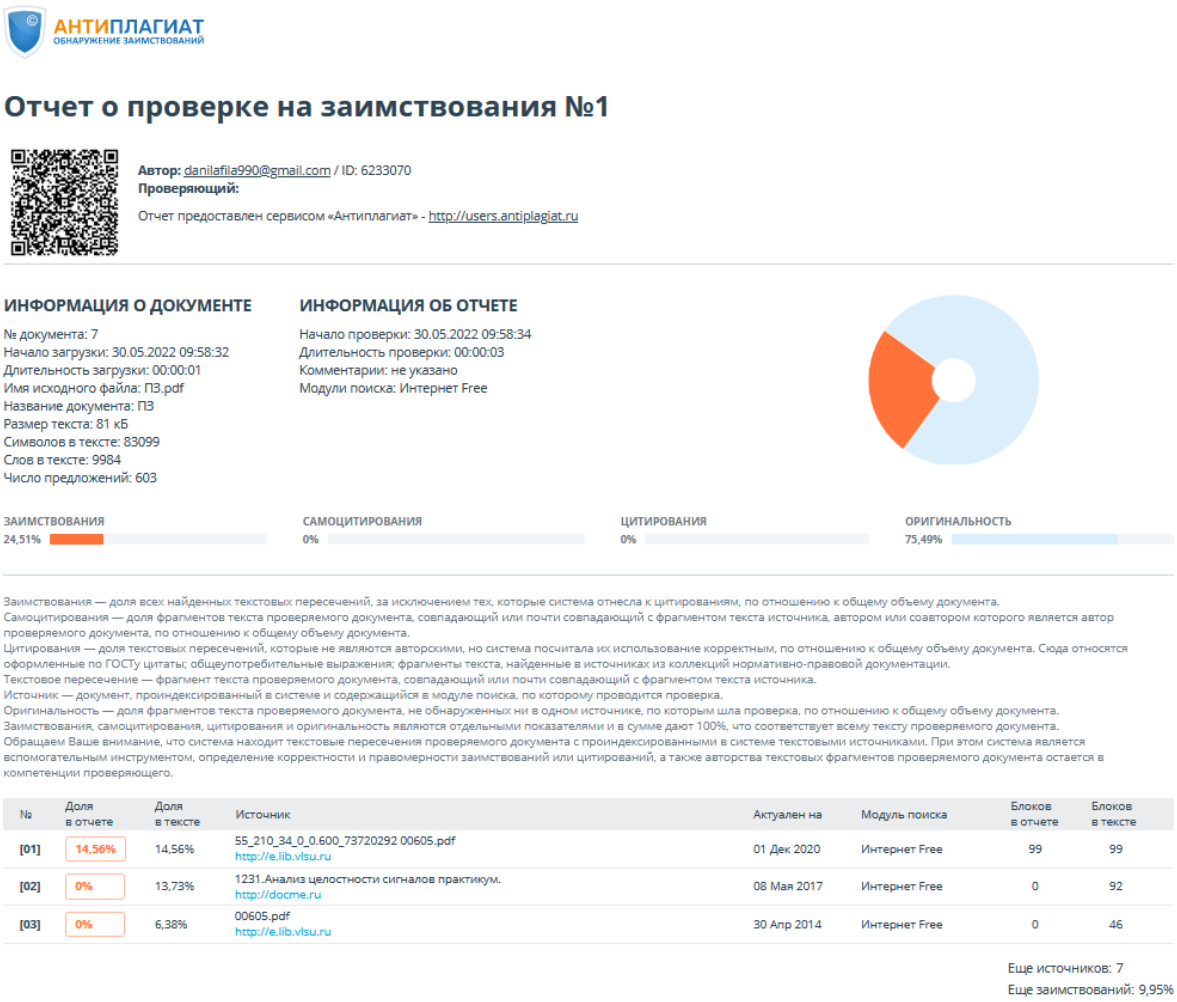
```
          // Проверка, нажата ли кнопка PD4 сразу после включения питания
          if (buttonState == 0){ // Если нажата, необходимо ввести имя сервера
и ключ шифрования
               for(ia=0;ia<100;ia++){
                   addr=ia; EEPROM.write(addr, 0); // Обнуляем EEPROM
               }

               Serial.print("Input server name: ");
               while(Serial.available() <= 0){ ;} // Ожидание ввода
               delay(1000); // Задержка для передачи в буфер

               for(i = 0; i < len; i++) //считывание ключа с последовательного
порта при запуске
               {
                   while(Serial.available() == 0){} // ожидание начала передачи
данных
                   key[i]  =  Serial.read();  //  считывание  ключа  шифрования
посимвольно
               }

               i=0;
               while(Serial.available() > 0 && addr_len < 100) {
                   server[i] = Serial.read(); addr_len=i; i++; // Чтение адреса
сервера
               }

               for(i = 0; i < 16; i++)
                   EEPROM.write(i, key[i]); // Запись ключа в EEPROM
               for(i = 0; i < addr_len; i++)
                   EEPROM.write(i + 16, server[i]); // Запись адреса в EEPROM

               Serial.println();
          }
          else {
               for(i = 0; i < 16; i++)
                   key[i] = EEPROM.read(i); // чтениею ключа с EEPROM

               for(i = 0; i < 100; i++)
                   server[i]=EEPROM.read(i  +  16);  //  Чтение  имени  сервера  с
EEPROM
          }

          // инициализация состояния шифратора
          beltCTRStart(&stateEncr, key, len, iv);

          // инициализация состояния дешифратора
          beltCTRStart(&stateDecr, key, len, iv);

          Serial.print("Server name: ");
          Serial.println(server);
          // Определение IP параметров с помощью DHCP
          if (Ethernet.begin(mac) == 0) {
               Serial.println("Failed to configure Ethernet using DHCP");
               // initialize the ethernet device not using DHCP:
               Ethernet.begin(mac, ip, myDns, gateway, subnet);
          }
          // start the Ethernet connection using a fixed IP address and DNS
server:
          // print the Ethernet board/shield's IP address:
          Serial.print("My IP address: ");
          Serial.println(Ethernet.localIP());
          Serial.print("My DNS address: ");
```

```
        Serial.println(Ethernet.dnsServerIP());
        Serial.print("Gateway address: ");
        Serial.println(Ethernet.gatewayIP());
        Serial.print("SubnetMask: ");
        Serial.println(Ethernet.subnetMask());
    }


    void loop() {
        // Получены ли данные по сети
        if (client.available()) {

            char c = client.read(); // чтение данных
            beltCTRStepE(&c, 1, &stateDecr); //расшифровка байта данных
            Serial.print(c); // отладочный вывод

            buf[ii-1]=c; ij=ii;
            if(c=='\n')
                ii=0;
            ii++;
        }

        // обработка данных от сервера
        if (!client.connected() && lastConnected) {
            Serial.println();
            Serial.print("disconnecting");

            if(ij<=1) goto lab;
            for(ii=0;ii<ij;ii++)
                Serial.write(buf[ii]);
            Serial.println();
            if( buf[1]=='1') {digitalWrite(3,HIGH);}
            if( buf[1]=='0') {digitalWrite(3,LOW);}
            if( buf[2]=='1') {digitalWrite(5,HIGH);}
            if( buf[2]=='0') {digitalWrite(5,LOW);}
            if( buf[3]=='1') {digitalWrite(7,HIGH);}
            if( buf[3]=='0') {digitalWrite(7,LOW);}
            lab:
            client.stop();
        }

        // Если клиент не подключен и прошло 10 секунд с последнего
подключения
        // Тогда подключаемся снова и отправляем данные
        if(!client.connected()  &&  (millis()  -  lastConnectionTime  >
postingInterval)) {
            sensors.requestTemperatures();
            int temp=sensors.getTempCByIndex(0);
            httpPost(&temp, 2); // шифрование и отправка данных от сенсора
        }
        if(  !client.connected()  &&  (millis()  -  lastConnectionTime  >
postingInterval/2)) {
            httpGet(); // опрашиваем сервер
        }
        // сохранить последнее состояние (подключен/не подключен)
        lastConnected = client.connected();
    }


    // Шифрование и отправка данных на сервер
    void httpPost(char * data, size_t len) {
        if (client.connect(server, 80)) {
            Serial.println("connecting (POST)...");
```

```
        // send the HTTP PUT request:
        client.println("POST /command.php HTTP/1.1");
        client.println("User-Agent: arduino-ethernet");
        client.println("Connection: close");
        client.println();

        for(int i = 0; i < len; i++) {
            // шифрование
            beltCTRStepE(enc_data + i, 1, &stateEncr);

            // отправка данных
            client.print(data[i]);
        }

        // note the time that the connection was made:
        lastConnectionTime = millis();
    }
    else {
        // if you couldn't make a connection:
        Serial.println("connection failed");
        Serial.println("disconnecting.");
        client.stop();
    }
}

// опрос сервера
void httpGet() {
    if (client.connect(server, 80)) {
        Serial.println("connecting (GET)...");

        client.println("GET /command.php HTTP/1.1");
        client.println("Connection: close");
        client.println();
    }
}
```

ПРИЛОЖЕНИЕ Б
(обязательное)
Отчёт о проверке на заимствование

Отчёт о проверке на заимствование материалов пояснительной записки представлен на рисунке Б.1.



АНТИПЛАГИАТ
ОБНАРУЖЕНИЕ ЗАИМСТВОВАНИЙ

Отчет о проверке на заимствования №1

Автор: danilafila990@gmail.com / ID: 6233070
Проверяющий:

Отчет предоставлен сервисом «Антиплагиат» - http://users.antiplagiat.ru

ИНФОРМАЦИЯ О ДОКУМЕНТЕ

№ документа: 7
Начало загрузки: 30.05.2022 09:58:32
Длительность загрузки: 00:00:01
Имя исходного файла: ПЗ.pdf
Название документа: ПЗ
Размер текста: 81 кБ
Символов в тексте: 83099
Слов в тексте: 9984
Число предложений: 603

ИНФОРМАЦИЯ ОБ ОТЧЕТЕ

Начало проверки: 30.05.2022 09:58:34
Длительность проверки: 00:00:03
Комментарии: не указано
Модули поиска: Интернет Free

ЗАИМСТВОВАНИЯ	САМОЦИТИРОВАНИЯ	ЦИТИРОВАНИЯ	ОРИГИНАЛЬНОСТЬ
24,51%	0%	0%	75,49%

Заимствования — доля всех найденных текстовых пересечений, за исключением тех, которые система отнесла к цитированиям, по отношению к общему объему документа.
Самоцитирования — доля фрагментов текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника, автором или соавтором которого является автор проверяемого документа, по отношению к общему объему документа.
Цитирования — доля текстовых пересечений, которые не являются авторскими, но система посчитала их использование корректным, по отношению к общему объему документа. Сюда относятся оформленные по ГОСТу цитаты; общеупотребительные выражения; фрагменты текста, найденные в источниках из коллекций нормативно-правовой документации.
Текстовое пересечение — фрагмент текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника.
Источник — документ, проиндексированный в системе и содержащийся в модуле поиска, по которому проводится проверка.
Оригинальность — доля фрагментов текста проверяемого документа, не обнаруженных ни в одном источнике, по которым шла проверка, по отношению к общему объему документа.
Заимствования, самоцитирования, цитирования и оригинальность являются отдельными показателями и в сумме дают 100%, что соответствует всему тексту проверяемого документа.
Обращаем Ваше внимание, что система находит текстовые пересечения проверяемого документа с проиндексированными в системе текстовыми источниками. При этом система является вспомогательным инструментом, определение корректности и правомерности заимствований или цитирований, а также авторства текстовых фрагментов проверяемого документа остается в компетенции проверяющего.

№	Доля в отчете	Доля в тексте	Источник	Актуален на	Модуль поиска	Блоков в отчете	Блоков в тексте
[01]	14,56%	14,56%	55_210_34_0_0.600_73720292 00605.pdf http://e.lib.vlsu.ru	01 Дек 2020	Интернет Free	99	99
[02]	0%	13,73%	1231.Анализ целостности сигналов практикум. http://docme.ru	08 Мая 2017	Интернет Free	0	92
[03]	0%	6,38%	00605.pdf http://e.lib.vlsu.ru	30 Апр 2014	Интернет Free	0	46

Еще источников: 7
Еще заимствований: 9,95%

Рисунок Б.1 – Отчёт о проверке на заимствование

_____ Филипцов Д. А.