МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Политехнический институт Факультет приборостроения, информационных технологий и электроники	Кафедра «Информационная безопасность систем и технологий»
Специальность 10.05.02 –	Информационная безопасность телекоммуникационных систем
Специализация –	Разработка защищенных телекоммуникационных систем

ДИПЛОМНЫЙ ПРОЕКТ

на тему

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, РЕАЛИЗУЮЩЕЕ ФУНКЦИИ КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ В МАЛОГАБАРИТНОЙ ВСТРАИВАЕМОЙ АППАРАТУРЕ ПЕРЕДАЧИ ДАННЫХ

Студентка _____ Тарханова Юлия Олеговна

Руководитель ведущий инженер

АО «ПНИЭИ» _____ Прокуров А.С.

Нормоконтролёр к.т.н., доцент _____ Иванов А.П.

Рецензент начальник НИС-103

АО «ПНИЭИ» _____ Акмаев А.Ж.

Работа допущена к защите (протокол заседания кафедры от _____ № ___)

Заведующий кафедрой _____ С.Л. Зефиров

Работа защищена с отметкой _____(протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____ А.Г. Фатеев

Пенза 2018

Оглавление

Введение 4
1 Описание рассматриваемой проблемы 6
1.1 Актуальность разработки программного обеспечения, реализующего функции криптографической защиты информации в малогабаритном встраиваемом устройстве передачи данных 6
1.2 Обеспечение информационной безопасности циркулирующей в канале связи информации 8
1.3 Основные способы реализации методов криптографической защиты информации 9
1.4 Требования, предъявляемые к разрабатываемым средствам криптографической защиты информации 14
2 Выбор и обоснование алгоритма криптографической защиты информации . 16
2.1 Обзор существующих методов шифрования информации 16
2.2 Анализ ГОСТ 28147-89 с точки зрения криптографической стойкости 19
2.3 Алгоритм криптографического преобразования ГОСТ 28147-89 20
3 Описание встраиваемой аппаратуры передачи данных 36
3.1 Обзор основных сведений и технических характеристик устройства 36
3.2 Описание структурной схемы малогабаритной встраиваемой аппаратуры передачи данных 37
3.3Аппаратно-программные возможности микроконтроллера STM32F415OGY6 39
4 Разработка программного обеспечения 42
4.1 Описание языка программирования, выбранного для разработки программного обеспечения, реализующего функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных 42
4.2 Программное обеспечение, используемое для разработки программы, реализующей функции криптографической защиты информации 43

4.3 Разработка алгоритма работы программного обеспечения, реализующего функции криптографической защиты информации 46
4.4 Описание функционирования программного обеспечения, реализующего функции криптографической защиты информации 47
4.5 Тестирование программного обеспечения, реализующего функции криптографической защиты информации, на соответствие предъявленным требованиям 50
5 Вредные психофизиологические факторы, влияющие на сердечно-сосудистую систему специалиста по защите информации. Методы и средства физической культуры, снижающие их воздействие 54
5.1 Вредные психофизиологические факторы, влияющие на сердечно-сосудистую систему специалиста по защите информации 54
5.2 Методы и средства физической культуры, снижающие воздействие психофизиологических факторов, влияющих на сердечно-сосудистую систему специалиста по защите информации 57
Заключение 59
Список использованных источников 60
Приложение А. Схема электрическая структурная малогабаритной встраиваемой аппаратуры передачи данных 62
Приложение Б. Исходный код программного обеспечения, реализующего функции криптографической защиты информации 64
Приложение В. Протокол проверки ВКР в системе «Антиплагиат» 74

Введение

Фундаментом всех разрабатываемых средств криптографической защиты информации являются криптографические методы защиты информации. Средства криптографической защиты информации реализуют криптографические алгоритмы преобразования информации, подразделяясь на программные, аппаратные или программно-аппаратные комплексы, средства и системы. Данные средства и системы необходимы для обеспечения безопасности информации, циркулирующей в каналах связи, обрабатываемой и хранимой в технических средствах и системах, включая устройства шифрования данных.

Средства криптографической защиты информации, реализованные в качестве малогабаритной встраиваемой аппаратуры передачи данных с программно-аппаратной реализацией криптографических функций, изготавливаются в виде плат с разъёмом для подключения в каналообразующую аппаратуру, а также в виде отдельных блоков с внешними интерфейсами для подключения к радиостанциям. Рассматриваемые средства криптографической защиты информации обладают такими преимуществами как: небольшой размер, низкое энергопотребление, возможность разработки для них программного обеспечения, осуществляющего криптографическое закрытие информации на достаточно высокой скорости, предотвращающей потерю значимости данных, передаваемых по каналу связи.

Целью выпускной квалификационной работы является создание программного обеспечения, реализующего функции криптографической защиты информации программного обеспечения в малогабаритной встраиваемой аппаратуре передачи данных. Разработанное программное обеспечение должно обеспечивать криптографическую обработку информационных пакетов, предназначенных для приёма/передачи через канальных стык.

В выпускной квалификационной работе должны быть рассмотрены следующие вопросы:

– описание структурной схемы малогабаритной встраиваемой аппаратуры передачи данных;

– описание функционирования реализующего функции криптографической защиты информации программного обеспечения;

– разработка алгоритмов работы функциональных частей реализующего функции криптографической защиты информации программного обеспечения;

– разработка и отладка реализующего функции криптографической защиты информации программного обеспечения;

– тестирование реализующего функции криптографической защиты информации программного обеспечения на соответствие заявленным требованиям;

– вредные психофизиологические факторы, влияющие на сердечно-сосудистую систему специалиста по защите информации, методы и средства физической культуры, снижающие их воздействие.

1 Описание рассматриваемой проблемы

1.1 Актуальность разработки программного обеспечения, реализующего функции криптографической защиты информации в малогабаритном встраиваемом устройстве передачи данных

Целью данной выпускной квалификационной работы является разработка программного обеспечения, реализующего функции криптографической защиты информации в малогабаритном встраиваемом устройстве передачи зашифрованной информации. Данное устройство предполагается к использованию в портативной аппаратуре связи низового звена управления.

Портативное оборудование (portable equipment): техническое средство радиосвязи и/или вспомогательное оборудование, носимые пользователем (располагаемые на теле пользователя) при их применении, получающие электропитание от собственной встроенной батареи [1]. В низовых звеньях управления циркулирует информация, не содержащая сведений, составляющих государственную тайну. Но, не смотря на это, для передаваемой информации должны сохраняться такие качественные характеристики (свойства) как: конфиденциальность, целостность, другими словами – достоверность, и доступность. Осуществление криптографической защиты информации, передаваемой по каналу связи, – главная задача рассматриваемого устройства передачи информации. Функциональное выполнение данной задачи возлагается на реализуемое в данном дипломном проекте программное обеспечение.

Разрабатываемое программное обеспечение, должно реализовывать функции криптографической защиты информации и предполагается к применению в малогабаритном встраиваемом устройстве передачи данных, обуславливаемое недолговременной актуальностью информации циркулирующей в канале связи. Актуальность передаваемых данных составляет от 12 часов до 1-2 суток. Таким образом, возникает необходимость в создании программного обеспечения, осуществляющего криптографическое

закрытие информации на достаточно высокой скорости, предотвращающей потерю значимости передаваемых данных.

Основные требования к криптографическому преобразованию информации:

– сложность и стойкость криптографического преобразования данных должны выбираться в зависимости от объема и степени секретности данных;

– надежность закрытия должна быть такой, чтобы секретность не нарушалась даже в том случае, когда злоумышленнику становится известен метод шифрования;

– метод закрытия, набор используемых ключей и механизм их распределения не должны быть слишком сложными;

– выполнение процедур прямого и обратного преобразований должно быть формальны, эти процедуры не должны зависеть от длины сообщений;

– ошибки, возникающие в процессе преобразования, не должны распространяться по системе;

– вносимая процедурами защиты избыточность должна быть минимальной [2].

Потребность в разработке нового программного обеспечения для устройств передачи зашифрованной информации обусловлена научно-техническими достижениями в области технологии производства элементной базы с точки зрения её миниатюризации, энергоэффективности, производительности и функциональности, а также с быстрым устареванием и прекращением поддержки производителем вычислительных устройств, которые всего несколько лет назад считались лучшими по быстродействию, растет. Новое программное обеспечение должно быть конкурентоспособным, соответствовать возможностям современных устройств и требованиям заказчика.

1.2 Обеспечение информационной безопасности циркулирующей в канале связи информации

Как было отмечено выше, малогабаритное встраиваемое устройство передачи зашифрованной информации предполагается к использованию в портативной аппаратуре связи военнослужащих нижнего тактического звена управления. Таким образом, для информации, передаваемой по каналу связи, присущи следующие особенности:

– данная информация не может быть отнесена к сведениям, составляющим государственную тайну, но имеет ценность для потенциального противника, который заинтересован в нарушении свойств ее безопасности, в частности конфиденциальности и целостности;

– стратегическую важность информация сохраняет в течение небольшого отрезка времени от нескольких часов до пары суток;

– утечка информации по каналу с использованием побочных электромагнитных излучений и наводок крайне маловероятна;

– высока вероятность попыток потенциального противника дешифровать передаваемую информацию, а также попыток перехвата ключевой информации;

– высока вероятность хищения устройств передачи информации и последующих попыток раскрытия ключевой информации.

Исходя из вышеперечисленного, средства криптографической защиты информации для военных низового звена управления должны разрабатываться с использованием криптографических алгоритмов, обладающих достаточной математической стойкостью. Не смотря на то, что актуальность передаваемых данных составляет не более пары суток, сохранять целостность криптографической системы необходимо на протяжении всего времени ее использования. Даже одного нарушения целостности системы может хватить для компрометации ключевых данных и последующего раскрытия информации

1.3 Основные способы реализации методов криптографической защиты информации

1.3.1 Аппаратное шифрование, принцип построения аппаратных средств криптографической защиты информации.

Аппаратное средство криптографической защиты информации – специализированный блок, компонент средства вычислительной техники или отдельное устройство, выполняющее шифрование информации [3].

Шифрование, реализованное аппаратно, должно быть физически исполнено на плате. При проектировании аппаратных средств криптографической защиты должны быть соблюдены некоторые условия и требования. Базовая схема аппаратного средства криптографической защиты информации приведена на рисунке 1.

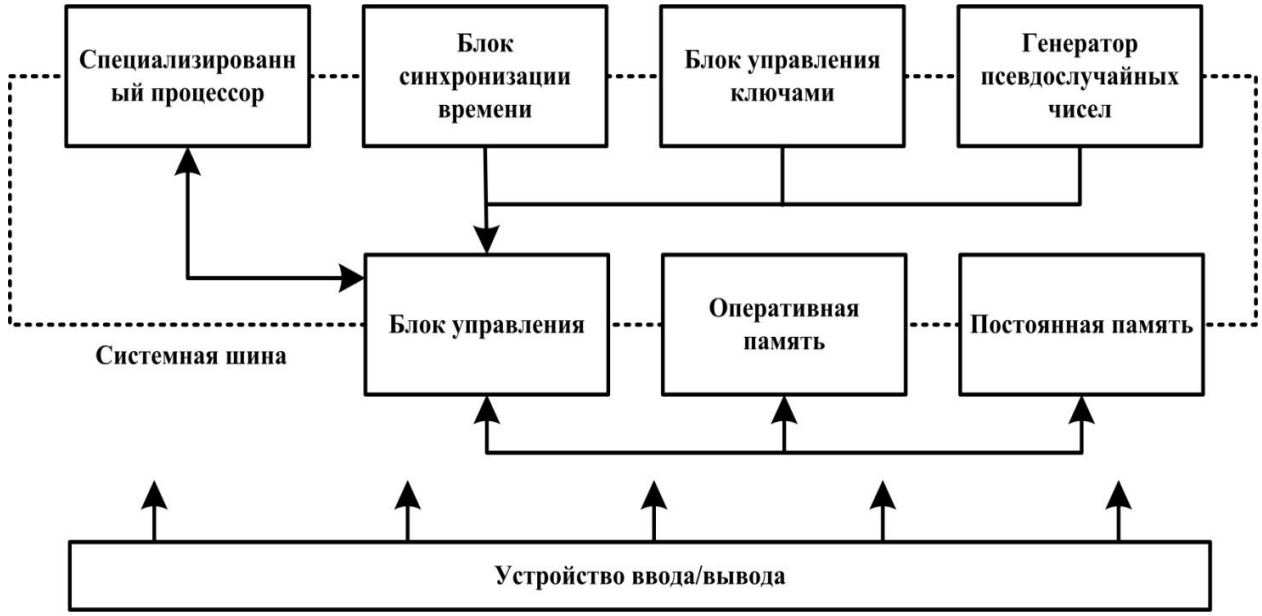


Рисунок 1 – Базовая схема аппаратного средства криптографической защиты информации

Блок управления координирует выполнение функций компонентов СКЗИ, коммутирует устройства и внешние потоки, управляет данными, которые передаются по системной шине, показанной на рисунке 1 пунктиром.

В аппаратном средстве криптографической защиты информации память делится на две части:

– оперативная память, которая содержит в себе программное обеспечение средства криптографической защиты информации, различные программируемые компоненты, переменные и другое;

– постоянная память, которая содержит в себе реализующий защитные функции набор команд, а также загрузку программного обеспечения средства криптографической защиты информации.

Также существуют два блока памяти, являющиеся дополнительными:

– память журнала;

– память хранения ключей.

В аппаратных средствах криптографической защиты информации для ускорения вычислений и выполнения большого количества криптографических операций применяются специализированные процессоры. Данные процессоры являются программируемыми блоками или блоками с предустановленным списком команд.

Для получения псевдослучайных последовательностей применяется генератор случайных чисел. Он должен отвечать требованиям криптографического алгоритма, который используется средством криптографической защиты информации. В аппаратной реализации криптографической защиты информации используются генераторы случайных чисел, которые базируются на разнообразных физических процессах.

Блок управления ключами принимает, обрабатывает и выдает ключевую информацию, и, при необходимости, функции идентификации и аутентификации.

В операциях контроля функционирования средства криптографической защиты информации, при синхронизации потока данных, а также для процедур расчета и согласования операций применяется блок синхронизации времени.

Обработка сигналов, которые исходят от внешних устройств, осуществляется устройством ввода-вывода. В аппаратных средствах криптографической защиты информации для устройств ввода-вывода по-

настоящему важным является обеспечение возможности взаимодействия со следующими устройствами:

– системной шиной компьютерной системы или приемо-передатчиком линии передачи информации, данный пункт крайне важен для реализации сквозного шифрования;

– устройствами хранения ключевой информации, которые используются в задачах идентификации и/или аутентификации и для контроля за функционированием аппаратного средства криптографической защиты информации;

– контроллерами жестких дисков и устройствами резервного копирования информации, выполнение данного пункта позволит реализовать безопасное хранение данных и их резервных копий.

Основными преимуществами аппаратных шифраторов являются:

– неизменность алгоритма шифрования;

– при генерации криптографических ключей используется аппаратных датчик случайных чисел;

– возможна прямая загрузка ключей шифрования в специализированный процессор аппаратного средства криптографической защиты информации;

– ключи хранятся в памяти криптопроцессора;

– процедуры идентификации и аутентификации происходят до загрузки операционной системы;

– возможна высокая скорость шифрования и расшифрования информации.

Также существуют и другие достоинства аппаратной реализации:

– возможность использования средства криптографической защиты в принципиально важных узлах автоматизированных систем, так как аппаратная реализация обладает высокой надежностью;

– можно реализовать в виде отдельного блока для более гибкого построения топологии защищенной автоматизированной системы в общем случае;

– возможность искажения ключевой информации исключается, что стабилизирует систему.

1.3.2 Программное шифрование, принцип построения программных средств криптографической защиты информации.

Программные средства шифрования реализовывают криптографические алгоритмы на различных языках программирования, высокоуровневых или низкоуровневых. Зачастую требуется выполнение комплекса вычислительных операций стандартными аппаратными средствами компьютерной системы для функционирования подобных средств криптографической защиты.

Способы программной реализации криптографических алгоритмов включают похожие базовые элементы. Схема программного СКЗИ представлена на рисунке 2.

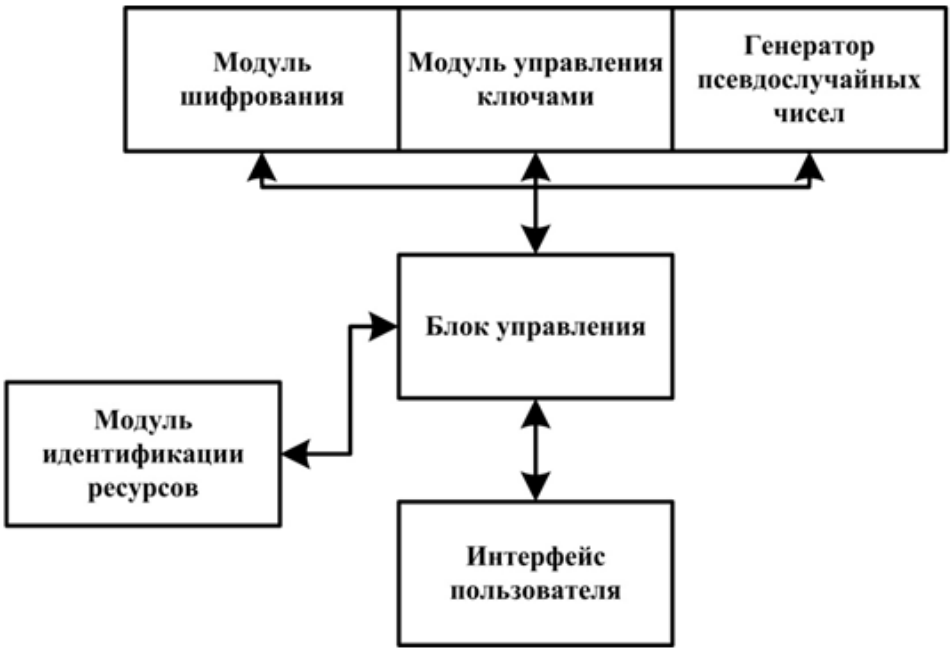


Рисунок 2 – Схема программного СКЗИ

Блок управления коммутирует модули, внешние потоки информации и внешние команды, координирует выполнение функций компонентов средства криптографической защиты информации.

Задачи обработки ключевой информации и идентификацию и/или аутентификацию пользователей выполняет модуль управления ключами.

Также в качестве элемента защитной подсистемы может дополнительно использоваться идентификация устройств.

В программных средствах криптографической защиты информации генератор случайных чисел реализуется программно. Используются разнообразные алгоритмы генерации псевдослучайных битовых последовательностей.

С помощью интерфейса происходит взаимодействие с пользователем. Интерфейс позволяет контролировать функционирование программной реализацией алгоритма.

Технология реализации криптографических алгоритмов программными средствами имеет следующую специфику:

– необходимо дополнительно контролировать качество функционирования, так как более вероятно нарушение программного средства криптографической защиты информации, чем подобного ему аппаратного средства;

– с помощью введения избыточности возможен контроль ошибок в закрытом тексте при шифровании;

– необходимо обеспечить безопасное хранение ключей;

– возможно увеличение или уменьшение масштаба и дополнение средства криптографической защиты информации другими программными блоками, а также изменение уже используемых;

– при шифровании информации в частных целях для облегчения общей схемы защиты существует возможность использования программного средства криптографической защиты информации с открытым кодом.

Таким образом, программное средство криптографической защиты информации отличает способность использования в распределенных и глобальных системах, более гибкая реализация, способность масштабирования и высокая мобильность [3].

Программно-аппаратные комплексы криптографической защиты информации представляют собой наиболее сложную и эффективную разновидность средств обеспечения информационной безопасности с использованием криптографических алгоритмов.

Под программно-аппаратными средствами криптографической защиты информации понимаются специальным образом организованные комплексы, содержащие взаимосвязанные программные и аппаратные блоки [3].

1.4 Требования, предъявляемые к разрабатываемым средствам криптографической защиты информации

Для современных криптографических систем защиты информации сформулированы следующие общепринятые требования:

– зашифрованное сообщение должно поддаваться чтению только при наличии ключа;

– число операций, необходимых для определения использованного ключа шифрования по фрагменту шифрованного сообщения и соответствующего ему открытого текста, должно быть не меньше общего числа возможных ключей;

– число операций, необходимых для расшифровывания информации путем перебора всевозможных ключей, должно иметь строгую нижнюю оценку и выходить за пределы возможностей современной аппаратуры;

– знание алгоритма шифрования не должно влиять на надежность защиты;

– незначительное изменение ключа должно приводить к существенному изменению вида зашифрованного сообщения даже при использовании одного и того же ключа;

– структурные элементы алгоритма шифрования должны быть неизменными;

– дополнительные биты, вводимые в сообщение в процессе шифрования, должен быть полностью и надежно скрыты в шифрованном тексте;

– в идеале длина шифрованного текста должна быть равной длине исходного текста;

– не должно быть простых и легко устанавливаемых зависимостей между ключами, последовательно используемыми в процессе шифрования;

– любой ключ из множества возможных ключей должен обеспечивать надежную защиту информации;

– алгоритм должен допускать как программную, так и аппаратную реализацию, при этом изменение длины ключа не должно вести к качественному ухудшению алгоритма шифрования.

Не для всех алгоритмов шифрования перечисленные требования выполняются полностью. В частности, требование отсутствия слабых ключей (ключей, которые позволяют злоумышленнику легче дешифровать зашифрованное сообщение) не выполняется для некоторых «старых» блочных шифров. Однако, все вновь разрабатываемые системы шифрования удовлетворяют перечисленным требованиям.

Стандартом современных алгоритмов симметричного шифрования являются несколько раундов шифрования с разными ключами, которые генерируются на основе одного общего ключа.

Надежный алгоритм шифрования должен удовлетворять двум свойствам: диффузии и конфузии.

Диффузия – каждый бит открытого текста должен влиять на каждый бит зашифрованного текста. Суть диффузии заключается в рассеянии статистических характеристик открытого текста внутри шифрованного текста.

Конфузия – отсутствие статистической взаимосвязи между ключом и зашифрованным текстом. Даже если противник определит какие-то статистические особенности зашифрованного текста, их будет недостаточно, чтобы получить информацию о ключе [4].

2 Выбор и обоснование алгоритма криптографической защиты информации

2.1 Обзор существующих методов шифрования информации

Криптографическая защита информации – это защита информации путем её преобразования с использованием криптографических алгоритмов.

Из всей совокупности криптографических методов защиты информации основным видом является шифрование. Под шифрованием понимается процесс преобразования открытой информации в зашифрованную информацию или процесс обратного преобразования зашифрованной информации в открытую с помощью ключа. Процесс преобразования открытой информации в закрытую получил название зашифрование, а процесс преобразования закрытой информации в открытую – расшифрование.

Ключ – это конкретное секретное состояние некоторых параметров алгоритма криптографического преобразования данных, обеспечивающее выбор одного преобразования из совокупности возможных для данного алгоритма преобразования [5].

Преобразование шифрования может быть симметричным и асимметричным относительно преобразования расшифрования. Это важное свойство определяет два класса криптосистем:

– симметричные (одноключевые) криптосистемы;

– асимметричные (двухключевые) криптосистемы (с открытым ключом).

Криптосистема представляет собой совокупность алгоритма, всех возможных открытых текстов, закрытых текстов и ключей шифрования.

В симметричных криптосистемах, также называемых криптосистемами с секретным (или закрытым) ключом, зашифрование и расшифрование осуществляется с помощью одного и того же ключа. Отправитель, зашифровывающий сообщение, должен передать ключ шифрования адресату, получающему и расшифровывающему зашифрованное сообщение.

Стойкость симметричного алгоритма определяется ключом, компрометация или раскрытие ключа означает, что кто угодно сможет зашифровывать и расшифровывать сообщения. Криптосистемы с секретным ключом получили название симметричных криптосистем из-за использования одного и того же ключа для зашифрования и расшифрования. Общая структура симметричной криптосистемы приведена на рисунке 3.

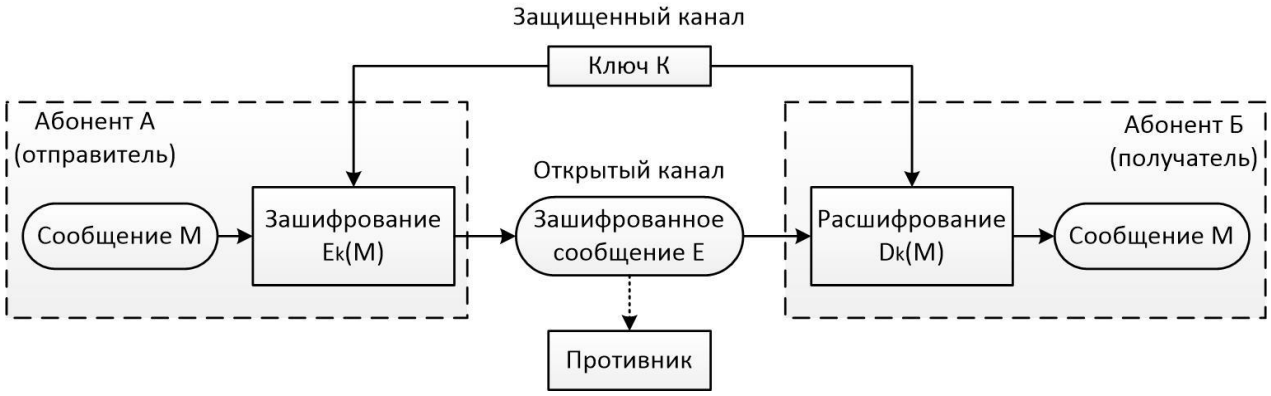


Рисунок 3 – Общая структура симметричной криптосистемы

С методом симметричного шифрования связаны следующие проблемы:

– необходимо часто менять секретные ключи, поскольку всегда существует риск их компрометации;

– достаточно сложно обеспечить безопасность секретных ключей при их генерировании, распространении и хранении.

Именно поэтому в криптографических системах, особенно в симметричных, большое внимание уделяют управлению ключами шифрования.

В ассиметричных криптосистемах, также называемых криптосистемами с открытым ключом, для зашифрования и расшифрования используются разные ключи, открытый и секретный, соответственно. Для каждого открытого существует секретный ключ, причем они отличаются друг от друга и взаимосвязаны. Секретный ключ не может быть (по крайней мере в течение разумного интервала времени) рассчитан по открытому ключу. Ассиметричные алгоритмы называются криптографическими алгоритмами с открытым ключом, потому что ключ зашифрования может быть открытым – кто угодно может использовать открытый ключ для зашифрования сообщений, но только

получатель с соответствующим секретным ключом сможет расшифровать сообщение. Общая структура асимметричной криптосистемы приведена на рисунке 4.

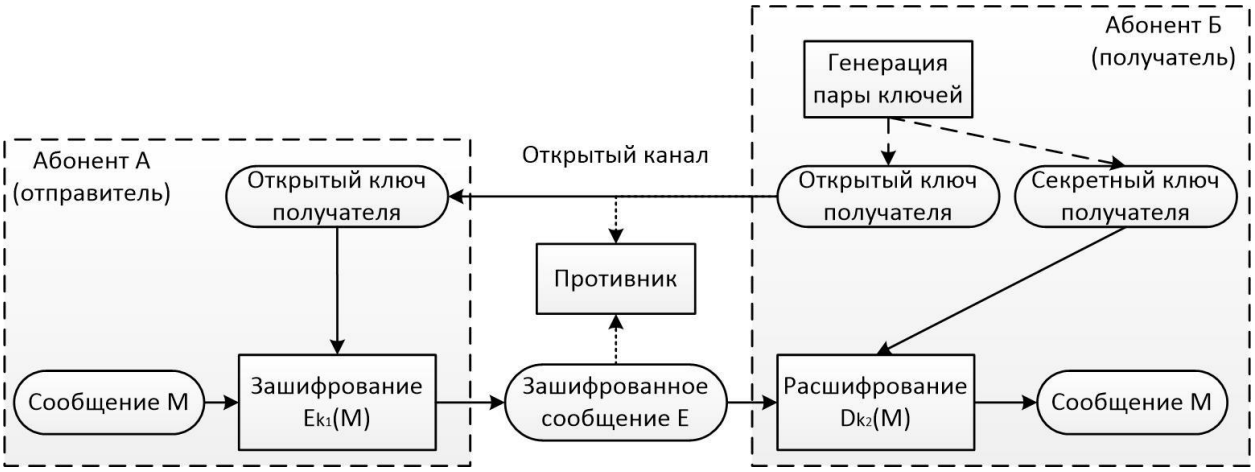


Рисунок 4 – Общая структура асимметричной криптосистемы

Асимметричные алгоритмы гораздо больше основаны на свойствах математических функций, чем алгоритмы симметричного шифрования, использующие в основном только операции перестановки и замены.

Преимуществом асимметричных криптосистем, по сравнению с симметричными, является возможность распространения открытых ключей шифрования по незащищенным каналам связи. Не имеет значения, получит ли кто-то кроме адресата открытый ключ, расшифровать сообщение, зашифрованное с помощью данного открытого ключа, сможет только получатель, у которого есть соответствующий секретный ключ. В асимметричных криптосистемах секретный ключ будет знать только получатель и никто кроме него.

Криптосистемы с открытым ключом решает задачу распределения ключей. Однако, как правило, криптографические алгоритмы с открытым ключом действуют в очень крупных алгебраических системах [6], т.е. связаны с выполнением весьма затратных, с точки зрения вычислений, алгебраических операций. С точки зрения быстродействия симметричные криптографические алгоритмы гораздо эффективнее.

2.2 Анализ ГОСТ 28147-89 с точки зрения криптографической стойкости

Одним из стандартов симметричного блочного шифрования является российский ГОСТ 28147-89 [5].

Первая публикация алгоритма ГОСТ 28147-89 [5] появилась в 1994 году, после чего значительное время не было найдено атак на него, которые приближались бы к практически осуществимым. Высокая стойкость алгоритма достигается за счет двух основных факторов. В первую очередь, это - большая длина ключа, которая составляет 256 бит. Вместе с синхропосылкой эффективная длина ключа увеличивается до 320 бит. Вторым фактором является наличие 32 раундов преобразований. После первых восьми из них достигается состояние, при котором изменение одного бита блока открытого текста влияет на все биты зашифрованного текста, и наоборот. Таким образом, имеется многократный запас стойкости.

ГОСТ 28147-89 [5] устойчив к методам линейного и дифференциального криптоанализа. Примечательно наличие алгоритма выработки имитовставки, который обеспечивает защиту от навязывания ложных данных.

В алгоритме имеется средство обеспечения защиты от атак скольжения и отражения. Для этого в последних восьми из тридцати двух раундов работы алгоритма используется обратный порядок ключей.

В 2010 году ГОСТ 28147-89 [5] был предложен в качестве мирового стандарта шифрования. В процессе исследований было выявлено, что алгоритм не обеспечивает уровня стойкости, требуемого ISO. Примененная атака позволяет взломать его в 2^8 раз быстрее, чем простым перебором при наличии 2^{64} известных открытых текста и 2^{64} памяти для хранения пар «открытый текст/зашифрованный текст».

В таблице 1 приведены данные об известных международному криптографическому сообществу результатах строго описанных и обоснованных атак на ГОСТ 28147-89 [5]. Сложность приводится в операциях зашифрования алгоритма ГОСТ 28147-89 [5], а память и материал указаны в блоках алгоритма (64 бита = 8 байт).

Таблица 1 – Данные об атаках на ГОСТ 28147-89 [5]

Атака	Трудоемкость	Память	Требуемый материал
Исобе	2^{224}	2^{64}	2^{32}
Динур-Данкельман-Шамир, FP, 2DMitM	2^{192}	2^{36}	2^{64}
Динур-Данкельман-Шамир, FP, low-memory	2^{204}	2^{19}	2^{64}
Динур-Данкельман-Шамир, Reflection, 2DMitM	2^{224}	2^{36}	2^{32}
Динур-Данкельман-Шамир, Reflection, 2DMitM	2^{236}	2^{19}	2^{32}
Полный перебор	2^{256}		

Несмотря на достаточно масштабный цикл исследований в области стойкости алгоритма ГОСТ 28147-89 [5], на данный момент не известно ни одной атаки, условия для осуществления которой являлись бы достижимыми при сопутствующих длине блока в 64 бита эксплуатационных требованиях. Вытекающие из параметров шифра (битовая длина ключа, битовая длина блока) ограничения на объем материала, который может быть обработан на одном ключе, существенно строже минимального объема, который необходим для осуществления любой из известных на данный момент атак. Следовательно, при выполнении существующих эксплуатационных требований ни один из предложенных к настоящему моменту методов криптоанализа ГОСТ 28147-89 не позволяет определять ключ с трудоемкостью меньшей полного перебора [7].

Можно сделать вывод, что отечественный стандарт шифрования все еще отличается высокой криптографической стойкостью и подходит для использования в рамках данного дипломного проекта.

2.3 Алгоритм криптографического преобразования ГОСТ 28147-89

2.3.1 Сеть Фейстеля.

Основные параметры алгоритма криптографического преобразования ГОСТ 28147-89 [5] следующие: размер блока составляет 64 бита, размер ключа – 256 бит, количество раундов – 32.

В основу алгоритма положена сеть Фейстеля, которая имеет следующую структуру. Входной блок делится на несколько подблоков равной длины, называемых ветвями. В случае если блок имеет длину 64 бита, используются две ветви по 32 бита каждая. Каждая ветвь обрабатывается независимо, после чего осуществляется циклический сдвиг всех ветвей влево. Такое преобразование выполняется несколько циклов (раундов). В случае двух ветвей каждый раунд имеет структуру, представленную на рисунке 4.

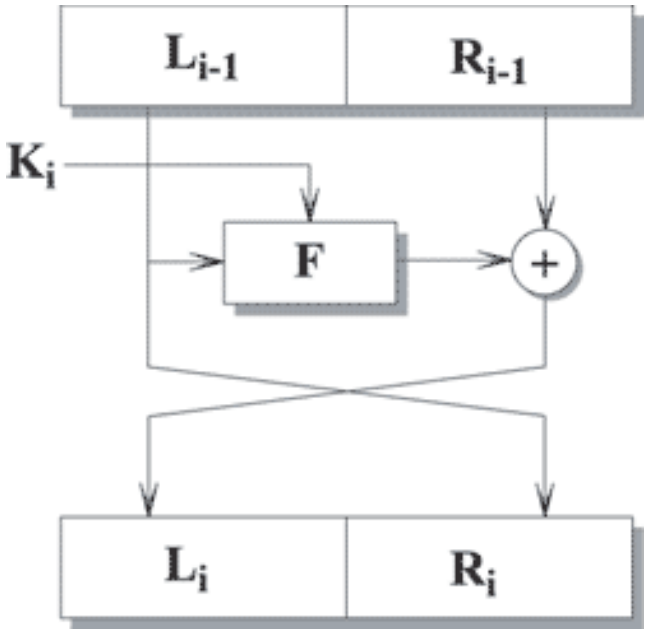


Рисунок 4 – Один раунд сети Фейстеля

Функция *F* называется образующей. Каждый раунд состоит из вычисления функции *F* для одной ветви и побитового выполнения операции *XOR* (побитовое сложение по модулю 2) результата *F* с другой ветвью. После этого ветви меняются местами.

Сеть Фейстеля является обратимой даже в том случае, если функция *F* не является таковой, так как для расшифрования не требуется вычислять F^{-1}. Для расшифрования используется тот же алгоритм, но на вход подается зашифрованный текст, и ключи используются в обратном порядке.

2.3.2 Режим гаммирования алгоритма криптографического преобразования ГОСТ 28147-89.

ГОСТ 28147-89 [5] предусматривает четыре режима работы:

– зашифрование/расшифрование в режиме простой замены;

– зашифрование/расшифрование в режиме гаммирования;

– зашифрование/расшифрование в режиме гаммирования с обратной связью;

– формирование имитовставки.

В данном дипломном проекте в соответствии с техническим заданием предполагается реализация алгоритма криптографического преобразования ГОСТ 28147-89 [5] в режиме гаммирования без обратной связи с выработкой имитовставки, используя ресурсы встраиваемой малогабаритной аппаратуры передачи данных.

Структурная схема, реализующая алгоритм зашифрования (расшифрования) в режиме гаммирования, изображена на рисунке 5.

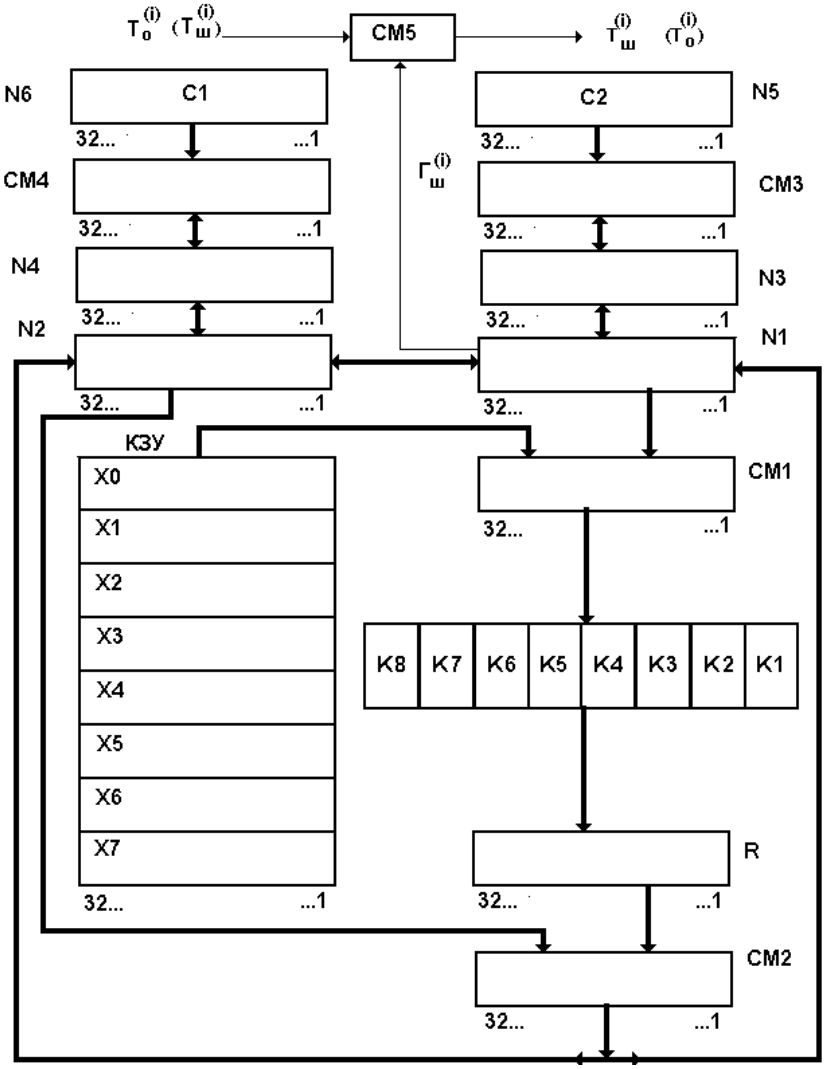


Рисунок 5 – Структурная схема, реализующая алгоритм зашифрования (расшифрования) в режиме гаммирования

Структурная схема содержит следующие элементы:

– To – открытые данные, представленные в виде блоков по 64 разряда каждый ($To^{(i)}$);

– Тш – шифрованные данные, представленные в виде блоков по 64 разряда каждый ($Тш^{(i)}$);

– КЗУ – ключевое запоминающее устройство на 256 бит, состоящее из восьми 32-разрядных накопителей (X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7);

– N_1, N_2, N_3, N_4 – четыре 32-разрядных накопителя;

– N_5, N_6 – два 32-разрядных накопителя с записанными в них постоянными заполнениями C_2, C_1;

– СМ1, СМ3 – два 32-разрядных сумматора по модулю 2^{32};

– СМ2 – 32-разрядный сумматор поразрядного суммирования по модулю 2;

– СМ3 – 32-разрядный сумматор по модулю (2^{32}-1);

– СМ5 – сумматор по модулю 2 без ограничений по разрядности;

– К – блок подстановки: восемь узлов замены $К_1$, $К_2$, $К_3$, $К_4$, $К_5$, $К_6$, $К_7$, $К_8$ каждый с памятью на 64 бита;

– R – 32-разрядный регистр циклического сдвига на 11 шагов в сторону старших разрядов.

2.3.3 Ключевая информация.

Ключи, определяющие заполнения КЗУ и таблиц блока подстановки К, являются секретными элементами. Заполнение таблиц блока подстановки К является долговременным ключевым элементом.

Поступающий на блок подстановки 32-разрядный вектор разбивается на восемь последовательно идущих 4-разрядный векторов, каждый из которых преобразуется в 4-разрядный вектор соответствующим узлом замены, представляющим собой таблицу из шестнадцати строк, содержащих по четыре бита заполнения в строке. Входной вектор определяет адрес строки в таблице, заполнение данной строки является выходным вектором. Затем 4-разрядные выходные векторы последовательно объединяются в 32-разрядный вектор.

КЗУ отведено 256 бит, в ГОСТ 28147-89[5] используется ключ длиной 256 бит. Ключ разбивается на восемь блоков по 32 бита, и каждый бит каждого блока последовательно вводится в накопитель *X* соответствующего порядка. То есть, 1-й бит ключа вводится в 1-й разряд накопителя X_0, 2-й — во 2-й разряд накопителя X_0, 33-й — в 1-й разряд накопителя X_1, 65-й — в 1-й разряд накопителя X_2, и так далее, 224-й бит ключа вводится в 1-й разряд накопителя X_7, 256-й бит ключа вводится в 32-й разряд накопителя X_7.

Считывание ключа из КЗУ выполняется в соответствии с выбранным режимом работы алгоритма.

2.3.4 Основной цикл криптографического преобразования.

Основной цикл криптографического преобразования по своей сути является оператором, определяющим преобразование 64 битового блока данных. Дополнительным параметром данного оператора является 32-битовый блок, в качестве которого используется один из элементов ключа.

Основной цикл криптографического преобразования приведен на рисунке 6.

Основной цикл криптографического преобразования выполняется в несколько шагов:

– определение исходных данных для основного цикла криптопреобразования, где N1 и N2 – преобразуемый 64-битовый блок данных, разделенный на две части по 32 бита обрабатываются как отдельные 32-битовые целые числа без знака, а key_n – 32-битовый элемент ключа;

– сложение с ключом. Младшая половина преобразуемого блока (N1) складывается по модулю 2^{32} с используемым на шаге элементом ключа, результат передается на следующий шаг;

– 32-битовое значение, полученное на предыдущем шаге, интерпретируется как массив из восьми 4-битовых векторов (CM1 = $CM1_0$, …, $CM1_7$);

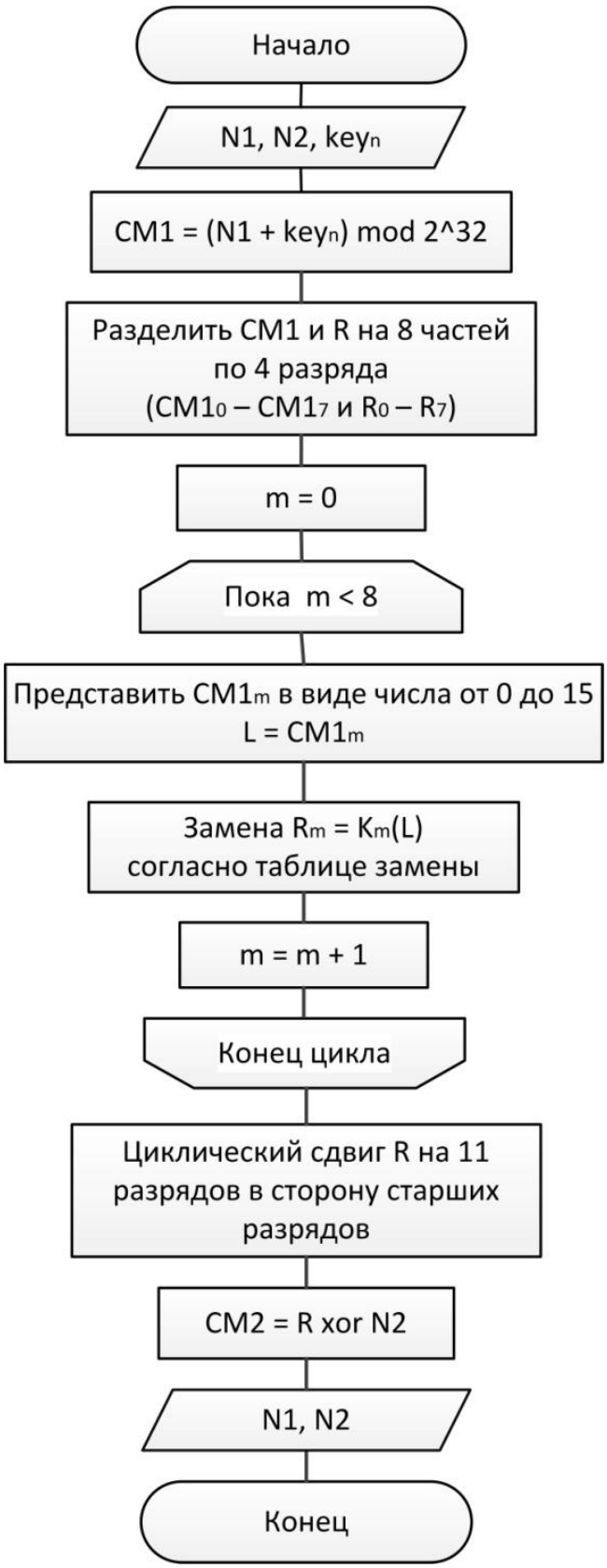


Рисунок 6 – Схема основного цикла шифрования

– поблочная замена - значение каждого из восьми 4-разрядных векторов меняется на новое, которое выбирается по таблице замен по правилам описанным ранее;

– циклический сдвиг на 11 бит влево: результат предыдущего шага сдвигается циклически на 11 бит в сторону старших разрядов и передается на следующий шаг;

– побитовое сложение: значение, полученное предыдущем шаге, побитно складывается по модулю 2 со старшей половиной (N2) преобразуемого блока;

– сдвиг по цепочке: младшая часть преобразуемого блока (N1) сдвигается на место старшей (N2), а на ее место (N1) помещается результат выполнения предыдущего шага;

– полученное значение преобразуемого блока возвращается как результат выполнения основного цикла криптографического преобразования.

2.3.5 Базовые циклы криптографического преобразования.

Базовые циклы построены из основных циклов криптографического преобразования. В процессе выполнения основного цикла используется только один элемент ключа, в то время как ключ ГОСТ содержит восемь таких элементов.

Базовые циклы заключаются в многократном выполнении основного цикла с использованием разных элементов ключа и отличаются друг от друга только порядком использования ключевых элементов. Порядок использования ключевых элементов при зашифровании и расшифровании представлен в Таблицах 2 и 3 соответственно.

Таблица 2 – Последовательность использования элементов ключа при зашифровании (32-З)

Раунд	1	2	3	4	5	6	7	8
Элемент ключа	K_0	K_1	K_2	K_3	K_4	K_5	K_6	K_7
Раунд	9	10	11	12	13	14	15	16
Элемент ключа	K_0	K_1	K_2	K_3	K_4	K_5	K_6	K_7
Раунд	17	17	19	20	21	22	23	24
Элемент ключа	K_0	K_1	K_2	K_3	K_4	K_5	K_6	K_7
Раунд	25	26	27	28	29	30	31	32
Элемент ключа	K_7	K_6	K_5	K_4	K_3	K_2	K_1	K_0

Таблица 3 – Последовательность использования элементов ключа при расшифровании (32-Р)

Раунд	1	2	3	4	5	6	7	8
Элемент ключа	$К_0$	$К_1$	$К_2$	$К_3$	$К_4$	$К_5$	$К_6$	$К_7$
Раунд	9	10	11	12	13	14	15	16
Элемент ключа	$К_7$	$К_6$	$К_5$	$К_4$	$К_3$	$К_2$	$К_1$	$К_0$
Раунд	17	17	19	20	21	22	23	24
Элемент ключа	$К_7$	$К_6$	$К_5$	$К_4$	$К_3$	$К_2$	$К_1$	$К_0$
Раунд	25	26	27	28	29	30	31	32
Элемент ключа	$К_7$	$К_6$	$К_5$	$К_4$	$К_3$	$К_2$	$К_1$	$К_0$

Схемы базовых циклов приведены на рисунках 7 и 8.

Каждая из них принимает в качестве аргумента и возвращает в качестве результата 64-битный блок данных, обозначенный на схеме, как два блока по 32 бита N1 и N2. Блок «Раунд шифрования key_n» представляет собой выполнение основного цикла криптографического преобразования для блоков N1 и N2 с использованием ключевого элемента key_n.

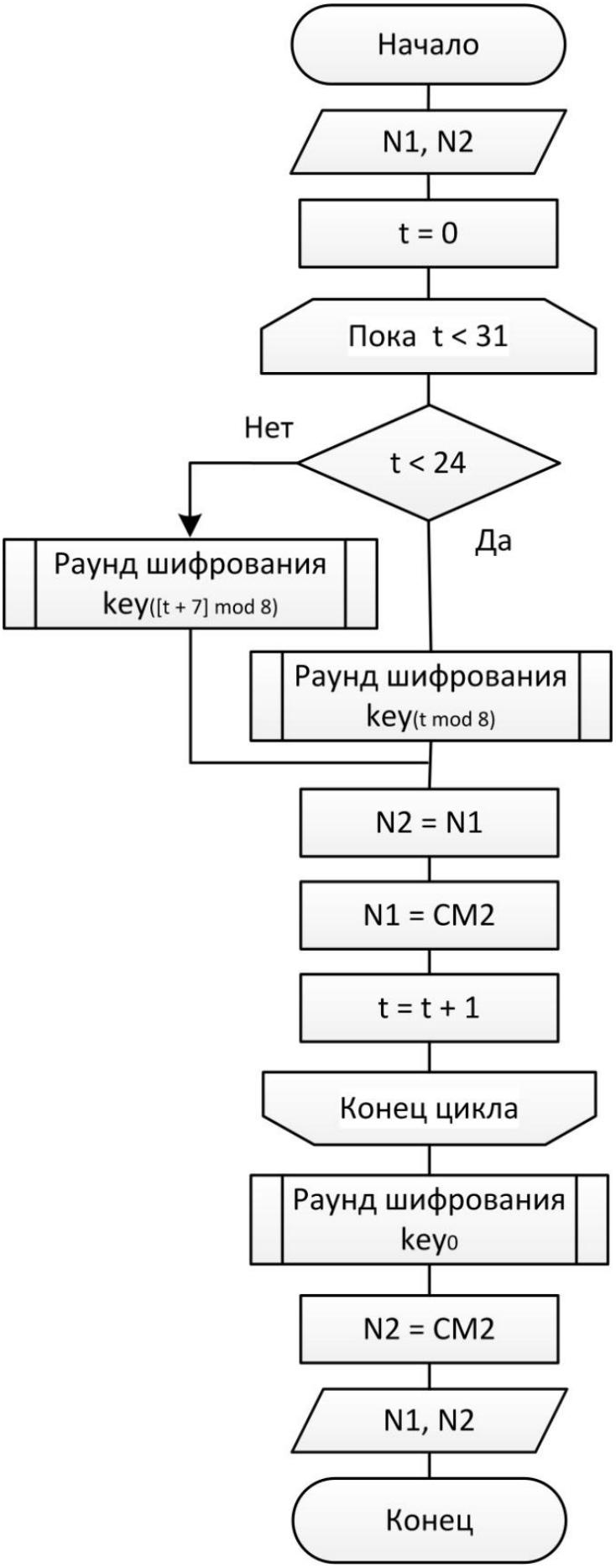


Рисунок 7 – Схема 32-х циклов зашифрования (32-З)

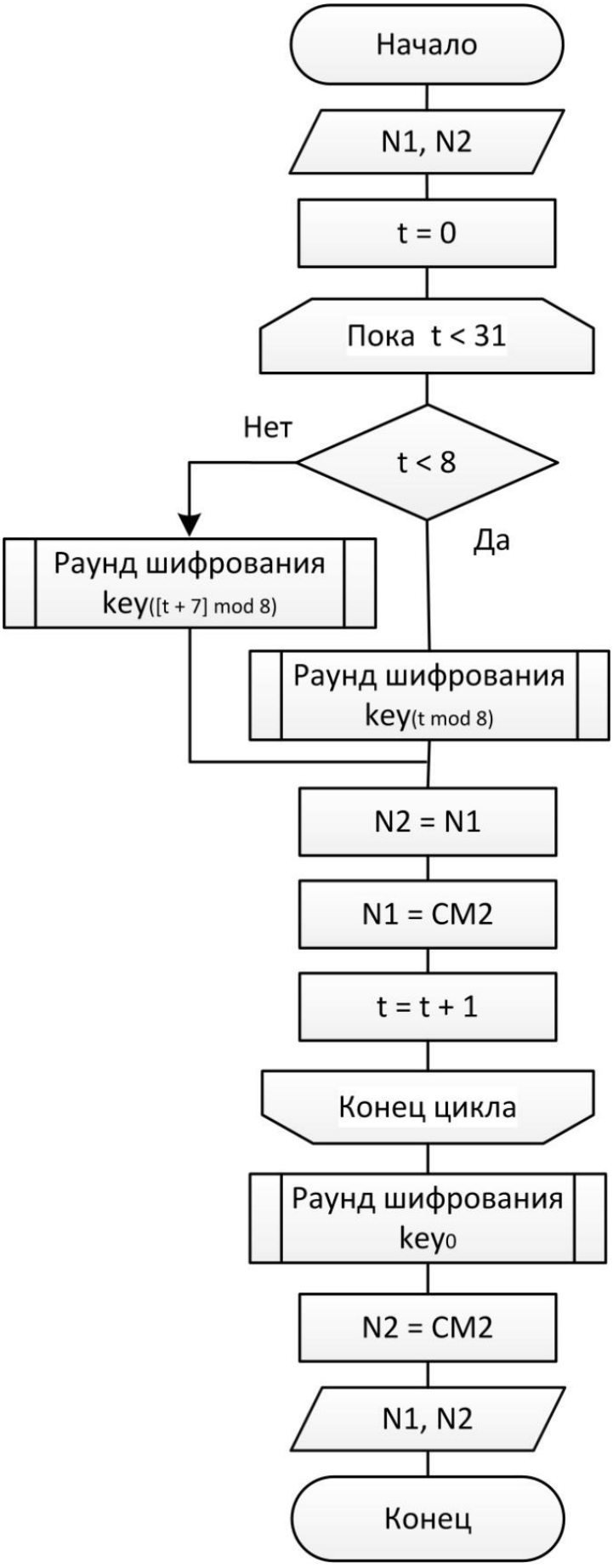


Рисунок 8 –Схема 32-х циклов расшифрования (32-Р)

2.3.6 Режим гаммирования алгоритма криптографического преобразования ГОСТ 28147-89.

Гаммирование – это наложение (снятие) на открытые (зашифрованные) данные криптографической гаммы, то есть последовательности элементов данных, вырабатываемых с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Для наложения гаммы при зашифровании и ее снятия при расшифровании должны использоваться взаимно обратные бинарные операции, например, сложение и вычитание по модулю 2^{64} для 64-битовых блоков данных. В ГОСТ 28147-89 для этой цели используется операция побитового сложения по модулю 2, поскольку она является обратной самой себе и, к тому же, наиболее просто реализуется аппаратно. При использовании гаммы для зашифрования результат зашифрования даже двух одинаковых блоков в одном массиве данных будет различным. Гамма вырабатывается блоками по 64 бита каждый, но использоваться может только часть блока размером, равным размеру шифруемого блока.

Гамма для этого режима получается следующим образом: с помощью некоторого алгоритмического рекуррентного генератора последовательности чисел вырабатываются 64-битные блоки данных, которые далее подвергаются зашифрованию в режиме простой замены, в результате получаются блоки гаммы.

Последовательность элементов гаммы для использования в режиме гаммирования однозначно определяется ключевыми данными и синхропосылкой. Для обратимости процедуры шифрования в процессах зашифрования и расшифрования должна использоваться одна и та же синхропосылка. Для шифрования двух различных массивов данных на одном ключе необходимо обеспечить использование различных синхропосылок. Это приводит к необходимости хранить или передавать синхропосылку по каналам связи вместе с зашифрованными данными, хотя в отдельных особых случаях

она может быть предопределена или вычисляться особым образом, если исключается шифрование двух массивов на одном ключе.

Характеристики рекуррентного генератора последовательности чисел:

– в 64-битовом блоке старшая и младшая части обрабатываются независимо друг от друга, фактически, существуют два независимых РГПЧ для старшей и младшей частей блока;

– период повторения последовательности для младшей части составляет 2^{32}, для старшей части $2^{32}–1$, для всей последовательности период составляет $2^{32}(2^{32}–1)$.

Синхропосылка шифруется с помощью базового цикла зашифрования (32-З), полученные значения записываются во вспомогательные 32-разрядные регистры N_3 и N_4 - младшие и старшие биты соответственно. N_3 суммируется по модулю 2^{32} с константой $C_2 = 1010101_{16}$, N_4 суммируется по модулю 2^{32}-1 с константой $C_1 = 1010104_{16}$. N_3 и N_4 переписываются соответственно в N_1 и N_2, которые затем шифруются с помощью базового цикла зашифрования (32-З). Полученные блоки N_1 и N_2 накладываются на блоки открытых (зашифрованных данных), используя сложение по модулю 2. Процедура зашифрования (расшифрования) заканчивается, когда все блоки открытых (зашифрованных данных) зашифруются (расшифруются) соответственно алгоритму.

Схемы алгоритмов зашифрования и расшифрования в режиме гаммирования приведены на рисунках 9 и 10 соответственно.

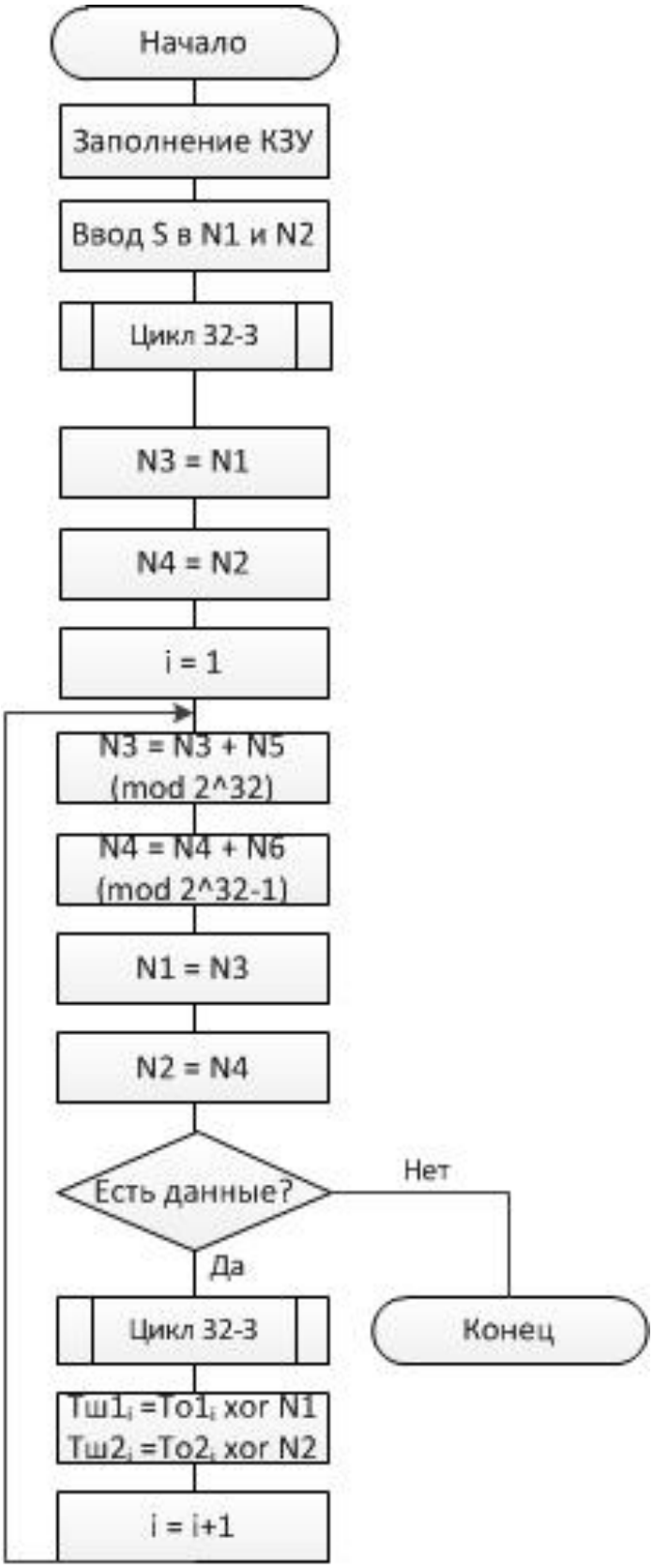


Рисунок 9 – Схема алгоритма зашифрования в режиме гаммирования

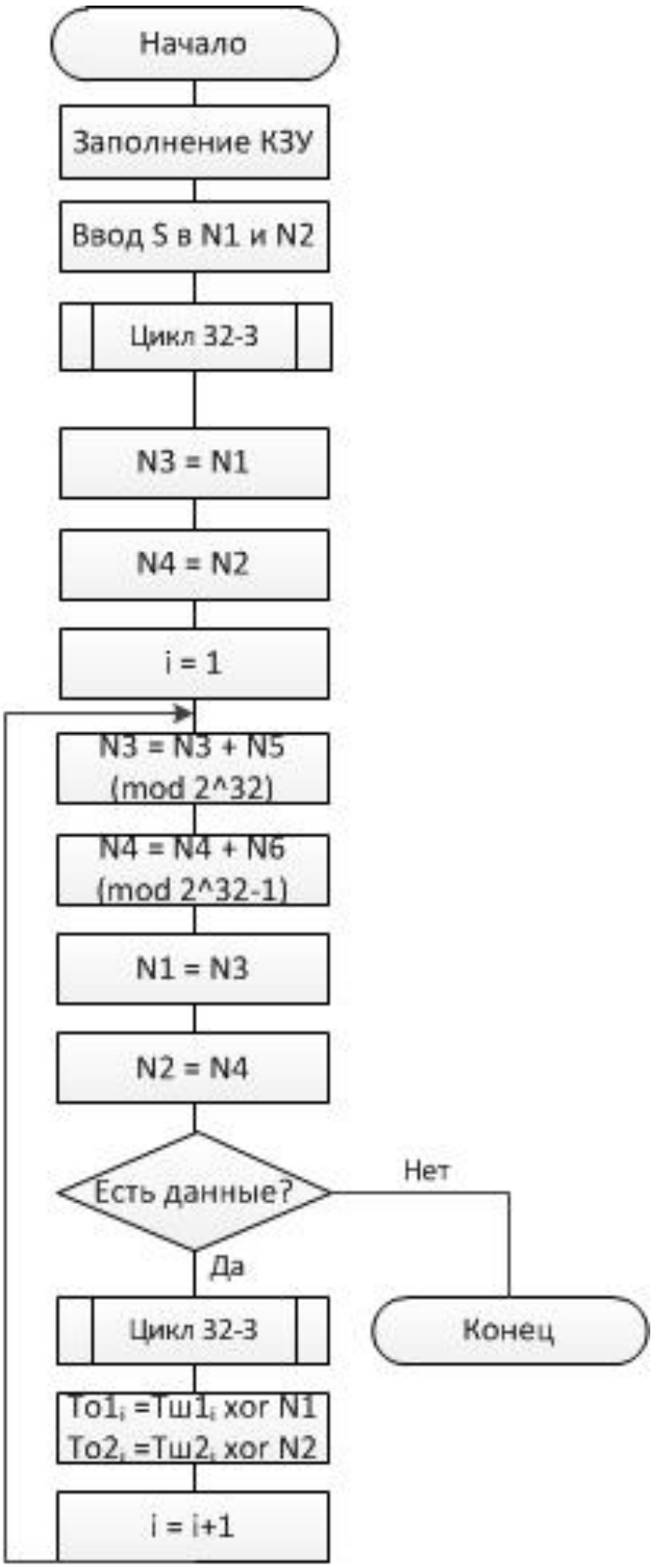


Рисунок 10 – Схема алгоритма расшифрования в режиме гаммирования

2.3.7 Режим выработки имитовставки алгоритма криптографического преобразования ГОСТ 28147-89.

Имитовставка – это контрольная комбинация, зависящая от открытых данных и секретной ключевой информации. Целью использования имитовставки является обнаружение всех случайных или преднамеренных изменений в массиве информации. Это особенно важно для шифрования в режиме гаммирования, где злоумышленник может изменить конкретные биты, даже не зная ключа.

Имитовставка вырабатывается для $M \geq 2$ блоков открытого текста по 64 бит. Блок открытых данных записывается в регистры N_1 и N_2, после чего подвергается преобразованию, соответствующему первым 16 раундам шифрования в режиме простой замены. К полученному результату побитно по модулю 2 прибавляется следующий блок открытых данных. Последний блок при необходимости дополняется нулями. Сумма также подвергается 16 раундовому шифрованию и суммируется по модулю 2 с третьим блоком открытой информации и так далее. После добавления и шифрования последнего блока из результата выбирается имитовставка длиной L бит: с бита номер 32-L до 32 (отсчёт начинается с 1). Имитовставка передается по каналу связи после зашифрованных блоков.

Схема алгоритма криптографического преобразования в режиме выработки имитовставки приведена на рисунке 11.

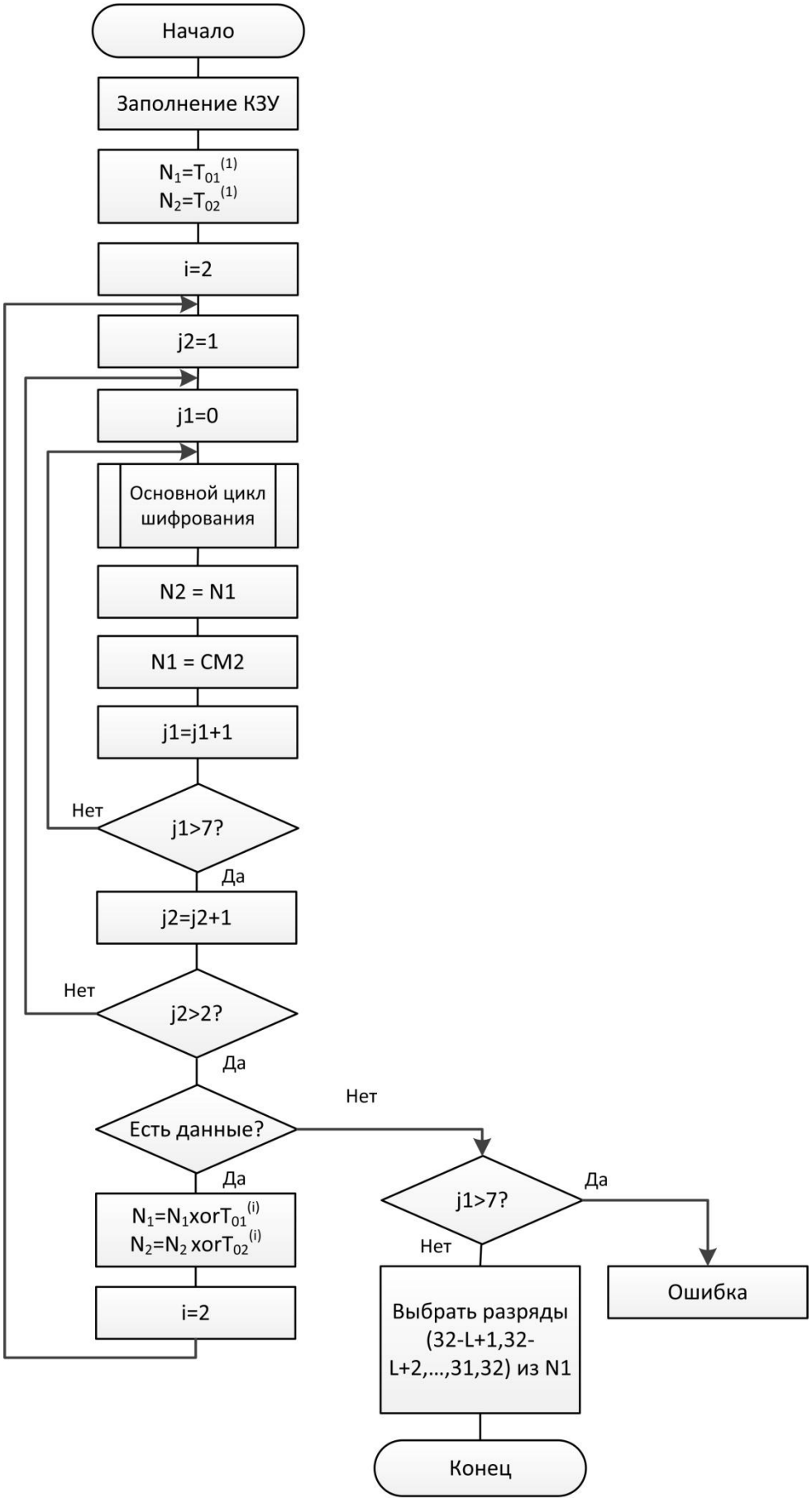


Рисунок 11 - Схема алгоритма криптографического преобразования в режиме выработки имитовставки

3 Описание встраиваемой аппаратуры передачи данных

3.1 Обзор основных сведений и технических характеристик устройства

Разработка программного обеспечения, реализующего функции криптографической защиты информации выполняется для малогабаритного встраиваемого устройства передачи данных (далее по тексту - устройство). Устройство предполагается для использования в качестве встраиваемого средства криптографической защиты информации в малогабаритной (носимой) аппаратуре связи. Обобщенная структурная схема устройства приведена на рисунке 12.

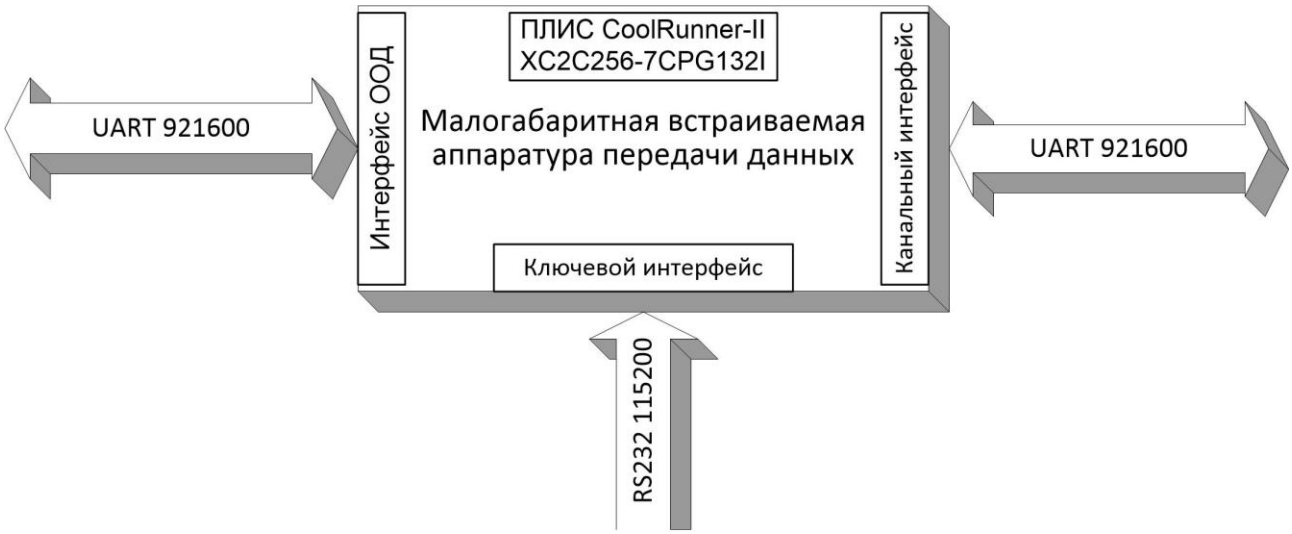


Рисунок 12 – Обобщенная структурная схема малогабаритной встраиваемой аппаратуры передачи данных

С оконечного оборудования данных в устройство поступает открытая информация. В микроконтроллере устройства происходит шифрование открытой информации, затем уже закрытая информация передается в канал связи с помощью канального интерфейса. Из канала связи поступает закрытая информация. В микроконтроллере происходит расшифрование закрытой информации. Открытая информация передается с устройства на оконечное оборудование. Таким образом, устройство работает в дуплексном режиме, так как информационный поток проходит через абонентский и канальный стыки в обе стороны.

Через интерфейс RS-232 вводится ключевая информация, используемая для шифрования информации.

Устройство должно обеспечивать криптографическую обработку информационного потока поступающего с максимальной скоростью 921600 бит/с.

Все стыки устройства реализованы в виде асинхронного последовательного интерфейса UART, при этом, ключевой стык имеет дополнительный преобразователь КМОП-уровней для прямого соединения с внешним устройством через порт RS-232.

3.2 Описание структурной схемы малогабаритной встраиваемой аппаратуры передачи данных

На рисунке 13 и в приложении А изображена структурная схема малогабаритной встраиваемой аппаратуры передачи данных.

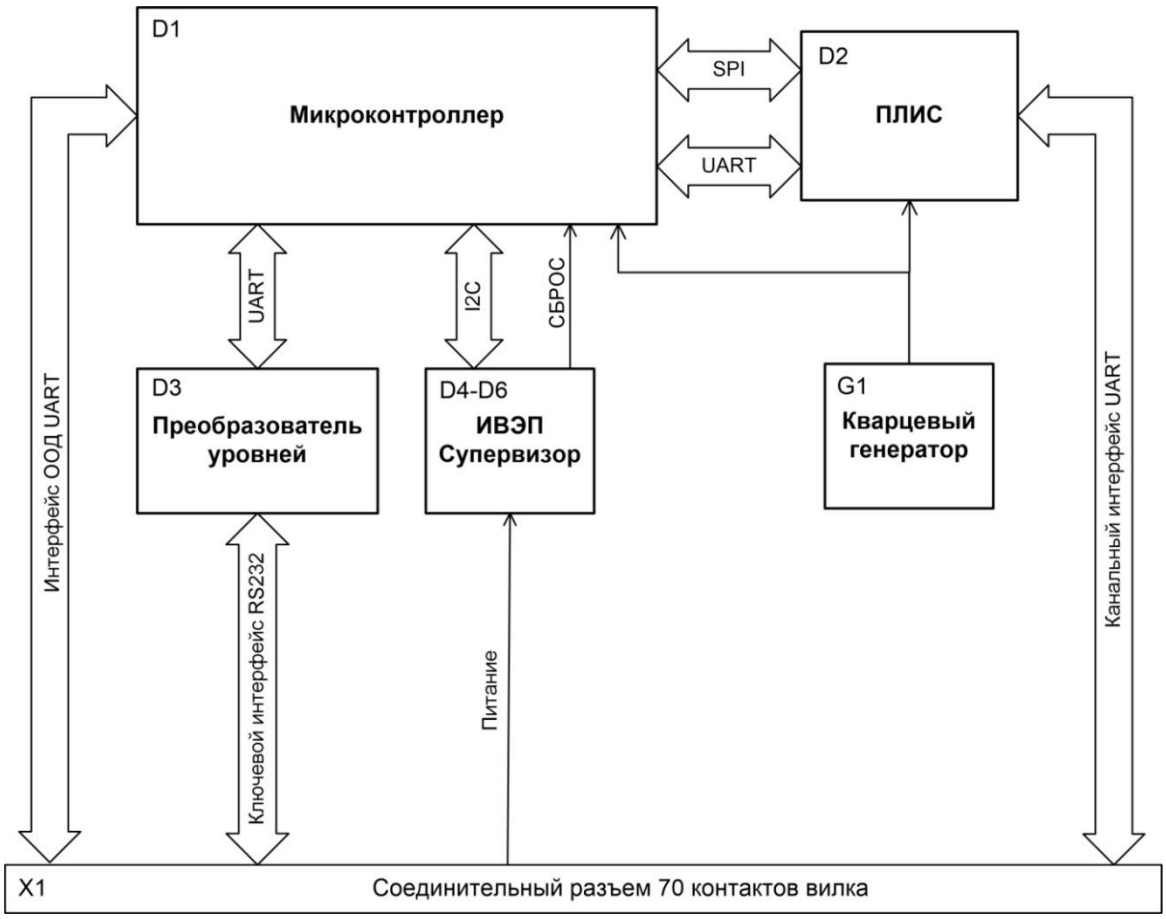


Рисунок 13 – Структурная схема малогабаритной встраиваемой аппаратуры передачи данных

В качестве центрального процессора в малогабаритной встраиваемой аппаратуре передачи данных используется микроконтроллер (МК) STM32F415OGY6 фирмы ST Microelectronics с ядром ARM Cortex-M4 разрядностью 32 бита, работающим на частоте 168 МГц. Ядро ARM Cortex-M4 поддерживает DSP-инструкции. Микроконтроллер STM32F415OGY6 обозначен на рисунке 5 как элемент D1. Данный микроконтроллер относится к семейству процессоров с RISC архитектурой с низким энергопотреблением. Инструкции DSP и модуль операций с плавающей точкой дают возможность применять данные контроллеры в широком спектре проектов. Оптимальное соотношение низкого энергопотребления, высокой производительности (до 210 MIPS) и низкой цены делает процессоры данного семейства идеальной платформой для ресурсоемких приложений обработки голоса и данных в портативной и мобильной аппаратуре (сотовые телефоны, УКВ-радиостанции и т.п.).

Генератор, изображенный на рисунке 5 как G1, предназначен для формирования сигнала опорной тактовой частоты 12 МГц, обеспечивающего синхронную работу всех схем устройства. От требуемого быстродействия зависит частота работы процессора, которая в свою очередь связана с его энергопотреблением.

Микроконтроллер связан последовательным периферийным интерфейсом (SPI) и интерфейсом USART с ПЛИС D2, в которой реализованы узел блока опорных частот (БОЧ) и узлы интерфейса USART.

К микроконтроллеру подключены ИМС D3, реализующая функции интерфейса RS-232. Интерфейс RS-232 предназначен для ввода ключевой информации, а также для целей диагностики и обновления ПО изделия с помощью персонального компьютера.

Оконечное оборудование данных (ООД) подключено через абонентский стык (интерфейс ООД) посредством интерфейса UART. Этот интерфейс позволяет осуществлять информационно-логический обмен командами управления и информационными данными, передаваемыми или принимаемыми

изделием. ООД является источником/получателем передаваемых/принимаемых данных. Физически данный интерфейс представляет собой трёхпроводной асинхронный интерфейс UART.

Источник вторичного электропитания (ИВЭП), реализованный на микросхемах D4-D6, предназначен для формирования напряжений питания всех элементов устройства. В состав ИВЭП входит супервизор D6, предназначенный для обеспечения контроля над уровнем напряжения питания устройства и формирования сигнала «СБРОС» при низких напряжениях вторичных источников питания. ИВЭП в части питания микроконтроллера и ПЛИС реализован на микросхеме D5.

Все цепи интерфейсов сопряжения, питания и управления выведены на соединительный разъём X1.

Рабочее напряжение питания устройства находится в диапазоне от 3,2 до 5,5 В. Максимальная мощность потребления устройства при питании от напряжения 5 В составляет 150 мВт.

3.3Аппаратно-программные возможности микроконтроллера STM32F415OGY6

Программное обеспечение, реализующее функции криптографической защиты информации, разрабатывается под аппаратно-программную вычислительную среду микроконтроллера семейства STM32F4 - STM32F415OGY6.

STM32F4 – богатое семейство, построенное на ядре Cortex M4 и объединяющее почти полторы сотни микроконтроллеров. Представители семейства объединены в несколько линеек, оптимизированных по различным параметрам: уровню интеграции, уровню потребления, производительности. Сейчас к услугам разработчиков предоставлены модели с рабочими частотами 168…180 МГц и объемом памяти до 2 Мбайт. Перечень периферии сильно отличается от линейки к линейке. Низкопроизводительные, но в то же время энерогоэффективные, микроконтроллеры имеют на борту только самое нужное:

коммуникационные интерфейсы, таймеры, порты ввода/вывода. Самые высокопроизводительные модели могут похвастаться графическим ускорителем, контроллером TFT, развитой аудиосистемой, продвинутым контроллером внешней памяти и т.д.

Микроконтроллер STM32F415OGY6 имеет 192 Кбайт встроенного ОЗУ, 1 Мбайт памяти программ, а также следующий набор внутренней периферии:

- три 12-разрядных АЦП;

- два 12-разрядных ЦАП;

- часы реального времени RTC (точность в секундах, аппаратный календарь);

- двенадцать универсальных 16-разрядных таймеров, включая два таймера ШИМ, два 32-разрядных таймера общего назначения;

- генератор случайных чисел (RNG);

- блок вычисления CRC;

- интегрированный криптографический процессор, обеспечивающий аппаратное шифрование по алгоритмам AES 128, 192, 256, Triple DES, и вычисления хеш сумм MD5 и SHA-1;

- универсальный ПДП (DMA): 16-канальный контроллер DMA с FIFO и поддержкой пакетного режима;

- два интерфейса USB OTG (один из них поддерживает USB HS);

- два интерфейса I²S в режиме дуплекса;

- 15 интерфейсов обмена данными (шесть USART со скоростью до 10,5 Мбит/с, три SPI интерфейса со скоростью до 42 Мбит/с, три I²C, два CAN и SDIO);

- 140 выводов общего назначения (GPIO).

Одной из главных причин популярности контроллеров фирмы STMicroelectronics является доступность и многообразие средств разработки и отладки – инструментов разработки и отладки ПО, готовых программных решений и библиотек, стартовых и оценочных наборов.

Существует большое количество платных и бесплатных сред разработки и отладки ПО для STM32F4 (Keil uVision, IAR, AtollicTrueSTUDIO, SystemWorkbench). Каждая из сред даёт разработчику возможность не только создавать ПО, но и отлаживать его как с помощью симуляторов, так и использовать различные программаторы и отладчики, и многое другое.

4 Разработка программного обеспечения

4.1 Описание языка программирования, выбранного для разработки программного обеспечения, реализующего функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных

Существует множество различных подходов и способов разработки программных средств, которые предлагают производители электронных компонентов. Они так же предлагают большое количество различных образцов, библиотек, программных решений, которые используются разработчиками программного обеспечения для осуществления своей деятельности. IAR Embedded Workbench представляет собой интегрированную среду разработки, которая применяется для разработки и отладки программ на языках Си и/или Ассемблер для различных сигнальных процессоров и микроконтроллеров.

Все чаще язык Си используется по следующим причинам:

– рост сложности архитектуры микроконтроллеров (на изучение которых тратится значительное время);

– с ростом объёма и сложности ПО;

– возможность переноса ПО на другую аппаратную платформу.

Программный код написанный при помощи языка Си удобен для чтения и имеет лаконичную структуру.

На данный момент язык Си применятся программирования большей части микроконтроллеров и сигнальных процессоров. Разработка прикладного программного обеспечения сводится к написанию исходного кода программы на языке Си, при использовании различных библиотечных функция, написанных на языке Ассемблер, предлагаемых собой интегрированной средой разработки.

Язык Си является гибким и универсальным языком программирования, что обуславливает его широкую распространенность. Универсальность данного языка программирования применятся для решения различных задач в области

разработки системного программного обеспечения, а так же написании прикладных программ.

Язык Си позволяет использовать программное обеспечение на компьютерах с различной архитектурой, имеет более лаконичную и удобную возможность написания и чтения программного кода, что повышает скорость разработки программ, по сравнению с программным кодом написанным на языке Ассемблер.

При разработке программного обеспечения язык Си удобен для разработчика по следующим причинам:

– наличие набора различных конструкций структурного программирования;

– поддержка блочной структуры программ;

– возможность раздельной компиляции модулей программы;

– наличие набора низкоуровневых средств для доступа к аппаратным средствам компьютера.

Разработка программного обеспечения шифратора в настоящей работе проведена на языке Си.

4.2 Программное обеспечение, используемое для разработки программы, реализующей функции криптографической защиты информации

Для разработки программы, реализующей функции криптографической защит информации в малогабаритной встраиваемой аппаратуре передачи данных, было использовано следующее программное обеспечение:

– STM32CubeMX;

– IAR Embedded Workbench.

STM32CubeMX – это графический редактор для конфигурирования микроконтроллеров STM32, позволяющий автоматически генерировать конфигурационный С-код и «скелет» прикладной программы с помощью набора визуальных утилит. Благодаря STM32CubeMX этап создания функций инициализации происходит очень быстро.

Графические генераторы кода позволяют уйти от рутинного написания программ. Вместо самостоятельного создания текстовых файлов разработчик использует графические утилиты, которые автоматически формируют программный код. Кросплатформенный графический редактор STM32CubeMX, работает на всех популярных ОС: Windows, Linux и OS X.

STM32CubeMX позволяет [8]:

– создавать и редактировать проекты для микроконтроллеров STM32 с последующей генерацией С-кода для конкретных интегрированных сред: IAR EWARM, Keil MDK-ARM, Atollic TrueSTUDIO и AC6 System Workbench (SW4STM32);

– производить визуализированную настройку выводов МК с помощью утилиты «Pin Wizard» с возможностью формирования табличного файла для трассировщиков печатных плат;

– производить визуализированную настройку дерева тактирования с помощью утилиты «Clock Wizard»;

– производить визуализированную настройку периферийных блоков и ПО промежуточного уровня, файловой системы, стеков протоколов, ОС с помощью утилиты «Perepherial and middleware Wizard»;

– производить оценку уровня потребления и срока службы аккумулятора при заданных настройках МК с помощью утилиты «Power consumption Wizard»;

– автоматизировать процесс миграции проектов при переходе от одной модели микроконтроллера к другой.

Программа STM32CubeMX значительно упрощает работу программистов, однако у нее также есть ограничения [8]:

– генерируемый С-код охватывает только настройку блоков и периферии микроконтроллера. Это значит, что алгоритмическую часть программы автоматически сгенерировать нельзя, ее нужно будет дописать вручную;

– STM32CubeMX поможет создать только стартовую конфигурацию. Иногда в процессе работы пользователю требуется изменить частоту работы периферийного блока или изменить конфигурацию вывода. Все это придется прописывать самостоятельно;

– для генерации кода используются стандартные, разработанные ST, библиотеки нижнего уровня (HAL и LL);

– в процессе генерации C-файл выстраивается таким образом, что для пользователя выделяются специальные секции, в которые он может помещать свой код. Если пользовательский код окажется вне этих рамок – он будет затерт при следующих генерациях.

IAR Embedded Workbench — комплексная среда разработки, многочисленные версии которой поддерживают большинство микроконтроллеров от различных производителей. Прежде всего, необходимо пояснить, что IAR EW является не просто программой, а именно комплексной средой разработки, поскольку состоит из целого комплекса программных и аппаратных инструментов. И хотя, как правило, аппаратные отладчики не поставляются в случае покупки только лицензии на пакет, так как в среду включена поддержка J-TAG-адаптеров различных фирм, тем не менее, компания IAR рекомендует использовать в комплекте аппаратное обеспечение выпускаемое ею. Программная составляющая среды состоит из следующих программных компонентов:

– C/C++ компилятор;
– компилятор ассемблера;
– компоновщик;
– библиотечный модуль;
– редактор;
– менеджер проектов;
– C-Spy-отладчик;
– сервер лицензий.

Внешний вид EWARM в режиме отладки представлен на рисунке 14.

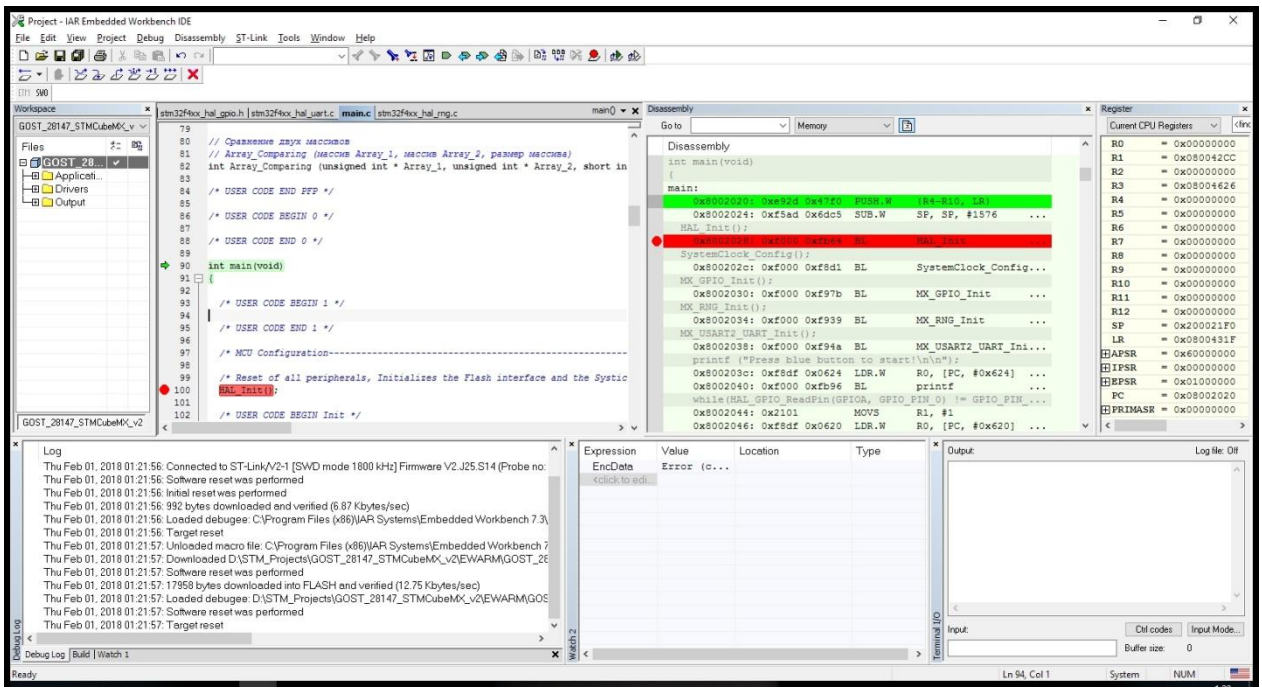


Рисунок 14 - Внешний вид EWARM в режиме отладки

Среда разработки обеспечивает все этапы создания программного обеспечения — от написания программы до загрузки отлаженного кода микропрограммы в целевой микроконтроллер. Ее достоинствами являются высокоэффективные компиляторы и удобный пользовательский графический интерфейс ко всем приложениям, входящим в пакет, наличие плагинов, поддерживающих различные операционные системы реального времени, и импорт данных из пакетов разных производителей. Эти функции позволяют легко использовать драйверы, сгенерированные для периферийных модулей (так называемые HAL, Hardware Abstraction Layer) и импортировать библиотеки, полученные с помощью этих программ [9].

4.3 Разработка алгоритма работы программного обеспечения, реализующего функции криптографической защиты информации

Для реализации программного обеспечения криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных была разработана соответствующая схема алгоритма. Разработанная схема алгоритма приведена на рисунке 15.

Рисунок 15 - Схема алгоритма программного обеспечения, реализующего функции криптографической защиты информации

4.4 Описание функционирования программного обеспечения, реализующего функции криптографической защиты информации

Исходя из того, что проверка работоспособности и тестирование разработанного программного обеспечения, реализующего функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных, будет использоваться отладочная плата STM32F407G-DISC1, то программа также разрабатывалась под МК STM32F407VGTx.

STM32F407G-DISC1 – это отладочная плата, которая позволяет пользователю быстро разрабатывать приложения для 32-битного ARM Cortex-M4 микроконтроллера STM32F407. Отладочная плата основана на МК STM32F407VGT6. Данная отладочная плата выбрана из-за того, что используемый на ней МК из того же семейства STM32F4, что и МК, применяемый в устройстве передачи зашифрованной информации. Это означает, что их функционал практически идентичен. Кроме того, на плате имеется встроенный программатор/отладчик ST-Link/V2-1, благодаря которому к МК можно подключаться по интерфейсу USB, то есть не требуются внешние программаторы.

По разработанной схеме алгоритма написана программа, реализующего функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных. Программа приведена в приложении Б.

Для разработки требуемой программы в конфигураторе для МК STMicroelectronics STM32CubeMX был создан проект для платформы STM32F407VGTx STM32F4-DISCOVERY. Далее в созданном проекте были определены назначение выводов МК, настроена система тактирования, выбрана целевая интегрированная среда разработки – IAR Embedded Workbench и сгенерирован С-код с использованием HAL-драйверов. В результате этого в указанной директории проекта был сгенерирован проект для среды разработки IAR Embedded Workbench.

В среде разработки IAR Embedded Workbench была написана программа по схеме алгоритма программы, реализующей функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных. При написании программы были использованы функции библиотеки HAL.

Функция, реализующая режим зашифрования/расшифрования в режиме простой замены, согласно схеме 32-х циклов зашифрования/расшифрования, GOST_ECB(). Функция GOST_ECB() в качестве входных параметров

принимает ключи, выбранный режим, зашифрование или расшифрование, и входные данные. Результатом работы функции GOST_ ECB() является зашифрованные или расшифрованные данные, в зависимости от входных данных и режима шифрования.

Функция, реализующая режим зашифрования/расшифрования в режиме гаммирования, согласно схеме алгоритма зашифрования в режиме гаммирования, GOST_Gamma(). Функция GOST_Gamma () в качестве входных параметров принимает ключи, синхропосылку, входные данные и размер данных. Результатом работы функции GOST_Gamma() является зашифрованные/расшифрованные данные, в зависимости от входных данных.

Функция, реализующая режим выработки имитовставки, согласно схеме алгоритма криптографического преобразования в режиме выработки имитовставки, GOST_MAC(). Функция GOST_MAC() в качестве входных параметров принимает ключи, входные данные и размер данных. Результатом работы функции GOST_MAC() является выработанная имитовставка размером 64 бита.

Для проверки правильности зашифрования и расшифрования также была реализована функция сравнения двух массивов данных - Array_Comparing(). Функция Array_Comparing() в качестве входных параметров принимает два массива для сравнения и их размер. Результатом работы функции является значение – совпадают эти массивы или нет.

Функция HAL_RNG_GetRandomNumber() из библиотеки HAL возвращает сгенерированное случайное число. Функция HAL_RNG_GetRandomNumber() в качестве входных параметров принимает структуру, которая содержит конфигурационные данные для генератора случайных чисел. Результатом работы функции является значение длиной 32 бита. Данная функция работает со встроенным в микроконтроллеры серии STM32F4 генератором случайных чисел. Встроенный генератор случайных чисел построен на аналоговой схеме. Он производит аналоговый шум, который подается в линейный регистр сдвига.

Аналоговая схема рассчитана на кольцевые генераторы, выход которых подвергается операции исключающего или (XOR).

Функция HAL_RNG_GetRandomNumber() в разработанном программном обеспечении используется для генерации синхропосылки, выработки тестовых ключей и тестовых открытых данных.

Для получения сведений о скорости криптопреобразований, а именно о скорости зашифрования и расшифрования в режиме гаммирования и скорости выработки имитовставки, МК по отладочному интерфейсу отправляет эту информацию.

Для проверки функционирования разработанной программы по отладочному интерфейсу также отправляются выработанные синхропосылка и имитовставка.

4.5 Тестирование программного обеспечения, реализующего функции криптографической защиты информации, на соответствие предъявленным требованиям

Проведено тестирование разработанной программы, реализующей криптографическую защиту информации. Для этого в FLASH-память МК отладочной платы была загружена разработанная программа.

Для получения сведений о скорости зашифрования в режиме гаммирования в массив с исходными открытыми данными OpenData[] записываются случайные значения, получаемые в результате работы функции HAL_RNG_GetRandomNumber(). Полученный массив с исходными данными зашифровывается на ключе и синхропосылке, полученных в результате работы функции HAL_RNG_GetRandomNumber(). После завершения расшифрования фиксируется затраченное время.

Для получения сведений о скорости расшифрования в режиме гаммирования массив с зашифрованными данными EncData[] расшифровывается на заданном ключе и синхропосылке и результат записываются в массив DecData[]. После завершения зашифрования фиксируется затраченное время.

Для получения сведений о скорости выработки имитовставки вызывется функция GOST_MAC(), которая на вход принимает массив с зашифрованными данными EncData[] и заданный ключ шифрования. После завершения выработки имитовставки фиксируется затраченное время.

Полученное время зашифрования, расшифрования и выработки имитовставки выводится по отладочному интерфейсу в терминал. Для проверки функционирования разработанной программы по отладочному интерфейсу также отправляются выработанные синхропосылка и имитовставка.

Окно с выводом сгенерированной синхропосылки и имитовставки, и информацией о времени, затраченного на зашифрование, расшифрование и выработку имитовставки приведено на рисунке 16.

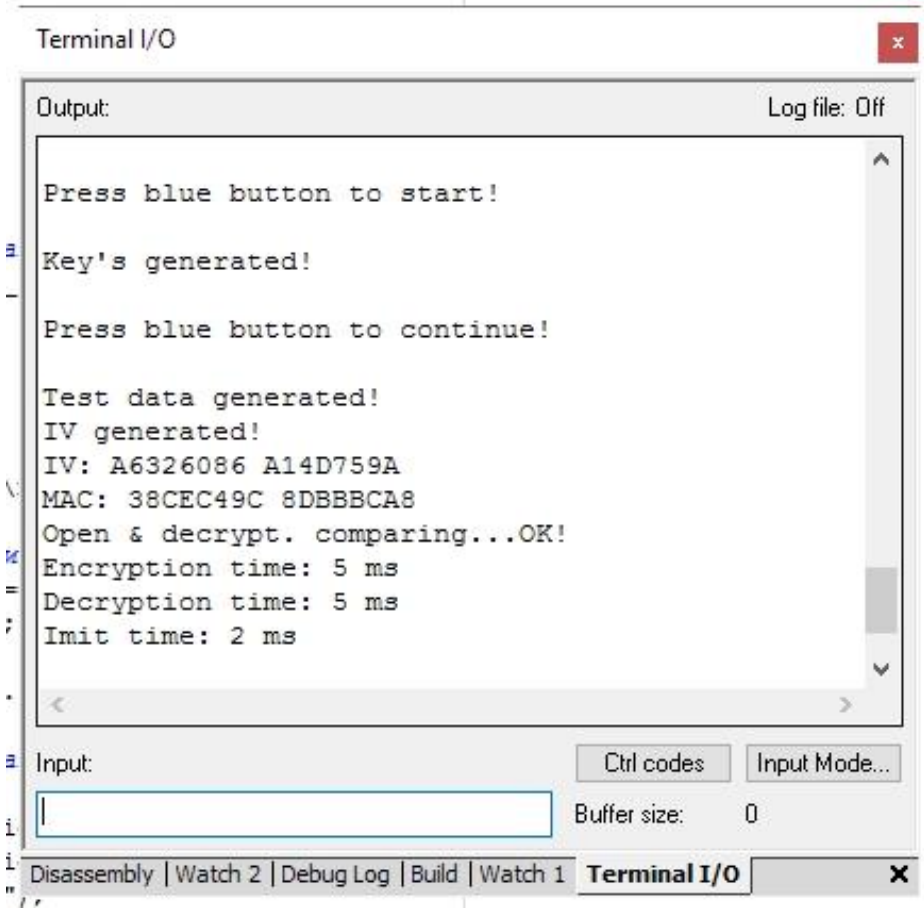


Рисунок 16 - Окно терминала с выводом отладочной информации

Исходя из полученных значений времени зашифрования и расшифрования и зная объем шифруемых данных, можно вычислить скорость зашифрования и расшифрования информации. Скорость зашифрования/расшифрования информации, полученная на отладочной плате,

составила 819200 бит/с. Полученная скорость криптопреобразования удовлетворяет требованиям, предъявленным в техническом задании.

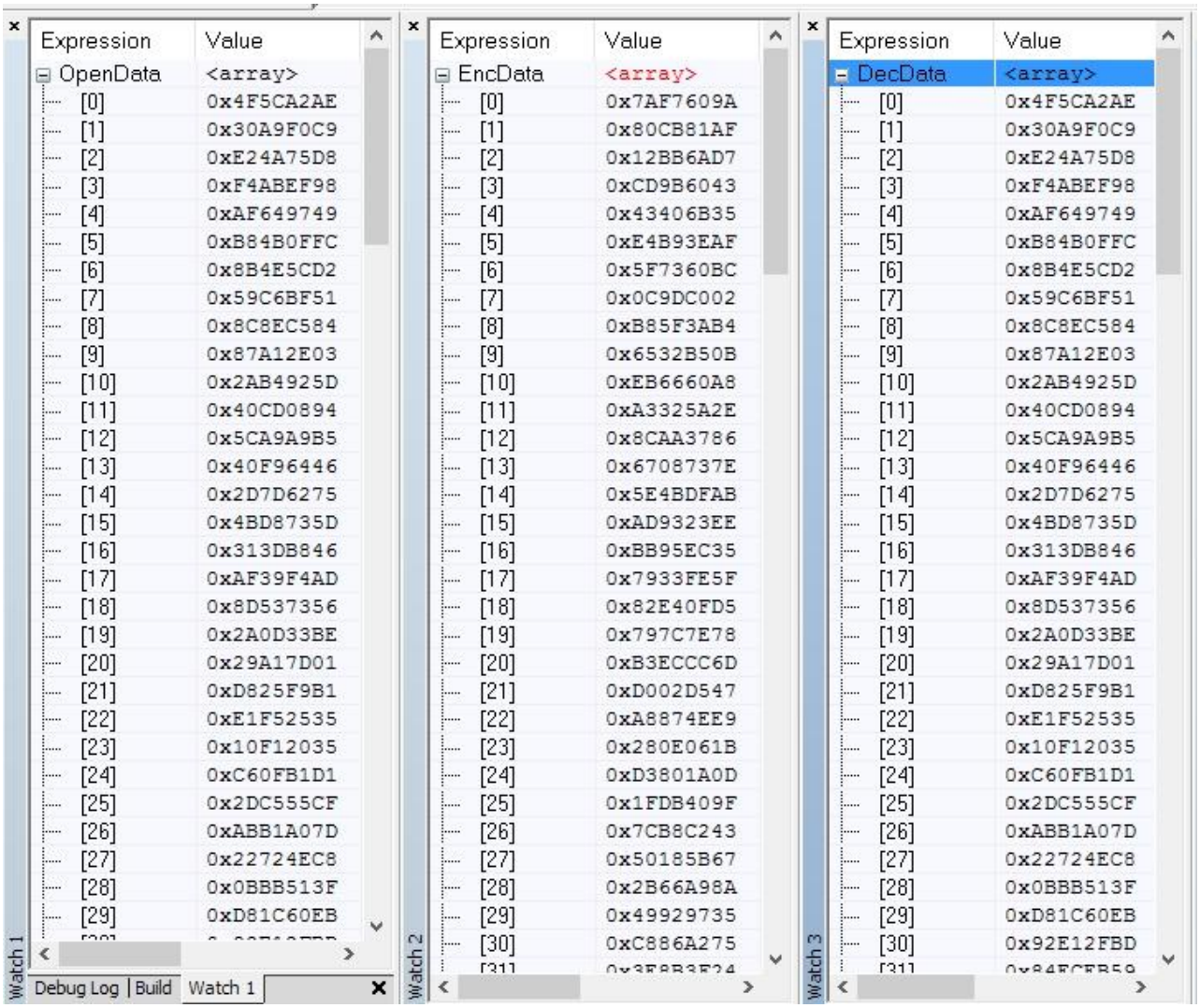
Скорость выработки имитовставки, полученная на отладочной плате, составила 256000 байт/с.

Для проверки правильности функционирования разработанного программного обеспечения было проверено содержимое памяти микроконтроллера, в которой содержатся массивы с исходными открытыми данными, зашифрованными и расшифрованными данными. Содержимое памяти микроконтроллера, в которой расположены массивы исходных, зашифрованных и расшифрованных данных, приведено на рисунке 17.

Исходя из сравнения исходного открытого и расшифрованного массивов, можно сделать вывод о правильности функционирования разработанного программного обеспечения.

Объем памяти микроконтроллера, занимаемого программным обеспечением, реализующим функции криптографической защиты информации, был получен из файла *.map. Общий объем программного обеспечения, который включает код, константы и переменные, составил 30,4 КБайта. Полученный объем программного обеспечения удовлетворяет требованиям технического задания.

Часть содержимого map-файла приведена на рисунке 18.



Expression	Value
OpenData	<array>
[0]	0x4F5CA2AE
[1]	0x30A9F0C9
[2]	0xE24A75D8
[3]	0xF4ABEF98
[4]	0xAF649749
[5]	0xB84B0FFC
[6]	0x8B4E5CD2
[7]	0x59C6BF51
[8]	0x8C8EC584
[9]	0x87A12E03
[10]	0x2AB4925D
[11]	0x40CD0894
[12]	0x5CA9A9B5
[13]	0x40F96446
[14]	0x2D7D6275
[15]	0x4BD8735D
[16]	0x313DB846
[17]	0xAF39F4AD
[18]	0x8D537356
[19]	0x2A0D33BE
[20]	0x29A17D01
[21]	0xD825F9B1
[22]	0xE1F52535
[23]	0x10F12035
[24]	0xC60FB1D1
[25]	0x2DC555CF
[26]	0xABB1A07D
[27]	0x22724EC8
[28]	0x0BBB513F
[29]	0xD81C60EB

Watch 1 | Debug Log | Build | Watch 1

Expression	Value
EncData	<array>
[0]	0x7AF7609A
[1]	0x80CB81AF
[2]	0x12BB6AD7
[3]	0xCD9B6043
[4]	0x43406B35
[5]	0xE4B93EAF
[6]	0x5F7360BC
[7]	0x0C9DC002
[8]	0xB85F3AB4
[9]	0x6532B50B
[10]	0xEB6660A8
[11]	0xA3325A2E
[12]	0x8CAA3786
[13]	0x6708737E
[14]	0x5E4BDFAB
[15]	0xAD9323EE
[16]	0xBB95EC35
[17]	0x7933FE5F
[18]	0x82E40FD5
[19]	0x797C7E78
[20]	0xB3ECCC6D
[21]	0xD002D547
[22]	0xA8874EE9
[23]	0x280E061B
[24]	0xD3801A0D
[25]	0x1FDB409F
[26]	0x7CB8C243
[27]	0x50185B67
[28]	0x2B66A98A
[29]	0x49929735
[30]	0xC886A275

Watch 2

Expression	Value
DecData	<array>
[0]	0x4F5CA2AE
[1]	0x30A9F0C9
[2]	0xE24A75D8
[3]	0xF4ABEF98
[4]	0xAF649749
[5]	0xB84B0FFC
[6]	0x8B4E5CD2
[7]	0x59C6BF51
[8]	0x8C8EC584
[9]	0x87A12E03
[10]	0x2AB4925D
[11]	0x40CD0894
[12]	0x5CA9A9B5
[13]	0x40F96446
[14]	0x2D7D6275
[15]	0x4BD8735D
[16]	0x313DB846
[17]	0xAF39F4AD
[18]	0x8D537356
[19]	0x2A0D33BE
[20]	0x29A17D01
[21]	0xD825F9B1
[22]	0xE1F52535
[23]	0x10F12035
[24]	0xC60FB1D1
[25]	0x2DC555CF
[26]	0xABB1A07D
[27]	0x22724EC8
[28]	0x0BBB513F
[29]	0xD81C60EB
[30]	0x92E12FBD

Watch 3

Рисунок 17 - Массивы исходных, зашифрованных и расшифрованных данных

```
[1] = D:\STM_Projects\GOST_28147_STMCubeMX_v2\EWARM\GOST_28147_STMCubeMX_v2\Obj
[2] = command line
[3] = dl7M_tlf.a
[4] = m7M_tl.a
[5] = rt7M_tl.a
[6] = shb_l.a

  16 356 bytes of readonly  code memory
   1 602 bytes of readonly  data memory
  13 160 bytes of readwrite data memory

Errors: none
Warnings: none
```

Рисунок 18 - Часть map-файла проекта в IAR EW

5 Вредные психофизиологические факторы, влияющие на сердечно-сосудистую систему специалиста по защите информации. Методы и средства физической культуры, снижающие их воздействие

5.1 Вредные психофизиологические факторы, влияющие на сердечно-сосудистую систему специалиста по защите информации

Сердце и кровеносные сосуды в совокупности образуют замкнутую систему кровообращения, снабжающую кровью органы и ткани в организме человека, которую называют сердечно-сосудистой системой.

Сердце является мышечным органом, который, благодаря ритмичным повторным сокращениям, обеспечивает циркуляцию крови по кровеносным сосудам. Кровеносные сосуды представляют собой полые трубки разного размера. Сердечно-сосудистая система доставляет к органам и тканям кислород и различные питательные вещества, необходимые для их жизнедеятельности, а также выводит из организма токсины, шлаки и углекислый газ.

В настоящее время заболевания сердечно-сосудистой системы в России являются основной причиной смертности. Только в год от болезней органов кровообращения, таких как цереброваскулярная патология и ишемическая болезнь, умирает более 17 миллионов людей по всему миру. По предположениям ученых в 2030 году из-за заболеваний сердечно-сосудистой системы могут уйти из жизни более 23 миллионов человек. Более половины жителей России, страдающих от болезней органов кровообращения, находятся в трудоспособном возрасте.

В соответствии с ГОСТ 12.0.003-2015 [10] опасные и вредные производственные факторы трудового процесса по источнику своего происхождения подразделяют:

- на психофизиологические:
- организационно-управленческие;
- личностно-поведенческие (то есть связанные с самим работающим);
- социально-экономические [10].

Психофизиологические опасные и вредные факторы обусловлены особенностями физиологии и психологии человека.

Опасные и вредные производственные факторы, обладающие свойствами психофизиологического воздействия на организм человека, подразделяют:

– на физические перегрузки, связанные с тяжестью трудового процесса;

– нервно-психические перегрузки, связанные с напряженностью трудового процесса [10].

В процессе своей работы специалист по защите информации в основном решает задачи, связанные с анализом большого количества информации.

Деятельность специалиста по защите информации включает в себя:

– проектирование, разработку и тестирование систем управления информационной безопасностью в команде с другими специалистами;

– определение необходимого набора мер по информационной безопасности, организация их введения и контроль исполнения всех предписаний;

– грамотное применение нормативно-правовой базы в области информационной безопасности в процессе своей работы;

– анализ и корректировка существующей системы организации комплексной защиты информации;

– проектирование организационно-распорядительной документации, которая регулирует процесс защиты информации в организации, что включает в себя разработку методических материалов, положений и инструкций.

Около 40 часов в неделю специалист по защите информации проводит за своим рабочим местом, сидя за компьютером. Таким образом, работники в сфере информационной безопасности практически не испытывают динамических физических нагрузок, зато в полной мере испытывают статическую физическую нагрузку, так как практически не встают со своего кресла весь рабочий день.

При сидячем образе жизни риск развития сердечно-сосудистых заболеваний увеличивается. Большую часть рабочего времени человек проводит в неподвижном состоянии, из-за нехватки физической активности снижается тонус системы кровоснабжения, что приводит к сбоям в работе сердца. Повышение триглицеридов в крови – показатель риска сердечно-сосудистых заболеваний. В условиях малой физической активности ферменты, разрушающие излишки триглицеридов, также снижают свою активность. Увеличение количества триглицеридов приводит к затруднению кровообращения, по краям кровеносных сосудов формируется налет, что может спровоцировать атеросклероз сосудов.

Риск сердечно-сосудистых заболеваний повышается также при лишнем весе. При недостаточной физической активности человека его обмен веществ замедляется, избыточные питательные вещества накапливаются в виде жировых отложений. Высокий уровень холестерина в крови, как следствие нарушенного обмена веществ, увеличивает риск возникновения гипертонии.

Исходя из основных обязанностей специалиста по защите информации, рассмотренных выше, можно отметить, что работник в сфере информационной безопасности испытывает также значительные нервно-психические перегрузки.

Нервно-психические перегрузки подразделяют:

– на умственное перенапряжение, в том числе вызванное информационной нагрузкой;

– перенапряжение анализаторов, в том числе вызванное информационной нагрузкой:

– монотонность труда, вызывающая монотонию:

– эмоциональные перегрузки [10].

В соответствии с исследованиями специалистов-физиологов нахождение центральной нервной системы в возбужденном состоянии, а также увеличение уровня ее возбужденности из-за непрекращающегося стресса приводят к спазму сосудов, что затрудняет кровообращение в организме, а в последствии может стать причиной заболевания сердечно-сосудистой системы. Эмоциональные

перегрузки в виде постоянного стресса напрямую связаны с артериальной гипертонией. При высоком уровне умственного перенапряжения на работе человек подвергается риску возникновения хронического повышенного давления, а также риску появления сердечной недостаточности.

5.2 Методы и средства физической культуры, снижающие воздействие психофизиологических факторов, влияющих на сердечно-сосудистую систему специалиста по защите информации

Гигиеническая гимнастика занимает важное место среди средств физической культуры, помогающих работникам поддерживать состояние своего организма в тонусе. Постоянство в выполнении упражнений необходимо для повышения трудоспособности работников умственного труда. Занятие гигиенической гимнастикой улучшает кровоснабжение организма, тренирует органы дыхательной системы.

В тренировку по гигиенической гимнастике необходимо включить упражнения на разминку суставов и мышц, упражнения на силу, а также заминку, в процессе которой уменьшится напряжение в мышцах.

Например, тренировочный комплекс может включать следующие упражнения:

– упражнения на растягивание мышц грудной клетки, может также включаться растягивание мышц передней части плеча;

– выполнение махов руками из стороны в сторону, круговые движения согнутыми в локтях руками (этот вид упражнений разомнет суставы и укрепит связки);

– приседания, а также приседания, чередуемые с наклонами туловища, круговые вращения чуть согнутыми в коленях ногами (укрепление суставов и связок ног и таза);

– наклоны туловища из стороны в сторону, вращение туловищем по часовой стрелке и против часовой стрелки поочередно;

– выполнение отжиманий (для начинающих возможно использование упора коленями);

– выполнение силовых упражнений на пресс, важно правильное дыхание и использование нескольких подходов вместо одного длительного;

– махи ногами вперед и назад с упором или без него, махи должны быть амплитудными для тренировки и разогрева связок и суставов таза;

– выполнение упражнения «Мельница» и различных его вариаций с усложнением;

– упражнения на тренировку сердечной мышцы, такие как: легкий бег, ходьба на высокой скорости, подскоки с захлестом голени или поднятием бедра, прыжки со скакалкой или без нее;

– выполнение заминки, в процессе которой растягиваются и расслабляются перенапряженные мышцы, напряжение от прошедшей тренировки должно уменьшиться или совсем исчезнуть.

После выполнения всех упражнений можно походить, восстанавливая дыхание, или посидеть несколько минут с закрытыми глазами.

Важно помнить, что самое главное – следить за дыханием. Ровное дыхание при физической активности – залог здоровья сердечно-сосудистой системы.

Заключение

В ходе выполнения выпускной квалификационной работы было разработано программное обеспечение, реализующее функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных.

В процессе работы была обоснована актуальность разработки программного обеспечения, реализующего функции криптографической защиты информации в малогабаритном встраиваемом устройстве передачи данных, определены основные способы реализации методов криптографической защиты информации и выявлены требования, предъявляемые к разрабатываемым средствам.

Обоснован выбор ГОСТ 28147-89 в качестве алгоритма криптографического преобразования, а также проведен анализ выбранного алгоритма с точки зрения криптографической стойкости.

Проведен анализ структурной схемы малогабаритной встраиваемой аппаратуры передачи данных.

В заключении было разработано и протестировано программное обеспечение, реализующее функции криптографической защиты информации.

Результатом выпускной квалификационной работы является программное обеспечение, реализующее функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных.

Таким образом, задание на выпускную квалификационную работу выполнено в полном объеме.

Список используемых источников

1 ГОСТ Р 52459.1-2009. Совместимость технических средств электромагнитная. Технические средства радиосвязи. Часть 1. Общие технические требования и методы испытаний. – Введ. 01-01-2010. – М.: Стандартинформ, 2010. – 46 с.

2 Теория информационной безопасности и методология защиты информации: Рабочая лекция по дисциплине, Санкт-Петербург: Санкт-Петербургский университет, 2014. – 30 с.

3 Методы и средства криптографической защиты информации: Учебное пособие / Жданов О.Н., Золотарев В.В., Красноярск: СГАУ им. М.Ф. Решетнева, 2007. – 72 с.

4 Информационная безопасность: Электронный учебник / Амелин Р.В. URL: http://nto.immpu.sgu.ru/sites/default/files/3/__77037.pdf (дата обращения 25.02.2018).

5 ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. – М.: Издательство стандартов, 1989.

6 Современная криптография: теория и практика / Мао В. Пер. с англ. – М.: Издательский дом «Вильямс», 2005. – 768 с.

7 ГОСТ 28147-89: «Не спеши его хоронить». Часть 1. Стойкость алгоритма // КриптоПро. URL: https://www.cryptopro.ru/blog/2013/08/27/gost-28147-89-ne-speshi-ego-khoronit-chast-1-stoikost-algoritma (дата обращения 25.02.2018).

8 User manual. STM32CubeMX for STM32 configuration and initialization C code generation (UM1718). ST Microelectronics, 2017.

9 Продукты компании IAR для разработки программного обеспечения встраиваемых устройств // Компоненты и технологии №7, 2014. URL: http://app.efo.ru/storage/art/micro/IAR_vstraivaemie.pdf (дата обращения 25.02.2018).

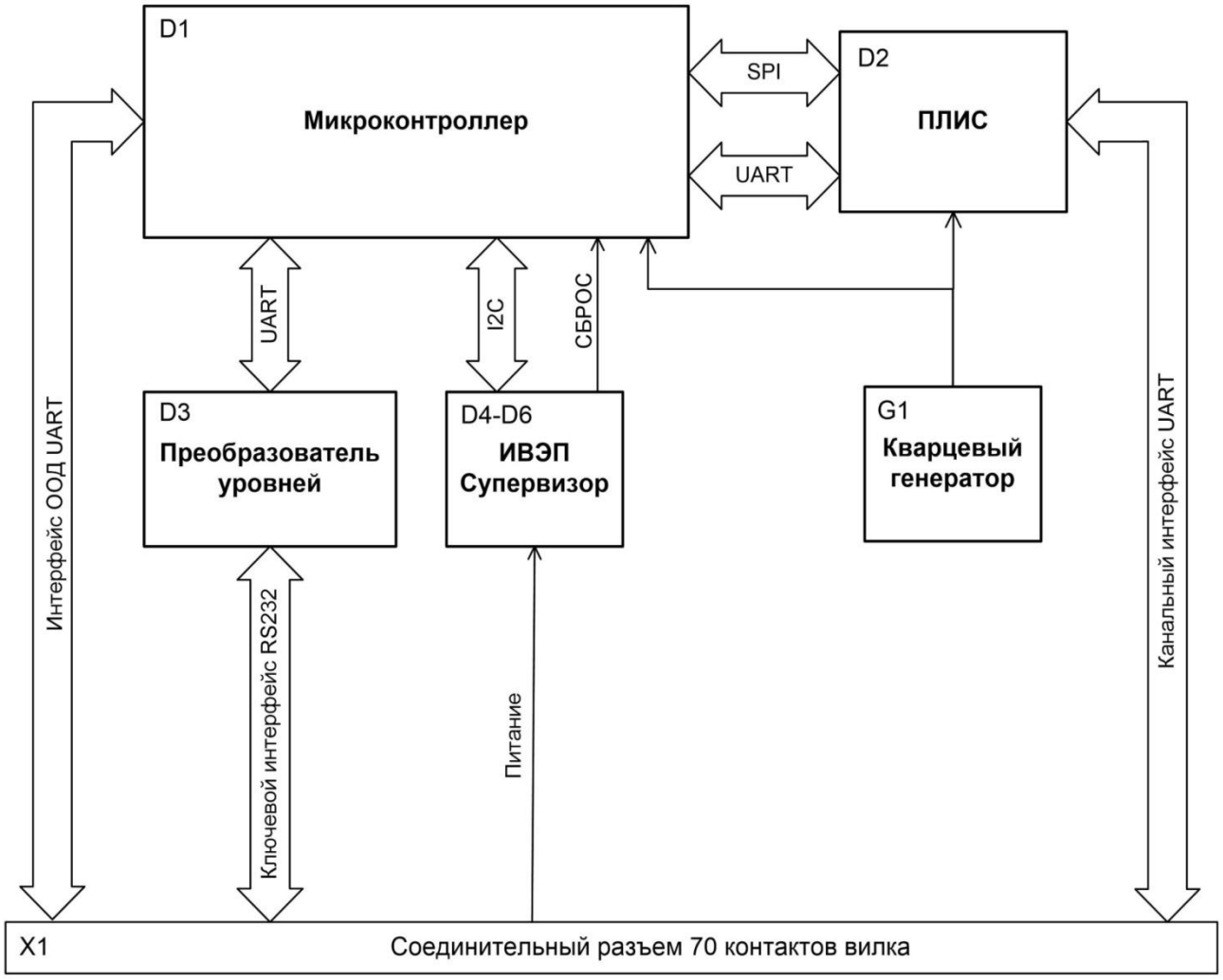
10 ГОСТ 12.0.003–2015. Система стандартов по безопасности труда. Опасные и вредные производственные факторы. Классификация. – Введ. 01-03-2017. – М.: Стандартинформ, 2016. – 10 с.

Приложение А

(обязательное)

Схема электрическая структурная малогабаритной встраиваемой аппаратуры передачи данных

ПГУ.100502.С32.ВКР.12ПТ131.01 Э1



					ПГУ.100502.С32.ВКР.12ПТ131.01 Э1	
Изм	Лист	№ докум.	Подпись	Дата	Аппаратура передачи данных Схема электрическая структурная	Лит. / Масса / Масштаб
Разраб.		Тарханова				
Проверил		Прокуров				Лист / Листов 1
Т. Контр.						
Н. Контр.		Иванов				
Утвердил						

Перв. примен. | Справочный № | Подп. И дата | Инв. № дубл. | Взам. инв. № | Подп. И дата | Инв. № подл.

Копировал

Формат А4

Приложение Б

(обязательное)

Исходный код программного обеспечения, реализующего функции криптографической защиты информации

Исходный код файла main.c

```
#include "main.h"
#include "stm32f4xx_hal.h"
#include <stdio.h>

RNG_HandleTypeDef hrng;

UART_HandleTypeDef huart2;

void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_RNG_Init(void);
static void MX_USART2_UART_Init(void);

// Режим простой замены
// GOST_ECB(ключи, режим(0 - зашифрование, 1 - расшифрование), Данные)
void GOST_ECB (unsigned int * key, short int work_mode, unsigned int * N);

// Режим гаммирования
// GOST_Gamma(ключи, синхропосылка 1, синхропосылка 1, Входные данные, Выходные данные, объем данных в байтах)
void GOST_Gamma (unsigned int * key, unsigned int S1, unsigned int S2, unsigned int * InData, unsigned int * OutData, short int size);

// Режим выработки имитовставки
// GOST_MAC(ключи, переменная для имитовставки, данные, объем данных в байтах)
void GOST_MAC (unsigned int * key, unsigned int * MAC, unsigned int * Data, short int size);

// Сравнение двух массивов
// Array_Comparing (массив Array_1, массив Array_2, размер массива)
int Array_Comparing (unsigned int * Array_1, unsigned int * Array_2, short int size);

int main(void)
{

  HAL_Init();

  SystemClock_Config();
```

```
MX_GPIO_Init();
MX_RNG_Init();
MX_USART2_UART_Init();

  // Синхропосылка
  unsigned int S1, S2;

while (1)
{

  // Нажми на кнопку
  printf ("Press blue button to start!\n\n");

  /*## Wait until Key button is pressed ##################################*/
  while(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) != GPIO_PIN_RESET)
  {
  }

  /*## Loop while Key button is maintained pressed #######################*/
  while(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) == GPIO_PIN_RESET)
  {
  }

  // Генерация ключей
  unsigned int key [8];
  for (int j = 0; j < 8; j++)
    key[j] = HAL_RNG_GetRandomNumber(&hrng);

  // Ключи сгенерированы
  printf ("Key's generated!\n\n");

  // Нажми на кнопку для продолжения
  printf ("Press blue button to continue!\n\n");

  /*## Wait until Key button is pressed ##################################*/
  while(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) != GPIO_PIN_RESET)
  {
  }

  /*## Loop while Key button is maintained pressed #######################*/
  while(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) == GPIO_PIN_RESET)
  {
  }
```

```
// Открытые данные
unsigned int OpenData [128];
for (int j = 0; j < 128; j++)
  OpenData[j] = HAL_RNG_GetRandomNumber(&hrng);

// Тестовые данные сгенерированы
printf ("Test data generated!\n");

short int size = sizeof(OpenData);

// Зашифрованные данные
unsigned int EncData [128] = {0};

// Расшифрованные данные
unsigned int DecData [128] = {0};



// Генерация синхропосылки
// Использование RNG STM32F4xx
S1 = HAL_RNG_GetRandomNumber(&hrng);
S2 = HAL_RNG_GetRandomNumber(&hrng);

// Синхропосылка сгенерирована
printf ("IV generated!\n");

// засекаем время начала зашифрования
unsigned int time_0 = HAL_GetTick();

// Вызов функции гаммирования - зашифрование
GOST_Gamma (key, S1, S2, OpenData, EncData, size);

// Время зашифрования в миллисекундах
unsigned int time_encryption = HAL_GetTick() - time_0;

time_0 = HAL_GetTick();

// Вызов функции гаммирования - расшифрование
GOST_Gamma (key, S1, S2, EncData, DecData, size);

// Время расшифрования в миллисекундах
unsigned int time_decryption = HAL_GetTick() - time_0;
```

```
    time_0 = HAL_GetTick();

    // Вызов функции выработки имитовставики
    unsigned int MAC[2] = {0};
    GOST_MAC (key, MAC, EncData, size);

    // Время выработки имитовставки в миллисекундах
    unsigned int time_imit = HAL_GetTick() - time_0;

    // Вывод синхопосылки
    printf("IV: %08X %08X %s", S1, S2, "\n");

    // Вывод имитовставки
    printf("MAC: %08X %08X %s", MAC[0], MAC[1], "\n");

    // Сравнение исходного и расшифрованного массивов
    if (Array_Comparing(OpenData, DecData, size) == 1)
      printf("Open & decrypt. comparing...OK!\n");
    else
      printf("Open and decryption data comparing... Error!\n\n");

    // Вывод времени криптографических преобразований в мс

    printf("Encryption time: %u %s", time_encryption, "ms\n");
    printf("Decryption time: %u %s", time_decryption, "ms\n");
    printf("Imit time: %u %s", time_imit, "ms\n\n");

  }

}

// Режим простой замены
// GOST_ECB(ключи, режим(0 - зашифрование, 1 - расшифрование), Данные)
void GOST_ECB (unsigned int * key, short int work_mode, unsigned int * N)
{
    //id-tc26-gost-28147-param-Z
    unsigned int S_box[8][16] =
    {   {0xC,0x4,0x6,0x2,0xA,0x5,0xB,0x9,0xE,0x8,0xD,0x7,0x0,0x3,0xF,0x1},
        {0x6,0x8,0x2,0x3,0x9,0xA,0x5,0xC,0x1,0xE,0x4,0x7,0xB,0xD,0x0,0xF},
        {0xB,0x3,0x5,0x8,0x2,0xF,0xA,0xD,0xE,0x1,0x7,0x4,0xC,0x9,0x6,0x0},
        {0xC,0x8,0x2,0x1,0xD,0x4,0xF,0x6,0x7,0x0,0xA,0x5,0x3,0xE,0x9,0xB},
        {0x7,0xF,0x5,0xA,0x8,0x1,0x6,0xD,0x0,0x9,0x3,0xE,0xB,0x4,0x2,0xC},
        {0x5,0xD,0xF,0x6,0x9,0x2,0xC,0xA,0xB,0x7,0x8,0x1,0x4,0x3,0xE,0x0},
```

```
    {0x8,0xE,0x2,0x5,0x6,0x9,0x1,0xC,0xF,0x4,0xB,0x0,0xD,0xA,0x3,0x7},
    {0x1,0x7,0xE,0xD,0x0,0x5,0x8,0x3,0x4,0xF,0xA,0x6,0x9,0xC,0xB,0x2}};

unsigned int temp;

//32 раунда шифрования
int t;
for (t=0; t<32; t++)
{
    temp = 0;

    //(N1 + key_roud) mod 2^32
    //Сложение блока N1 с раундовым ключом по mod 2^32

    //Для режима зашифрования
    if (!work_mode)
    {
        //Для 1 - 24 раундов K_0 - K_7
        if (t < 24)
        {
            temp = (N[0] + key[t  % 8]) & 0xFFFFFFFF;
        }
        //Для 25 - 32 раундов K_7 - K_0
        if (t >= 24)
        {
            temp = (N[0] + key[7 - (t  % 8)]) & 0xFFFFFFFF;
        }
    }
    //Для режима расшифрования
    if (work_mode)
    {
        //Для 1 - 8 раундов K_0 - K_7
        if (t < 8)
        {
            temp = (N[0] + key[t % 8]) & 0xFFFFFFFF;
        }
        //Для 9 - 32 раундов K_7 - K_0
        if (t >= 8)
        {
            temp = (N[0] + key[7 - (t  % 8)]) & 0xFFFFFFFF;
        }
    }
```

```
        //Подготовка блока размером 32 бита
        unsigned int Vector[8];

        int n;
        for (n=0; n<8; n++)
        {
            //Разбиение блока на 8 4-разрядных векторов
            Vector[n] = (temp >> n * 4) & 0xF;
            //Подстановка в S-блоке
            Vector[n] = S_box[n][Vector[n]];
        }

        temp = 0;
        for (n=0; n<8; n++)
        {
            //Объединение выходных векторов в блок размером 32 бита
            temp = temp | (Vector[n] << n * 4);
        }

        //Циклический сдвиг в сторону старших разрядов на 11 бит
        temp = (temp << 11) | (temp >> 21);

        //Побитовое сложение по mod2 результата с N2
        temp = temp ^ N[1];

        //Для раундов 1 - 31 шифрования
        if (t < 31)
        {
            N[1] = N[0];
            N[0] = temp;
        }
        //Для последнего раунда шифрования
        if (t == 31)
        {
            N[1] = temp;
        }
    } //Цикл: 32 раунда шифрования
}

// Режим гаммирования
// GOST_Gamma(ключи, синхропосылка 1, синхропосылка 1, Входные данные, Выходные
данные, объем данных в байтах)
```

```
void GOST_Gamma (unsigned int * key, unsigned int S1, unsigned int S2, unsigned
int * InData, unsigned int * OutData, short int size)
{
  // синхропосылка разделяется на два блока по 32 бита
  unsigned int N[2];
  N[0] = S1;
  N[1] = S2;

  unsigned int N3, N4;

  // Цикл 32-З
  GOST_ECB(key, 0, N);

  N3 = N[0];
  N4 = N[1];

  // Определение количества блоков по 32 бита
  int i = size / 4;

  while (i > 0)
  {
    // константы C2 и C1
    // 0x01010101;
    // 0x01010104;

    // суммирование по модулю (2^32)
    N3 = (N3 + 0x01010101) & 0xFFFFFFFF;

    // суммирование по модулю (2^32-1)
    N4 = (N4 + 0x01010104) % 0xFFFFFFFF;

    N[0] = N3;
    N[1] = N4;

    // Цикл 32-З
    GOST_ECB(key, 0, N);

    // Гаммирование
    OutData[(size/4) - i] = InData[(size / 4) - i] ^ N[0];
    OutData[(size/4) - i + 1] = InData[(size / 4) - i + 1] ^ N[1];

    // Декремент цикла
    i = i - 2;
```

```
  }
}

// Режим выработки имитовставки
// GOST_MAC(ключи, переменная для имитовставки, данные, объем данных в байтах)
void GOST_MAC (unsigned int * key, unsigned int * MAC, unsigned int * Data,
short int size)
{
  //id-tc26-gost-28147-param-Z
  unsigned int S_box[8][16] =
  {   {0xC,0x4,0x6,0x2,0xA,0x5,0xB,0x9,0xE,0x8,0xD,0x7,0x0,0x3,0xF,0x1},
      {0x6,0x8,0x2,0x3,0x9,0xA,0x5,0xC,0x1,0xE,0x4,0x7,0xB,0xD,0x0,0xF},
      {0xB,0x3,0x5,0x8,0x2,0xF,0xA,0xD,0xE,0x1,0x7,0x4,0xC,0x9,0x6,0x0},
      {0xC,0x8,0x2,0x1,0xD,0x4,0xF,0x6,0x7,0x0,0xA,0x5,0x3,0xE,0x9,0xB},
      {0x7,0xF,0x5,0xA,0x8,0x1,0x6,0xD,0x0,0x9,0x3,0xE,0xB,0x4,0x2,0xC},
      {0x5,0xD,0xF,0x6,0x9,0x2,0xC,0xA,0xB,0x7,0x8,0x1,0x4,0x3,0xE,0x0},
      {0x8,0xE,0x2,0x5,0x6,0x9,0x1,0xC,0xF,0x4,0xB,0x0,0xD,0xA,0x3,0x7},
      {0x1,0x7,0xE,0xD,0x0,0x5,0x8,0x3,0x4,0xF,0xA,0x6,0x9,0xC,0xB,0x2}};

  unsigned int N[2] = {0}, temp;

  for ( int i = 0; i < size / 4; i += 2 )
  {
    //Заполнение регистров N1 и N2
    N[i] = Data[i];
    N[i+1] = Data[i+1];

    //16 раунда шифрования
    int t;
    for (t=0; t<16; t++)
    {
        temp = 0;

        //(N1 + key_roud) mod 2^32
        //Сложение блока N1 с раундовым ключом по mod 2^32
        temp = (N[0] + key[t  % 8]) & 0xFFFFFFFF;

        //Подготовка блока размером 32 бита
        unsigned int Vector[8];

        int n;
        for (n=0; n<8; n++)
        {
```

```
          //Разбиение блока на 8 4-разрядных векторов
          Vector[n] = (temp >> n * 4) & 0xF;
          //Подстановка в S-блоке
          Vector[n] = S_box[n][Vector[n]];
        }

        temp = 0;
        //Объединение выходных векторов в блок размером 32 бита
        for (n=0; n<8; n++)
        {
          temp = temp | (Vector[n] << n * 4);
        }

        //Циклический сдвиг в сторону старших разрядов на 11 бит
        temp = (temp << 11) | (temp >> 21);

        //Побитовое сложение по mod2 результата с N2
        temp = temp ^ N[1];

        N[1] = N[0];
        N[0] = temp;

      } //Цикл: 16 раундов
    MAC[0] ^= N[0];
    MAC[1] ^= N[1];
  }
}

// Сравнение двух массивов
// Array_Comparing (массив Array_1, массив Array_2, размер массива)
int Array_Comparing (unsigned int * Array_1, unsigned int * Array_2, short int
size)
{
  for (int i = 0; i < size; i++)
  {
    if (Array_1[i] != Array_2[i])
      return 1;
  }

  return 0;
}
```

Приложение В

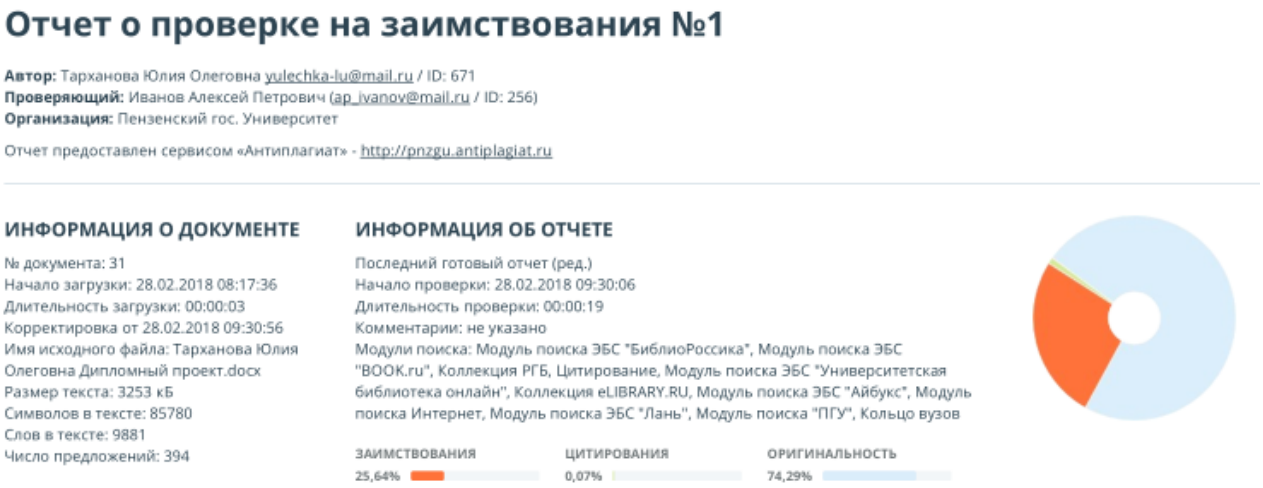
(обязательное)

Протокол проверки ВКР в системе «Антиплагиат»

Протокол

проверки на оригинальность в системе «Антиплагиат.ВУЗ» выпускной квалификационной работы на тему «Программное обеспечение, реализующее функции криптографической защиты информации в малогабаритной встраиваемой аппаратуре передачи данных» студентки 6 курса Ю.О. Тархановой специальности 10.05.02 – «Информационная безопасность телекоммуникационных систем».

Руководитель ВКР ведущий инженер НИО-05 АО «ПНИЭИ» А.С. Прокуров.



Отчет о проверке на заимствования №1

Автор: Тарханова Юлия Олеговна yulechka-lu@mail.ru / ID: 671
Проверяющий: Иванов Алексей Петрович (ap_ivanov@mail.ru / ID: 256)
Организация: Пензенский гос. Университет

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

ИНФОРМАЦИЯ О ДОКУМЕНТЕ

№ документа: 31
Начало загрузки: 28.02.2018 08:17:36
Длительность загрузки: 00:00:03
Корректировка от 28.02.2018 09:30:56
Имя исходного файла: Тарханова Юлия Олеговна Дипломный проект.docx
Размер текста: 3253 кБ
Символов в тексте: 85780
Слов в тексте: 9881
Число предложений: 394

ИНФОРМАЦИЯ ОБ ОТЧЕТЕ

Последний готовый отчет (ред.)
Начало проверки: 28.02.2018 09:30:06
Длительность проверки: 00:00:19
Комментарии: не указано
Модули поиска: Модуль поиска ЭБС "БиблиоРоссика", Модуль поиска ЭБС "BOOK.ru", Коллекция РГБ, Цитирование, Модуль поиска ЭБС "Университетская библиотека онлайн", Коллекция eLIBRARY.RU, Модуль поиска ЭБС "Айбукс", Модуль поиска Интернет, Модуль поиска ЭБС "Лань", Модуль поиска "ПГУ", Кольцо вузов

ЗАИМСТВОВАНИЯ	ЦИТИРОВАНИЯ	ОРИГИНАЛЬНОСТЬ
25,64%	0,07%	74,29%

_____ А.С. Прокуров