

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 100503 –

Информационная безопасность
автоматизированных систем

Специализация -

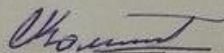
Защищенные автоматизированные
системы управления

ДИПЛОМНЫЙ ПРОЕКТ

на тему

СПЕЦИАЛЬНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ КОМПЛЕКСА СРЕДСТВ
ИНФОРМАЦИОННО-ТЕХНИЧЕСКОГО ВЗАИМОДЕЙСТВИЯ И
ИМИТАЦИИ СОПРЯЖЕНИЯ ПУНКТОВ УПРАВЛЕНИЯ
ПРОТИВОВОЗДУШНОЙ ОБОРОНЫ

Студент

 Калмыков Иван Олегович

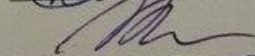
Руководитель

 Зефирова С.Л.

Консультант

 Савель А.И.

Нормоконтролёр

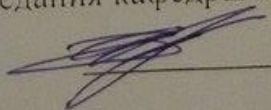
 Иванов А.П.

Рецензент начальник отдела 5800 АО «НПП «Рубин»

 Байкин А.А.

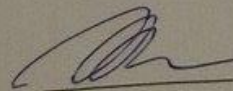
Работа допущена к защите (протокол заседания кафедры от 20.06.16 № 14)

Заведующий кафедрой

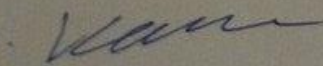
 С.Л. Зефиров

Работа защищена с отметкой Очень хорошо (протокол заседания ГЭК от 29.06.16 № 11)

Секретарь ГЭК

 А.П. Иванов

Пенза 2016

108кл. 

РЕФЕРАТ

Пояснительная записка содержит 60 с., 22 рис., 6 табл., 3 источника, 2 прил.

ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ, КОНФИДЕНЦИАЛЬНОСТЬ, ЦЕЛОСТНОСТЬ, СПЕЦИАЛЬНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, УГРОЗА, АЛГОРИТМ, КРИПТОГРАФИЧЕСКОЕ ПРЕОБРАЗОВАНИЕ.

Объектом исследования в дипломном проекте является специальное программное обеспечение для защищённого информационного обмена.

Цель дипломного проекта – разработка специального программного обеспечения комплекса средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны для защищённого информационного обмена, обеспечивающего поддержку таких свойств информационной безопасности как целостность и конфиденциальность.

В результате выполнения дипломного проекта был произведён анализ угроз, реализованы методы защиты, разработано специальное программное обеспечение для защищённого информационного обмена.

СОДЕРЖАНИЕ

Введение.....	6
1 Обоснование необходимости защиты информации, передающейся посредством специального программного обеспечения	8
2 Анализ угроз информационной безопасности	11
2.1 Идентификация угроз информационной безопасности	11
2.2 Актуализация угроз информационной безопасности	12
3 Оценка рисков информационной безопасности.....	15
3.1 Идентификация рисков.....	15
3.2 Анализ рисков	19
3.3 Сравнительная оценка рисков	19
4 Выбор методов защиты информации.....	21
4.1 Противодействие реализации угрозы прослушивания канала связи.....	21
4.2 Противодействие реализации угрозы искажения информации и навязывания ложной информации противником	21
4.3 Противодействие реализации угрозы повтора ранее переданной информации	22
4.4 Противодействие реализации угрозы непосредственного уничтожения БМ КП.....	23
5 Реализация выбранных методов защиты информации	24
5.1 Программная реализация алгоритма криптографического преобразования.....	24
5.2 Программная реализация алгоритма выработки имитовставки	38

5.3 Программная реализация алгоритма ключевого обмена	38
5.4 Программная реализация защиты от повтора ранее переданной информации	40
6 Реализация специального программного обеспечения	43
6.1 Работа по шлюзу сопряжения алгоритмов обмена данными «АККОРД СС-ПС»	43
6.2 Обеспечение заданной скорости обмена данными по шлюзу сопряжения	43
6.3 Обеспечение совместимости специального программного обеспечения с заданными устройствами	44
6.4 Обеспечение возможности установки нового специального программного обеспечения	46
6.5 Общий алгоритм работы	46
7 Реализация требований к пользовательскому интерфейсу	48
7.1 Реализация отображения информации о состоянии соединения	48
7.2 Реализация отображения статуса работы автоматического и пользовательского ФК специального программного обеспечения	48
7.3 Реализация требований к шрифту, используемого в специальном программном обеспечении	49
7.4 Реализация требований использования фотографических элементов в специальном программном обеспечении	50
7.5 Реализация требований к времени перехода между элементами интерфейса и навигационного меню	51
8 Реализация требований к функциональности программы	52
8.1 Обеспечение выбора имитационного устройства	52

9 Проверка работоспособности приложения на поддерживаемых операционных системах	54
9.1 Проверка работоспособности приложения на операционной системе Windows 7	54
9.2 Проверка работоспособности приложения на операционной системе Ubuntu 14.04	55
9.3 Проверка работоспособности приложения на операционной системе MCBC 5.0.....	55
9.4 Проверка работоспособности приложения на операционной системе Astra Linux	56
Заключение	59
Список использованных источников	60
Приложение А	61
Приложение Б.....	68

ВВЕДЕНИЕ

В связи с необходимостью обучения личного состава Вооруженных Сил Российской Федерации, проводятся специальные занятия на обучающих тренажерах. Обучение на тренажере помогает сформировать базу знаний и навыков, а также отработать порядок действий в экстренных ситуациях.

Тренажеры противовоздушной обороны предназначены для обучения в специальных условиях операторов зенитных ракетных комплексов основным операциям боевой работы при стрельбе по воздушным целям. В том числе: переводу боевого оборудования в дежурное, боевое, и походное состояние; вводу исходных данных для боевой работы; поиску, обнаружению, опознаванию, и захвату цели; определению параметров цели; сопровождению цели и определению момента пуска ракеты; контролю поражения цели.

Для полноты обучения создаются тренажеры и комплексы, которые работают со специальным программным обеспечением, которое обеспечивает полную симуляцию боевых действий.

Обучение на данных комплексах средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны значительно сокращает затраты на обучение личного состава, так как не приходится расходовать ресурс настоящих боевых машин.

Специальное программное обеспечение имитирует канал связи между аппаратурой передачи данных и боевыми машинами. Передача данных производится по защищенному протоколу. Шифрование данных выполняется программным методом, с целью получения наибольшей эргономии всего комплекса и уменьшения его веса.

Комплекс средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны

позволяет выявлять недочеты в работе на стадии обучения. Недочеты отражаются в отчетах о работе программы.

В настоящее время создание таких тренажерных комплексов первостепенная задача. Данные комплексы позволяют не только сократить затраты, но и увеличить скорость и качество обучения личного состава.

Развитие и внедрение новых информационных технологий в военном деле сильно изменяют порядок ведения современных боевых действий, так как повышение эффективности управления войсками достигается за счет увеличения скорости обработки и передачи информации или оперативности системы управления.

1 Обоснование необходимости защиты информации, передающейся посредством специального программного обеспечения

Типовая структура зенитного дивизиона представлена на рисунке 1 и в графической части проекта на плакате 1. Основной передаваемой информацией в данном формировании являются:

- служебная информация;
- радиолокационная информация;
- команды распределения усилий подчиненных средств;
- приказы, распоряжения и документы, отражающие вопросы развёртывания, размещения и функционирования систем противовоздушной обороны.



Рисунок 1 – Типовая структура зенитного дивизиона

Список передаваемой информации представлен в порядке возрастания её значимости. Таким образом, наиболее значимой информацией являются сведения о развёртывании, размещении и функционировании систем противовоздушной обороны.

Разрабатываемое специальное программное обеспечение комплекса средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны работает на закрытом канале связи. Обусловлено это тем, что комплекс работает с реальными боевыми машинами и аппаратами передачи данных. Информация, передаваемая по каналам связи, должна быть зашифрована.

В соответствии с техническим заданием, специальное программное обеспечение позволяет имитировать, как аппаратуру передачи данных, так и любой из имеющихся на вооружении комплекс противовоздушной обороны.

Передаваемые сигналы с пунктов управления секретны на момент передачи данных, потому что несут в себе данные о местоположении объектов, сопровождаемых целях, пунктах управления и т.п.

Также необходим механизм контроля целостности информации, чтобы полностью исключить возможность дистанционного полного дезориентирования операторов комплекса противовоздушной обороны с помощью сложных помех. Исключение «перехвата управления» комплекса противовоздушной обороны посредством инструментов радиоэлектронной борьбы является одной из основных задач.

Комплекс средств имитации информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны использует секретный протокол шифрования данных. Для сопряжения с реальными устройствами, которые располагаются на боевых машинах.

Таким образом, с целью обеспечения полного функционального взаимодействия разрабатываемого комплекса с учебными и боевыми машинами, которые находятся на вооружении Вооруженных Сил Российской Федерации, производится полная имитация настоящей аппаратуры передачи данных, данные которой необходимо шифровать и передавать по специальному протоколу.

Особенности передачи по данному протоколу будут учитываться, при разработке программного обеспечения комплекса средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны.

Необходимость обеспечения защиты информации, которая передается с использованием данного приложения, обусловлена:

- её важностью для обеспечения управления боевыми единицами;
- степенью её секретности (грифом секретности для документов: приказов, распоряжений командования, планов);
- наличием источника угроз – противника, и предпринимаемых им действий, направленных на снижение боеспособности подразделений противовоздушной обороны.

2 Анализ угроз информационной безопасности

2.1 Идентификация угроз информационной безопасности

Рассмотрены угрозы, источником которых является подразделения армии вероятного противника.

Идентификация угроз, исходящих от подразделений армии вероятного противника, производилась с использованием схемы сопрягаемых объектов, связывающей боевые машины (БМ).

В каждой БМ специальное программное обеспечение объединено в локальную сеть коммутатором, подключенным к аппаратуре передачи данных (АПД), через которую производится выдача информации в канал связи. В качестве канала связи может быть использован радиоканал или канал, образованный проводными линиями связи.

На основании данной схемы можно выделить основные угрозы ИБ:

- прослушивание каналов связи противником (СИ1);
- искажение информации и навязывание ложной информации (СИ2);
- повтор ранее переданной информации (СИ3);
- непосредственное уничтожение БМ (СИ4);
- использование противником средств радиоэлектронной борьбы (СИ5).

Реализация угрозы прослушивания канала связи повлечёт за собой раскрытие противником секретной информации о расположении БМ, авиации, безопасность пролёта которой они обеспечивают, а также о целях БМ. Таким образом, будет нарушено свойство конфиденциальности, сохранения которого прописано в техническом задании. Получив такую информацию, противник сможет организовать скрытый беспрепятственный пролёт своей авиации или

уничтожить как БМ противовоздушной обороны, так и авиацию, безопасность полёта которой они обеспечивают.

Реализация угроз искажения и навязывания ложной информации противником и повтора ранее переданной информации способна привести к некорректным действиям экипажей БМ и, как следствие, потере их боеспособности.

Угроза непосредственного уничтожения БМ весьма серьёзна и существует всегда. В условиях постоянно меняющейся боевой обстановки вероятность её реализации высока. Возникновение данной угрозы не связано с некорректными действиями экипажа или командования, и зависит от применяемых противником новейших средств разведки. Реализация данной угрозы повлечёт потерю связи у подчинённых подразделений, и, как следствие, невозможность выполнения ими своих боевых задач.

Использованием противником средств радиоэлектронной борьбы (РЭБ) – угроза, также способная привести к потере связи между БМ. Средства РЭБ всё чаще применяются вооружёнными силами большинства стран, ведётся постоянная работа по их усовершенствованию, поэтому реализация угрозы использования противником средств РЭБ является весьма вероятной.

Схема сопрягаемых объектов представлена на рисунке 2 и в графической части проекта на плакате 2.

Схема взаимодействия имитационного устройства с аппаратурой передачи данных представлена на рисунке 2 и в графической части проекта на плакате 3.

2.2 Актуализация угроз информационной безопасности

По требованиям технического задания сетевое приложение должно обеспечивать конфиденциальность и целостность передаваемых данных, а также защиту от навязывания ложной информации, однако для него не

предусмотрено закрытого канала, поэтому угрозы прослушивания канала связи противником, искажения информации и навязывания им ложной информации являются актуальными.



Рисунок 2 — Схема сопрягаемых объектов

Противник всегда имеет возможность перехвата информации, передающейся по радиоканалу, и, следовательно, возможность её повторной передачи в канал независимо от того закрытый он или нет. Таким образом, угроза повтора ранее переданной информации является актуальной.

Угрозу непосредственного уничтожения БМ необходимо рассматривать как две самостоятельные угрозы: уничтожение батарейного командного пункта (БКП) и командного пункта (КП).

Угроза использования противником РЭБ актуальной не является. В БМ КП и БКП существуют эффективные средства радиоэлектронной защиты (РЭЗ).

К тому же в случае применения противником средств РЭБ при расположении дивизиона на месте есть возможность перехода на проводные линии связи.

Список угроз и их актуальность отражены в таблице 1.

Таблица 1 – Список угроз и их актуальность

Угроза	Актуальность
Прослушивание каналов связи	Актуальна
Искажение информации и навязывание ложной информации	Актуальна
Повтор ранее переданной информации	Актуальна
Непосредственное уничтожение БМ: уничтожение КП уничтожение БКП	
	Актуальна
	Неактуальна
Использование противником средств РЭБ	Неактуальна

3 Оценка рисков информационной безопасности

3.1 Идентификация рисков

В ходе работы были идентифицированы информационные активы, задействованные при работе специального программного обеспечения.

Информационные активы включают информацию, необходимую для осуществления основной задачи специального программного обеспечения, сбор, хранение, обработка и передача которой требует много времени и связана со значительными затратами на приобретение.

Затем для каждого из информационных активов были определены значимые свойства информационной безопасности (ИБ).

Документированный перечень информационных активов представлен в таблице 2.

Таблица 2 – Перечень информационных активов и их значимых свойств

Информационный актив	Значимые свойства информационной безопасности		
	Конфиденциальность	Целостность	Доступность
Приказ о проведении испытаний (учений)	+	+	+
Отчет об испытаниях (учениях)	+	+	+
База данных подключенных устройств	+	+	+
Информация об имитируемых устройствах	+	+	+

Для каждого значимого свойства каждого информационного актива была определена величина нарушения ИБ в соответствии со следующей качественной шкалой:

- минимальная;
- средняя;
- высокая;
- критическая.

Величина потенциального ущерба от нарушения свойств ИБ активов оценивалась по следующим направлениям:

- степень влияния на непрерывность деятельности специального программного обеспечения(направление 1);
- объем финансовых и материальных затрат, необходимых для восстановления свойств ИБ для информационного актива и ликвидации последствий нарушения ИБ (направление 2);
- объем дополнительных людских ресурсов, необходимых для восстановления свойств ИБ для информационного актива и ликвидации последствий нарушения ИБ (направление 3);
- объем дополнительных временных затрат, необходимых для восстановления свойств ИБ для информационного актива и ликвидации последствий нарушения ИБ (направление 4);
- степень нарушения законодательных требований и(или) специальных договорных обязательств (направление 5).

Оценка величины нарушения ИБ была выполнена по каждому из указанных выше направлений.

Далее было произведено оценивание величин нарушения ИБ. Результаты приведены в таблице 3.

Далее был определён перечень вспомогательных активов, необходимых для обработки, хранения и использования основных информационных ресурсов. Результат представлен в таблице 4.

Таблица 3 – Оценивание результатов величин нарушения ИБ

Информационный актив	Свойства безопасности	Величины нарушения ИБ по направлениям оценивания					Обобщенная величина нарушения ИБ	Ценность актива
		1	2	3	4	5		
Информационный актив №1	Конфиденциальность.	с	с	в	с	с	в	В
	Целостность	в	с	с	в	с	в	
	Доступность	м	м	с	с	с	с	
Информационный актив №2	Конфиденциальность.	с	с	в	в	в	в	В
	Целостность	с	с	с	в	в	в	
	Доступность	м	с	в	в	в	в	
Информационный актив №3	Конфиденциальность.	с	с	в	с	в	в	В
	Целостность	с	с	с	с	с	с	
	Доступность	с	с	с	в	в	в	
Информационный актив №4	Конфиденциальность.	с	в	с	с	с	в	В
	Целостность	с	с	с	с	с	с	

	Доступность	с	с	с	с	с	с	
--	-------------	---	---	---	---	---	---	--

Таблица 4 – Перечень вспомогательных активов

Краткое обозначение вспомогательного актива	Описание
BA1	Инсталляционный файл специального программного обеспечения
BA2	Сотрудники и лица, работающие со специальным программным обеспечением
BA3	Автоматизированные рабочие места сотрудников
BA4	База данных подключенных устройств
BA5	Сервер хранения информации

Результатом идентификации основных и вспомогательных активов представлены в таблицах 5 и 6.

Таблица 5 – Идентификация основных активов

Основные активы	Форма хранения активов
Приказ о проведении испытаний (учений)	Бумажный вид
Отчет об испытаниях (учениях)	Электронный и бумажный вид
База данных подключенных устройств	Электронный вид
Информация об имитируемых устройствах	Электронный вид

Таблица 6 – Идентификация вспомогательных активов

Вспомогательные активы	Количество экземпляров
Инсталляционный файл специального программного обеспечения	2 (каждый на отдельном USB носителе)

Сотрудники и лица, работающие со специальным программным обеспечением	30 человек
--	------------

Продолжение таблицы 6 – Идентификация вспомогательных активов

Вспомогательные активы	Количество экземпляров
Автоматизированные рабочие места сотрудников	30
База данных подключенных устройств	1
Сервер хранения информации	1

3.2 Анализ рисков

Анализ риска информационной безопасности

На данном этапе работы был определен уровень риска для всех значимых сценариев инцидентов.

Уровень риска - величина риска, выраженная с точки зрения комбинации последствий и их вероятности.

Для определения уровня рисков были оценены вероятности инцидентов, как сочетания возможностей реализации угроз (возможности угроз) и оценок возможностей использования уязвимостей.

Определение вероятностей инцидента представлено в таблице 5.

Таблица 5 – Определение вероятности инцидента

Возможность угрозы	Низкая			Средняя			Высокая		
Уровни уязвимости	Н	С	В	Н	С	В	Н	С	В
Вероятность сценария инцидента	0	1	2	1	2	3	2	3	4

3.3 Сравнительная оценка рисков

На данном этапе работы было проведено оценивание риска.

Оценивание риска - процесс сравнения результатов анализа риска с критериями риска с целью определения является ли риск и (или) его величина приемлемыми или допустимыми.

Оценивание рисков проводилось по отношению к следующим критериям:

- низкий риск: 1;

- средний риск: 2-3;
- высокий риск: 4.

Результат оценивания рисков представлен в таблице 6.

Таблица 6 - Результат оценивания рисков

Сценарий инцидента	Уровень риска	Оценка риска
СИ1	4	Высокий риск
СИ2	4	
СИ3	4	
СИ4	4	
СИ5	4	

4 Выбор методов защиты информации

4.1 Противодействие реализации угрозы прослушивания канала связи

Для противодействия угрозе прослушивания канала связи противником и сохранения конфиденциальности информации необходимо осуществлять шифрование информации, передаваемой по каналу.

Так как информация, передающаяся с помощью сетевого приложения, может содержать сведения, составляющие государственную тайну, для шифрования необходимо использовать действующий стандарт РФ. Действующим стандартом РФ в области шифрования является «ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования».

4.2 Противодействие реализации угрозы искажения информации и навязывания ложной информации противником

Для противодействия угрозе искажения информации и навязывания ложной информации противником в действующем стандарте ГОСТ 28147-89 предусмотрен алгоритм выработки имитовставки.

Имитовставка – отрезок информации фиксированной длины, полученной по определенному правилу из открытых данных и ключа и добавленной к зашифрованным данным для обеспечения имитозащиты [1].

Имитозащита – защита системы шифрованной связи от навязывания ложных данных [1].

Из определений следует, что имитовставка является механизмом контроля целостности и аутентичности данных.

4.3 Противодействие реализации угрозы повтора ранее переданной информации

Для противодействия реализации угрозы повтора ранее переданной информации в приложениях с клиент-серверной архитектурой обычно используются отметки времени и специальный код – попсе.

Использование отметки времени с её последующей имитозащитой позволяет исключить атаку повтора информации. Отметка времени прикрепляется к передаваемым данным и на приёмной стороне на её основе делается вывод: принята актуальная информация, или же противник провёл атаку повтора.

Nonce — это одноразовый код, выбранный случайным или псевдослучайным образом, который используется для безопасной передачи информации, предотвращая атаку её повторного воспроизведения.

Механизм использования выглядит следующим образом. На передающей стороне к сообщению добавляется случайно сгенерированный nonce, сообщение зашифровывается, вычисляется имитовставка. На приёмной стороне сообщение расшифровывается, проверяется корректность имитовставки и корректность nonce. Nonce является корректным, если он отсутствует в списке ранее принятых. Если nonce нет в списке, значит атаки повтора не было, и список обновляется путём добавления вновь принятого nonce.

Nonce используется один и только один раз, все последующие передачи данных с тем же nonce будут отвергнуты, поэтому злоумышленник, перехватив сообщение с зашифрованными данными, не сможет провести атаку, повторно отправляя перехваченное сообщение.

4.4 Противодействие реализации угрозы непосредственного уничтожения БМ КП

Полностью исключить уничтожение той или иной боевой единицы в условиях вооруженного конфликта невозможно. В данном случае следует уточнить, что будет рассматриваться противодействие угрозе потере связи вследствие уничтожения БМ КП.

По функционалу и возможностям БМ БКП близка к БМ КП и способна выполнять её функции. В связи с этим командиром может быть отдан приказ о порядке перехода функций КП к БКП этого дивизиона или же (по предварительному согласованию) к КП других дивизионов (зенитных или зенитных ракетных). Поэтому решено устанавливать на все БМ специальное программное обеспечение.

5 Реализация выбранных методов защиты информации

5.1 Программная реализация алгоритма криптографического преобразования

В ГОСТ 28147-89 приведена структурная схема алгоритма криптографического преобразования, представленная на рисунке 3 [1].

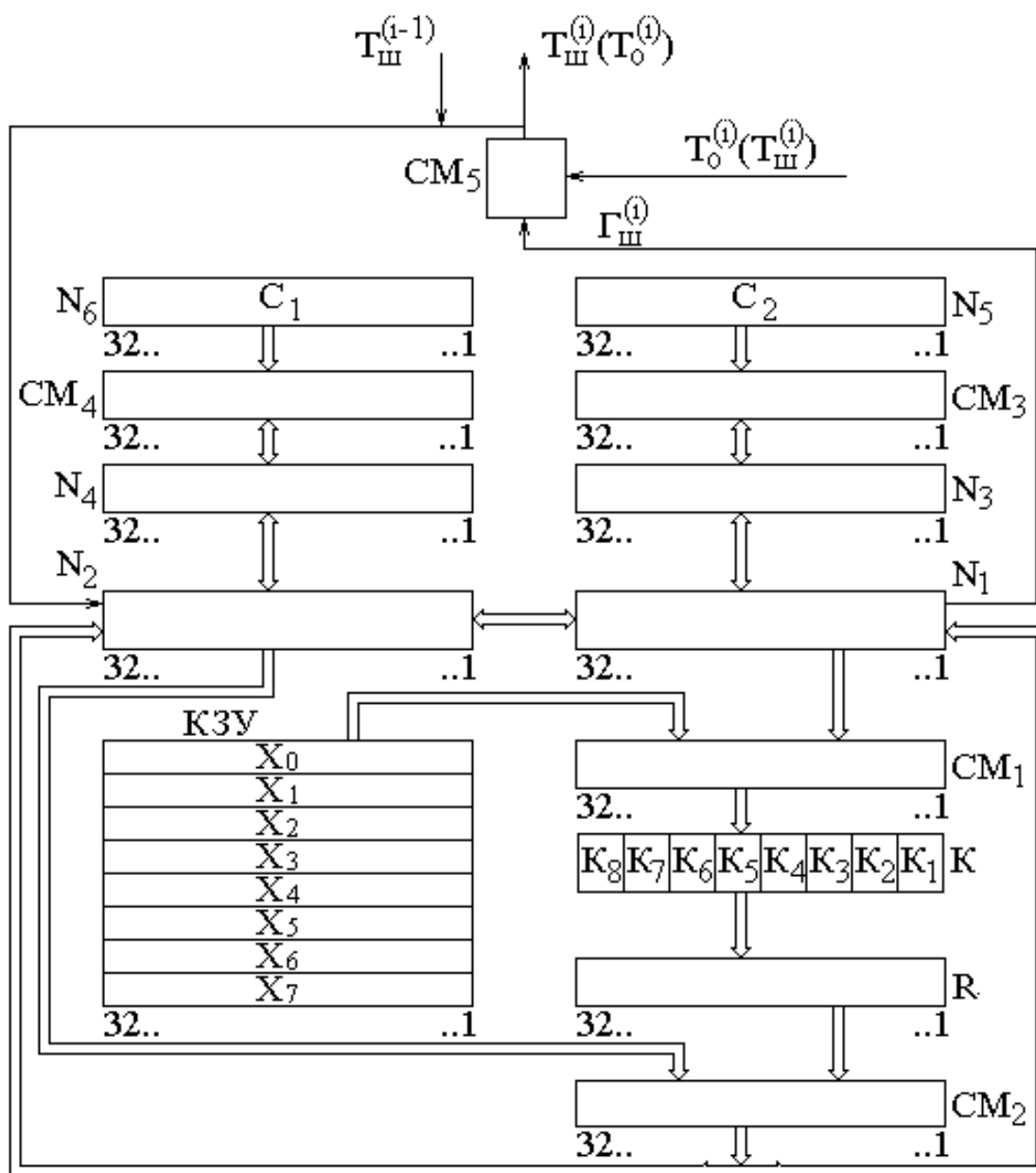


Рисунок 3 – Структура алгоритма криптографического преобразования ГОСТ 28147-89

Она содержит:

- ключевое запоминающее устройство (КЗУ) на 256 бит, состоящее из восьми 32-разрядных накопителей ($X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7$);
- четыре 32-разрядных накопителя (N_1, N_2, N_3, N_4);
- два 32-разрядных накопителя (N_1, N_2) с записанными в них постоянными заполнениями;
- два 32-разрядных сумматора по модулю 2^{32} (CM_1, CM_3);
- 32-разрядный сумматор поразрядного суммирования по модулю 2 (CM_2);
- 32-разрядный сумматор по модулю $(2^{32} - 1)$ (CM_4);
- сумматор по модулю 2, ограничение на разрядность не накладывается (CM_5);
- блок подстановки (K);
- регистр циклического сдвига на 11 шагов в сторону старшего разряда (R).

В криптосхеме предусмотрены четыре вида работы:

- зашифрование (расшифрование) данных в режиме простой замены;
- зашифрование (расшифрование) данных в режиме гаммирования;
- зашифрование (расшифрование) данных в режиме гаммирования с обратной связью;
- режим выработки имитовставки.

Решено создать класс GOST с набором методов, реализующих режимы работы криптосхемы. Класс описан в заголовочном файле GOST.h:

```
#ifndef GOST_H
#define GOST_H
#include<QtGui>
#include<QWidget>
class GOST : public QObject
{
```

```

    Q_OBJECT
public:
    GOST();
    quint32 * X, * secX, *K;
    void setKey(QByteArray key);
    quint32 reverse_block(quint32 X);
    quint64 reverse_block64(quint32 XH, quint32 XL);
    quint64 elementaryCrypt(quint8 mode, bool direction, quint32
N1, quint32 N2);
    QByteArray cryptMode1(bool direction, QByteArray dataIn);
    QByteArray cryptMode2(QByteArray dataIn, quint64 S);
    QByteArray cryptMode3(QByteArray dataIn);
    quint64 imit(QByteArray dataIn);
    QByteArray cryptKey(bool direction, QByteArray dataIn);
    //
    quint32 check;
};

#endif // GOST_H

```

Метод `QByteArray cryptMode1(bool direction, QByteArray dataIn)` выполняет криптографическое преобразование в первом режиме – режиме простой замены. Первый параметр влияет на порядок применения накопителей КЗУ – для зашифрования или для расшифрования. Второй параметр – открытые данные для зашифрования. Возвращает преобразованные данные.

Метод `QByteArray cryptMode2(QByteArray dataIn, quint64 S)` выполняет криптографическое преобразование во втором режиме – режиме гаммирования. Первый параметр – открытые данные для зашифрования, второй – синхропосылка. Возвращает преобразованные данные.

Метод `QByteArray cryptMode3(QByteArray dataIn)` выполняет криптографическое преобразование в первом режиме – режиме гаммирования с обратной связью. Имеет один параметр – открытые данные для зашифрования. Возвращает преобразованные данные.

Метод `quint32 reverse_block(quint32 X)` выполняет перестановку битов в байтовых блоках 32х разрядного числа, являющегося параметром.

Метод *quint64 reverse_block64(quint32 X)* выполняет перестановку битов в байтовых блоках 32х разрядного числа, параметрами являются старшая и младшая части этого числа.

Метод *quint64 elementaryCrypt(quint8 mode, bool direction, quint32 N1, quint32 N2)* выполняет криптографическое преобразование над блоком данных. Параметры: режим работы криптосхемы, порядок применения накопителей КЗУ, накопители N_1, N_2 , указатель на таблицу подстановки.

Метод *void setKey(QByteArray key)* устанавливает переданный параметр как секретный элемент (ключ) для выполнения криптографического преобразования.

Криптосхема, реализующая алгоритм зашифрования (расшифрования) в режиме простой замены представлена на рисунке 4 [1].

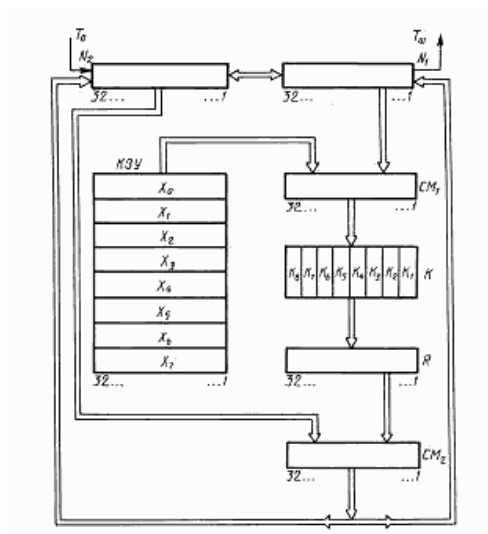


Рисунок 4 – Криптосхема, реализующая алгоритм зашифрования (расшифрования) в режиме простой замены

Открытые данные, подлежащие зашифрованию, разбивают на блоки по 64 бита в каждом. Ввод любого блока открытых данных $T_0 = (a_1(0), a_2(0), \dots, a_{31}(0), a_{32}(0), b_1(0), b_2(0), \dots, b_{31}(0), b_{32}(0))$ в накопители осуществляется таким образом, что в результате получают состояния $N_1 = (a_{32}(0), a_{31}(0), \dots, a_2(0), a_1(0))$ и $N_2 = (b_{32}(0), b_{31}(0), \dots, b_2(0), b_1(0))$.

В КЗУ вводятся 256 бит ключа. Содержимое восьми 32-разрядных накопителей имеет вид:

$$X_0 = (W_{32}, \dots W_1),$$

$$X_1 = (W_{64}, \dots W_{33}),$$

...

$$X_7 = (W_{256}, \dots W_{225}).$$

Алгоритм зашифрования 64-разрядного блока открытых данных в режиме простой замены состоит из 32 циклов.

В первом цикле начальное заполнение накопителя N_1 , суммируется по модулю 2^{32} в сумматоре CM , с заполнением накопителя X_0 при этом заполнение накопителя N_1 , сохраняется.

Результат суммирования преобразуется в блоке подстановки K и полученный вектор поступает на вход регистра R , где циклически сдвигается на одиннадцать шагов в сторону старших разрядов. Результат сдвига суммируется поразрядно по модулю 2 о сумматоре CM_2 с 32-разрядным заполнением накопителя N_2 . Полученный в CM_2 результат записывается в N_1 . при этом старое заполнение N_1 , переписывается в N_2 . Первый цикл заканчивается.

Последующие циклы осуществляются аналогично. при атом во 2-м цикле из КЗУ считывается заполнение X_1 , в 3-м цикле – заполнение X_2 и т.д., в 8-м – X_7 , в циклах с 9-ого по 16-й и с 17-ого по 24-й порядок тот же. В последних восьми циклах с 25-ого по 32-й порядок обратный. Таким обрядом, при зашифровании в 32 циклах осуществляется следующий порядок выбора заполнений накопителей:

$$X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7,$$

$X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7,$

$X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7,$

$X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0.$

В 32 цикле результат из сумматора SM_2 вводится в накопитель N_2 , а в накопителе N_1 сохраняется старое заполнение.

Полученные после 32-го цикла зашифрования заполнения накопителей N_1 и N_2 являются блоком зашифрованных данных, соответствующим блоку открытых данных.

Криптосхема, реализующая расшифрование в режиме простой замены имеет такой же вид. Расшифрование производится аналогичным образом с разницей в порядке выбора накопителей КЗУ – он обратный:

$X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7,$

$X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0,$

$X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0,$

$X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0.$

Решено выделить 32 цикла криптографического преобразования над 64-х битным блоком в отдельную функцию *quint64 elementaryCrypt(quint8 mode, bool direction, quint32 N1, quint32 N2, quint32* K)*, вызываемую внутри метода *cryptMode1(bool direction, QByteArray dataIn)* для зашифрования очередного блока открытых данных:

```
quint64 GOST::elementaryCrypt(quint8 mode, bool direction, quint32
N1, quint32 N2)
{
    quint8 cycles = 32;
    quint32 * locX;
    /*
    0 - data,
    1 - key,
```

```

    2 - imit
    */
    if(mode)
    {
        if(mode == 2)
            cycles = 16;
        locX = secX;
    }
    else
    {
        //cycles = 32;
        locX = X;
    }
    quint32 T;
    //-- 32 rounds -----
    for(quint8 i = 0; i < cycles; i++)
    {
        T = 0;
        if(direction)
        {
            if(i<24)
            {
                T = (N1+locX[i%8]); // % 0x100000000;
            }
            else
            {
                T = (N1+locX[7-(i%8)]); // % 0x100000000;
            }
        }
        else
        {
            if(i<8)
            {
                T = (N1+locX[i%8]); // % 0x100000000;
            }
            else
            {
                T = (N1+locX[7-(i%8)]); // % 0x100000000;
            }
        }
    }
    quint32 FourBitFragments[8] =
    {
        (T & 0xF0000000)>>28,
        (T & 0xF000000)>>24,
        (T & 0xF00000)>>20,
        (T & 0xF0000)>>16,
        (T & 0xF000)>>12,
        (T & 0xF00)>> 8,
        (T & 0xF0)>> 4,
        (T & 0xF)
    };
    for(int j = 0; j<8; j++)

```

```

    {
        FourBitFragments[j] = K[(j + 1)*FourBitFragments[j]];
    }
    T = (FourBitFragments[0]<<28) +
        (FourBitFragments[1]<<24) +
        (FourBitFragments[2]<<20) +
        (FourBitFragments[3]<<16) +
        (FourBitFragments[4]<<12) +
        (FourBitFragments[5]<< 8) +
        (FourBitFragments[6]<< 4) +
        FourBitFragments[7];
    T = (T<<11) | (T>>21);
    T ^= N2;
    if(i != 31)
    {
        N2 = N1;
        N1 = T;
    }
    else
    {
        N2 = T;
    }
} //-- end of rounds -----
quint64 Out64;
Out64 = N1;
Out64 = Out64<<32;
Out64 = Out64|N2;
return Out64;
}

```

Сам метод `QByteArray cryptModel1(bool direction, QByteArray dataIn)` получает в качестве параметров порядок применения заполнений КЗУ и данные для криптографического преобразования, дополняет эти данные нулями, если их размер не кратен восьми, выполняет преобразования над очередным блоком данных и возвращает результат преобразований:

```

QByteArray GOST::cryptModel1(bool direction, QByteArray dataIn)
{
    QByteArray dataOut;
    char *strbuff;
    int L = dataIn.length();
    //adding
    quint32 rest = 0;
    if (L%8 == 0)
        strbuff = (char*)calloc(L, 1);
    else
    {
        rest = 8 - L%8;
    }
}

```

```

        strbuff = (char*)calloc(L+rest, 1);
    }
    for(int h = 0; h < L; h++)
        strbuff[h] = dataIn[h];
    int offset = 0;
    while(offset < L)
    {
        quint32 N1 = 0;
        quint32 N2 = 0;
        N1 = *((quint32*)(strbuff + offset));
        offset +=4;
        N2 = *((quint32*)(strbuff + offset));
        offset +=4;
        N1 = reverse_block(N1);
        N2 = reverse_block(N2);
        //encryption
        quint64 Out = elementaryCrypt(0, direction, N1, N2);
        N1 = (quint32)(Out>>32);
        N2 = (quint32)(Out&0xFFFFFFFF);
        N1 = reverse_block(N1);
        N2 = reverse_block(N2);
        char tempbuff[8];
        tempbuff [7] = N2 >>24;
        tempbuff [6] = (N2 >>16)&0xFF;
        tempbuff [5] = (N2 >>8)&0xFF;
        tempbuff [4] = N2&0xFF;
        tempbuff [3] = N1 >>24;
        tempbuff [2] = (N1 >>16)&0xFF;
        tempbuff [1] = (N1 >>8)&0xFF;
        tempbuff [0] = N1&0xFF;;
        dataOut.append((char*)tempbuff, 8);
    }
    return dataOut;
}

```

Криптосхема, реализующая алгоритм зашифрования (расшифрования) в режиме гаммирования с обратной связью представлена на рисунке 5 [2].

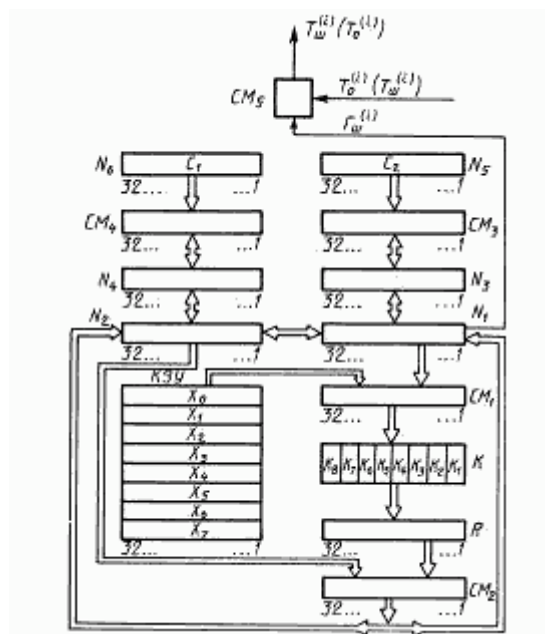


Рисунок 5 – Криптосхема, реализующая алгоритм зашифрования (расшифрования) в режиме гаммирования

Открытые данные, разбитые на 64-битные блоки, зашифровываются в режиме гаммирования путём поразрядного суммирования по модулю 2 в сумматоре CM_5 с гаммой шифра Γ_w , которая вырабатывается блоками по 64 бита. Число двоичных разрядов в очередном блоке открытых данных может быть меньше 64, при этом неиспользованная часть гаммы отбрасывается.

В КЗУ вводятся 256 бит ключа.

В накопители N_1, N_2 вводится 64-разрядная синхропосылка $S = (S_1, \dots, S_{64})$, являющаяся исходным заполнением этих накопителей для последующей выработки блоков гаммы шифра.

Исходное заполнение накопителей N_1, N_2 зашифровывается в режиме простой замены. Результат шифрования переписывается в накопители N_3, N_4 .

Заполнение накопителя N_4 суммируется по модулю $(2^{32} - 1)$ в сумматоре CM_4 с 32-разрядной константой C_1 из накопителя N_6 и записывается в N_4 . Заполнение накопителя N_3 суммируется по модулю $(2^{32} - 1)$ в сумматоре CM_3 с 32-разрядной константой C_3 из накопителя N_5 и записывается в N_3 .

Заполнение N_3 переписывается в N_1 , а заполнение N_4 – в N_2 , при этом заполнения N_3, N_3 сохраняются.

Заполнение N_1 , N_2 образует первый блок гаммы шифра, который суммируется поразрядно по модулю 2 с первым 64-разрядным блоком открытых данных. В результате получается первый 64-разрядный блок зашифрованных данных.

Для получения следующего 64-разрядного блока гаммы шифра заполнение N_4 суммируется по модулю $(2^{32} - 1)$ в сумматоре CM_4 с 32-разрядной константой C_1 из N_6 и результат записывается в N_2 , а заполнение N_3 суммируется по модулю $(2^{32} - 1)$ в сумматоре CM_3 с 32-разрядной константой C_1 из N_5 и результат записывается в N_2 . Аналогичным образом вырабатываются блоки гаммы шифра и зашифровываются блоки открытых данных.

В канал связи передаются синхропосылка и блоки зашифрованных данных.

Алгоритм расшифрования аналогичен алгоритму зашифрования.

Метод `QByteArray cryptMode2(QByteArray dataIn, quint64 S)` получает в качестве параметра данные для криптографического преобразования, дополняет эти данные нулями, если их размер не кратен восьми, выполняет преобразования над очередным блоком данных и возвращает результат преобразований, предварительно отбросив дополненные биты:

```
QByteArray GOST::cryptMode2(QByteArray dataIn, quint64 S)
{
    QByteArray dataOut;
    char *strbuff;
    quint32 L = dataIn.length();
    //adding
    quint32 rest = 0;
    if (L%8 == 0)
        strbuff = (char*)calloc(L, 1);
    else
    {
        rest = 8 - L%8;
        strbuff = (char*)calloc(L+rest, 1);
    }
    for (quint32 h = 0; h < L; h++)
        strbuff[h] = dataIn[h];
    quint64 N64;
    quint32 N1, N2, N3, N4;
    //encrypt S (synhromessage)
```

```

    N64 = elementaryCrypt(0, true, (quint32) S & 0xFFFFFFFF,
(quint32) (S >> 32));
    N3 = (quint32) (N64 >> 32);
    N4 = (quint32) (N64 & 0xFFFFFFFF);
    //initialize const's
    quint32 N6 = C1;
    quint32 N5 = C2;
    //-----
    quint32 offset = 0;
    //L += rest;
    while (offset < L)
    {
        N4 = (N4 + N6) % 0xFFFFFFFF; //mod 2^32-1
        N3 = (N3 + N5) % 0xFFFFFFFF;
        N1 = N3;
        N2 = N4;
        quint64 Data;
        Data = *((quint64*) (strbuff + offset));
        offset += 8;
        Data = reverse_block64((quint32) (Data >> 32),
(quint32) (Data & 0xFFFFFFFF)); //?
        //encryption
        quint64 Gamma;
        Gamma = elementaryCrypt(0, true, N1, N2);

        Gamma = reverse_block64((quint32) (Gamma >> 32),
(quint32) (Gamma & 0xFFFFFFFF));
        Data = Data ^ Gamma;
        Data = reverse_block64((quint32) (Data >> 32),
(quint32) (Data & 0xFFFFFFFF));
        char tempbuff[8];
        tempbuff[7] = (Data >> 56) & 0xFF;
        tempbuff[6] = (Data >> 48) & 0xFF;
        tempbuff[5] = (Data >> 40) & 0xFF;
        tempbuff[4] = (Data >> 32) & 0xFF;
        tempbuff[3] = (Data >> 24) & 0xFF;
        tempbuff[2] = (Data >> 16) & 0xFF;
        tempbuff[1] = (Data >> 8) & 0xFF;
        tempbuff[0] = Data & 0xFF;
        dataOut.append((char*) tempbuff, 8);
    }
    //cut data: delete rest
    dataOut.remove(L, rest);
    return dataOut;
}

```

Так алгоритмы зашифрования и расшифрования аналогичны, параметра, отвечающего за порядок применения заполнений КЗУ нет, и метод *quint64 elementaryCrypt(quint8 mode, bool direction, quint32 N1, quint32 N2)* в теле

данного метода всегда вызывается со вторым параметром, равным *true*, означающим работу по зашифрованию данных.

Криптосхема, реализующая алгоритм зашифрования (расшифрования) в режиме гаммирования с обратной связью представлена на рисунке 6 [3].

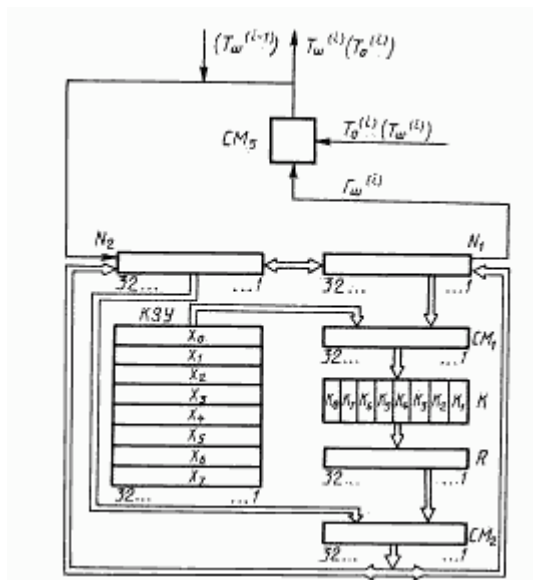


Рисунок 6 – Криптосхема, реализующая алгоритм зашифрования (расшифрования) в режиме гаммирования с обратной связью

Алгоритм зашифрования (расшифрования) в режиме гаммирования с обратной связью отличается от предыдущего алгоритма тем, что очередной блок зашифрованных данных одновременно является также исходным состоянием N_1, N_2 (записывается по обратной связи) для выработки следующего блока гаммы шифра.

```
QByteArray GOST::cryptMode3(QByteArray dataIn, quint64 S)
{
    QByteArray dataOut;
    char *strbuff;

    quint32 L = dataIn.length();

    //adding
    quint32 rest = 0;
    if (L%8 == 0)
        strbuff = (char*)calloc(L, 1);
    else
    {
        rest = 8 - L%8;
```

```

    strbuff = (char*)calloc(L+rest, 1);
}

for(uint32 h = 0; h < L; h++)
    strbuff[h] = dataIn[h];

uint32 N1, N2;

N1 = (uint32) (S>>32);
N2 = (uint32) (S&0xFFFFFFFF);

uint32 offset = 0;
while(offset < L)
{
    uint64 Data;

    Data = *((uint64*)(strbuff + offset));
    offset +=8;

    uint64 Gamma;
    Gamma = elementaryCrypt(0, true, N1, N2, (uint32*)K);

    uint64 mask;

    //cut Gamma
    if ((rest) && (offset > L))
    {
        mask = 0xFFFFFFFFFFFFFFFF >> (rest*8);
        Gamma &= mask;
    }
    /**/

    Data = Data ^ Gamma;

    //Data will next N1 N2
    N1 = (uint32) (Data>>32);
    N2 = (uint32) (Data&0xFFFFFFFF);

    char tempbuff[8];
    tempbuff [7] = (Data >>56);
    tempbuff [6] = (Data >>48)&0xFF;
    tempbuff [5] = (Data >>40)&0xFF;
    tempbuff [4] = (Data >>32)&0xFF;

    tempbuff [3] = (Data >>24)&0xFF;
    tempbuff [2] = (Data >>16)&0xFF;
    tempbuff [1] = (Data >> 8)&0xFF;
    tempbuff [0] = Data&0xFF;

    dataOut.append((char*)tempbuff, 8);
}

```

```
return dataOut;
}
```

5.2 Программная реализация алгоритма выработки имитовставки

В класс GOST добавлен метод выработки имитовставки *quint64 imit(QByteArray dataIn)*. Метод имеет один параметр – открытые данные, для которых будет осуществляться выработка имитовставки.

Описание класса в заголовочном файле GOST.h теперь выглядит следующим образом:

```
#ifndef GOST_H
#define GOST_H
#include<QtGui>
#include<QWidget>
class GOST : public QWidget
{
    Q_OBJECT
public:
    GOST();
    QLineEdit * openText, *cryptoText;
    QPushButton * encryptButton;
    QTextEdit * debug;
    QByteArray dataIn, dataOut;
    bool mode;//direction;
    int reverse_bit(int X);
    int reverse_block(int X);
private slots:
    void cryptMode1(/*bool direction*/);
    /*/
    void cryptMode2(bool direction);
    void cryptMode3(bool direction);
```

```

        void cryptMode4(bool direction); //may be joined with 2 or 3
        /**/
};

#endif // GOST_H

class GOST : public QObject
{
    Q_OBJECT
public:
    GOST();
    quint32 * X, * secX, *K;
    void setKey(QByteArray key);
    quint32 reverse_block(quint32 X);
    quint64 reverse_block64(quint32 XH, quint32 XL);
    quint64 elementaryCrypt(quint8 mode, bool direction, quint32
N1, quint32 N2);
    QByteArray cryptModel1(bool direction, QByteArray dataIn);
    QByteArray cryptMode2(QByteArray dataIn, quint64 S);
    QByteArray cryptMode3(QByteArray dataIn);
    quint64 imit(QByteArray dataIn);
};

#endif // GOST_H

```

Метод осуществляющий выработку имитовставки:

```

quint64 GOST::imit(QByteArray dataIn)
{
    char *strbuff;
    int L = dataIn.length();
    //adding
    quint32 rest = 0;
    if (L%8 == 0)
        strbuff = (char*)calloc(L, 1);
    else
    {

```

```

        rest = 8 - L%8;
        strbuff = (char*)calloc(L+rest, 1);
    }
    for(int h = 0; h < L; h++)
        strbuff[h] = dataIn[h];
    quint64 N64, T;
    int offset = 0;
    T = *((quint64*)(strbuff + offset));
    offset +=8;
    while(offset < L)
    {
        T = elementaryCrypt(2, true, (quint32)(T>>32),
(quint32)(T&0xFFFFFFFF));

        N64 = *((quint64*)(strbuff + offset));
        offset +=8;
        T = T ^ N64;
    }
    return T;
}

```

Полный исходный код модуля gost (файлы gost.h и gost.cpp) представлен в Приложении А.

5.3 Программная реализация алгоритма ключевого обмена

Согласно рассматриваемому стандарту может быть организована следующая ключевая система:

- выработка ключей (заполнений КЗУ) в режиме простой замены,
- зашифрование ключей в режиме простой замены с обеспечением имитозащиты,
- передача по каналу связи.

При этом заполнение блока подстановки K и заполнение КЗУ является долговременным ключевым (а значит секретным) элементом, общим для сети ЭВМ.

Структура данных представлена в графической части на плакате 4.

Добавлены методы для установки общего секретного элемента и его зашифрования (расшифрования): *void setSecretKey(quint32 * key, quint32 *pK)* и *QByteArray cryptKey(bool direction, QByteArray dataIn)*.

Описание класса в заголовочном файле GOST.h теперь выглядит следующим образом:

```
#ifndef GOST_H
#define GOST_H
#include<QtGui>
#include<QWidget>
class GOST : public QObject
{
    Q_OBJECT
public:
    GOST();
    quint32 * X, * secX, *K;
    void setKey(QByteArray key);
    void setSecretKey(quint32 * key, quint32 *pK);
    quint32 reverse_block(quint32 X);
    quint64 reverse_block64(quint32 XH, quint32 XL);
    quint64 elementaryCrypt(quint8 mode, bool direction, quint32
N1, quint32 N2);
    QByteArray cryptModel(bool direction, QByteArray dataIn);
    QByteArray cryptMode2(QByteArray dataIn, quint64 S);
    QByteArray cryptMode3(QByteArray dataIn);
    quint64 imit(QByteArray dataIn);
    QByteArray cryptKey(bool direction, QByteArray dataIn);
    //
```

```
};
#endif // GOST_H
```

Ключ вырабатывается сервером по запросу от клиента в соответствие с приведёнными выше требованиями стандарта:

```
//get encrypted key
quint64 X[8];
for(int j = 0; j < 8; j++)
{
    qsrand(QTime::currentTime().msec() + j);
    X[j] = (qrand() << 16);
    qsrand(QTime::currentTime().second() +
QTime::currentTime().msec() - j);
    X[j] |= qrand();
}
QByteArray keyArray((char*)X, 8*8);
//make key
keyArray = gostItem.cryptKey(true, keyArray);
//imit
quint64 imit = gostItem.imit(keyArray);
//encrypt key
keyArray = gostItem.cryptKey(true, keyArray);
```

5.4 Программная реализация защиты от повтора ранее переданной информации

Для реализации защиты от повтора данных используется отметка времени и поппе. Функции генерации поппе на серверной и клиентской частях приложения аналогичны:

```
inline quint64 myServer::synhroNonce()
{
    quint64 sn;

    qsrand(QTime::currentTime().msec());
```

```

    sn = (qrand() << 32);

    qsrand(QTime::currentTime().second() +
QTime::currentTime().msec());

    sn |= qrand();

    return sn;
}

```

Сгенерированный попсе добавляется к передаваемым данным, при этом обеспечивается имитозащита самого попсе и отметки времени.

Генерация ключа и защита его от повтора ключа:

```

quint64 X[8];
for(int j = 0; j < 8; j++)
{
    qsrand(QTime::currentTime().msec() + j);
    X[j] = (qrand() << 16);
    qsrand(QTime::currentTime().second() +
QTime::currentTime().msec() - j);
    X[j] |= qrand();
}
QByteArray keyArray((char*)X, 8*8);
//make key
keyArray = gostItem.cryptKey(true, keyArray);
//synhro nonce
quint64 snonce = synhroNonce();
//imit
time = QTime::currentTime();
quint64 imit = gostItem.imit(time.toString().toAscii() + keyArray
+ QByteArray::number(snonce));
//encrypt key
keyArray = gostItem.cryptKey(true, keyArray);
//send to initiator

```

```

sendKeyToClient(pclientSocket, time, keyArray, *addrCorrTo,
snonce, imit);
//searching reciever's socket in the list
for (int i=0; i < users.size(); i++)
{
    if (users[i].pSocket->peerAddress() == *(addrCorrTo))
    {
        pclientSocket = users[i].pSocket;
        sendKeyToClient(pclientSocket, time, keyArray,
*addrCorrFrom, snonce, imit);
        break;
    }
    else //response to sender if correspondent not online
        if (i == (users.size()-1))
        {
            cmd = 127;
            sendToClient(pclientSocket, NULL, NULL);
        }
}

```

Алгоритм ключевого обмена представлен на плакате 5.

Защита от повтора сообщения:

```

quint64 snonce = synhroNonce();

time =QTime::currentTime();

gostItem.setKey(seances[j].key);

quint64 imit = gostItem.imit(time.toString().toAscii() + msg +
QByteArray::number(snonce));

msg = gostItem.cryptMode2(msg, snonce);

```

Защита от повтора файла:

```

time = QTime::currentTime();

gostItem.setKey(seances[j].key);

```

```
imit = gostItem.imit(time.toString().toAscii() +  
fileInfo.fileName().toAscii() + fileData +  
QByteArray::number(snonce));  
  
fileData = gostItem.cryptMode2(fileData, snonce);
```

6 Реализация специального программного обеспечения

6.1 Работа по шлюзу сопряжения алгоритмов обмена данными «АККОРД СС-ПС»

Работа по шлюзу сопряжения алгоритмов обмена данными «АККОРД СС-ПС» представлена на рисунке 7.

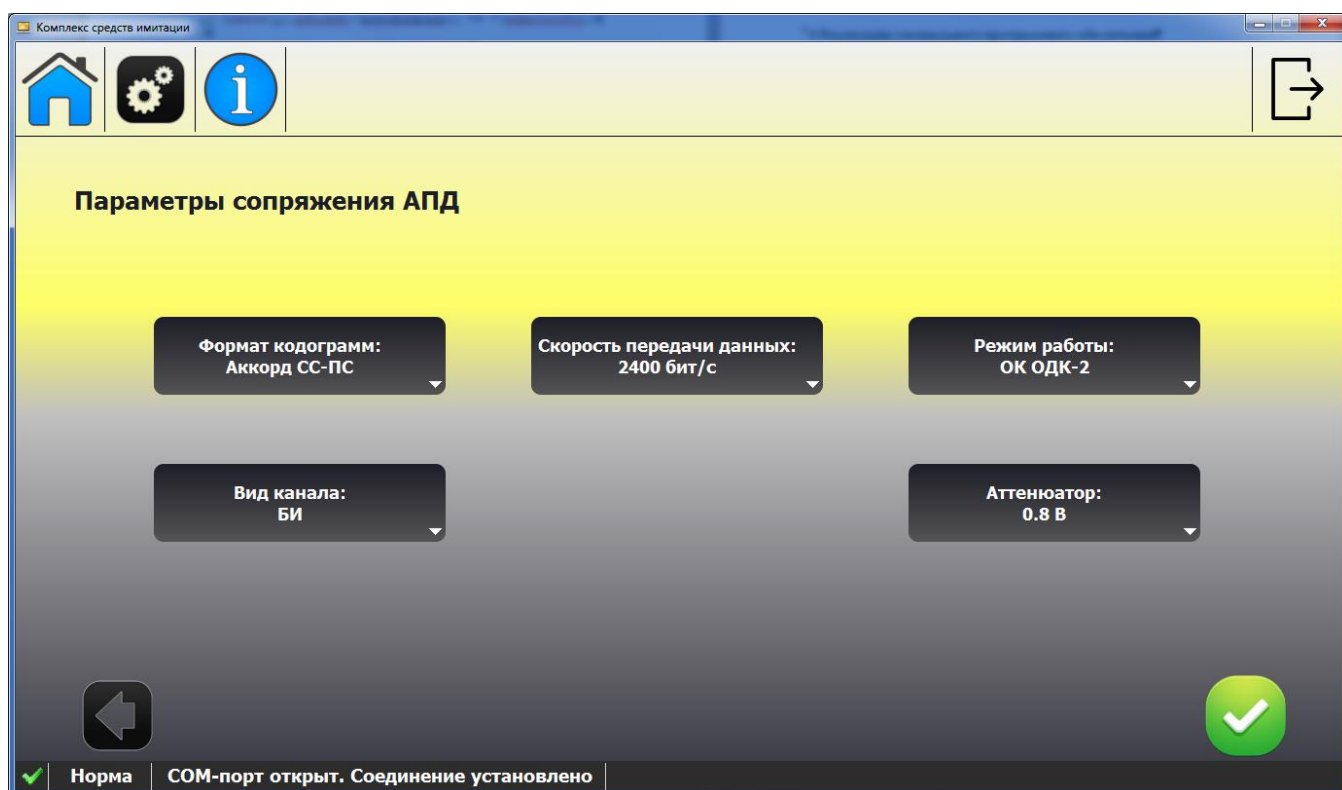


Рисунок 7 - Работа по шлюзу сопряжения алгоритмов обмена данными «АККОРД СС-ПС».

Специальное программное обеспечение позволяет выбрать формат кодограмм, необходимую (заданную) скорость передачи данных, режим работы, вид канала.

Программная реализация модуля сопряжения алгоритмов обмена данными «АККОРД СС-ПС» представлена в приложении Б.

6.2 Обеспечение заданной скорости обмена данными по шлюзу сопряжения

Обеспечение заданной скорости обмена данными по шлюзу сопряжения представлено на рисунке 8.

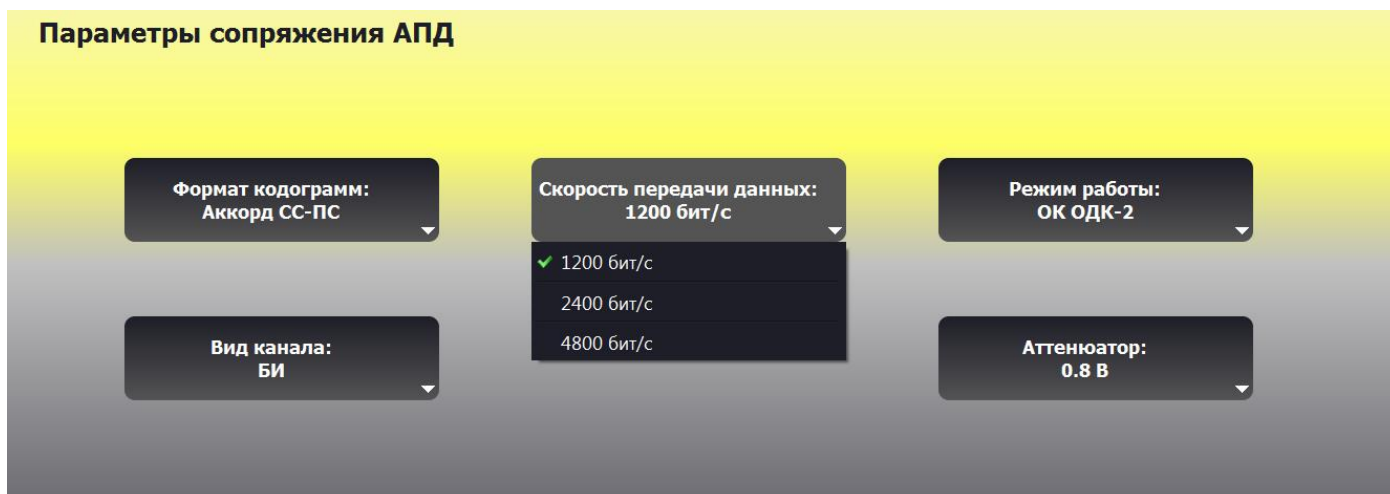


Рисунок 8 - Обеспечение заданной скорости обмена данными по шлюзу сопряжения.

Специальное программное обеспечение позволяет настраивать необходимую скорость для передачи данных по шлюзу сопряжения.

Скорость выбирается в ручном режиме. Задать скорость можно значениями:

- 1200 бит/с;
- 2400 бит/с;
- 4800 бит/с.

Программная реализация модуля обеспечения заданной скорости обмена данными по шлюзу сопряжения представлена в приложении Б.

6.3 Обеспечение совместимости специального программного обеспечения с заданными устройствами

Обеспечение совместимости специального программного обеспечения с заданными устройствами представлено на рисунке 9. Совместимость изделий является тестовым вариантом. При прохождении тестирования программы у заказчика изделия необходимо будет заменить на действующие.

Выбор имитируемого и проверяемого устройства происходит на экране выбора изделий. Проверка сопряжения двух одинаковых изделий представлена на рисунке 10.

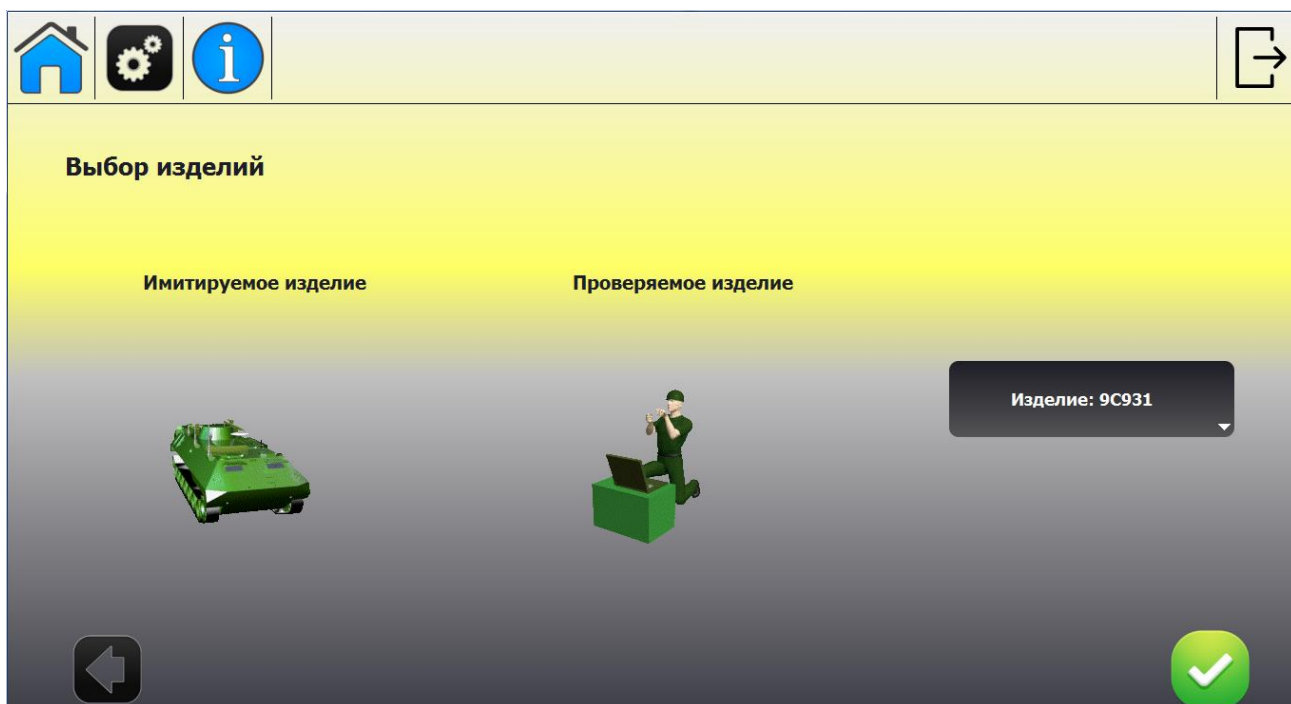


Рисунок 9 - Обеспечение совместимости специального программного обеспечения с заданными устройствами

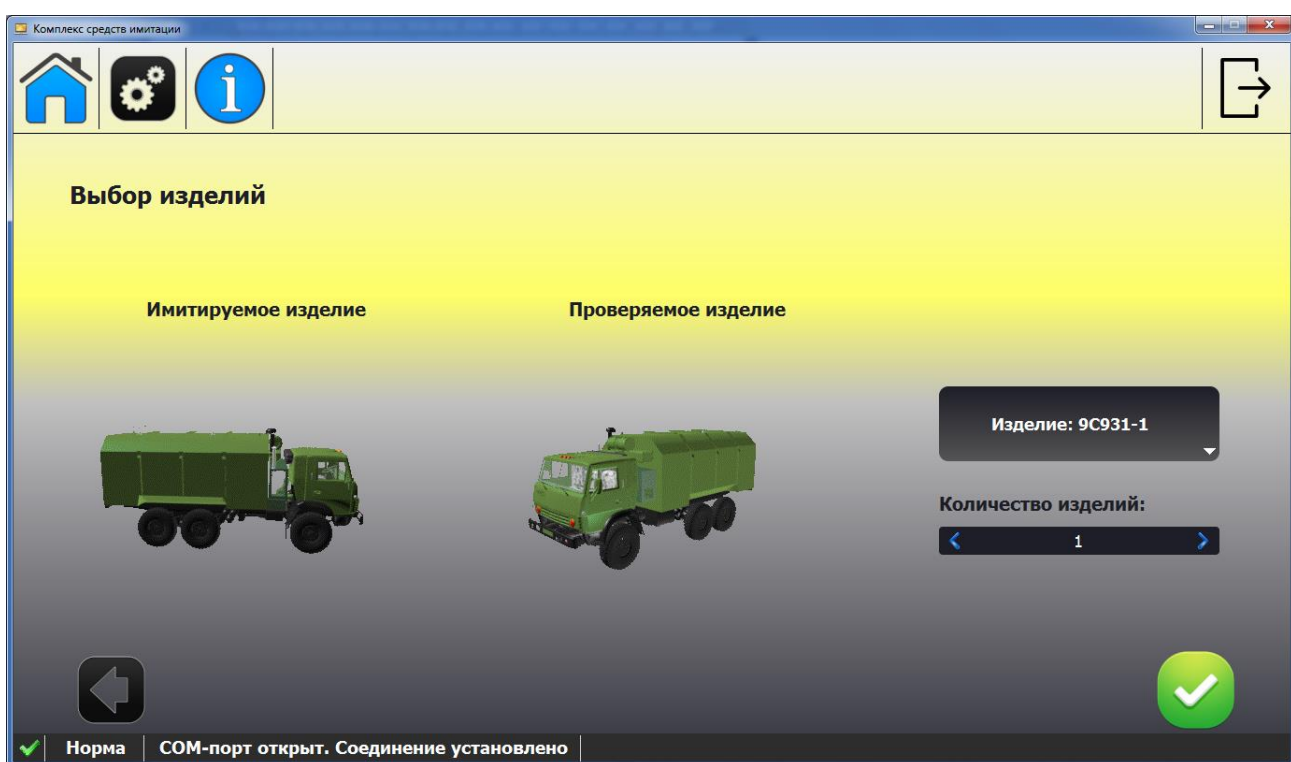


Рисунок 10 - Обеспечение совместимости специального программного обеспечения с заданными устройствами

6.4 Обеспечение возможности установки нового специального программного обеспечения

Обеспечение возможности установки нового специального программного обеспечения представляется путём запуска `install.bat` файла, где указывается путь установки нового специального программного обеспечения на устройство.

`install.bat:`

```
start /min A:\install\newVersionInstaller.exe "ftp://admin:pas@serv"
```

6.5 Общий алгоритм работы

Сеанс работы по закрытому каналу можно обобщить в следующий словесный алгоритм:

- запуск специального программного обеспечения;
- проверка наличия общего секретного элемента;
- старт специального программного обеспечения на выбранном порте;
- подсоединение аппаратуры передачи данных изделия к специальному программному обеспечению;
- высылка специальным программным обеспечением тестовой комбинации;
- выбор имитируемого изделия;
- запрос к специальному программному обеспечению на сеансовый ключ;
- получение сеансового ключа, переход в закрытый канал;
- обмен данными;
- разъединение.

UML-диаграмма компонентов специального программного обеспечения представлена на плакате 6.

Разъединение происходит при нажатии на соответствующую кнопку или при закрытии приложения. Также рассмотрен вариант аварийного завершения клиентской части приложения. Для этого на сервере предусмотрен слот, срабатывающий по таймеру (500 мс) и производящий проверку состояния соединений с аппаратурой передачи данных, и в случае обрыва соединения сообщаящий об этом.

7 Реализация требований к пользовательскому интерфейсу

7.1 Реализация отображения информации о состоянии соединения

Отображение информации о состоянии соединения представлено на рисунке 11.

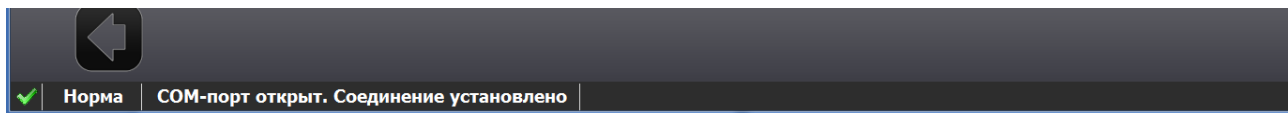


Рисунок 11 - Отображение информации о состоянии соединения.

В нижнем навигационном меню находится строка состояния. В строке состояния отображается состояние соединения, статус проверки линии связи и статус проверки COM-порта.

7.2 Реализация отображения статуса работы автоматического и пользовательского ФК специального программного обеспечения

Отображение статуса работы автоматического и пользовательского ФК специального программного обеспечения представлено на рисунке 12 и рисунке 13

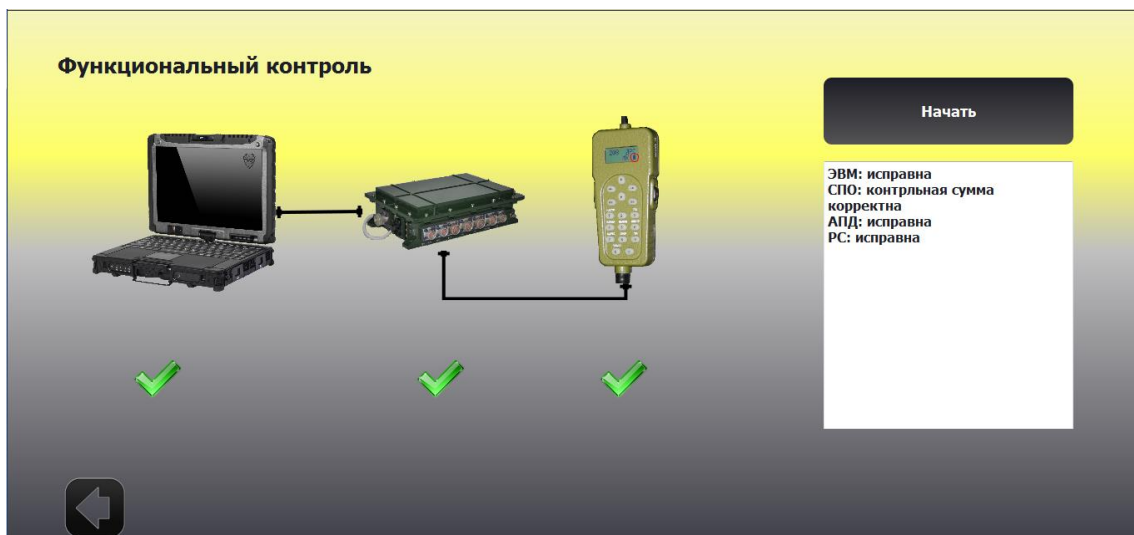


Рисунок 12- Отображение статуса работы автоматического ФК специального программного обеспечения.

Функциональный контроль автоматического режима проверяет ЭВМ на исправность. специального программного обеспечения проверяется на корректность контрольной суммы. Аппаратура передачи данных проверяется на исправность. Радиостанция проверяется на исправность.

В ручном режиме, при выборе имитируемых устройств необходимо самостоятельно перепроверить все исходные заданные данные. Исходные данные располагаются в верхнем навигационном меню. Исходные данные включают в себя:

- Имитируемое изделие;
- Проверяемое изделие;
- Алгоритм АПД;
- Тип канала;
- Скорость передачи данных;
- Параметры радиостанции;
- Тип радиостанции;
- Кодограммы.

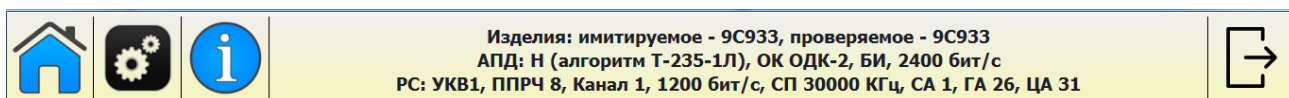


Рисунок 13 – Пользовательский функциональный контроль.

7.3 Реализация требований к шрифту, используемого в специальном программном обеспечении

Требования к шрифту реализованы в полном объёме. Размер не менее 18 пикселей. Результат представлен на рисунке 14.

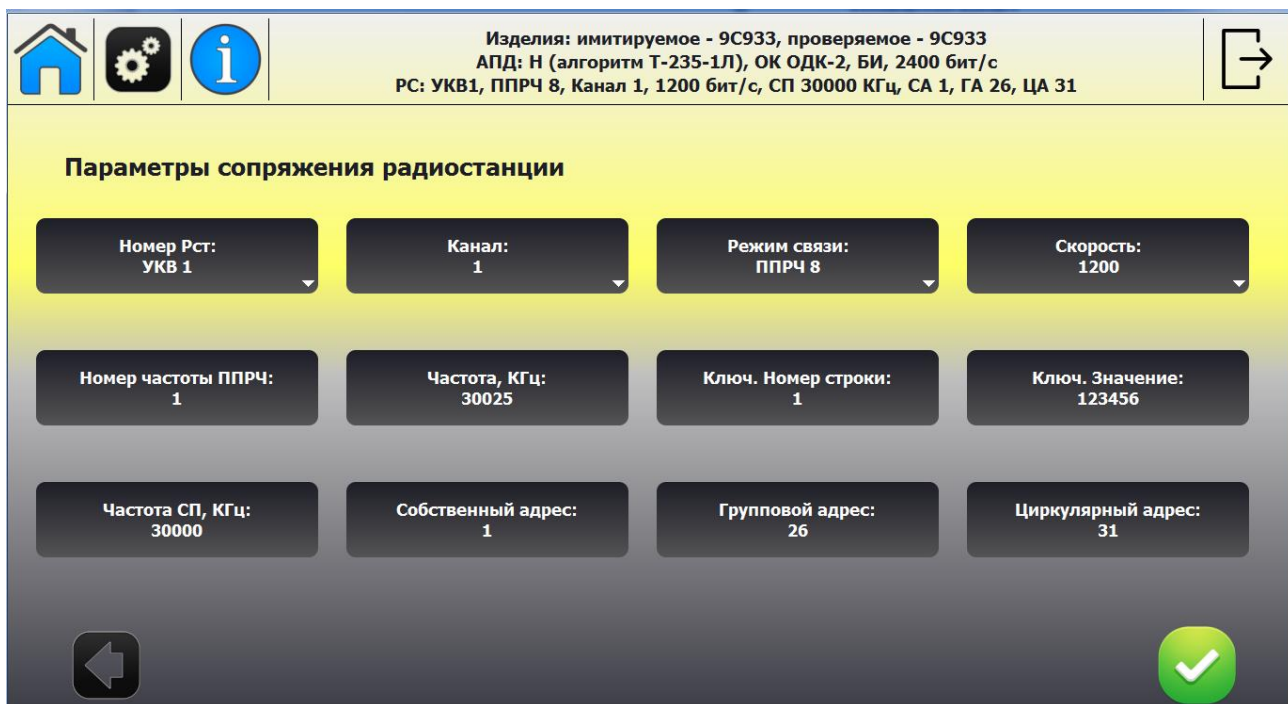


Рисунок 14 – Демонстрация шрифта, используемого в специальном программном обеспечении.

Программная реализация представлена в приложении Б.

7.4 Реализация требований использования фотографических элементов в специальном программном обеспечении

Реализация требований к использованию фотографических элементов в специальном программном обеспечении представлена на рисунке 15.



Рисунок 15 - Реализация требований к использованию фотографических элементов в специальном программном обеспечении.

На каждое имитируемое и проверяемое изделие соответствует один фотографический анимированный элемент. Все элементы располагаются внутри программы и загружаются вместе с запуском программы, что позволяет повысить эффективность работы специального программного обеспечения.

7.5 Реализация требований к времени перехода между элементами интерфейса и навигационного меню

Панель навигации открывается быстрее, чем установлено в техническом задании из-за простоты интерфейса. Исходный код с проверкой времени представлен в приложении Б.

Интерфейс представлен на рисунке 16.

Навигация

Имитируемое изделие

Координаты точки стояния:	X	<input type="text" value="500000"/>	Номер зоны	<input type="text" value="7"/>	Y	<input type="text" value="500000"/>
Координаты реперной точки:	X	<input type="text" value="500000"/>	Номер зоны	<input type="text" value="7"/>	Y	<input type="text" value="500000"/>

Проверяемые изделия

1 Координаты точки стояния:	X	<input type="text" value="н/д"/>	Номер зоны	<input type="text" value="н/д"/>	Y	<input type="text" value="н/д"/>
2 Координаты точки стояния:	X	<input type="text" value="н/д"/>	Номер зоны	<input type="text" value="н/д"/>	Y	<input type="text" value="н/д"/>
3 Координаты точки стояния:	X	<input type="text" value="н/д"/>	Номер зоны	<input type="text" value="н/д"/>	Y	<input type="text" value="н/д"/>

Navigation buttons: Back (left arrow) and Confirm (green checkmark).

Рисунок 16 – Навигационное меню.

8 Реализация требований к функциональности программы

8.1 Обеспечение выбора имитационного устройства

Специальное программное обеспечение позволяет производить выбор имитационного устройства. Выбор устройства представлен на рисунке 17.

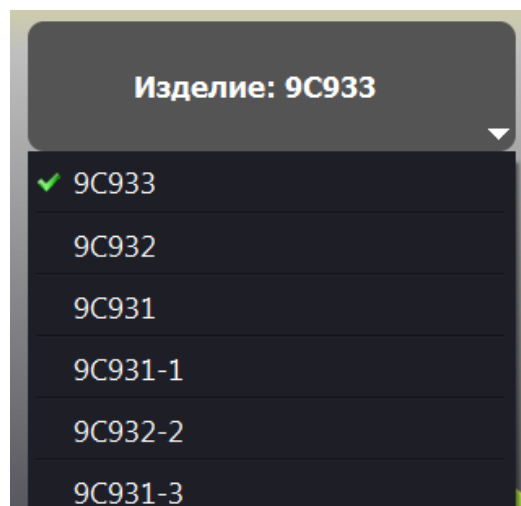


Рисунок 17 – Выбор имитационного устройства.

Исходный код выбора имитационного устройства представлен в приложении Б.

8.2 Обеспечение настройки параметров аппаратуры передачи данных и радиостанции

Специальное программное обеспечение позволяет настраивать параметры аппаратуры передачи данных и радиостанции. Настройка представлена на рисунке 19.

В настройках параметров радиостанции указываются:

- номер радиостанции;
- номер канала;
- режим связи;

- скорость передачи данных;
- номер частоты псевдослучайной перестройки радиочастоты;
- частота;
- ключевой номер строки;
- ключевое значение;
- собственный адрес;
- групповой адрес;
- циркулярный адрес.

Параметры сопряжения радиостанции

Номер Рст: УКВ 1	Канал: 1	Режим связи: ППРЧ 8	Скорость: 1200
Номер частоты ППРЧ: 1	Частота, КГц: 30025	Ключ. Номер строки: 1	Ключ. Значение: 123456
Частота СП, КГц: 30000	Собственный адрес: 1	Групповой адрес: 26	Циркулярный адрес: 31

←

✓

Норма | СОМ-порт открыт. Соединение установлено

Рисунок 19 - Настройка параметров аппаратуры передачи данных

Исходный код настройки параметров аппаратуры представлен в приложении Б.

9 Проверка работоспособности приложения на поддерживаемых операционных системах

9.1 Проверка работоспособности приложения на операционной системе Windows 7

Одним из требований технического задания является возможность работы приложения на компьютерах под управлением ОС Windows 7. Для осуществления проверки, специальное программное обеспечение было запущено в системе Windows 7. Результат проверки представлен на рисунке 19.

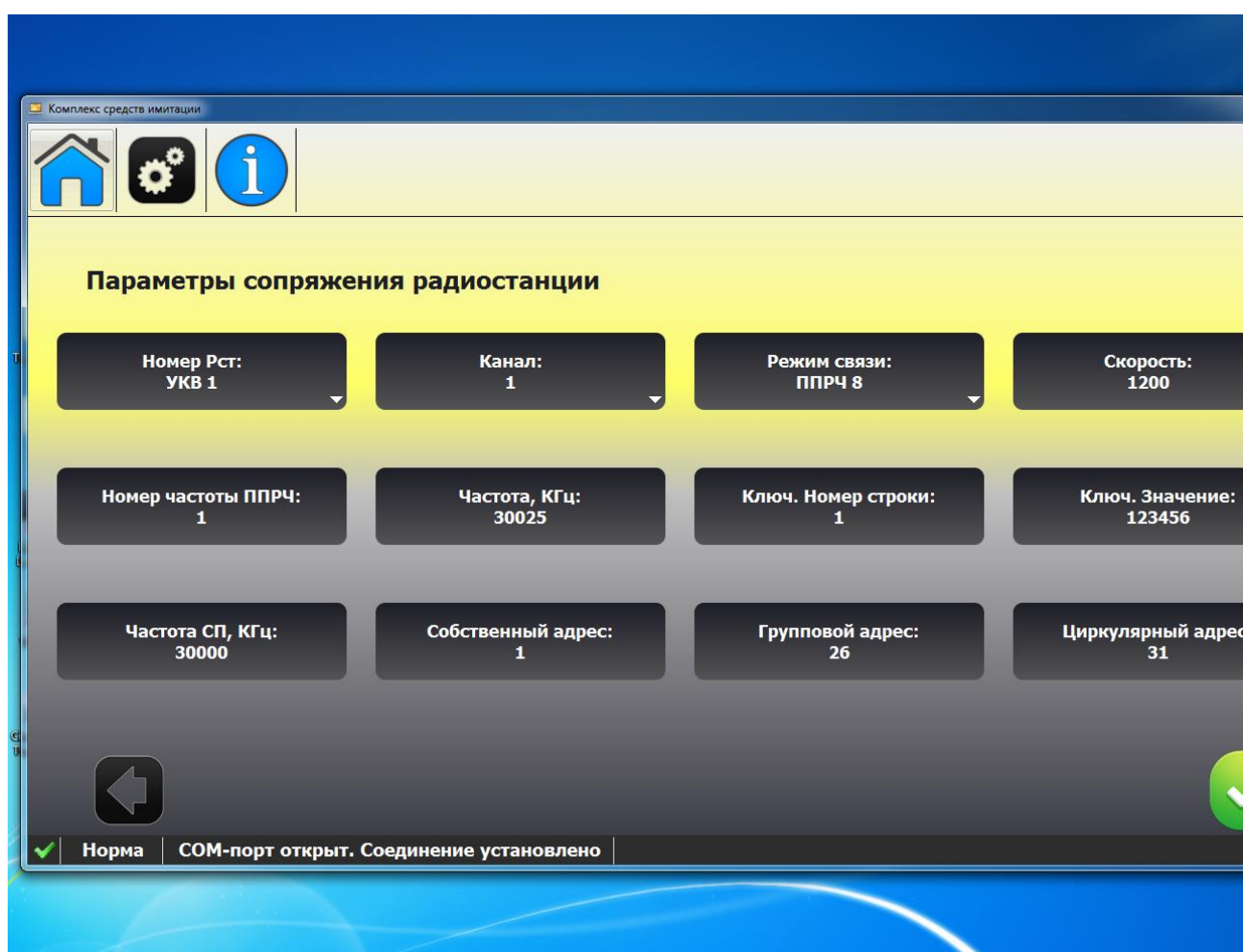


Рисунок 19 — Работоспособность специального программного обеспечения в ОС Windows 7

9.2 Проверка работоспособности приложения на операционной системе Ubuntu 14.04

Одним из требований технического задания является возможность работы приложения на компьютерах под управлением Ubuntu 14.04. Для осуществления проверки, специальное программное обеспечение было запущено в системе Ubuntu 14.04. Результат проверки представлен на рисунке 20.

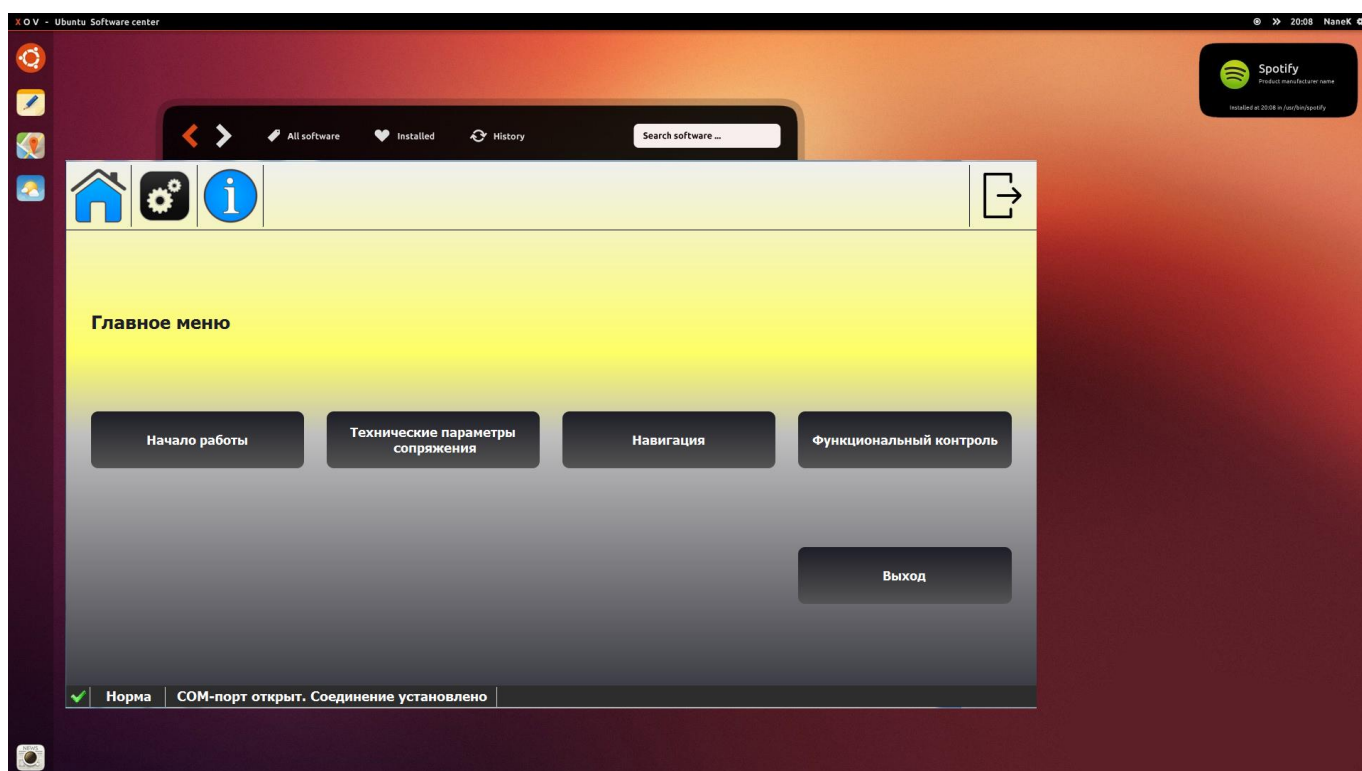


Рисунок 20 - Работоспособность специального программного обеспечения в ОС Ubuntu 14.04

9.3 Проверка работоспособности приложения на операционной системе MCBC 5.0

Одним из требований технического задания является возможность работы приложения на компьютерах под управлением MCBC 5.0. Для осуществления проверки, специальное программное обеспечение было запущено в системе MCBC 5.0. Результат проверки представлен на рисунке 21.

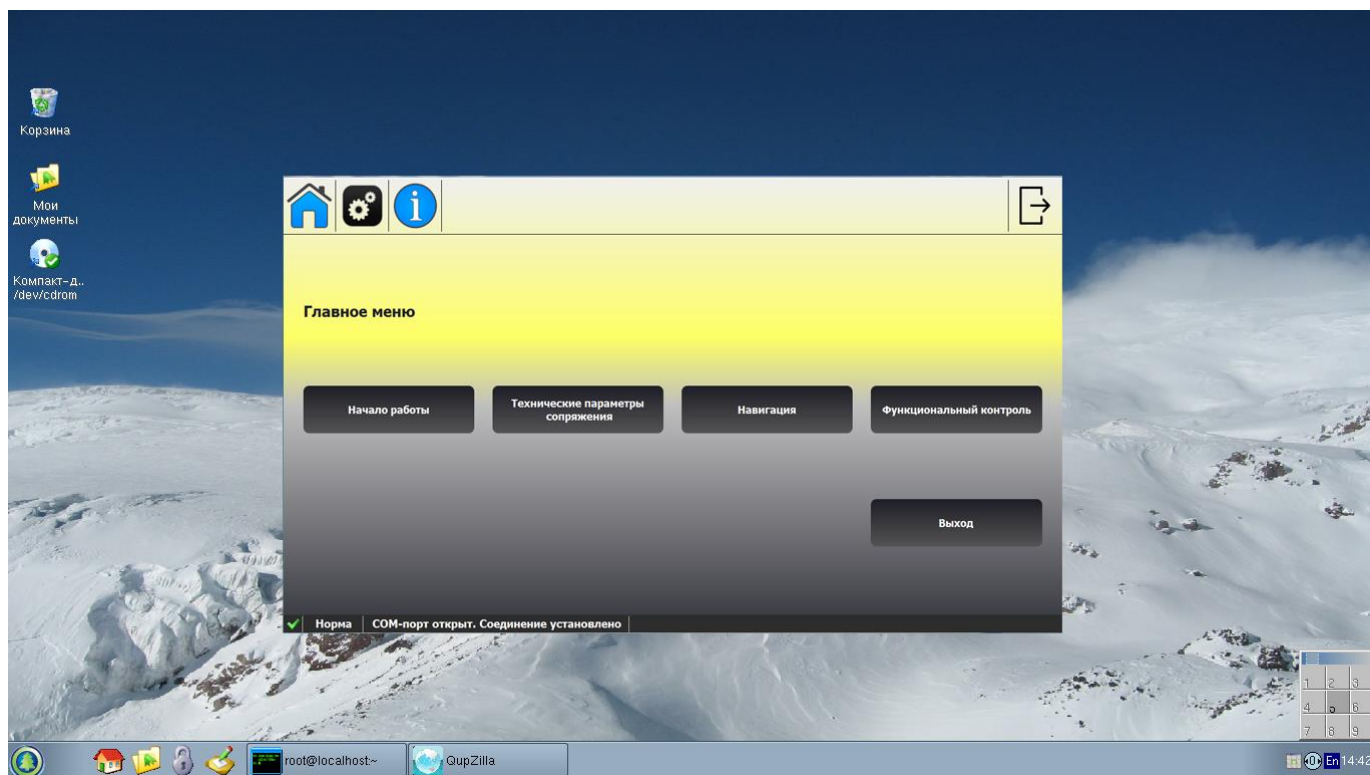


Рисунок 21 - Работоспособность специального программного обеспечения в ОС MCBC 5.0

9.4 Проверка работоспособности приложения на операционной системе Astra Linux

Одним из требований технического задания является возможность работы приложения на компьютерах под управлением Astra Linux. Для осуществления проверки, специальное программное обеспечение было запущено в системе Astra Linux. Результат проверки представлен на рисунке 22.

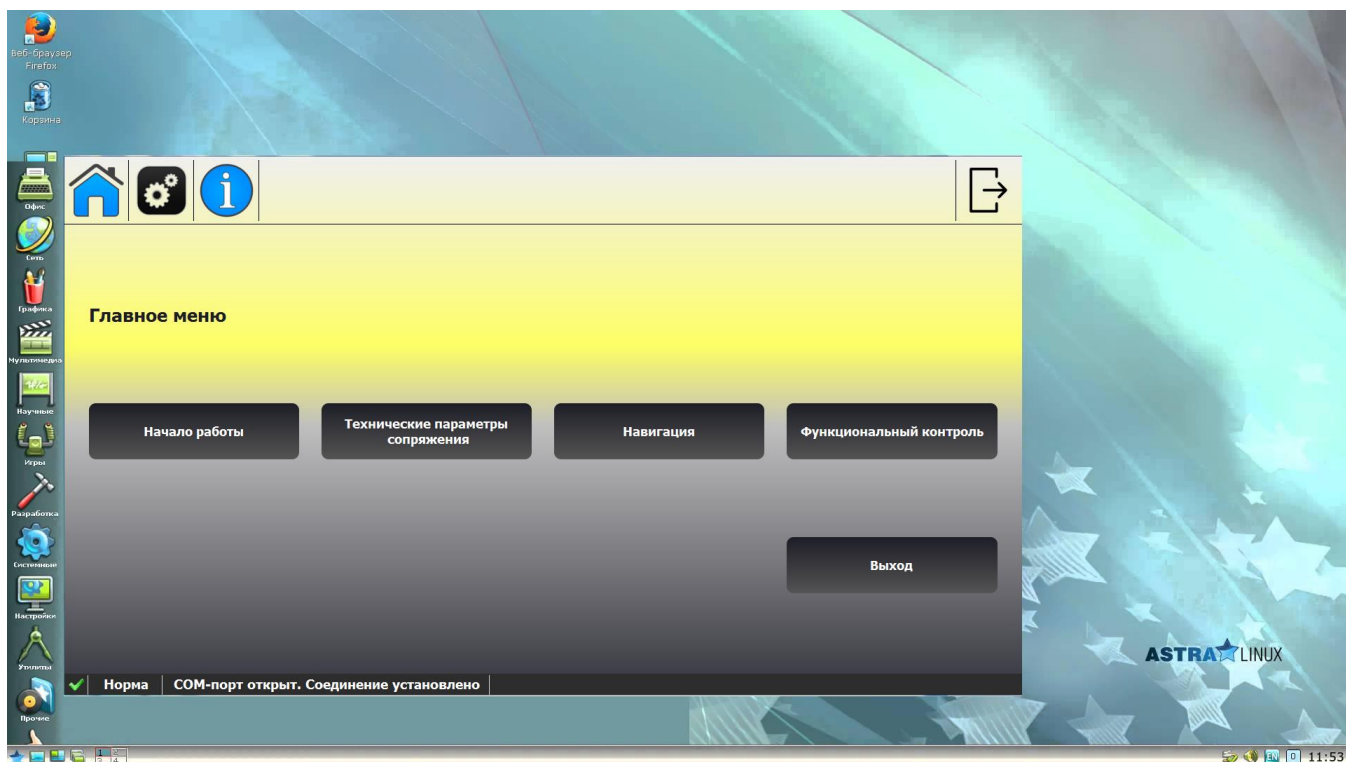


Рисунок 22 - Работоспособность специального программного обеспечения в ОС
Astra Linux

ЗАКЛЮЧЕНИЕ

В результате выполнения ВКР была обоснована необходимость защиты информации, передающейся с использованием специального программного обеспечения в тренажёрных системах и боевых машинах противовоздушной обороны, были произведены идентификация и оценка угроз. идентификация уязвимостей, и на основании этого сделан выбор методов защиты.

Контроль целостности решено осуществлять путём вычисления и проверки имитовставки. Конфиденциальность достигается путём использования алгоритма симметричного криптографического преобразования. Для рассылки ключей используется ключевая система симметричного типа. Выработка имитовставки, алгоритм симметричного криптографического преобразования и ключевая система описаны в действующем стандарте ГОСТ 28147-89.

Таким образом, техническое задание на ВКР выполнено в полном объёме.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 «ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования» – Федеральное агентство по техническому регулированию и метрологии [Электронный ресурс]: Режим доступа: <http://protect.gost.ru/v.aspx?control=7&id=139177>, свободный.

2 «Системы обработки информации. Защита криптографическая. Рекомендации по стандартизации. Задание узлов замены блока подстановки алгоритма шифрования ГОСТ 28147-89» – Технический комитет по стандартизации «Криптографическая защита информации» (ТК 26) [Электронный ресурс]: Режим доступа: <https://tc26.ru/methods/recommendation/%D0%A2%D0%9A26%D0%A3%D0%97.pdf>, свободный.

3 ГОСТ 28147-89 Standard – Википедия [Электронный ресурс]: Режим доступа: https://ru.wikipedia.org/wiki/%D0%93%D0%9ED0%A1%D0%A2_28147-89, свободный.

ПРИЛОЖЕНИЕ А

Исходный код модулей gost.h и gost.cpp

gost.cpp

```
#include "gost.h"
#define SEKRETKKEY "98765432109876543210987654321098"
#define SPACE ' ' //32

GOST::GOST()
{
    //codec
    QTextCodec * codec = QTextCodec::codecForName("CP1251");//KOI8R
    if (codec)
    {
        QTextCodec::setCodecForCStrings(codec);
        QTextCodec::setCodecForLocale(codec);
        QTextCodec::setCodecForTr(codec);
    }
    //*/

    openText = new QLineEdit;
    cryptoText = new QLineEdit;
    debug = new QTextEdit;

    encryptButton = new QPushButton("Encrypt/Decrypt");
    connect(encryptButton, SIGNAL(clicked()), this, SLOT(cryptModel()));
    connect(openText, SIGNAL(returnPressed()), this, SLOT(cryptModel()));
    mode = true;

    //connect

    QVBoxLayout * layout = new QVBoxLayout;
    layout->addWidget(new QLabel("GOST 28147-89"));
    layout->addWidget(openText);
    layout->addWidget(encryptButton);
    layout->addWidget(debug);
    //layout->addWidget(cryptoText);

    setLayout(layout);
}

void GOST::cryptModel()
{
    //--8 S-box-----
    quint32 Sbox[8][16] =
    {
        {4,10,9,2,13,8,0,14,6,11,1,12,7,15,5,3},
        {14,11,4,12,6,13,15,10,2,3,8,1,0,7,5,9},
        {5,8,1,13,10,3,4,2,14,15,12,7,6,0,9,11},
        {7,13,10,1,0,8,9,15,14,4,6,12,11,2,5,3},
        {6,12,7,1,5,15,13,8,4,10,9,14,0,3,11,2},
        {4,11,10,0,7,2,1,13,3,6,8,5,9,12,15,14},
        {13,11,4,1,3,15,5,9,0,10,14,7,6,8,2,12},
        {1,15,13,0,5,7,10,4,9,2,3,14,6,11,8,12}
    };

    //-----
    //char s[33] = SEKRETKKEY;
    quint32 key[8] =
    {
        0x1234,
        0xFF56,
        0x7890,
        0xFF56,
```

```

        0x12FF,
        0xFF56,
        0x1FFF,
        0xFF56,
    };

    /*/
    for(int i = 0; i<8; i++)
    {
        key[i] = (s[4*i]<<24) + (s[1+4*i]<<16) + (s[2+4*i]<<8) + s[3+4*i];

        key[i] = reverse_bit(key[i]);
    }
    /**/

//-----

char *strbuff;
QString instring;
QString resultString;
QByteArray temp;

instring = openText->displayText();
temp = instring.toAscii();

quint32 L = temp.length();

debug->append("instring length "+QString::number(L));

//abcdefgh qwertyyy

quint32 rest;

if (L%8 == 0)
{
    strbuff = (char*)temp.data();
}
else
{
    quint32 k;
    /*/
    if(mode)
    {
        if (L%4 == 0)
        {
            strbuff = (char*)calloc(L+4, 1);
            //filling by data
            for(k = 0; k < L; k++)
                strbuff[k] = temp[k];
            //filling by nulls
            for(k = L; k < L+4; k++)
                strbuff[k] = 0;
        }
        else
        {
            rest = 4 - L%4;
            strbuff = (char*)calloc(L+rest+4, 1);

            //filling by data
            for(k = 0; k < L; k++)
                strbuff[k] = temp[k];

```

```

        //filling by spaces
        for(k = L; k < L+rest; k++)
            strbuff[k] = SPACE;
        //filling by nulls
        for(k = L+rest; k < L+rest+4; k++)
            strbuff[k] = 0;
    }
}
/**/
//
//else
{
    rest = 8 - L%8;
    strbuff = (char*)calloc(L+rest, 1);

    //filling by data
    for(k = 0; k < L; k++)
        strbuff[k] = temp[k];
    //filling by nulls or spaces?
    for(k = L; k < L+rest; k++)
        strbuff[k] = SPACE;
}

/**/
}
/**/

//-----
quint32 offset = 0;

while(offset < L)
{
    quint32 A = 0;
    quint32 B = 0;

    A = *((quint32*)(strbuff + offset));
    offset +=4;
    debug->append(QString::number(A));
    B = *((quint32*)(strbuff + offset));
    offset +=4;
    debug->append(QString::number(B));

    A = reverse_block(A);
    debug->append(QString::number(A));
    B = reverse_block(B);
    debug->append(QString::number(B));

    quint32 T;
    quint64 T64 = 0, A64 = 0, B64 = 0;

    //--32 cycles-----
    for(int i = 0; i < 32; i++)
    {
        T = 0;
        //A64 = A;
        //B64 = B;
        if(mode)
        {
            if(i < 24)
                T = (A+key[i%8]); // % 0x100000000;
            else

```

```

        T = (A+key[7-(i%8)]); // % 0x100000000;
    }
    else
    {
        if(i < 8)
            T = (A+key[i%8]); // % 0x100000000;
        else
            T = (A+key[7-(i%8)]); // % 0x100000000;
    }

quint32 Fragments[8] =
{
    (T & 0xF0000000)>>28,
    (T & 0xF000000)>>24,
    (T & 0xF00000)>>20,
    (T & 0xF0000)>>16,
    (T & 0xF000)>>12,
    (T & 0xF00)>> 8,
    (T & 0xF0)>> 4,
    (T & 0xF)
};

for(int j = 0; j < 8; j++)
{
    Fragments[j] = Sbox[j][Fragments[j]]; //Sbox
}

T = (Fragments[0]<<28) +
    (Fragments[1]<<24) +
    (Fragments[2]<<20) +
    (Fragments[3]<<16) +
    (Fragments[4]<<12) +
    (Fragments[5]<< 8) +
    (Fragments[6]<< 4) +
    Fragments[7];

T = (T<<11) | (T>>21);
T ^= B;

if(i != 31)
{
    B = A;
    A = T;
}
else
{
    B = T;
}
} //-----

debug->append(QString::number(A));
debug->append(QString::number(B));

A = reverse_block(A);
B = reverse_block(B);

debug->append(QString::number(A));
debug->append(QString::number(B));

char tempbuff[9];

```

```

        tempbuff [8] = '\0';
        tempbuff [7] = B >>24;
        tempbuff [6] = (B >>16)&0xFF;
        tempbuff [5] = (B >>8)&0xFF;
        tempbuff [4] = B&0xFF;

        tempbuff [3] = A >>24;
        tempbuff [2] = (A >>16)&0xFF;
        tempbuff [1] = (A >>8)&0xFF;
        tempbuff [0] = A&0xFF;;

        QString outstring(tempbuff);
        debug->append("outstring length "+QString::number(outstring.length()));
        resultString.append(outstring);
    }
    openText->setText(resultString);
    debug->append("resultString length
"+QString::number(resultString.length())+"\n");
    mode = !mode;
}

int GOST::reverse_bit(int X)
{
    return
    X =
        ((X & 0x1)          <<31)+((X & 0x2)          <<29)+((X & 0x4)
<<27)+((X & 0x8)          <<25)+
        ((X & 0x10)         <<23)+((X & 0x20)         <<21)+((X & 0x40)
<<19)+((X & 0x80)         <<17)+
        ((X & 0x100)        <<15)+((X & 0x200)        <<13)+((X & 0x400)
<<11)+((X & 0x800)        << 9)+
        ((X & 0x1000)       << 7)+((X & 0x2000)       << 5)+((X & 0x4000)       <<
3)+((X & 0x8000)       << 1)+
        ((X & 0x10000)      >> 1)+((X & 0x20000)      >> 3)+((X & 0x40000)      >>
5)+((X & 0x80000)      >> 7)+
        ((X & 0x100000)    >> 9)+((X & 0x200000)    >>11)+((X & 0x400000)    >>
>>13)+((X & 0x800000)    >>15)+
        ((X & 0x1000000)   >>17)+((X & 0x2000000)   >>19)+((X & 0x4000000)   >>
>>21)+((X & 0x8000000)   >>23)+
        ((X & 0x10000000)  >>25)+((X & 0x20000000)  >>27)+((X & 0x40000000)  >>
>>29)+((X & 0x80000000) >>31);
}

int GOST::reverse_block(int X)
{
    return
    X =
        ((X & 0x1)          << 7)+((X & 0x2)          << 5)+((X & 0x4)          <<
3)+((X & 0x8)          << 1)+
        ((X & 0x10)         >> 1)+((X & 0x20)         >> 3)+((X & 0x40)         >>
5)+((X & 0x80)         >> 7)+
        ((X & 0x100)        << 7)+((X & 0x200)        << 5)+((X & 0x400)        <<
3)+((X & 0x800)        << 1)+
        ((X & 0x1000)       >> 1)+((X & 0x2000)       >> 3)+((X & 0x4000)       >>
5)+((X & 0x8000)       >> 7)+
        ((X & 0x10000)      << 7)+((X & 0x20000)      << 5)+((X & 0x40000)      <<
3)+((X & 0x80000)      << 1)+
        ((X & 0x100000)    >> 1)+((X & 0x200000)    >> 3)+((X & 0x400000)    >>
5)+((X & 0x800000)    >> 7)+
        ((X & 0x1000000)   << 7)+((X & 0x2000000)   << 5)+((X & 0x4000000)   <<
3)+((X & 0x8000000)   << 1)+

```

```

        ((X & 0x100000000) >> 1)+((X & 0x200000000) >> 3)+((X & 0x400000000) >>
5)+((X & 0x800000000) >> 7);
}

```

gost.h:

```

#ifndef GOST_H
#define GOST_H
#include<QtGui>
#include<QWidget>

class GOST : public QWidget
{
    Q_OBJECT

public:
    GOST();

    QLineEdit * openText, *cryptoText;
    QPushButton * encryptButton;

    QTextEdit * debug;

    QByteArray dataIn, dataOut;

    bool mode;//direction;

    int reverse_bit(int X);
    int reverse_block(int X);

private slots:
    void cryptMode1(/*bool direction*/);
    /*
    void cryptMode2(bool direction);
    void cryptMode3(bool direction);
    void cryptMode4(bool direction);//may be joined with 2 or 3
    */
};

#endif // GOST_H

```

ПРИЛОЖЕНИЕ Б
Листинг программы imitationsoft

FrameWindow.cpp

```
#include "FrameWindow.h"
```

```
FrameWindow::FrameWindow()
```

```
{
    //title
    setWindowTitle("лПННМЕЛН УТЕДУФЧ ЙННФБГЙЙ");
    setFixedSize(MONITOR_WIDTH, MONITOR_HEIGHT);

    //tool buttons
    setIconSize(QSize(TOOLBUTTON_SIZE, TOOLBUTTON_SIZE));

    QToolButton * gotoMainMenu = new QToolButton;
    gotoMainMenu->setFixedSize(TOOLBUTTON_SIZE, TOOLBUTTON_SIZE);
    gotoMainMenu->setToolTip(tr("ЗМЕЧОНЕ НЕОН"));
    gotoMainMenu->setIcon(QIcon(":/new/icons/home"));
    connect(gotoMainMenu, SIGNAL(clicked()), this, SLOT(gotoMainMenu()));

    QToolButton * params = new QToolButton;
    params->setFixedSize(TOOLBUTTON_SIZE, TOOLBUTTON_SIZE);
    params->setToolTip(tr("РБТБНЕФТЦ УПРТЦЕОЙС"));
    params->setIcon(QIcon(":/new/icons/settings"));
    connect(params, SIGNAL(clicked()), this, SLOT(gotoParameters()));

    QToolButton * info = new QToolButton;
    info->setFixedSize(TOOLBUTTON_SIZE, TOOLBUTTON_SIZE);
    info->setToolTip(tr("ЙОЖПНТБГЙС"));
    info->setIcon(QIcon(":/new/icons/info_sign"));
    connect(info, SIGNAL(clicked()), this, SLOT(infoClicked()));

    QToolButton * exit = new QToolButton;
    exit->setFixedSize(TOOLBUTTON_SIZE, TOOLBUTTON_SIZE);
    exit->setToolTip(tr("ЧЩПД"));
    exit->setIcon(QIcon(":/new/icons/exit"));
    connect(exit, SIGNAL(clicked()), this, SLOT(close()));

    //make context
    pContext = new context();

    //toolbar
    QToolBar * toolBar = addToolBar("Controls");
    toolBar->setFixedHeight(TOOLBUTTON_SIZE+10);
    toolBar->setMovable(false);
    //toolBar->addSeparator();
    toolBar->addWidget(gotoMainMenu);
    toolBar->addSeparator();
    toolBar->addWidget(params);
    toolBar->addSeparator();
    toolBar->addWidget(info);
    toolBar->addSeparator();
    toolBar->addWidget(pContext);
    toolBar->addSeparator();
    toolBar->addWidget(exit);

    ///! status bar
    //////////////////////////////////////
    QStatusBar * statusbar = new QStatusBar(this);
    statusbar->setFixedHeight(STATUSBAR_SIZE);
    statusbar->setSizeGripEnabled(false);
}
```

```

    statusBar->setStyleSheet("QStatusBar {background: rgb(45,45,45); color:
white; font: bold 18px;}");// QStatusBar::item {border: 2px solid white;}");

    iconLabel = new QLabel();//icon with green or red circum
    //QPixmap pixmap(SMALL_OK_ICON);
    //pixmap = pixmap.scaled(STATUSBAR_SIZE-5, STATUSBAR_SIZE-5);
    //iconLabel->setPixmap(pixmap);
    iconLabel->setFixedSize(STATUSBAR_SIZE, STATUSBAR_SIZE);
    iconLabel->setStyleSheet(CSS_INFOTEXT_W);
    iconLabel->setAlignment(Qt::AlignHCenter);

    statusLabel = new QLabel(/*"оПТНБ"*/);//Norma/Error (green/red)
    statusLabel->setFixedSize(100, STATUSBAR_SIZE);
    statusLabel->setStyleSheet(CSS_INFOTEXT_W);
    statusLabel->setAlignment(Qt::AlignHCenter);

    messageLabel = new QLabel(/*" оБЮБМШОЩК ЖХОЛГЙПОВМШОЩК ЛПОФТПМШ РТПКДЕО
ХУРЕБЮП  "*/);//detail's message
    messageLabel->setFixedHeight(STATUSBAR_SIZE);
    messageLabel->setStyleSheet(CSS_INFOTEXT_W);
    messageLabel->setAlignment(Qt::AlignHCenter);

    statusBar->addWidget(iconLabel);
    statusBar->addWidget(statusLabel);
    statusBar->addWidget(messageLabel);

    connect(pContext, SIGNAL(statusChangedSignal(bool,QString)), this,
    SLOT(statusChanged(bool,QString)));

    setStatusBar(statusBar);

    //make main menu widget
    pmainMenu = new mainMenu;
    activeWidget = 1;

    //connection signals from main menu
    connection(activeWidget);

    setCentralWidget(pmainMenu);
    setWindowIcon(QIcon(":/new/icons/windowIcon"));

    //open com port, establish connection with apparatus
    pContext->establishConnection();

    //+func control

    //launch read thread
    pContext->transfer->launchReadThread();
}

void FrameWindow::connection(quint8 newWidget)
{
    switch(newWidget)
    {
    case 1:
    {
        connect(pmainMenu->devicesButton, SIGNAL(clicked()), this,
        SLOT(gotoDevices()));
        connect(pmainMenu->paramButton, SIGNAL(clicked()), this,
        SLOT(gotoParameters()));
        connect(pmainMenu->funcControl, SIGNAL(clicked()), this,
        SLOT(gotoFuncControl()));
    }
    }
}

```

```

        connect(pmainMenu->geolocation, SIGNAL(clicked()), this,
SLOT(gotoGeolocation()));
        connect(pmainMenu->exitButton, SIGNAL(clicked()), this, SLOT(close()));
    }
    break;
case 2:
{
    connect(pParameters->DTAbutton, SIGNAL(clicked()), this,
SLOT(gotoParametersDTA()));
    connect(pParameters->RSbutton, SIGNAL(clicked()), this,
SLOT(gotoParametersRS()));
    connect(pParameters->OK, SIGNAL(clicked()), this,
SLOT(gotoMainMenu())); //or to modes?
    connect(pParameters->cancel, SIGNAL(clicked()), this,
SLOT(gotoMainMenu()));
}
    break;
case 21:
{
    connect(pParameters->DTAbutton, SIGNAL(clicked()), this,
SLOT(gotoParametersDTA()));
    connect(pParameters->RSbutton, SIGNAL(clicked()), this,
SLOT(gotoParametersRS()));
    connect(pParameters->OK, SIGNAL(clicked()), this, SLOT(gotoModes()));
    connect(pParameters->cancel, SIGNAL(clicked()), this,
SLOT(gotoDevices()));
}
    break;
case 3:
{
    connect(pParametersDTA->OK, SIGNAL(clicked()), this,
SLOT(gotoParameters()));
    connect(pParametersDTA->cancel, SIGNAL(clicked()), this,
SLOT(gotoParameters()));
}
    break;
case 4:
{
    connect(pParametersRS->OK, SIGNAL(clicked()), this,
SLOT(gotoParameters()));
    connect(pParametersRS->cancel, SIGNAL(clicked()), this,
SLOT(gotoParameters()));
}
    break;
case 5:
{
    connect(pdevices->OK, SIGNAL(clicked()), this, SLOT(gotoParameters()));
    connect(pdevices->cancel, SIGNAL(clicked()), this,
SLOT(gotoMainMenu()));
}
    break;
case 6:
{
    connect(pModes->OK, SIGNAL(clicked()), this, SLOT(gotoResults()));
    connect(pModes->cancel, SIGNAL(clicked()), this,
SLOT(gotoParameters()));
}
    break;
case 7:
{
    connect(pResults->OK, SIGNAL(clicked()), this, SLOT(gotoMainMenu()));
    connect(pResults->cancel, SIGNAL(clicked()), this, SLOT(gotoModes()));
}

```

```

        }
        break;
    case 8:
    {
        connect(pFuncControl->cancel, SIGNAL(clicked()), this,
        SLOT(gotoMainMenu()));
    }
    break;
    case 9:
    {
        connect(pGeolocation->OK, SIGNAL(clicked()), this,
        SLOT(gotoMainMenu()));
        connect(pGeolocation->cancel, SIGNAL(clicked()), this,
        SLOT(gotoMainMenu()));
    }
    break;
}

activeWidget = newWidget;
}

////////// goto slots ////////////////////////////////////////////
void FrameWindow::gotoMainMenu()
{
    if (activeWidget != 1)
    {
        pmainMenu = new mainMenu();
        connection(1);
        setCentralWidget(pmainMenu);
    }
}

void FrameWindow::gotoParameters()
{
    //switch: 2, 21, or nothing
    pParameters = new parameters(pContext);

    if(activeWidget == 1)
        connection(2);
    else//5, 6
        connection(21);

    setCentralWidget(pParameters);
}

void FrameWindow::gotoParametersDTA()
{
    pParametersDTA = new parametersDTA(pContext);
    connection(3);
    setCentralWidget(pParametersDTA);
}

void FrameWindow::gotoParametersRS()
{
    pParametersRS = new parametersRS(pContext);
    connection(4);
    setCentralWidget(pParametersRS);
}

void FrameWindow::gotoDevices()
{
    pdevices = new Devices(pContext);

```

```

        connection(5);
        setCentralWidget(pdevices);
    }

void MainWindow::gotoModes()
{
    pModes = new Modes(pContext);
    connection(6);
    setCentralWidget(pModes);
}

void MainWindow::gotoResults()
{
    pResults = new Results(pContext);
    connection(7);
    setCentralWidget(pResults);
}

void MainWindow::gotoFuncControl()
{
    pFuncControl = new funcControl(pContext);
    connection(8);
    setCentralWidget(pFuncControl);
}

void MainWindow::gotoGeolocation()
{
    pGeolocation = new Geolocation(pContext);
    connection(9);
    setCentralWidget(pGeolocation);
}

void MainWindow::infoClicked()
{
    pContext->hideOrShowInfo();
}

void MainWindow::statusChanged(bool status, QString message)
{
    QString space = "  ";
    if (status)
    {
        QPixmap pixmap(SMALL_OK_ICON);
        pixmap = pixmap.scaled(STATUSBAR_SIZE-5, STATUSBAR_SIZE-5);
        iconLabel->setPixmap(pixmap);

        statusLabel->setText("ОПТНБ");
    }
    else
    {
        QPixmap pixmap(CROSS_ICON);
        pixmap = pixmap.scaled(STATUSBAR_SIZE-5, STATUSBAR_SIZE-5);
        iconLabel->setPixmap(pixmap);

        statusLabel->setText("НПНБНБ");
    }

    messageLabel->setText(space + message + space);
}

MainWindow::~MainWindow()
{

```

```

        pContext->deleteLater();
    }

```

funcControl.cpp

```

#include "funcControl.h"

```

```

funcControl::funcControl(context * pContext)
{
    //
    contextPointer = pContext;

    //form
    form.setupUi(this);
    info = form.info;
    info->setStyleSheet(CSS_TEXTEDIT);
    cancel = form.cancelButton;

    form.titleLabel->setFixedSize(TITLE_WIDTH, TITLE_HEIGHT);
    form.titleLabel->setStyleSheet(CSS_TITLE);
    //form.pictureLabel->setPixmap(QPixmap("icons/funcControlAlpha.png"));

    computerStateLabel = form.computerStateLabel;
    DTASStateLabel = form.DTASStateLabel;
    RSStateLabel = form.RSStateLabel;

    //connect start button
    form.startButton->setStyleSheet(CSS_BUTTON);
    connect(form.startButton, SIGNAL(clicked()), this, SLOT(start()));

    //Cancel button
    cancel->setIcon(QIcon(CANCEL_ICON));
    cancel->setStyleSheet(CSS_CONTROLBUTTON);
    cancel->setFixedSize(CONTROLBUTTON_WIDTH, CONTROLBUTTON_HEIGHT);
    cancel->setIconSize(QSize(CONTROLBUTTON_WIDTH-5, CONTROLBUTTON_HEIGHT-5));
}

void funcControl::start()
{
    info->clear();

    //computer checking
    if(computerCheck())
    {
        info->append("ЪЧН: ЙУРТВЧОБ");
        info->append("урп: ЛПОФТМШОБС УХННБ ЛПТТЕЛФОБ");
        computerStateLabel->setPixmap(QPixmap(SMALL_OK_ICON));
    }
    else
    {
        computerStateLabel->setPixmap(QPixmap(CROSS_ICON));
    }

    //DTA checking
    if(DTACheck())
    {
        info->append("брд: ЙУРТВЧОБ");
        DTASStateLabel->setPixmap(QPixmap(SMALL_OK_ICON));
    }
    else
    {

```

```

        computerStateLabel->setPixmap(QPixmap(CROSS_ICON));
    }

    //RS checking
    if(RScheck())
    {
        info->append("ty: ЙУПТБЧОБ");
        RSStateLabel->setPixmap(QPixmap(SMALL_OK_ICON));
    }
    else
    {
        computerStateLabel->setPixmap(QPixmap(CROSS_ICON));
    }

    contextPointer->changeStatus(false, "OLOLOLO");
}

bool funcControl::computerCheck()
{
    return true;
}

bool funcControl::DTAcheck()
{
    return true;
}

bool funcControl::RScheck()
{
    return true;
}

```

geolocation.cpp

```

#include"geolacation.h"

Geolocation::Geolocation(context *pContext)
{
    //form
    form.setupUi(this);
    OK = form.okButton;
    cancel = form.cancelButton;
    form.groupBox->setStyleSheet("QGroupBox {color: rgb(30,30,39); font: bold 20px;}");
    form.groupBox_2->setStyleSheet("QGroupBox {color: rgb(30,30,39); font: bold 20px;}");

    //context
    contextPointer = pContext;

    //menu title
    form.titleLabel->setFixedSize(TITLE_WIDTH, TITLE_HEIGHT);
    form.titleLabel->setStyleSheet(CSS_TITLE);

    // labels
    //////////////////////////////////////
    //SP imit
    form.labelSP->setStyleSheet(CSS_TEXT);
    //form.labelSP->setFixedSize(TITLE_WIDTH, TITLE_HEIGHT);
    form.labelSP_X->setStyleSheet(CSS_TEXT);
}

```

```

form.labelSP_zone->setStyleSheet(CSS_TEXT);
form.labelSP_Y->setStyleSheet(CSS_TEXT);

//RP imit
form.labelRP->setStyleSheet(CSS_TEXT);
//form.labelRP->setFixedSize(TITLE_WIDTH, TITLE_HEIGHT);
form.labelRP_X->setStyleSheet(CSS_TEXT);
form.labelRP_zone->setStyleSheet(CSS_TEXT);
form.labelRP_Y->setStyleSheet(CSS_TEXT);

//check
form.labelSP_check1->setStyleSheet(CSS_TEXT);
form.labelSP_check2->setStyleSheet(CSS_TEXT);
form.labelSP_check3->setStyleSheet(CSS_TEXT);

form.labelSP_X_check1->setStyleSheet(CSS_TEXT);
form.labelSP_X_check2->setStyleSheet(CSS_TEXT);
form.labelSP_X_check3->setStyleSheet(CSS_TEXT);

form.labelSP_zone_check1->setStyleSheet(CSS_TEXT);
form.labelSP_zone_check2->setStyleSheet(CSS_TEXT);
form.labelSP_zone_check3->setStyleSheet(CSS_TEXT);

form.labelSP_Y_check1->setStyleSheet(CSS_TEXT);
form.labelSP_Y_check2->setStyleSheet(CSS_TEXT);
form.labelSP_Y_check3->setStyleSheet(CSS_TEXT);

form.inputSP_X_check1->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_X_check2->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_X_check3->setStyleSheet(CSS_LINEEDIT_W);

form.inputSP_zone_check1->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_zone_check2->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_zone_check3->setStyleSheet(CSS_LINEEDIT_W);

form.inputSP_Y_check1->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_Y_check2->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_Y_check3->setStyleSheet(CSS_LINEEDIT_W);

//validator
QValidator * zoneValidator = new QIntValidator(0, 120, this);

// line edits
////////////////////////////////////
//
form.inputSP_X->setText(QString::number(contextPointer->SP.X));
form.inputSP_X->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_X->setWhatsThis("Input coordinates");

form.inputSP_Zone->setText(QString::number(contextPointer->SP.zone));
form.inputSP_Zone->setValidator(zoneValidator);
form.inputSP_Zone->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_Zone->setWhatsThis("Input coordinates");

form.inputSP_Y->setText(QString::number(contextPointer->SP.Y));
form.inputSP_Y->setStyleSheet(CSS_LINEEDIT_W);
form.inputSP_Y->setWhatsThis("Input coordinates");

form.inputRP_X->setText(QString::number(contextPointer->RP.X));
form.inputRP_X->setStyleSheet(CSS_LINEEDIT_W);
form.inputRP_X->setWhatsThis("Input coordinates");

```

```

form.inputRP_Zone->setText(QString::number(contextPointer->RP.zone));
form.inputRP_Zone->setValidator(zoneValidator);
form.inputRP_Zone->setStyleSheet(CSS_LINEEDIT_W);
form.inputRP_Zone->setWhatsThis("Input coordinates");

form.inputRP_Y->setText(QString::number(contextPointer->RP.Y));
form.inputRP_Y->setStyleSheet(CSS_LINEEDIT_W);
form.inputRP_Y->setWhatsThis("Input coordinates");

//OK&Cancel buttons
OK->setIcon(QIcon(OK_ICON));
OK->setStyleSheet(CSS_CONTROLBUTTON);
OK->setFixedSize(CONTROLBUTTON_WIDTH, CONTROLBUTTON_HEIGHT);
OK->setIconSize(QSize(CONTROLBUTTON_WIDTH-5, CONTROLBUTTON_HEIGHT-5));
connect(OK, SIGNAL(clicked()), this, SLOT(OKclicked()));
cancel->setIcon(QIcon(CANCEL_ICON));
cancel->setStyleSheet(CSS_CONTROLBUTTON);
cancel->setFixedSize(CONTROLBUTTON_WIDTH, CONTROLBUTTON_HEIGHT);
cancel->setIconSize(QSize(CONTROLBUTTON_WIDTH-5, CONTROLBUTTON_HEIGHT-5));
}

void Geolocation::OKclicked()
{
    bool ok = false;
    quint32 temp;

    //SP
    temp = form.inputSP_X->text().toInt(&ok, 10);
    if (ok)
        contextPointer->SP.X = temp;

    temp = form.inputSP_Zone->text().toInt(&ok, 10);
    if (ok)
        contextPointer->SP.zone = temp;

    temp = form.inputSP_Y->text().toInt(&ok, 10);
    if (ok)
        contextPointer->SP.Y = temp;

    //RP
    temp = form.inputRP_X->text().toInt(&ok, 10);
    if (ok)
        contextPointer->RP.X = temp;

    temp = form.inputRP_Zone->text().toInt(&ok, 10);
    if (ok)
        contextPointer->RP.zone = temp;

    temp = form.inputRP_Y->text().toInt(&ok, 10);
    if (ok)
        contextPointer->RP.Y = temp;
}

```

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

УДК 004.056

Политехнический институт
Факультет приборостроения,
информационных технологий и
электроники

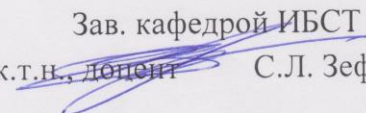
Выпускающая кафедра:

Информационная безопасность систем
и технологий

Учебная группа 11ПИ1

УТВЕРЖДАЮ

Зав. кафедрой ИБСТ

к.т.н., доцент  С.Л. Зефилов

«27» 04 2016 г.

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ
РАБОТУ**

студента группы 11ПИ1 Калмыкова Ивана Олеговича

Специальность 100503 – Информационная безопасность
автоматизированных систем

Тема дипломного проекта: Специальное программное обеспечение
комплекса средств информационно-технического взаимодействия и
имитации сопряжения пунктов управления противовоздушной обороны.

Руководитель ВКР
к.т.н, зав. каф. ИБСТ



Зефилов С.Л.

Консультанты по разделам:

- технологическая часть

начальник отдела «АО «НПП
Рубин»



Савель А.И.

Нормоконтролер

к.т.н., доцент



Иванов А.П.

1 Цели и задачи ВКР

Цели работы:

- разработка специального программного обеспечения комплекса средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны зенитного дивизиона.
- обеспечение защищенного обмена служебной информацией в тренажёрных системах средств управления противовоздушной обороной и боевых машинах противовоздушной обороны зенитного дивизиона.

2 Тактико-технические требования к выполнению ВКР

Требования к программе:

- скорость обмена данными по шлюзу сопряжения алгоритмов обмена данными должна быть 1200, 2400 и 4800 бит/с;
- формат кодограмм должен быть совместим с работой по шлюзу сопряжения алгоритмов обмена данными «АККОРД СС-ПС»;
- специальное программное обеспечение должно быть совместимо с использованием на устройстве с разрешением экрана 1366x768 пикселей;
- должна быть возможность установки нового специального программного обеспечения для имитации работы вновь разрабатываемых пунктов управления.

Требования к пользовательскому интерфейсу:

- специальное программное обеспечение должно отображать информацию о состоянии соединения;
 - специальное программное обеспечение должно отображать статус автоматического и пользовательского функционального контроля;
 - размер шрифта в тексте должен быть не менее 18 пикселей;
- 

- для удобства в использовании необходимо пользоваться фотографическими элементами в интерфейсе;

- задержка перехода между элементами интерфейса и навигационным меню не должна превышать 0.5 секунд;

- специальное программное обеспечение должно обладать унифицированным в соответствии с требованиями заказчика графическим интерфейсом.

Требования к функциональности программы:

- специальное программное обеспечение должно обеспечивать выбор имитационного устройства;

- специальное программное обеспечение должно обеспечивать целостность передаваемых данных и защиту от навязывания ложной информации;

- специальное программное обеспечение должно обеспечивать конфиденциальность передаваемых данных;

- специальное программное обеспечение должно обеспечивать настройку параметров аппаратуры передачи данных.

Требования к средствам разработки:

- разработка приложения должна вестись с использованием кроссплатформенного инструментария разработки на языке программирования C++ (библиотеки Qt или GTK+);

- для работы приложения необходимо следующее программно-аппаратное обеспечение: компьютер с установленной операционной системой Ubuntu 14.04, Windows 7, MCBC 5.0, Astra Linux;

В ВКР должны быть рассмотрены следующие вопросы:

- обоснование необходимости защиты информации, передающейся посредством сетевого приложения;

- анализ угроз информационной безопасности;



- оценка рисков информационной безопасности;
- выбор методов защиты информации;
- реализация выбранных методов защиты информации;
- технология проектирования защищенного программного обеспечения.

3 Требования к разрабатываемой документации

Отчет по ВКР выполняется в соответствии с требованиями ГОСТ 7.32.

Перечень и содержание графической части ВКР:

- структура зенитного дивизиона – 1 л., ф. А1 (плакат);
- схема сети – 1 л., ф. А1 (плакат);
- схема взаимодействия имитационного устройства с аппаратурой передачи данных – 1 л., ф. А1 (плакат);
- структура данных – 1 л., ф. А1 (плакат);
- UML-диаграмма компонентов специального программного обеспечения – 1 л., ф. А1 (плакат);
- алгоритм ключевого обмена – 1 л., ф. А1 (плакат).

Электронная копия отчета и графической части ВКР должна быть представлена на магнитном или оптическом носителе.

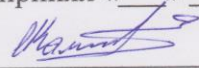
4 Сроки выполнения ВКР

Тема утверждена приказом ректора ПГУ № ____/О от «_» __ 2016 г.

Дата выдачи задания «__» _____ 2016 г.

Время дипломного проектирования с 18.04.2016 г. по 10.07.2016 г.

Задание к исполнению принял «__» _____ 2016 г.

Исполнитель ВКР  Калмыков И.О.



Ректору Пензенского
государственного университета
А. Д. Гулякову
студента 5 курса ФПИТЭ
очной формы обучения
Калмыкова Ивана Олеговича

заявление.

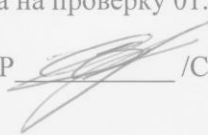
Я, Калмыков Иван Олегович, студент 5 курса очной формы обучения специальности 10.05.03 – «Информационная безопасность автоматизированных систем», прошу провести проверку с использованием системы «Антиплагиат» дипломного проекта на тему «Специальное программное обеспечение комплекса средств информационно-технического взаимодействия и имитации пунктов управления противовоздушной обороны», выполненной мной самостоятельно, на содержание элементов плагиата.

Все прямые заимствования из печатных и электронных источников, а также из защищенных ранее выпускных квалификационных работ, кандидатских и докторских диссертаций имеют соответствующие ссылки.

Я ознакомлен с действующим Стандартом университета «Выпускная квалификационная работа обучающихся по образовательным программам высшего образования – программам бакалавриата, программам специалитета и программам магистратуры», согласно которому обнаружение плагиата является основанием для недопуска ВКР к защите и отчисления из университета.

 /И.О. Калмыков/
01.06.2016 г.

ВКР представлена на проверку 01.06.2016 г.

Руководитель ВКР  /С.Л. Зефилов/



Протокол

проверки на оригинальность в системе «Антиплагиат.ВУЗ» выпускной квалификационной работы на тему «Специальное программное обеспечение комплекса средств информационно технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны» студента 5 курса И.О. Калмыкова специальности 10.05.03 – «Информационная безопасность автоматизированных систем».

Руководитель ВКР к.т.н., доцент, зав. кафедры ИБСТ Зефилов Сергей Львович

Отчет о проверке № 2

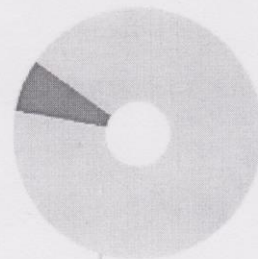
ФИО: Иванов Алексей Петрович
дата выгрузки: 02.06.2016 23:53:06
пользователь: ap_ivanov@mail.ru / ID: 1267386
отчет предоставлен сервисом «Антиплагиат»
на сайте <http://www.antiplagiat.ru>

Информация о документе

№ документа: 34
Имя исходного файла: ВКР_Калмыков_ИО_11ПИ1.pdf
Размер текста: 2171 кБ
Тип документа: Не указано
Символов в тексте: 77739
Слов в тексте: 8589
Число предложений: 233

Информация об отчете

Дата: Отчет от 03.06.2016 07:17:59 - Последний готовый отчет
Комментарии: не указано
Оценка оригинальности: 92,78%
Заимствования: 7,22%
Цитирование: 0%



Оригинальность: 92.78%
Заимствования: 7.22%
Цитирование: 0%

09.06.2016 г.

Секретарь ГЭК

С.Л. Зефилов

А.П.Иванов

ОТЗЫВ

на выпускную квалификационную работу студента

Пензенского государственного университета

Калмыкова Ивана Олеговича

Тема проекта: «Специальное программное обеспечение комплекса средств информационно технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны»

Целью выпускной квалификационной работы Калмыкова Ивана Олеговича является разработка специального программного обеспечения комплекса средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны для защищённого информационного обмена, обеспечивающего поддержку таких свойств информационной безопасности как целостность, доступность и конфиденциальность. Специальное программное обеспечение имитирует полное функциональное взаимодействие разрабатываемого комплекса с учебными и боевыми машинами, которые находятся на вооружении Вооружённых Сил Российской Федерации.

Во время выполнения задания автором был изучен большой объём теоретического и практического материала по различным аспектам проектирования защищённого программного обеспечения. Автором было разработано специальное программное обеспечение для обучения в специальных условиях операторов зенитных ракетных комплексов основным операциям боевой работы при стрельбе по воздушным целям. А также был разработан и реализован программный модуль криптозащиты в соответствии с алгоритмом ГОСТ 28147-89.

Следует отметить, что такая работа требует большой работоспособности, терпения и упорства, которые Калмыков Иван Олегович проявил в полной мере. Ранее дипломник зарекомендовал себя как грамотный программист, теперь же он проявил талант перспективного проектировщика защищённой информационной системы.

Работы по всем пунктам технического задания выполнены полностью. Пояснительная записка и плакаты содержат необходимые сведения о проделанной работе.

Считаю, что дипломный проект выполнен в полном соответствии с заданием и заслуживает оценки «отлично», а дипломник Калмыков Иван Олегович достоин присвоения квалификации специалиста по информационной безопасности по специальности 10.05.03 «Информационная безопасность автоматизированных систем»

Количество баллов: 15

Научный руководитель:

к.т.н., доцент, зав. кафедры ИБСТ



Зефиров Сергей Львович

РЕЦЕНЗИЯ
на выпускную квалификационную работу
студента
Пензенского государственного университета
Калмыкова Ивана Олеговича

Тема проекта: "Специальное программное обеспечение комплекса средств информационно технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны"

Представленная на рецензию ВКР содержит пояснительную записку в объеме 77 страниц и графическую часть на 6 листах. В техническом задании на ВКР требовалось разработать специальное программное обеспечение комплекса средств информационно-технического взаимодействия и имитации сопряжения пунктов управления противовоздушной обороны для защищенного информационного обмена, обеспечивающего поддержку таких свойств информационной безопасности как целостность и конфиденциальность.

Актуальность работы заключается в повседневной необходимости обучения личного состава вооруженных сил Российской Федерации на тренажерных системах, которые имитируют работу современного оборудования и вооружения. Разработанное специальное программное обеспечение позволит не только сократить затраты, но и увеличить скорость и качество обучения личного состава.

В пояснительной записке приведено описание работы специального программного обеспечения, а также описание алгоритма шифрования данных и ключевого обмена.

Требования технического задания выполнены полностью. Работа выполнена на современном теоретическом и техническом уровне, имеет практическую значимость. В пояснительной записке материалы работы изложены полностью.

К достоинствам работы можно отнести детальную проработку функциональных возможностей специального программного обеспечения, а так же высокую производительность в целом. Отдельно стоит отметить, что разработка является частью реального проекта.

В качестве недостатка работы следует отметить, что не рассмотрена модель внутреннего нарушителя.

Выпускная квалификационная работа Калмыкова Ивана Олеговича может быть представлена к защите на заседании государственной аттестационной комиссии и заслуживает оценки "отлично".

Количество баллов: 9

Рецензент:
начальник отдела 5800
АО "НПП "Рубин"

Байкин Александр Анатольевич

