

СОДЕРЖАНИЕ

СПИСОК ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ	4
ВВЕДЕНИЕ	5
1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ. МЕТОДЫ И СРЕДСТВА ПЛАНИРОВАНИЯ КАНАЛЬНЫХ РЕСУРСОВ.....	7
1.1.Вычислительные сети.....	7
1.1.1. Бортвые вычислительные сети	8
1.2.Стандартная модель OSI. Протоколы и спецификации	8
1.3.Сети SpaceWire.....	9
1.4.Методы и средства управления канальными ресурсами	10
1.4.1. Алгоритмы и методы управления канальными ресурсами	10
1.4.1.1.Flow control.....	11
1.4.1.2.GPS.....	11
1.4.1.3.Virtual clock	12
1.4.1.4.Алгоритмы и методы технологии TDMA	12
1.4.1.4.1. Round Robin	12
1.4.1.4.2. Deficit Round Robin	13
1.4.1.4.3. Ликвидное построение расписания.....	13
1.4.1.4.4. Эвристическая раскраска графа	13
1.4.1.4.5. Новый алгоритм планирования	14
1.4.2. Средства управления канальными ресурсами.....	14
1.4.2.1.Планировщик Maui.....	14
1.4.2.2.Вычислительный кластер Condor	16
1.4.2.3.Планировщик Moab.....	16
1.4.2.4.Компонент №3 программного комплекса SANDS.....	17
ВЫВОДЫ ПО РАЗДЕЛУ 1.....	17
2. КАЧЕСТВО СЕРВИСА «ПЛАНИРОВАНИЕ» ПРОТОКОЛА СТП-ИСС-14. ТАБЛИЦЫ РАСПИСАНИЯ.....	19
2.1.Протокол СТП-ИСС-14. Типы качества сервиса протокола СТП-ИСС-14	19
2.2.Качество сервиса «Планирование»	20
2.2.1. Основные определения. Эпоха и тайм-слоты	20
2.2.2. Прием меток времени	20
2.2.3. Передача данных в соответствии с расписанием.....	22
2.3.Новый метод планирования канальных ресурсов	23
2.3.1. Шаг 1. Загрузка и проверка корректности входных данных.	24
2.3.2. Шаг 2. Проверка возможности передачи пакетов данных с максимально допустимыми задержками.....	24

2.3.3. Шаг 3. Определение длительности эпохи и временных интервалов	24
2.3.4. Шаг 4. Проверка пропускной способности сети	25
2.3.5. Шаг 5. Проверка необходимости планирования трафиков	25
2.3.6. Шаг 6. Планирование	25
2.3.6.1. Шаг 6.1. Планирование неконфликтных трафиков	26
2.3.6.2. Шаг 6.2. Планирование конфликтных трафиков	26
2.3.7. Шаг 7. Вывод результатов работы	27
2.4. Система автоматизированного проектирования бортовых космических сетей	27
2.4.1. Работа с Компонентом 3. Генерация таблиц расписания	29
ВЫВОДЫ ПО РАЗДЕЛУ 2	31
3. ПРИЛОЖЕНИЕ ДЛЯ ПРОВЕРКИ КОРРЕКТНОСТИ ТАБЛИЦЫ РАСПИСАНИЯ	33
3.1. Назначение приложения для проверки корректности таблицы расписания	33
3.2. Входные и выходные данные	33
3.2.1. Входные данные	33
3.2.1.1. Топология сети	33
3.2.1.1.1. Структура Xml-файла	34
3.2.1.2. Параметры таблицы расписания	35
3.2.1.3. Таблица расписания	35
3.2.1.4. Задержка перед отправкой	36
3.2.2. Выходные данные	36
3.2.2.1. Сообщения пользователю	36
3.2.2.2. Списки трафиков, узлов и коммутаторов для корректировки	36
3.2.2.3. Рекомендуемая длина эпохи	36
3.3. Интерфейс приложения для проверки корректности таблицы расписания	36
3.3.1. Окно «Проверка корректности таблицы расписания СТП-ИСС-14»	36
3.3.2. Окно «Результаты проверки корректности таблицы расписания СТП-ИСС-14»	38
3.4. Проверки таблиц расписания	39
3.4.1. Проверки корректности входных данных	39
3.4.2. Проверки корректности сети	40
3.4.3. Проверки возможности построения расписания для заданной сети	41
3.4.4. Проверки корректности таблицы расписания	41
3.4.4.1. Проверка возможности передачи первых пачек трафиков	42
3.4.4.2. Проверка возможности построения расписания для бесконфликтных трафиков в выделенных тайм-слотах	43
3.4.4.3. Проверка возможности построения расписания для конфликтных трафиков в выделенных тайм-слотах	43

3.5.Всплывающие окна приложения для проверки корректности таблицы расписания..	45
3.6.Результаты проверок	48
ВЫВОДЫ ПО РАЗДЕЛУ 3.....	50
4. ПРОВЕРКА КОРРЕКТНОСТИ ТАБЛИЦ РАСПИСАНИЯ	51
4.1.Проверка таблиц расписания для сети с конфликтными трафиками	51
4.1.1. Входная топология сети с конфликтными трафиками	51
4.1.2. Сгенерированное расписание.....	53
4.1.3. Примеры проверки таблиц расписания для сети с конфликтными трафиками..	53
4.1.3.1.Пример проверки таблицы расписания, сгенерированной Компонентом 3 для сети с конфликтными трафиками	53
4.1.3.2.Пример проверки таблицы расписания с неиспользуемым разрешенным тайм-слотом.....	55
4.1.3.3.Пример проверки таблицы расписания с запретом одно тайм-слота, используемого узлом.....	56
4.1.3.4.Пример проверки таблицы расписания со сдвигом влево расписания для одного узла	58
4.1.3.5.Пример проверки таблицы расписания со всеми разрешенными тайм- слотами	60
4.1.3.6.Пример проверки таблицы расписания с увеличенным числом тайм-слотов	61
4.2.Проверка таблиц расписания для сети с неконфликтными трафиками	62
4.2.1. Входная топология сети с неконфликтными трафиками	62
4.2.2. Сгенерированное расписание.....	63
4.2.3. Примеры проверки таблиц расписания для сети с неконфликтными трафиками	64
4.2.3.1.Пример проверки таблицы расписания, сгенерированной Компонентом 3 для сети с неконфликтными трафиками	64
4.2.3.2.Пример проверки таблицы расписания со сдвигом вправо расписания для одного узла	65
ВЫВОДЫ ПО РАЗДЕЛУ 4.....	66
ЗАКЛЮЧЕНИЕ.....	68
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	69
ПРИЛОЖЕНИЕ А	71
ПРИЛОЖЕНИЕ Б	72
ПРИЛОЖЕНИЕ В.....	74

СПИСОК ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ

OSI – Open System Interconnection;

ВОС – Взаимодействие Открытых Систем;

TCP – Transmission Control Protocol;

RR – Round Robin;

DRR – Deficit Round Robin;

GPS – Generalized Processor Sharing;

HOL – Head-Of-Line;

TDMA – Time Division Multiple Access;

САПР – Система автоматизированного проектирования;

БВС – Бортовая вычислительная сеть;

КА – Космический аппарат;

СТП-ИСС – Сетевой транспортный протокол для КА ОАО «ИСС»;

СПО – Системы пакетной обработки;

SANDS – SpaceWire Automated Network Design and Simulation;

ВКиСТ – Высокопроизводительные Компьютерные и Сетевые Технологии;

QoS – Quality of Service, качество сервиса.

ВВЕДЕНИЕ

В настоящее время существует большое количество сетей и сетевых транспортных протоколов. Существуют сети с различной архитектурой, сети разного назначения, для которых необходимо обеспечивать корректную передачу данных. Одной из важнейших задач транспортных протоколов является управление канальными ресурсами. С этой задачей транспортные протоколы справляются посредством различных качеств сервиса (QoS). Одним из таких качеств сервиса является «Планирование». Данное качество сервиса реализуется в различных протоколах. В частности, сетевая технология SpaceWire, предназначенная для работы в бортовых сетях аэрокосмического назначения, поддерживает данное качество сервиса. «Планирование» реализовано в таких протоколах как SpaceWire-D и СТП-ИСС-14.

Ключевым параметром качества сервиса «Планирование» является таблица расписания. С помощью данной таблицы в сети осуществляется распределение канальных ресурсов во времени.

Целью данной выпускной квалификационной работы является разработка приложения, реализующего графический интерфейс для проверки корректности таблицы расписания. В качестве протокола, для которого будет осуществляться проверка корректности таблицы расписания, был выбран протокол СТП-ИСС-14, работающий поверх стека протоколов SpaceWire. Для достижения поставленной цели необходимо решить следующие задачи:

1. изучение методов и средств управления канальными ресурсами, в том числе методов планирования;
2. выбор средства, которое будет дополнено графическим интерфейсом проверки корректности таблицы расписания;
3. изучение выбранного средства проектирования сетей;
4. разработка метода проверки корректности таблицы расписания;
5. реализация приложения, обеспечивающего графический интерфейс проверки корректности таблицы расписания;
6. оптимизация приложения для работы в выбранном средстве проектирования сетей;
7. проверка работоспособности реализованного интерфейса;
8. проверка различных таблиц расписания для спроектированных сетей.

Выпускная квалификационная работа состоит из введения, четырех разделов, заключения и трех приложений.

В разделе 1 содержится обзор различных методов и средств управления канальными ресурсами, а также выбор средства проектирования сетей, для которого будет разработано приложение проверки корректности таблиц расписания.

Раздел 2 посвящен изучению выбранного средства проектирования сетей, в частности компонента данного средства, реализующего построение таблицы расписания.

Раздел 3 содержит описание разработанного приложения для проверки корректности таблицы расписания. А также описание метода проверки корректности таблицы расписания.

В разделе 4 проведено тестирование с целью выявления ошибок в работоспособности приложения, а также ошибок в функционировании алгоритма проверки корректности таблицы расписания.

В заключении сформулированы основные выводы.

В приложениях приведена публикация по теме выпускной квалификационной работы, таблицы с описанием текстовых полей и кнопок приложения, а также листинг разработанного приложения.

1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ. МЕТОДЫ И СРЕДСТВА ПЛАНИРОВАНИЯ КАНАЛЬНЫХ РЕСУРСОВ

1.1. Вычислительные сети

Вычислительная сеть — это система, обеспечивающая обмен данными между вычислительными устройствами, такими как: коммутаторы, маршрутизаторы, компьютеры, серверы и другое оборудование, находящееся в данной сети. В настоящее время вычислительные сети распространены во многие сферы деятельности человека. Единой общепринятой системы, которой удовлетворяют все сети не существует, но есть два важнейших параметра: технология передачи и размеры.

По типу технологии передачи вычислительные сети делятся на:

- *Сети с передачей от узла к узлу.* Сети с передачей от узла к узлу состоят из соединенных пар машин. Чтобы дойти от источника до места назначения в сети, коротким сообщением, называемым пакетами, нужно пройти, в большинстве случаев, один или несколько промежуточных узлов. Часто возможны несколько маршрутов различных длин, поэтому в данных сетях можно найти лучший из них.
- *Широковещательные сети.* Широковещательные сети отличаются тем, что обладают единым каналом связи, совместно используемом всеми узлами сети. Пакеты посылаются одним узлом, а получаются всеми остальными узлами сети. При получении пакета узел проверяет адресное поле и, в случае обнаружения своего адреса, узел обрабатывает пакет.

Протяженность	Расположение процессоров	Пример
1 м	На одном квадратном метре	Персональная сеть
10 м	комната	Локальная сеть
100 м	здание	
1 км	кампус	
10 км	город	Муниципальная сеть
100 км	страна	Глобальная сеть
1000 км	континент	
10000 км	планета	Интернет

Рисунок 1 – Классификация сетей по протяженности

Другим признаком классификации вычислительных сетей является их протяженность. Протяженность сетей является важным классификационным фактором, потому что в сетях различного размера применяются различные сетевые технологии.

На Рисунок 1 приведена классификация сетей в зависимости от их протяженности. В верхней части таблицы помещаются персональные сети, предназначенные для одного человека. Далее в таблице следуют более протяженные сети, и замыкают таблицу объединения двух и более сетей [1].

1.1.1. Бортовые вычислительные сети

Бортовые сети – это локальные сети обеспечения взаимодействия элементов системы жизнедеятельности аппаратов. Такими аппаратами являются средства наземного передвижения, такие как автомобили, средства передвижения по воздуху, например, самолеты, космические аппараты (КА), спутники, ракеты и другое.

Для обеспечения взаимодействия в бортовых сетях применяется согласованный набор программных и аппаратных средств (кабели, драйверы, сетевые адаптеры, разъёмы), а также механизмы передачи данных по линиям связи, достаточный для построения вычислительной сети. Такой согласованный набор называется сетевой технологией [2].

1.2. Стандартная модель OSI. Протоколы и спецификации

Разработчики сетевого оборудования сталкиваются с вопросом об организации сетевого взаимодействия. Чаще всего для этого используется многоуровневый подход. При таком подходе сложная задача взаимодействия разбивается на более простые – модули. Такие модули упорядочиваются и группируются по уровням, образуя иерархию. Каждый модуль может взаимодействовать с примыкающими к нему нижележащим и вышележащим уровнем иерархии, в которой он находится, а также с модулями другого узла, расположенном на том же уровне иерархии. Каждый уровень поддерживает интерфейсы двух типов: интерфейс – формализованное описание правил взаимодействия модулей соседних уровней на одном узле; протокол – формализованное описание правил взаимодействия модулей одного уровня в разных узлах. Программный модуль, реализующий работу протокола, также называется протоколом.

Прикладной уровень
Уровень представления
Сеансовый уровень
Транспортный уровень
Сетевой уровень
Канальный уровень
Физический уровень

Рисунок 2 – Иерархия стандартной модели OSI

Набор протоколов, образующий иерархию, способный реализовать взаимодействие узлов в сети, образует стек протоколов. Существует ряд моделей взаимодействия систем, отличающихся друг от друга названием, функцией уровней, самой распространенной такой моделью является стандартная модель взаимодействия открытых систем – модель OSI (Open System Interconnection). Модель OSI определяет 7 уровней взаимодействия систем, иерархия уровней представлена на Рисунок 2 [3].

Протоколы любого уровня должны быть формализованы, то есть для каждого протокола должна существовать спецификация. Спецификация – это формализованное описание аппаратных или программных компонентов, способов их функционирования, взаимодействия с другими компонентами, условий эксплуатации, особых характеристик. Спецификации делятся по типу доступа на открытые и закрытые. К открытым спецификациям есть пользовательский доступ, так как данные спецификации публикуются, являются общедоступными и соответствующими стандартам закрытые наоборот не являются общедоступными.

Любой программный или аппаратный продукт, который построен в соответствии с открытыми спецификациями, называется открытой системой, а протокол – протоколом взаимодействия открытых систем (протокол ВОС) [2]. Системы, представляющие собой сетевое оборудование, не всегда используют протоколы ВОС с открытыми спецификациями на всех уровнях. Они могут быть представлены в различных конфигурациях, таких как:

- семиуровневые открытые системы, используют ВОС на всех уровнях;
- частично открытые системы, используют протоколы ВОС только на открытых уровнях;
- открытые ретрансляционные системы, используют ВОС на уровнях 1 – 3 (ретрансляционные системы на сетевом уровне) или 1 – 7 (ретрансляционные системы на прикладном уровне).

1.3. Сети SpaceWire

SpaceWire – это наиболее передовая и активно развиваемая сетевая технология, применяемая при реализации бортовых вычислительных сетей, преимущественно аэрокосмического назначения. SpaceWire разрабатывалась в соответствии с такими требованиями аэрокосмических применений, как высокие скорости передачи информации, устойчивость к отказам и сбоям, низкое энергопотребление, малые задержки доставки сообщений, поддержка систем реального времени и системных функций бортовых комплексов. Стандарт SpaceWire регламентирует логические протоколы, физические разъёмы и кабели, электрические свойства соединений, которые определяют канал связи,

архитектуру коммуникационной сети и обеспечивают средства передачи пакетов информации от исходного узла до требуемого узла назначения через масштабируемую коммуникационную сеть [4].

Стек протоколов SpaceWire реализован в виде шестиуровневой иерархии. На Рисунок 3 представлено сравнение иерархии протоколов SpaceWire с семиуровневой моделью OSI. Стек протоколов SpaceWire функционирует на 4 уровнях модели OSI: транспортном, сетевом, соединения, физическом. Для функционирования транспортного уровня SpaceWire существуют различные протоколы, такие как RMAP, PTP, SpW-D, STUP, JRDDP, STP, СТП-ИСС и другие.

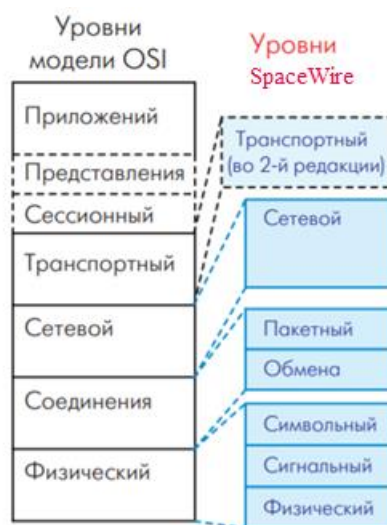


Рисунок 3 – Сопоставление иерархии протоколов SpaceWire с моделью OSI

1.4. Методы и средства управления канальными ресурсами

Бортовые сети КА, в том числе и сети SpaceWire, имеют возможность управления, распределения и планирования канальных ресурсов. Канальные ресурсы являются передаточными возможностями линий связи сети, иными словами с помощью технологии SpaceWire можно определять какие трафики данных будут занимать канал в данный момент и, при необходимости, планировать очередность трафиков. Планирование канальных ресурсов необходимо для достижения корректной передачи данных по линиям связи сети.

Существуют различные методы планирования канальных ресурсов, а также средства, реализующие данные методы.

1.4.1. Алгоритмы и методы управления канальными ресурсами

Для описания алгоритмов и методов планирования канальных ресурсов необходимо описать некоторые ключевые понятия планирования, управления и распределения ресурсов. Необходимыми понятиями являются:

- *Принцип справедливого распределения ресурсов* – это один из основных принципов функционирования сетей. Основная идея данного принципа заключается в равнозначном распределении полосы пропускания между конкурирующими пакетами или потоками вне зависимости от нагрузки.
- *Критерий max-min* – это базовый критерий обеспечения принципа справедливого распределения ресурсов. Данный критерий означает, что пропускная способность для каждого потока должна быть как минимум равна пропускной способности каждого другого потока, проходящего через тот же критический ресурс. Для каждого потока ресурсы могут быть распределены справедливо как на всем пути, так и локально в рамках одного сетевого узла. Для справедливого распределения ресурсов необходимо и достаточно, чтобы этот принцип был реализован в каждом сетевом узле маршрута.
- *Справедливая буферизация на уровне пакетов* – это трансформация принципа справедливого распределения ресурсов. В алгоритмах обслуживания очередей с приоритетами при организации передачи пакетов, различающихся длиной и приоритетом в обслуживании, реализация принципа справедливого распределения ресурсов существенно осложняется. В связи с этим при обслуживании пакетов различной длины, принадлежащих различным классам качества обслуживания, формируется принцип справедливой буферизации на уровне пакетов. Данный принцип заключается в том, что каждый пакет, независимо от длины, должен быть передан на обслуживание в соответствии с требованиями по качеству обслуживания того класса, к которому данный пакет принадлежит [5].

1.4.1.1. Flow control

Одним из наиболее распространённых методов планирования канальных ресурсов является flow control (управление потоком). Данный метод позволяет настраивать скорость поступления данных в буфер приемника в зависимости от скорости передачи данных прикладному уровню. Управление скоростью, на которой данные поступают в буфер приемника осуществляется путем уменьшения скорости передачи сегментов передатчиком. Наиболее известный пример применения flow control – контроль потока в протоколе TCP.

1.4.1.2. GPS

GPS (Generalized Processor Sharing) – это непрерывно работающий алгоритм, разделяющий пропускную способность между потоками. GPS обеспечивает в непрерывном времени принцип справедливого распределения ресурсов в соответствии с критерием max-min для классов трафика с различными требованиями по качеству обслуживания. Алгоритм обеспечивает обработку всех потоков независимо друг от друга.

Для реализации принципа справедливой буферизации на уровне пакетов» алгоритм GPS для каждого поступающего в маршрутизатор пакета предполагает вычисление значения параметра «время окончания обслуживания» (finished tag). Каждый раз, когда заканчивается обслуживание некоторого пакета, планировщик передает на обслуживание следующий пакет с наименьшим значением параметра «время окончания обслуживания» из некоторой очереди. В данном алгоритме анализируются только пакеты, стоящие первыми на обслуживание в каждой из очередей, называемые HOL (Head-Of-Line) пакетами.

1.4.1.3. Virtual clock

Virtual clock (виртуальные часы) используют концепцию виртуального времени. С помощью реального времени каждому пакету, каждого потока, назначается метка времени «виртуальное время окончания обслуживания». Виртуальные часы существуют для каждого потока независимо. Если поток отправляет пакеты в соответствии со своей средней скоростью – он помечается штампом времени и помещается в очередь обслуживания, общую для всех потоков. Принцип обработки пакетов из каждого потока состоит в том, что временная метка каждого пакета всегда меньше или равна реальному времени. В случае, когда поток отправляет больше пакетов, чем может обработать система, временные метки становятся большими по значению, чем реальное время, и планировщик, путем снижения количества отводимого потоку процессорного времени, уменьшает пропускную способность для данного потока [5].

1.4.1.4. Алгоритмы и методы технологии TDMA

Технология TDMA – один из методов планирования канальных ресурсов, на основе которого в последствии были спроектированы другие методы, скомпенсированные для определенных сетей. Принцип действия данной технологии заключается в разделении канала на слоты, ограниченные по времени, называемые тайм-слотами. Количество и длительность тайм-слотов определяется количеством подключенных абонентов. Подобное соединение характеризуется надежностью и высокой стабильностью, поскольку каждый абонент может отправлять данные только в рамках отведенного ему времени. В момент использования канала абонентом в свой разрешенный тайм-слот, узел имеет доступ к всей полосе пропускания. Перед отправкой пакет данных в промежуточном буфере разделяется на блоки, заполняя временный слот в очередном порядке. Как только время отправки данных для одного абонента заканчивается, прекращается передачи для данного пользователя и осуществляется переход к следующему клиенту.

1.4.1.4.1. Round Robin

Round Robin (RR) – это алгоритм циклического планирования. В ходе выполнения данного алгоритма для каждого потока из множества потоков формируется отдельный временной интервал (тайм-слот). При этом, потоки в тайм-слотах не должны повторяться и использовать общие каналы. Основополагающей идеей этого алгоритма является обеспечение одинакового доступа каждой из очередей к ресурсам системы, то есть, когда система освобождается, планировщик циклически выбирает очередь, из которой принимается пакет на обслуживание. Планирование происходит до тех пор, пока все потоки трафика не будут использованы. Этот алгоритм обладает существенным недостатком, делающим его непригодным для реализации в реальном оборудовании – RR не обеспечивает принцип «справедливого распределения ресурсов» для случаев, когда пакеты имеют переменную длину [6].

1.4.1.4.2. Deficit Round Robin

Данный алгоритм является модификацией алгоритма RR для пакетов переменной длины. В DRR добавлена функция накопления квантов выделяемого процессорного времени. Обслуживание «слишком длинных» пакетов происходит следующим образом: когда некоторой очереди выделено определенное количество процессорного времени для обработки находящегося в ней пакета, но пакет, из-за своего размера требует большего кванта процессорного времени на обработку, данный пакет не передается на обработку; При обращении планировщика к данной очереди в следующий раз квант выделяемого времени суммируется с неиспользованным временем, необработанный пакет большого размера поступает на обработку. Такое решение позволяет обслуживать пакеты переменной длины, обеспечивая исполнение принципа «справедливого распределения ресурсов» [7].

1.4.1.4.3. Ликвидное построение расписания

Ликвидное построение расписания использует понятие «загруженность канала». Загруженность канала – это ситуация, в которой группа неконфликтующих трафиков использует максимальную пропускную способность канала. Ликвидность определяется отношением количества потоков в трафике к числу потоков, проходящих через канал с максимальной загруженностью, умноженное на пропускную способность канала. Для того чтобы составить ликвидное расписание, необходимо разбиение трафика на такие группы, чтобы максимальная загруженность каналов была постоянной на протяжении всего цикла отправки данных [8].

1.4.1.4.4. Эвристическая раскраска графа

Данный алгоритм формирования таблицы расписания учитывает параметры сети и требование виртуальных каналов, в отличие от RR и Ликвидного построения расписания. Для построения таблицы расписания исследуемая сеть представляется в виде матрицы

соединений и набора физических и виртуальных каналов. Формирование таблицы расписания осуществляется в несколько этапов:

- Проведение предварительных расчетов дополнительных параметров для упрощения дальнейших вычислений;
- Формирование групп из множества маршрутов;
- Расчет времени, необходимого для передачи данных по маршрутам виртуальных каналов, участвующих в группе;
- Удаление групп, являющихся подмножеством других групп из множества, учитывая время передачи данных удаляемой группы;
- Формирование таблицы расписаний из множества групп [9].

1.4.1.4.5. Новый алгоритм планирования

Новый алгоритм планирования строит таблицу расписания, в которой время доставки всех пакетов всех трафиков не превышает максимально допустимую задержку, заданную пользователем для каждого трафика.

Данный алгоритм учитывает множество параметров, таких как параметры трафиков, максимальное количество тайм-слотов, максимальная длительность тайм-слота шаг времени между тайм-слотами, максимальная загрузка каналов, запас размера пакета, и другие. В основе нового алгоритма планирования лежит концепция генетического алгоритма и поиска с возвратом. Результатом работы данного метода являются таблицы расписания, рекомендации, длительности рассчитанной эпохи и другое [10]. Данный алгоритм подробно рассмотрен в разделе 2.3.

1.4.2. Средства управления канальными ресурсами

Существуют различные программные средства, реализующие методы и алгоритмы управления канальными ресурсами.

1.4.2.1. Планировщик Maui

Maui реализует возможность практического применения резервирования ресурсов для выполнения задач. Данная возможность позволяет корректно осуществлять управление трафиком, исключая ситуации соревнования задач за одни и те же ресурсы. Данный планировщик является одним из компонентов системы пакетной обработки заданий (СПО).

Maui реализует схожие с другими планировщиками функции, например при заданном множестве готовых к исполнению задач из очереди и для текущего состояния ресурсов определяет очередность запуска заданий на счет и поиск подходящих ресурсов для тех из них, которые могут быть запущены в данный момент. В этих рамках Maui реализует новые механизмы, основанные на «предвидении» и направленные на

оптимизацию управления. Помимо этого, в Maui есть ряд индивидуальных параметров, отличающих его от других планировщиков. К таким параметрам относятся:

- Резервирование ресурсов.
- Алгоритмы обратного распределения и справедливого распределения ресурсов.
- Система автоматического определения приоритетов заданий.

Аппарат резервирования реализуется следующим образом: физически резервирование представляет собой запись в базе данных планировщика. С другой стороны, интерпретация резервирований предполагает рассмотрение состояний ресурсов на моменты начала резервирований. Изображается это в виде временной развертки состояния ресурсов, представленной на Рисунк 4. Для каждого ресурса на оси времени ставятся метки, определяющие, какому заданию в этот момент могут быть доступны. Совокупность некоторого количества меток образуют одно резервирование.



Рисунок 4 – Временная развертка резервирования каналных ресурсов

Алгоритм планирования Maui работает итерационно, как и другие планировщики. Каждый цикл начинается при осуществлении одного из следующих событий:

- Меняется состояние задания или ресурса.
- Достигнута граница резервирования.
- Получена внешняя команда.
- Максимальная длительность цикла достигнута.

В процессе планирования применяется резервирование ресурсов для доступности под выполняющиеся задания, а также для предупреждения простоя высокоприоритетных заданий. После выполнения резервирования начинает работу обратного распределения. После опроса узлов алгоритм получает информацию о том, какие узлы свободны начиная с текущего времени и объединяет их в окна. Затем среди всех окон выбирается самое «широкое», среди всех оставшихся заданий выбирается наиболее точно удовлетворяющее этому «окну» задание и запускается. Если есть более одного задания, наиболее

удовлетворяющее окну, они все запускаются. При этом, учитывая резервирование ресурсов, запущенные задания не мешают высокоприоритетным заданиям [11].

1.4.2.2. Вычислительный кластер Condor

Condor – это система планирования загрузки вычислительного кластера. Была создана группой разработчиков университета города Висконсин. В настоящее время система находится в свободном доступе и функционирует практически на всех платформах.

Одним из компонентов данного средства является condor_schedd, который строит расписание для узлов, включенных в сеть, поверх которой работает Condor. Расписание строится неявно для пользователя, оно остается на служебном уровне программы. С помощью него программа выполняет поставленные пользователем задачи [12]. На Рисунок 5 представлена архитектура Condor. Компонент планирования расположен в выполняющей машине.



Рисунок 5 – Архитектура Condor

1.4.2.3. Планировщик Moab

Moab – это система выполнения задач, работающая в кластерах. Существует так же архитектура Moab Grid, работающая поверх нескольких кластеров, в которых работает Moab.

Планировщик строит таблицу расписания выполнения задач в зависимости от их приоритета. Приоритетность определяется несколькими пунктами:

- Владелец задачи;
- Размер задачи;
- Простой задачи в очереди.

Приоритезация может меняться в зависимости от независимых целей, например, максимизация загрузки ресурсов, особый приоритет пользователей в отдельных проектах.

Определенное качество сервиса, например “run now”, может быть добавлено определенному заданию, что вызывает его немедленное исполнение, тогда во время

составления расписания: планировщик принимает решение на основе информации о ресурсах и настроенных политик; влияющая на решения планировщика информация может быть изменена для того, чтобы быть рассмотренной на следующем цикле планирования; параметры политик могут быть изменены на лету, например подстройка заполнения таблицы; инфраструктура встраиваемых модулей планировщика, которая делает возможным перехват и взаимодействие с конкретным циклом планирования, может быть доступна пользователю. Во время исполнения задания могут: быть уничтоженными; обновлять время исполнения; вытесняться и повторно ставиться в очередь [13]. На Рисунок 6 представлен пример таблицы расписания, построенной планировщиком Moab.

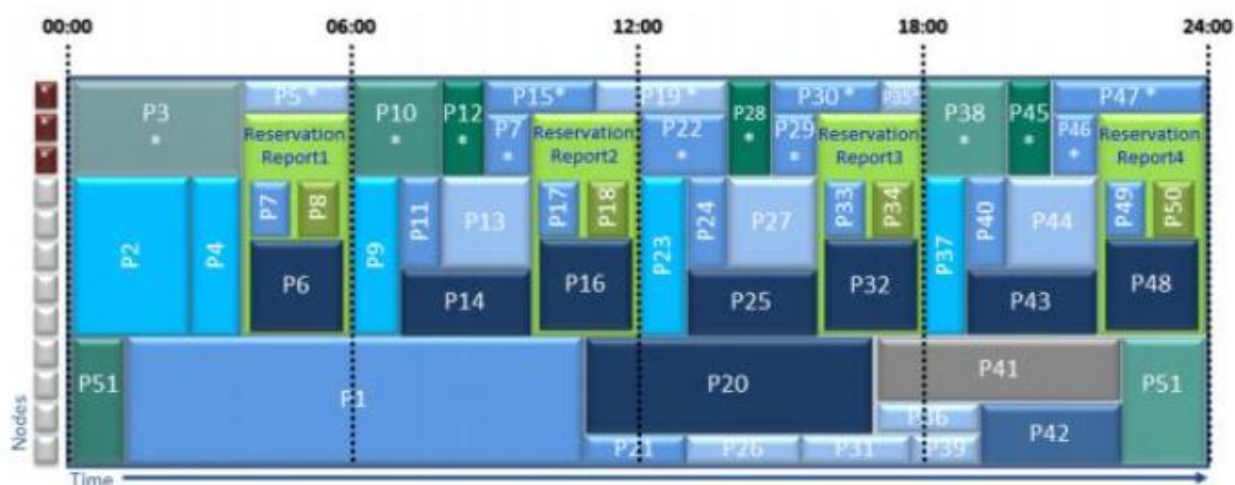


Рисунок 6 – Таблица расписания планировщика Moab

1.4.2.4. Компонент №3 программного комплекса SANDS

Одним из средств управления канальными ресурсами является 3 компонент программного комплекса SANDS, предназначенный для генерации таблиц расписания протокола СТП-ИСС-14. Данный компонент подробно описан в разделе 2.4.1.

ВЫВОДЫ ПО РАЗДЕЛУ 1

Задача управления канальными ресурсами является сложной. Особенно если речь идет о планировании канальных ресурсов. Планирование канальных ресурсов зачастую осуществляется посредством различных качеств сервиса, таких как «Планирование». Существует большое количество методов и средств обеспечения качества сервиса «Планирование» для различных коммуникационных протоколов. Но, так как протоколы имеют различный подход к планированию в зависимости от специфики и сферы применения, задача реализации новых средств планирования канальных ресурсов становится актуальной.

В результате изучения методов планирования канальных ресурсов было принято решение о том, что основное внимание будет уделено новому, перспективному методу планирования. В качестве средства, реализующего данный метод планирования, был выбран компонент 3 программного комплекса SANDS. Данный метод и средство проектирования сетей будут рассмотрены подробнее в разделе 2.

2. КАЧЕСТВО СЕРВИСА «ПЛАНИРОВАНИЕ» ПРОТОКОЛА СТП-ИСС-14. ТАБЛИЦЫ РАСПИСАНИЯ

2.1. Протокол СТП-ИСС-14. Типы качества сервиса протокола СТП-ИСС-14

Протокол СТП-ИСС-14 – это протокол транспортного уровня, работающий поверх стека протоколов SpaceWire. Данный протокол определяет транспортное взаимодействие узлов в сети, регламентирует форматы передаваемых данных и правила передачи сообщений между абонентами бортовой сети SpaceWire. Протокол СТП-ИСС-14 обеспечивает транспортировку данных между удаленными узлами сети с предоставлением требуемого качества сервиса в соответствии с приоритетами потоков данных. Важной особенностью протокола СТП-ИСС-14 является его гибкость. Протокол имеет ряд конфигурационных параметров, которые позволяют настраивать протокол в зависимости от потребностей разработчиков.

Протокол СТП-ИСС-14 поддерживает следующие типы качества сервиса.

– *Приоритетный тип качества сервиса.*

Данный тип качества сервиса для протокола СТП-ИСС-14 является основным. Согласно приоритетному типу качества сервиса, данные с более высокими приоритетами должны быть переданы первыми. СТП-ИСС-14 поддерживает 9 уровней приоритетов. Самый высокий приоритет имеют пакеты подтверждения и пакеты подтверждения транспортного соединения, далее – пакеты команд управления, повторные пакеты команд управления, служебные пакеты режима с установкой соединения, служебные пакеты синхронизации кредитов, пакеты срочных сообщений, повторные пакеты срочных сообщений, повторные пакеты обычных сообщений, а также пакеты обычных сообщений.

– *Качество сервиса «Гарантированная доставка данных».*

Данный тип качества сервиса обеспечивает подтверждение корректной доставки данных посредством отправки пакетов подтверждения. Пакеты подтверждения отправляются, если в пакете отсутствуют ошибки. Протокол обеспечивает повторную пересылку данных источником в случае отсутствия пакета подтверждения.

– *Качество сервиса «Негарантированная доставка данных»*

Данный тип качества сервиса обеспечивает передачу данных без подтверждения приема. При приеме пакета, не требующего подтверждения, приемник также проверяет корректность пакета, но не отправляет пакеты подтверждения.

– *Качество сервиса «Планирование»*

Данный тип качества сервиса подразумевает наличие единого расписания для всей сети, которое дает возможность отправлять данные только во время конкретных временных интервалов [14]. Данный тип качества сервиса подробнее описан в разделе 2.2.

2.2. Качество сервиса «Планирование»

2.2.1. Основные определения. Эпоха и тайм-слоты

Качество сервиса «Планирование» предназначено для передачи пакетов по сети в соответствии с заданным расписанием.

Данный тип качества сервиса обеспечивается следующим образом. На этапе конфигурации протокола задаются два конфигурационных параметра: «Количество временных интервалов» и «Длительность временного интервала». *Временные интервалы*, также называемые *тайм-слотами* – это промежутки времени, на которые разбито расписание работы сети. Расписание работает циклично, один цикл расписания называется *эпоха*. Длительность эпох вычисляется как произведение длительности тайм-слота и количества тайм-слотов. На Рисунок 7 представлена первая эпоха расписания, разбитая на N тайм-слотов.



Рисунок 7 – Эпоха и тайм-слоты

Расписание является единым для всей сети. Оно регламентирует передачу пользовательских данных для каждого узла в заданные для него тайм-слоты. При этом служебные пакеты СТП-ИСС-14 отправляются вне расписания, в соответствии со своим приоритетом. Расписание задается на этапе конфигурации протокола и хранится в оконечных узлах сети. Также в оконечных узлах сети работает таймер тайм-слота T_{TS} . Данный таймер отсчитывает время до окончания текущего тайм-слота на узле. Срабатывание таймера означает начало нового тайм-слота.

2.2.2. Прием меток времени

В каждом оконечном узле сети работает локальный счетчик времени. Для корректной передачи данных с качеством сервиса «Планирование» необходимо, чтобы данные счетчики были синхронизированы. Для этого в сети один из узлов становится *мастером времени*. Данный узел отправляет метки времени с помощью time-кодов SpaceWire. Остальные узлы сети получают данные метки времени и корректируют при необходимости локальные счетчики. При этом значение меток времени опускается, важным является только факт их получения (для мастера времени – отправки). Первая метка времени, в отличие от последующих, предназначена для оповещения узлов о начале

первой эпохи. После прихода первой метки времени сеть начинает работать в соответствии с расписанием. До момента прихода первой метки данные не отправляются.

Еще одним параметром, который должен быть задан на этапе конфигурации протокола, является «Окно ожидания метки времени». *Окно ожидания метки времени* определяет количество тайм-слотов в начале и конце эпохи. Метка времени, поступившая в данном окне, считается *актуальной*.

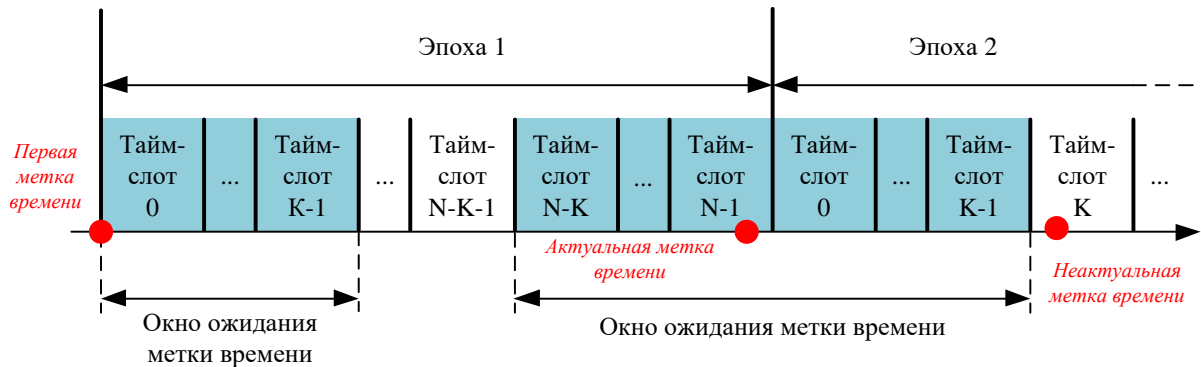


Рисунок 8 – Расписание и метки времени

На Рисунок 8 представлен пример расписания. Каждая эпоха данного расписания состоит из N тайм-слотов. Окно ожидания метки времени установлено в K тайм-слотов. Это означает, что метки времени, поступившие в первые K и последние K тайм-слотов эпохи будут считаться актуальными. Метки времени, поступившие вне окна ожидания метки времени считаются неактуальными. На Рисунок 8 приведены примеры первой метки времени, актуальной метки времени, а также неактуальной метки времени.

При приеме метки времени возможны следующие ситуации:

- *Актуальная метка времени принята в первые K тайм-слотов эпохи.* Прием метки времени в первые K интервалов эпохи означает, что произошла рассинхронизация локального счетчика времени и счетчика мастера времени. При приеме такой метки выполняется корректировка значения длительности тайм-слота. Узел, принявший актуальную метку, останавливает таймер T_{TS} и вычисляет новое значение длительности тайм-слота D_{TS_new} по формуле:

$$D_{TS_new} = D_{TS} + (\Delta t / N_{TS}),$$

где D_{TS} – это значение длительности тайм-слота до корректировки, Δt – время, прошедшее с начала текущей эпохи, N_{TS} – число тайм-слотов в эпохе.

После корректировки значения длительности тайм-слота, взводится таймер T_{TS} с новым значением D_{TS_new} .

- *Актуальная метка времени принята в последние K тайм-слотов эпохи.* Прием метки времени в последние K интервалов эпохи также означает, что произошла

рассинхронизация локального счетчика времени и счетчика мастера времени. При приеме такой метки выполняется корректировка значения длительности тайм-слота. Узел, принявший актуальную метку, останавливает таймер T_{TS} и вычисляет новое значение длительности тайм-слота D_{TS_new} по формуле:

$$D_{TS_new} = \Delta t / N_{TS},$$

где Δt – время, прошедшее с начала текущей эпохи, N_{TS} – число тайм-слотов в эпохе. После корректировки значения длительности тайм-слота, взводится таймер T_{TS} с новым значением D_{TS_new} .

- *Актуальная метка времени принята одновременно с завершением последнего тайм-слота.* Прием метки времени в момент срабатывания таймера T_{TS} для последнего тайм-слота в эпохе означает, что локальный счетчик времени и счетчик мастера времени синхронизированы. Корректировка значения длительности тайм-слота не производится, таймер T_{TS} продолжает работу с прежним значением.
- *Принята неактуальная метка времени.* Прием неактуальной метки времени не означает начало новой эпохи. При приеме неактуальной метки времени таймер T_{TS} продолжает работу. При этом в узле ведется учет принятых неактуальных меток времени. Прием трех неактуальных меток означает, что произошла значительная рассинхронизация локального счетчика времени и счетчика мастера времени. Прием третьей неактуальной метки определяет начало новой эпохи. Узел останавливает таймер T_{TS} и ожидает прием метки времени. Узел не будет передавать данные до прихода новой метки времени. После приема метки времени выполняется корректировка значения длительности тайм-слота. Новое значение длительности тайм-слота D_{TS_new} вычисляется по формуле:

$$D_{TS_new} = \Delta t / N_{TS},$$

где Δt – время, прошедшее с момента приема третьей неактуальной метки времени, N_{TS} – число тайм-слотов в эпохе. После корректировки значения длительности тайм-слота, взводится таймер T_{TS} с новым значением D_{TS_new} .

2.2.3. Передача данных в соответствии с расписанием

Каждый оконечный узел сети работает в соответствии с расписанием. Для каждого узла разрешается для отправки определенное число тайм-слотов. В такие тайм-слоты узел может отправлять пользовательские данные: информационные сообщения и команды управления. При этом узел может отправлять служебные пакеты в любое время, независимо от расписания.

Если узлу необходимо передать данные, он ожидает наступления разрешенного тайм-слота и начинает передачу. При наступлении неразрешенного тайм-слота узел

передает целиком пакет, предаваемый в данный момент, завершает передачу данных и ожидает наступления очередного разрешенного тайм-слота [14].

2.3. Новый метод планирования канальных ресурсов

При обеспечении качества сервиса «Планирование» ключевую роль играют таблицы расписания. Таблицу расписания необходимо задать таким образом, чтобы все узлы сети могли передать все пользовательские данные за отведенное время. Данная задача является сложной, так как при построении таблицы важно учитывать следующие параметры:

- ограничения, накладываемые на время доставки пакетов данных;
- периодичность генерации пакетов данных;
- тип маршрутизации в сети (червячная маршрутизация);
- маршруты передачи пакетов данных;
- приоритеты пакетов данных;
- порядок отправки пакетов данных с узлов;
- необходимость отправки пакетов подтверждения;
- наличие шага времени между тайм-слотами;
- максимально возможную загрузку каналов.

Алгоритмы построения таблиц расписания, рассмотренные в разделе 1.4.1.4, учитывают только часть вышеперечисленных параметров. Поэтому, для составления таблиц расписания для сети, работающей в соответствии с протоколом СТП-ИСС-14, был разработан новый метод планирования канальных ресурсов, учитывающий все параметры, необходимые для построения корректной таблицы расписания.

Входными параметрами данного метода являются:

- параметры трафиков, такие как: маршруты передачи трафиков в сети, размеры пакетов данных, период генерации пакетов данных, тип качества сервиса, допустимые задержки передачи пакетов данных, допустимые задержки передачи пакетов подтверждения, порядок отправки пакетов;
- число тайм-слотов в эпохе;
- максимальная длительность тайм-слота;
- максимальная длительность эпохи;
- шаг времени между тайм-слотами;
- максимальная загрузка каналов;
- запас размера пакета.

Метод планирования канальных ресурсов состоит из 7 последовательно выполняющихся шагов.

2.3.1. Шаг 1. Загрузка и проверка корректности входных данных.

На первом шаге метода планирования канальных ресурсов происходит загрузка всех входных данных и проверка их корректности. В случае, если какие-либо из входных данных некорректны, метод переходит к шагу 7 (раздел 2.3.7).

Также на данном шаге выполняются следующие действия:

- Формирование трафиков пакетов подтверждения (ответных трафиков). В случае, если узел передает пакеты данных с качеством сервиса «Гарантированная доставка данных», узел-получатель данных пакетов должен отправлять пакеты подтверждения на принятые пакеты. Наличие пакетов подтверждения важно учитывать при построении таблиц расписания. Поэтому на данном шаге формируются трафики пакетов подтверждений.
- Для каждого трафика в сети определяются маршруты на основании таблиц маршрутизации коммутаторов.
- На основании заданной очередности отправки пакетов формируются группы трафиков. В случае, если для трафиков не задана очередность, группы трафиков формируются согласно приоритетам.
- Скорости каналов корректируются исходя из заданной максимальной загрузки каналов;
- Размеры пакетов корректируются исходя из заданного запаса размера пакета.

2.3.2. Шаг 2. Проверка возможности передачи пакетов данных с максимально допустимыми задержками

На втором шаге метода планирования канальных ресурсов происходит проверка принципиальной возможности передачи всех пакетов данных с максимально допустимыми задержками. На данном шаге выполняются следующие действия:

- Для каждого пакета каждого трафика в сети вычисляются минимальные задержки, необходимые для передачи данных пакетов по маршрутам.
- Рассчитанные минимальные задержки сравниваются с заданными максимально допустимыми задержками. В случае, если для какого-либо трафика минимальная задержка превышает максимальную, это означает, что технических характеристик устройств в сети недостаточно для передачи данного трафика за заданную максимально допустимую задержку. В этом случае метод переходит к шагу 7 (раздел 2.3.7).

2.3.3. Шаг 3. Определение длительности эпохи и временных интервалов

На третьем шаге метода планирования канальных ресурсов происходит вычисление длительности эпохи и временных интервалов. На данном шаге выполняются следующие действия:

- Значение длительности эпохи вычисляется исходя из периодичности трафиков и шага времени между тайм-слотами.
- Рассчитанная длительность эпохи сравнивается с максимальной длительностью эпохи. В случае, если рассчитанная длительность превышает максимальную, метод переходит к шагу 7 (раздел 2.3.7).
- Длительность временных интервалов вычисляется исходя из рассчитанной длительности эпохи.

2.3.4. Шаг 4. Проверка пропускной способности сети

На четвертом шаге метода планирования канальных ресурсов проверяется, достаточно ли пропускной способности сети для передачи всех пакетов всех трафиков за время одной эпохи. На данном шаге выполняются следующие действия:

- Для каждого порта в узлах и коммутаторах вычисляется интенсивность поступления пакетов от всех трафиков за одну эпоху.
- Исходя из тактов работы узлов и коммутаторов за одну эпоху, вычисляется максимальная интенсивность обслуживания пакетов всех трафиков.
- Рассчитанные интенсивности поступления пакетов сравниваются с максимальной интенсивностью. В случае, если рассчитанная интенсивность для какого-либо пакета превышает максимальную, метод переходит к шагу 7 (раздел 2.3.7).

2.3.5. Шаг 5. Проверка необходимости планирования трафиков

На пятом шаге метода планирования канальных ресурсов проверяется, существует ли необходимость в планировании трафиков. На данном шаге выполняются следующие действия:

- Для каждого трафика выполняется поиск конфликтов. Конфликт может происходить между рассматриваемым трафиком и другими трафиками в сети за доступ к порту одного из устройств на маршруте рассматриваемого трафика.
- Для каждого трафика определяется худшее время доставки пакета по маршруту.
- Трафики разделяются на две группы. В первой группе находятся трафики, худшее время доставки пакетов которых не превышает максимально допустимое. Во второй группе находятся остальные трафики. В случае, если в первой группе нет трафиков, это означает, что необходимости в планировании нет.

2.3.6. Шаг 6. Планирование

Шестой шаг является главным в методе планирования канальных ресурсов. На данном шаге происходит непосредственно планирование трафиков и построение таблицы расписания.

В основе планирования трафиков и построения расписания лежит концепция генетического алгоритма и поиска с возвратом. Данная концепция заключается в итеративном процессе разрешения конфликтов, найденных на предыдущем шаге метода. Шестой шаг разделяется на два шага.

2.3.6.1. Шаг 6.1. Планирование неконфликтных трафиков

На данном шаге происходит построение расписания для трафиков, не конфликтующих с другими трафиками в сети. Это трафики, помещенные на шаге 5 во вторую группу.

На данном шаге выполняются следующие действия:

- Определяется расстановка начальных моментов генерации пакетов трафиков с учетом очередности, заданной пользователем.
- От начальной расстановки происходит расчет и планирование по времени передачи пакетов по сети с учетом пакетов подтверждения.
- Для того, чтобы учесть приоритеты пакетов при отправке с узла, выполняется корректировка расчета.
- Производится оценка полученного расчета. Если по результатам оценки очередность при отправке с узла соблюдается, а время доставки пакетов укладывается в максимально допустимое, то решение найдено. На основании этого расчета формируется таблица расписания. Если очередность не соблюдается, или время доставки превышает допустимое, то расстановка изменяется, а расчет выполняется заново. В случае, если получить расстановку, удовлетворяющую всем требованиям, невозможно, то получить расписание невозможно. Метод переходит к шагу 7 (раздел 2.3.7).

2.3.6.2. Шаг 6.2. Планирование конфликтных трафиков.

На данном шаге происходит построение расписания для трафиков, конфликтующих с другими трафиками в сети. Это трафики, помещенные на шаге 5 в первую группу. На данном шаге выполняются следующие действия:

- Определяется расстановка начальных моментов генерации пакетов трафиков с учетом очередности, заданной пользователем.
- Вычисляются верхние и нижние границы расстановки;

- От начальной расстановки происходит расчет и планирование по времени передачи пакетов по сети с учетом пакетов подтверждения.
- Для того, чтобы учесть приоритеты пакетов при отправке с узла, выполняется корректировка расчета.
- Производится оценка полученного расчета. Если по результатам оценки конфликты отсутствуют, очередность при отправке с узла соблюдается, а время доставки пакетов укладывается в максимально допустимое, то решение найдено. На основании этого расчета формируется таблица расписания. Если не все конфликты были разрешены, очередность не соблюдается, или время доставки превышает допустимое, то расстановка изменяется с учетом верхних и нижних границ, а расчет выполняется заново. В случае, если получить расстановку, удовлетворяющую всем требованиям, невозможно, то получить расписание невозможно. Метод переходит к шагу 7 (раздел 2.3.7).

2.3.7. Шаг 7. Вывод результатов работы

Седьмой шаг является последним шагом метода планирования канальных ресурсов. На данном шаге происходит вывод результатов работы. В случае, если метод прервался на одном из шагов, выводятся сообщение об ошибке и рекомендации по корректировке входных данных. Если алгоритм завершился успешно и удалось построить таблицу расписания выводится данная таблица [10].

2.4. Система автоматизированного проектирования бортовых космических сетей

Рассмотренный в разделе 2.3 метод планирования канальных ресурсов реализован в системе автоматизированного проектирования бортовых космических сетей (САПР БВС КА). Данная система автоматизированного проектирования разработана в 2016-2018 гг. институтом ВКиСТ в рамках гранта с министерства образования Российской Федерации совместно с промышленным партнером АО «Информационные спутниковые системы» [15, 16].

Разработанная САПР представляет собой программную систему SpaceWire Automated Network Design and Simulation (SANDS).

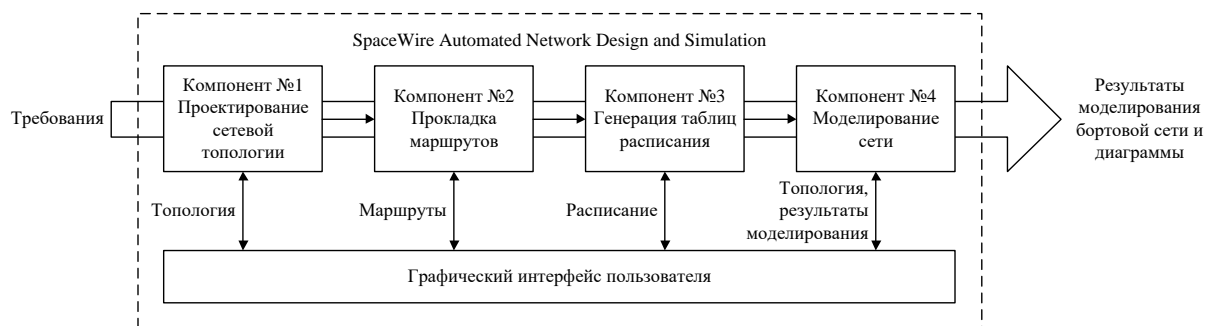


Рисунок 9 – Архитектура SANDS

SANDS поддерживает полный цикл разработки и моделирования бортовых сетей, начиная с проектирования и анализа топологии сети и заканчивая получением результатов моделирования работы сети, статистики, а также различных диаграмм. Архитектура SANDS представлена на Рисунок 9. SANDS состоит из четырех основных компонентов, а также графического интерфейса, обеспечивающего визуализацию проектируемой сети и возможности управления данной сетью. Графический интерфейс представлен на Рисунок 10.

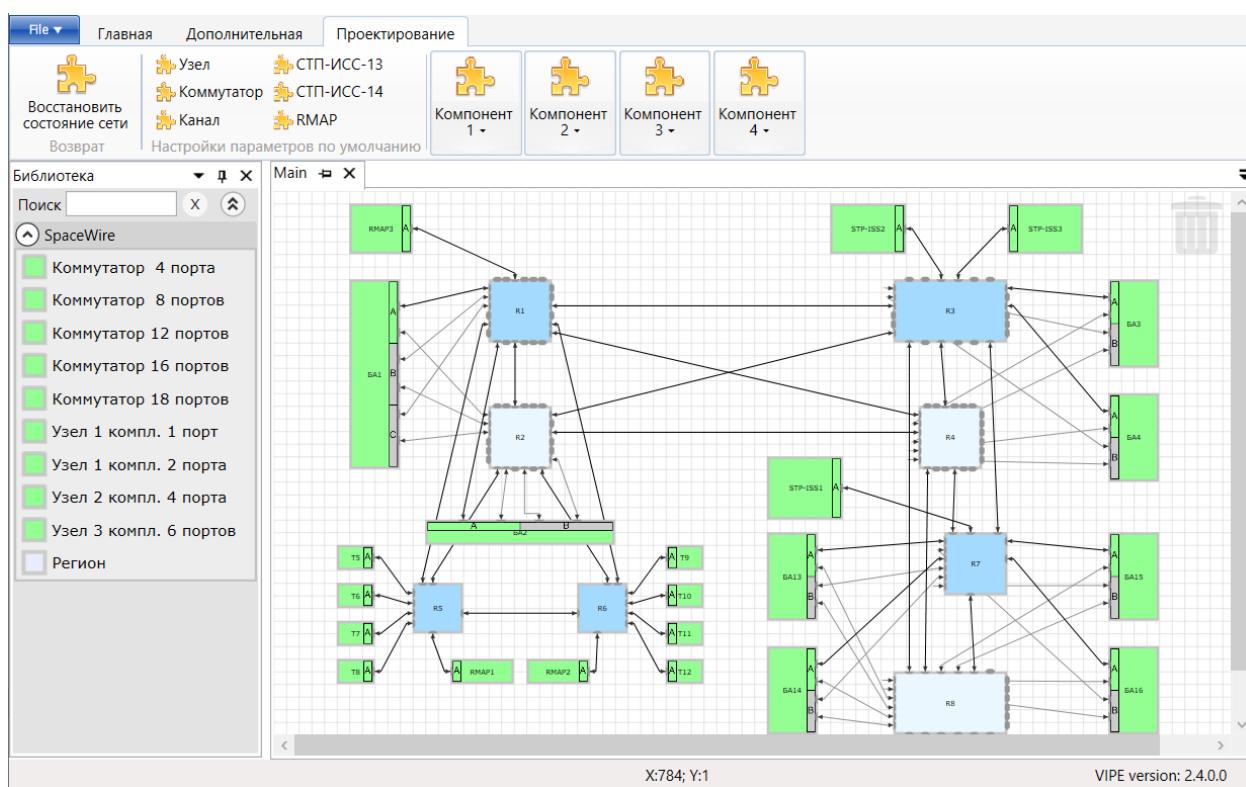


Рисунок 10 – Графический интерфейс SANDS

Основными компонентами SANDS являются:

- *Компонент 1. Проектирование сетевой топологии.* Первый компонент предназначен для проектирования топологии сети при помощи графического интерфейса. При проектировании сети для каждого узла, коммутатора и канала задаются различные параметры и характеристики. Данный компонент также

позволяет провести оценку физических характеристик спроектированной топологии сети, а также трансформацию сети для достижения требуемого уровня отказоустойчивости.

- *Компонент 2. Прокладка маршрутов.* Второй компонент предназначен для прокладки беступиковых маршрутов передачи данных в сети. Данный компонент производит поиск последовательности коммутаторов между источником и приемником и составляет таблицы маршрутизации для данных коммутаторов. Также компонент предоставляет расчет задержек для наихудшего случая передачи данных по сети.
- *Компонент 3. Генерация таблиц расписания.* Третий компонент предназначен для создания таблиц расписания для протокола СТП-ИСС-14, работающего с качеством сервиса «Планирование». При генерации таблиц принимается во внимание спроектированная топология сети, а также проложенные в ней маршруты. Компонент 3 программной системы SANDS является реализацией метода, представленного в разделе 2.3.
- *Компонент 4. Моделирование сети.* Четвертый компонент предназначен для моделирования работы сети. Данный компонент позволяет производить моделирование с двумя уровнями детальности: битовым и пакетным. По результатам моделирования пользователю предоставляется детальная статистическая информация о всех событиях, произошедших в сети, диаграммы и графическое представление результатов симуляции [15].

2.4.1. Работа с Компонентом 3. Генерация таблиц расписания

В данном разделе будет рассмотрен Компонент 3, являющийся реализацией метода планирования канальных ресурсов, представленного в разделе 2.3.

Работа с Компонентом 3 происходит через графический интерфейс, являющийся частью программной системы SANDS. Для запуска компонента на вкладке «Компонент 3» необходимо выбрать элемент «Построение расписаний», представленный на Рисунок 11.

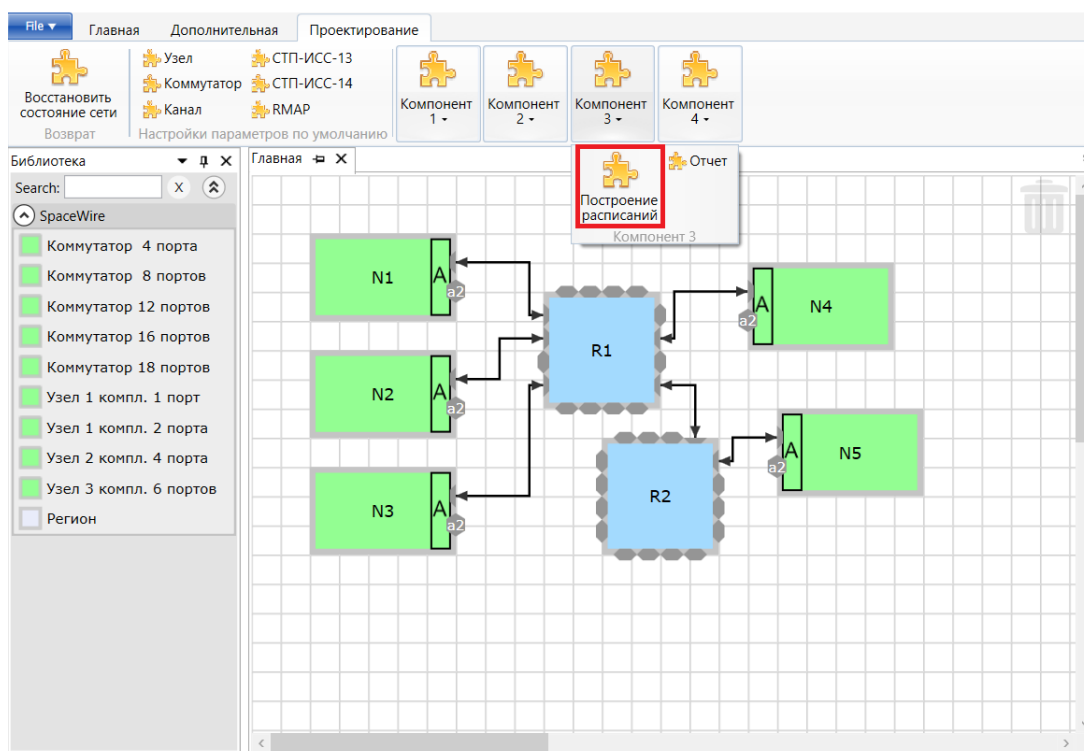


Рисунок 11 – Запуск компонента 3

После запуска компонента необходимо заполнить входные параметры Компонента 3 в появившемся окне «Окно запуска Компонента 3». Данное окно представлено на Рисунок 12.

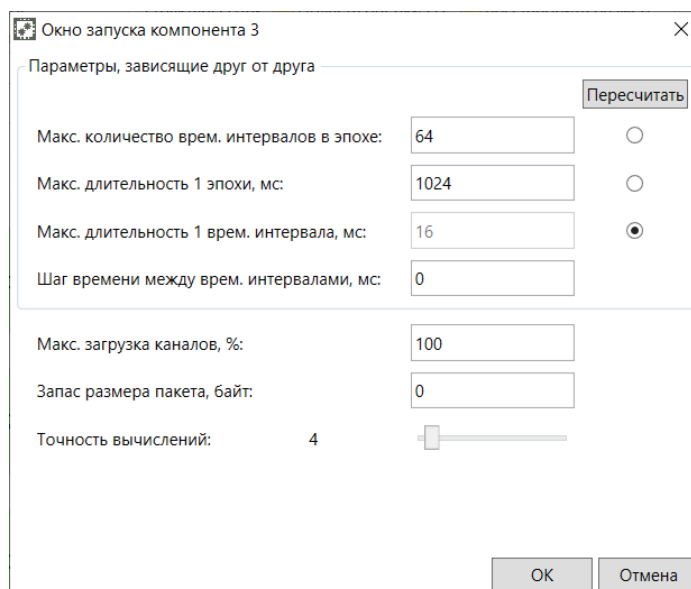


Рисунок 12 – Окно запуска Компонента 3

Результатом работы Компонента 3 являются:

- *Сообщение пользователю.* В случае, если Компоненту 3 не удалось построить расписание, пользователю выдается сообщение об ошибке с рекомендациями по корректировке настроек потоков данных, узлов или коммутаторов.

- *Рассчитанные параметры.* Для узлов с поддержкой протокола СТП-ИСС-14 с включенным качеством сервиса «Планирование» рассчитываются длительность эпохи с учетом шага времени между тайм-слотами, длительность тайм-слота с учетом рассчитанной длительности эпохи, окно ожидания метки времени, период отправки меток времени, а также расписание отправки данных.
- *Таблица расписания.* В данной таблице отмечаются тайм-слоты, выделенные для передачи данных для каждого узла.

Результаты работы Компонента 3 выводятся в отдельном окне «Отчет Компонента 3». Пример результатов работы Компонента 3 представлен на Рисунок 13 [17].

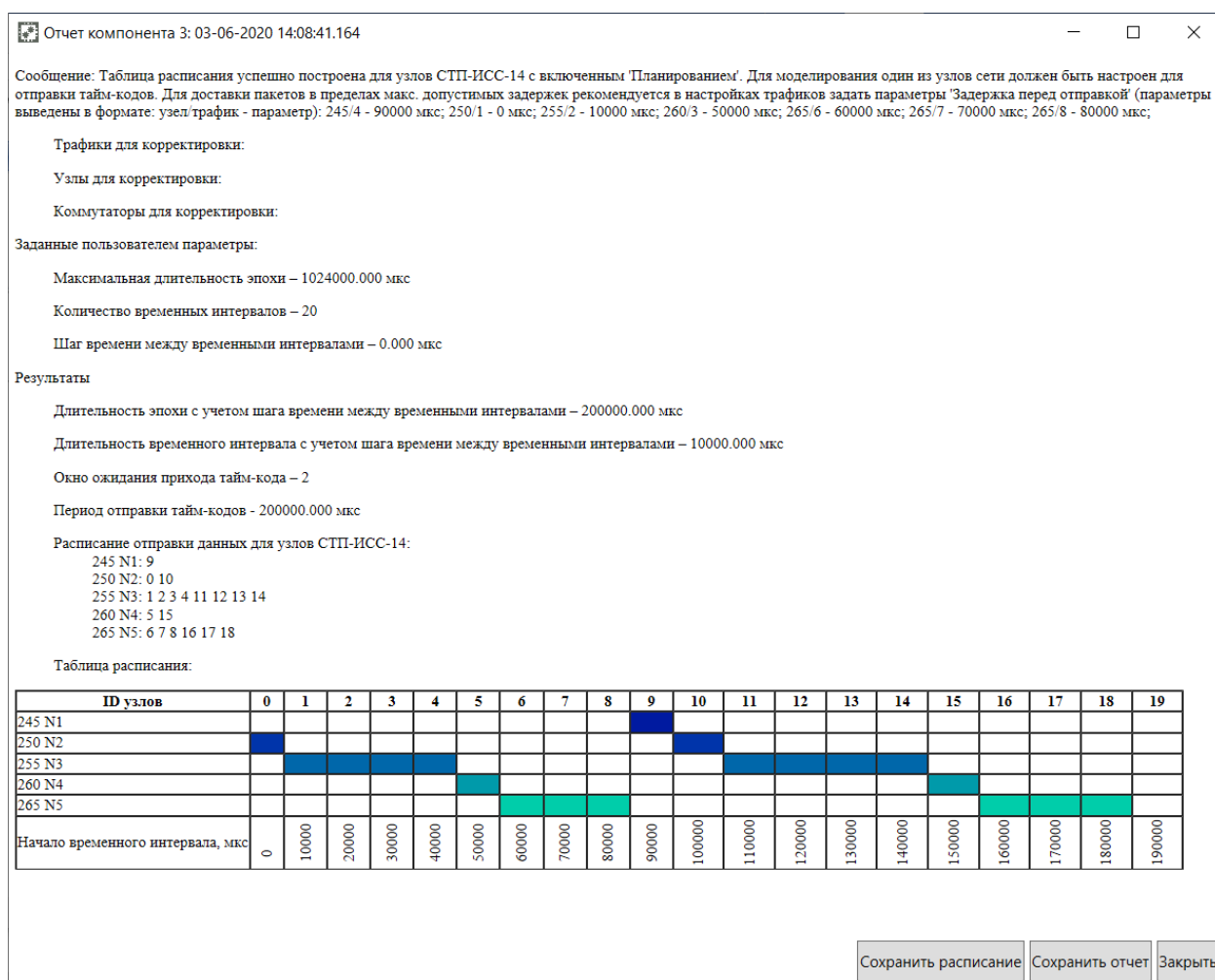


Рисунок 13 – Результаты работы Компонента 3

ВЫВОДЫ ПО РАЗДЕЛУ 2

В данном разделе было проведено исследование нового метода планирования канальных ресурсов и средства, реализующего данный метод – компонента 3 программного комплекса SANDS. Данный компонент в полной мере реализует рассмотренный метод и является удобным для пользователя. Компонент 3 позволяет сгенерировать таблицу расписания для сети, спроектированной в программном комплексе SANDS. Однако

возможна ситуация, в которой пользователю необходимо будет создать и проверить собственную таблицу расписания. Компонент 3 не реализует данную функцию, поэтому был сделан вывод о необходимости реализации интерфейса, позволяющего проверить пользовательскую таблицу расписания. В данной выпускной квалификационной работе был разработано приложение, реализующее графический интерфейс для проверки таблицы расписания. Подробно приложение описано в разделе 3.

3. ПРИЛОЖЕНИЕ ДЛЯ ПРОВЕРКИ КОРРЕКТНОСТИ ТАБЛИЦЫ РАСПИСАНИЯ

3.1. Назначение приложения для проверки корректности таблицы расписания

Разработанное приложение для проверки корректности таблицы расписания предназначено для проверки таблиц расписания, заданных пользователем. Данное приложение позволяет проверять таблицы расписания для бортовых сетей SpaceWire, включающих узлы с поддержкой протокола СТП-ИСС-14 и включенным качеством сервиса «Планирование». Приложение проводит ряд проверок, в результате которых можно сделать вывод о корректности таблицы расписания.

Основой данного приложения послужил экспериментальный программных компонент SANDS «Подсистема автоматизированного трафика в BBC КА» (Компонент 3), представленный в разделе 2.3.

3.2. Входные и выходные данные

3.2.1. Входные данные

Входными данными приложения для проверки корректности таблицы расписания являются следующие данные.

3.2.1.1. Топология сети

Разработанное приложение позволяет совершать проверку корректности таблицы расписания для сетей, спроектированных в SANDS. Пример сети, спроектированной в SANDS представлен на Рисунок 14.

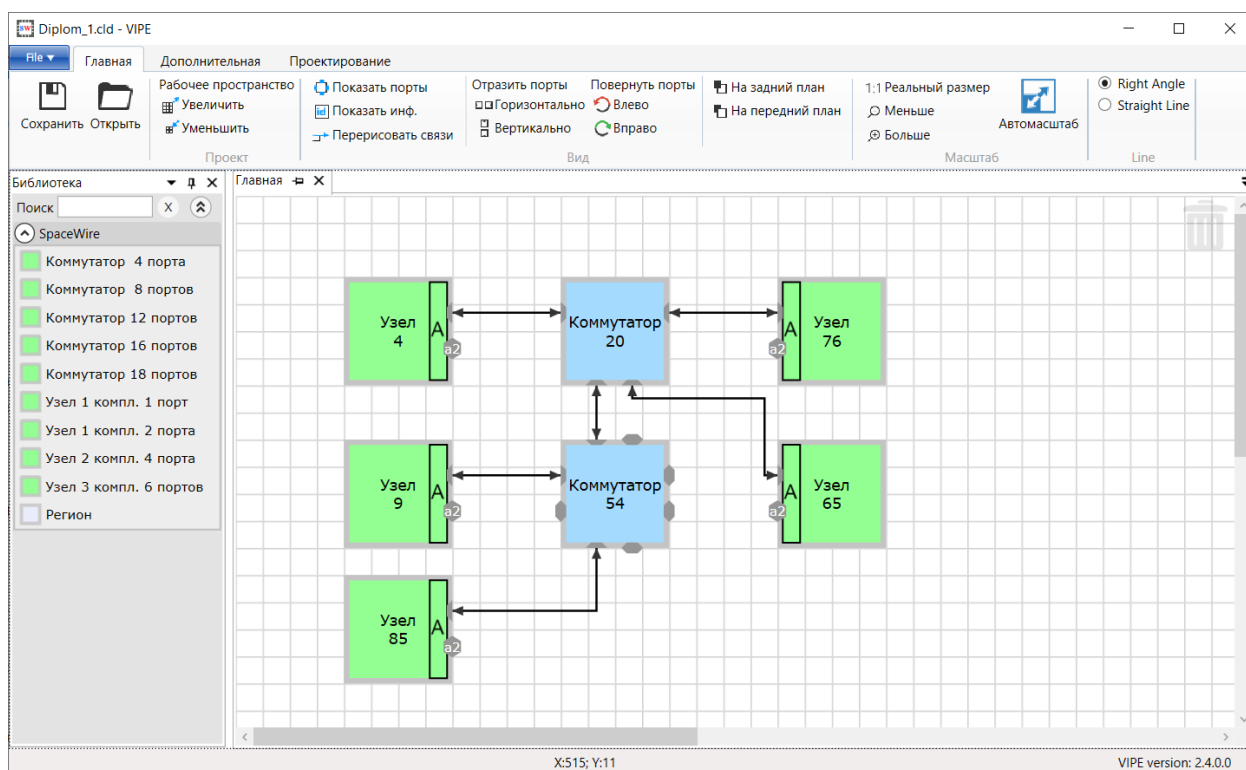


Рисунок 14 – Сеть, спроектированная в SANDS

SANDS генерирует xml-файл, содержащий информацию о топологии сети, параметрах узлов, коммутаторов, трафиков, маршрутах в сети и другую информацию о спроектированной сети. Данный xml-файл является входным файлом приложения для проверки корректности таблицы расписания.

3.2.1.1.1. Структура Xml-файла

Xml-файл, сгенерированный SANDS, состоит из двух основных частей:

a. *Описание топологии сети (тег <setting name="Topology">)*

Данная часть содержит информацию о всех элементах, из которых состоит спроектированная сеть: узлах (тег <node_list>), коммутаторах (тег <switch_list>) и каналах (тег <channel_list>). На Рисунок 15 представлен пример xml-файла.

```
<setting name="Topology">
  <node_list>
    <node node_id="269" name="A" reservation_units="1" region_id="0">
      <node_unit node_unit_id="1" isActive="true">
        <port port_id="1" isActive="true" />
      </node_unit>
    </node>
    <node node_id="275" name="B" reservation_units="1" region_id="0">
      <node_unit node_unit_id="1" isActive="true">
        <port port_id="1" isActive="true" />
      </node_unit>
    </node>
  </node_list>
  <switch_list>
    <switch switch_id="259" name="Sw1" region_id="0" isActive="true" active_switch_id="">
      <port port_id="1" reg_log_address="" />
      <port port_id="2" reg_log_address="" />
      <port port_id="3" reg_log_address="" />
      <port port_id="4" reg_log_address="" />
    </switch>
  </switch_list>
  <channel_list>
    <channel channel_id="277" name="" device1="node" device1_id="269" unit1_id="1" port1_id="1" device2="switch" device2_id="259"
      unit2_id="" port2_id="1" from_device1_to_device2="true" from_device2_to_device1="true" />
    <channel channel_id="278" name="" device1="switch" device1_id="259" unit1_id="" port1_id="2" device2="node" device2_id="275"
      unit2_id="1" port2_id="1" from_device1_to_device2="true" from_device2_to_device1="true" />
  </channel_list>
</setting>
```

Рисунок 15 – Описание топологии сети в xml-файле

b. *Описание элементов сети (тег <setting name="Network Elements">)*

Данная часть содержит подробное описание каждого элемента сети. Список узлов (тег <node_list>) содержит подробную информацию о настройках каждого узла (тег <node>), комплектах (тег <node_unit>) и приложениях (тег <application_list>), о сообщениях, передаваемых каждым узлом (теги <stp_iss_message> и <trmap_message>), описание транспортного уровня узла (тег <transport_layer_list>) и другое. На Рисунок 16 представлен пример описания одного узла в xml-файле. Помимо описания узлов данная часть содержит подробное описание коммутаторов и каналов в сети.

```

<setting name="Network Elements">
  <node_list>
    <node node_id="269" name="A" do_evaluate_fault_tolerant="true" do_evaluate_completion_area="true" allow_adding_ports_and_units="false" isSwitching="false" startSwitching="10 ms" switchingTime="10 ms" switchingCount="2">
      <node_unit node_unit_id="1">
        <application_list>
          <application id="1" name="trafgen_stp_iss_14" log_address="32" stp_delay_for_command_messages="0 ms" stp_delay_for_common_messages="0 ms" stp_delay_for_transport_messages="0 ms">
            <stp_iss_message id="1" msg_type="2" type="EXPRESS_MESSAGE" dest_path_address="" dest_logical_address="33" source_path_address="" source_logical_address="32" data_type="EXPRESS_DATA" secondary_hdr_flag="NO" data_length="512" amount_messages="60" packet_period="4 ms" sending_delay="6 ms" max_acceptable_latency="0 ms" max_acceptable_ack_latency="0 ms" is_critical="false" bundle_size="1" group_id="" traf_index_in_group="">
              <flow total="" optimal="" max_delay="" need_ack="False" addressing_type="0" packet_size="526" dest_node_id="275" />
            </stp_iss_message>
          </application>
        </application_list>
        <transport_layer_list>
          <transport_layer id="1" app_id="1" name="stp_iss_14" command_packet_life_timer="1.25 ms" express_packet_life_timer="2 ms" general_packet_life_timer="4 ms" tc_express_packet_life_timer="64 ms" tc_general_packet_life_timer="128 ms" tc_retry_timer="64 ms" retry_timer="1 ms" guaranteed_qos="false" general_tx_buffer_size="131158" general_tx_buffer_packets="16" express_tx_buffer_size="131158" express_tx_buffer_packets="16" command_tx_buffer_size="96" command_tx_buffer_packets="2" rx_buffer_size="4184" rx_buffer_packets="2000" tc_rx_buffer_size="131158" tc_rx_buffer_packets="2" scheduling_qos="false" time_slots_number="32" time_slot_duration="31.25 ms" scheduling_table="0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31" window_size="2" stand_by_timer="525 ms" />
        </transport_layer_list>
        <network_layer clock="100">
          <ports_list>
            <port port_id="1" name="spacewire" rate="50" data_clock="3" code_clock="2" null_clock="4" fct_clock="16" output_buffer_size="64" isFail="false" failTime="10 ms" />
          </ports_list>
        </network_layer>
        <failure_properties failure_state="false" delay="0 ms" />
      </node_unit>
    </node>
  </node_list>

```

Рисунок 16 – Описание узла в xml-файле

3.2.1.2. Параметры таблицы расписания

Для создания таблицы расписания пользователь должен задать значения следующих параметров:

- *Длина эпохи, мс.* Данный параметр должен принимать значение в диапазоне от 1 мс до 1024 мс. Данное ограничение обусловлено возможностями аппаратуры, для которой составляется таблица расписания.
- *Шаг между тайм-слотами, мс.* Данный параметр должен принимать значение от 0 мс до 1 мс. Шаг времени между тайм-слотами позволяет строить расписания для реальной бортовой аппаратуры, работающей на основании тактовых сигналов от тактового генератора.
- *Загрузка каналов, %.* Данный параметр должен принимать значение от 1% до 100%.
- *Запас размера пакета, байт.* Данный параметр может принимать значение от 0 байт.
- *Количество тайм-слотов.* Данный параметр должен принимать значение от 0 до 256 тайм-слотов.

Параметры заполняются пользователем в окне приложения. Процесс заполнения параметров подробно описан в разделе 3.3.1.

3.2.1.3. Таблица расписания

После задания параметров, генерируется таблица расписания. Размер таблицы вычисляется исходя из заданного пользователем параметра «количество тайм-слотов» и числа узлов с включенным качеством сервиса «планирование». Данная таблица

заполняется пользователем в окне приложения путем закрашивания ячеек таблицы. Подробнее описание заполнения таблицы представлено в разделе 3.3.1.

3.2.1.4. Задержка перед отправкой

Для каждого трафика пользователь может задать задержку перед отправкой. Для этого пользователь указывает ID трафика и номер тайм-слота. Задержка устанавливается на начало выбранного тайм-слота. По умолчанию, для каждого трафика задержка перед отправкой установлена в 0 мс. Подробнее описание процесса задания задержек описан в разделе 3.3.1.

3.2.2. Выходные данные

3.2.2.1. Сообщения пользователю

Основными выходными данными приложения для проверки корректности таблицы расписания являются сообщения пользователю. Данные сообщения содержат информацию о результатах проверки таблицы. В случае, если таблица некорректна, сообщения могут содержать рекомендации для пользователя по корректировке таблицы или сети.

3.2.2.2. Списки трафиков, узлов и коммутаторов для корректировки

В случае обнаружения ошибок в трафиках, узлах, либо коммутаторах пользователю выдаются списки узлов, трафиков и коммутаторов, в которые необходимо внести изменения для того, чтобы проверяемая таблица расписания стала корректной.

3.2.2.3. Рекомендуемая длина эпохи

Также, в результате проверок таблицы, пользователю выдается рекомендуемая для данной таблицы длина эпохи, рассчитанная на основе параметров трафиков.

3.3. Интерфейс приложения для проверки корректности таблицы расписания

Интерфейс представляет собой состоит из двух основных окон и ряда окон-сообщений, всплывающих при возникновении различных ошибок. Основное окно «Проверка корректности таблицы расписания СТП-ИСС-14» появляется при запуске приложения. Второе окно «Результаты проверки таблицы расписания СТП-ИСС-14» появляется в случае, если в результате проверок обнаружено, что заданное пользователем расписание некорректно.

3.3.1. Окно «Проверка корректности таблицы расписания СТП-ИСС-14»

Окно «Проверка корректности таблицы расписания СТП-ИСС-14» предназначено для ввода параметров, задания таблицы расписания и задержек.

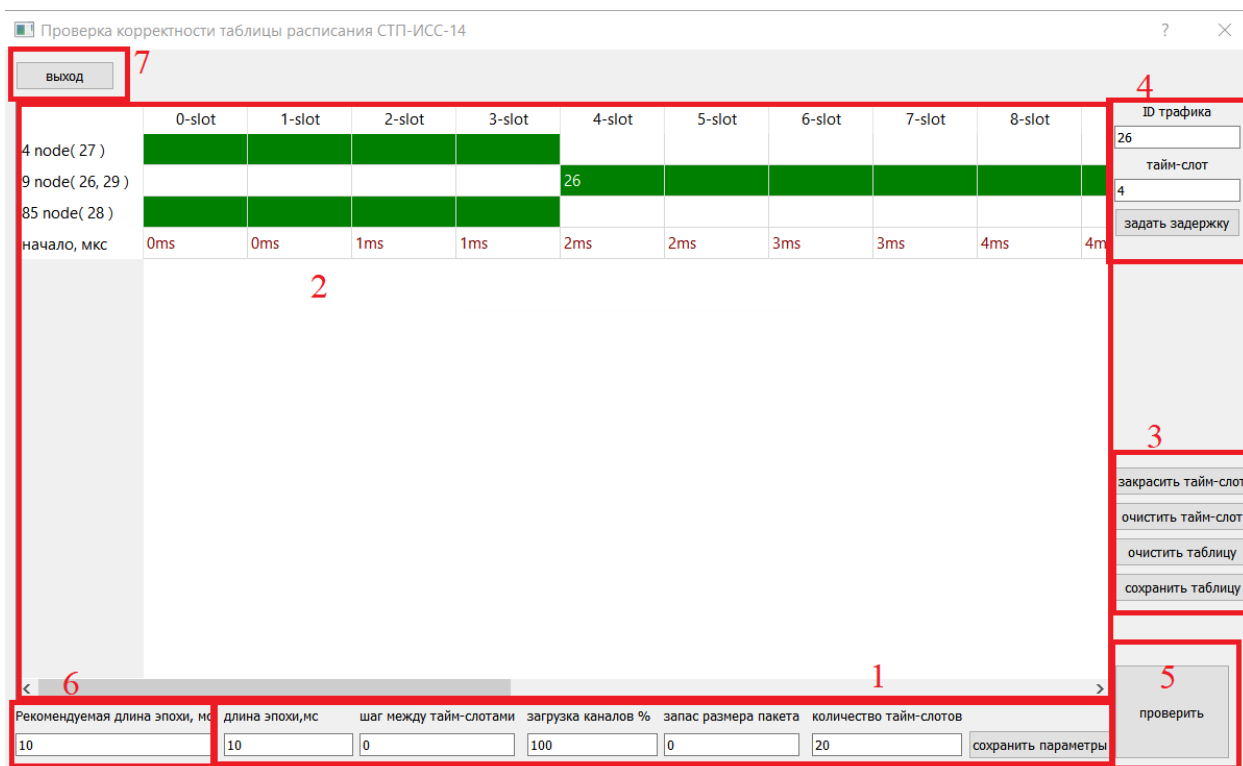


Рисунок 17 – Окно «Проверка корректности таблицы расписания СТП-ИСС-14»

Окно «Проверка корректности таблицы расписания СТП-ИСС-14», представленное на Рисунок 17, включает в себя следующие области:

- *Параметры (область №1)*. Данная область состоит из 5 полей, предназначенных для задания пользователем параметров таблицы. Значения данных параметров должны лежать в диапазонах, описанных в разделе. При вводе некорректного значения приложение генерирует всплывающее окно с предупреждением. Пример всплывающего окна с предупреждением представлен на Рисунок 18. Полный перечень всплывающих окон и ситуаций, при которых они генерируются, представлен в таблице.

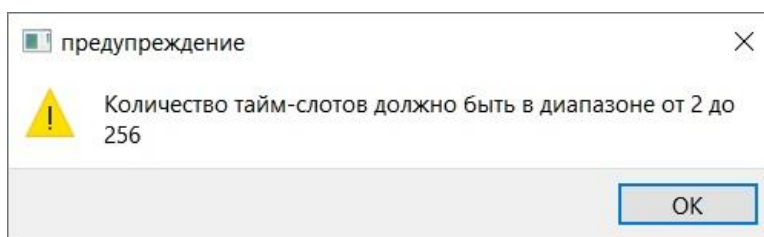


Рисунок 18 – Всплывающее окно с предупреждением о некорректном значении параметра "Количество тайм-слотов"

После заполнения всех параметров пользователь должен нажать кнопку «сохранить параметры», также расположенную в данной области. При нажатии кнопки параметры сохраняются в памяти таблицы. Также, после нажатия кнопки в области №2 генерируется пустая таблица расписания.

- *Таблица расписания (область №2).* Данная таблица генерируется пустой. Число столбцов таблицы зависит от параметра «количество тайм-слотов», а число строк – от числа узлов СТП-ИСС-14 с включенным качеством сервиса «Планирование» в сети. Данная таблица заполняется пользователем с помощью кнопок, расположенных в области №3. Для того, чтобы разрешить какой-либо тайм-слот для отправки, необходимо выделить данный тайм-слот в таблице и нажать кнопку «закрасить тайм-слот». Для того, чтобы запретить какой-либо тайм-слот для отправки, необходимо выделить данный тайм-слот в таблице и нажать кнопку «очистить тайм-слот». Также пользователь может очистить заполненную таблицу с помощью кнопки «очистить таблицу». При нажатии на данную кнопку все тайм-слоты в таблице становятся запрещенными для отправки.
- *Работа с таблицей (область №3).* Данная область состоит из 4 кнопок, предназначенных для работы с таблицей расписания. Помимо ранее упомянутых кнопок «закрасить тайм-слот», «очистить тайм-слот» и «очистить таблицу» в данной области присутствует кнопка «сохранить таблицу», предназначенная для сохранения таблицы в памяти приложения.
- *Параметры задержки (область №4).* Данная область предназначена для задания пользователем задержек перед отправкой для трафиков. Для того, чтобы задать задержку, необходимо ввести в поле «ID трафика» ID трафика, присутствующего в данной сети, а в поле «тайм-слот» - номер тайм-слота, на начало которого необходимо установить задержку. После ввода значений в поля пользователь должен нажать кнопку «Задать задержку». После нажатия на кнопку в заданном тайм-слоте появится цифра, означающая ID трафика, для которого была задана задержка.
- *Кнопка «Проверить» (область №5).* Данная кнопка предназначена для запуска процесса проверки заданной таблицы расписания. Пользователь должен запустить проверку только после ввода и сохранения параметров, заполнения и сохранения таблицы, а также, при необходимости, после задания задержек.
- *Рекомендуемая длина эпохи (область №6).* Данная область представляет собой текстовое поле. После проверки корректности таблицы расписания данное поле будет содержать в себе рекомендуемую длину эпохи для проверяемого расписания.
- *Кнопка «Выход» (область №7).* Данная кнопка предназначена для выхода из приложения.

3.3.2. Окно «Результаты проверки корректности таблицы расписания СТП-ИСС-14»

Окно «Результаты проверки таблицы расписания СТП-ИСС-14» предназначено для вывода информации о выявленных в заданной таблице расписания ошибках.

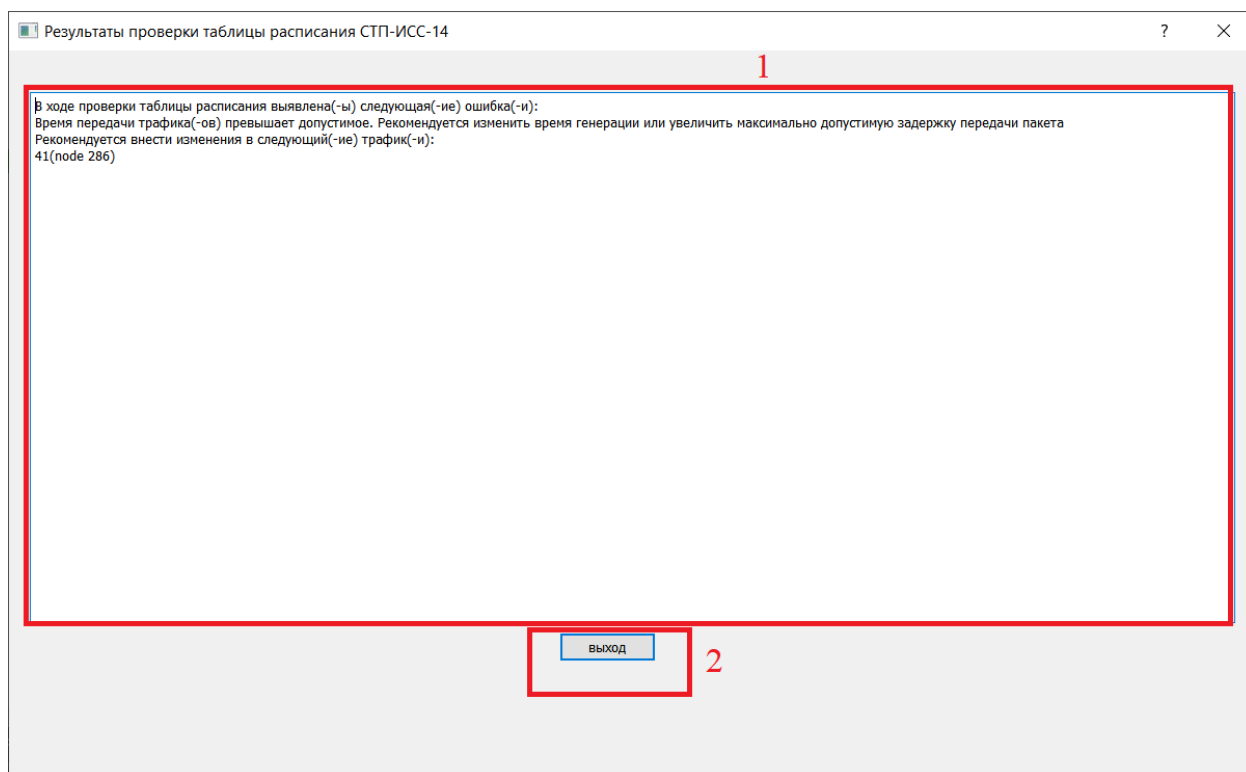


Рисунок 19 – Окно «Результаты проверки корректности таблицы расписания СТП-ИСС-14»

Окно «Результаты проверки корректности таблицы расписания СТП-ИСС-14», представленное на Рисунок 19, включает в себя две области:

- *Сообщение пользователю (Область №1).* Данная область представляет собой текстовое поле. В данном поле выводится сообщение с описанием ошибки и рекомендациями по корректировке ошибки для пользователя. Также в данном поле может быть выведено три списка: список трафиков для корректировки, список узлов для корректировки, и список коммутаторов для корректировки.
- *Кнопка «Выход» (Область №2).* Данная кнопка предназначена для закрытия окна.

3.4. Проверки таблиц расписания

3.4.1. Проверки корректности входных данных

До запуска проверки расписания, заданного пользователем, приложение проверяет корректность входных данных:

- Проверка параметров таблицы на соответствие допустимым значениям. Данная проверка производится на этапе заполнения пользователем полей в области №1 окна «Проверка корректности таблицы расписания СТП-ИСС-14». Подробнее данная проверка описана в разделе 3.3.1.

- Проверка числа узлов в сети. Для того, чтобы построение расписания было возможным, необходимо чтобы в сети было как минимум два узла, между которыми будет осуществляться передача данных. Проверка числа узлов в сети производится при нажатии кнопки «сохранить параметры», в процессе генерации пустой таблицы.
- Проверка числа трафиков. Для того, чтобы построение расписания было возможным, необходимо чтобы в сети было как минимум два трафика СТП-ИСС-14 с включенным качеством сервиса «Планирование». Проверка числа трафиков производится при нажатии кнопки «сохранить параметры», в процессе генерации пустой таблицы.
- Проверка заполнения таблицы. Пользователь должен задать таблицу расписания таким образом, чтобы для каждого узла в таблице был разрешен для отправки данных как минимум один тайм-слот. Данная проверка производится при нажатии кнопки «сохранить таблицу».

В случае, если одна из проверок не будет пройдена, приложение сгенерирует всплывающее окно с текстом ошибки. Полный перечень возможных всплывающих окон представлен в Таблица 1 – Всплывающие окна. Если все проверки пройдены, пользователь может нажать кнопку «проверить». По нажатию данной кнопки последовательно запускаются следующие проверки.

3.4.2. Проверки корректности сети

Данная группа проверок предназначена для проверки сети, спроектированной в SANDS. В нее входят проверки:

- Проверка коммутаторов. В спроектированной в SANDS пользователем сети обязательно должны присутствовать коммутаторы, соединенные с узлами каналами. Также в коммутаторах должна быть настроена таблица маршрутизации.
- Тип транспортного протокола. Для каждого трафика тип транспортного протокола на узле-получателе должен совпадать с типом транспортного протокола на узле-отправителе.

В случае, если одна из проверок не будет пройдена, приложение завершит проверку и сгенерирует всплывающее окно, представленное на Рисунок 20.

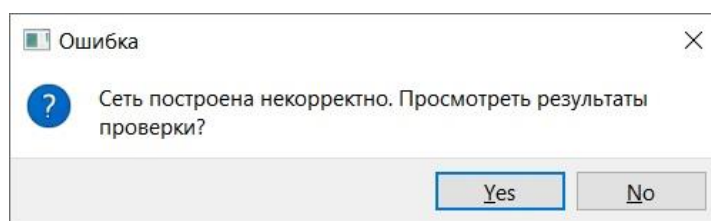


Рисунок 20 – Всплывающее окно, генерируемое в результате обнаружения ошибки при проверке корректности сети

При нажатии в окне кнопки «Yes», открывается окно «Результаты проверки таблицы расписания СТП-ИСС-14», содержащее сообщение с текстом ошибки. Полный перечень возможных сообщений представлен в Таблица Б.1. Также данное окно может содержать списки трафиков, узлов и коммутаторов, в которых данная ошибка была обнаружена. Если все проверки пройдены, приложение производит следующие проверки.

3.4.3. Проверки возможности построения расписания для заданной сети

Данная группа проверок предназначена для проверки возможности построения расписания на заданной спроектированной пользователем сети. Данные проверки являются реализацией шагов 2, 4 и 5 алгоритма Компонента 3. Описание данного алгоритма представлено в разделе 2.3. В группу входят следующие проверки:

- Проверка возможности передачи пакетов с допустимыми задержками. Представляет собой реализацию шага 2 алгоритма Компонента 3.
- Проверка пропускной способности сети. Представляет собой реализацию шага 4 алгоритма Компонента 3.
- Проверка возможности разрешения конфликтов. Представляет собой реализацию шага 5 алгоритма Компонента 3.

В случае, если одна из проверок не будет пройдена, приложение завершит проверку и сгенерирует всплывающее окно, представленное на Рисунок 21.

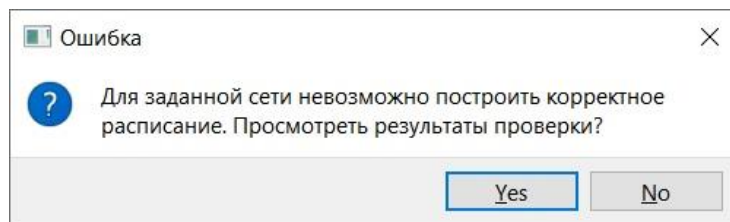


Рисунок 21 – Всплывающее окно, генерируемое в результате обнаружения ошибки при проверке возможности построения расписания

При нажатии в окне кнопки «Yes», открывается окно «Результаты проверки таблицы расписания СТП-ИСС-14», содержащее сообщение с текстом ошибки. Полный перечень возможных сообщений представлен в Таблица Б.1. Также данное окно может содержать списки трафиков, узлов и коммутаторов, в которых данная ошибка была обнаружена.

Если все проверки пройдены, значит для рассматриваемой сети возможно построить корректное расписание. Приложение переходит непосредственно к проверке корректности заданного расписания

3.4.4. Проверки корректности таблицы расписания

Данная группа проверок направлена на проверку заданной пользователем таблицы расписания.

3.4.4.1. Проверка возможности передачи первых пачек трафиков

На данном этапе проверяется возможность передачи первых пачек трафиков. Данная проверка осуществляется следующим образом:

- а. Каждая пачка трафика характеризуется следующими параметрами:
- GenTime – задержка перед отправкой первой пачки пакета;
 - SendTime – время, когда пачка начинает фактически передаваться;
 - EndTime – время, за которое пачка полностью передастся и освободит свой маршрут;
 - DeliveryTime – полное время передачи пачки от узла-отправителя до узла-получателя;
 - DeadlineTime – максимально допустимая задержка передачи пакета.

Для первой пачки каждого трафика данные параметры принимают следующие значения:

- Задержка GenTime первой пачки трафика может быть задана пользователем в окне «Проверка корректности таблицы расписания СТП-ИСС-14» приложения. По умолчанию задержка устанавливается на начало тайм-слота 0, в значение 0 ms.
- SendTime первой пачки устанавливается на начало первого разрешенного для данного трафика тайм-слота. Первым разрешенным для трафика тайм-слотом считается первый разрешенный тайм-слот для узла-отправителя данного трафика из таблицы расписания. Таблица расписания задается пользователем в окне «Проверка корректности таблицы расписания СТП-ИСС-14» приложения.
- В случае, если GenTime больше, чем SendTime, в SendTime записывается значение GenTime.
- EndTime вычисляется по формуле:

$$\text{EndTime} = \text{SendTime} + \text{PTime},$$

где PTime – время передачи пачки текущего трафика. Данное время вычисляется при проведении проверки возможности передачи пакетов с допустимыми задержками.

- DeliveryTime вычисляется по формуле:

$$\text{DeliveryTime} = \text{EndTime} - \text{GenTime}$$

- Задержка DeadlineTime первой пачки трафика может быть задана пользователем с помощью SANDS.
- б. Полное время передачи пакета DeliveryTime не должно превышать максимально допустимую задержку передачи пакета DeadlineTime. Данное условие проверяется для каждого трафика. В случае, если условие не выполняется для одного или нескольких трафиков, алгоритм завершается.

- с. Также проверяется, достаточно ли тайм-слотов выделено для передачи первой пачки каждого трафика. Если время EndTime превышает размер первой позиции пользовательской таблицы расписания, то тайм-слотов выделено недостаточно. Позицией пользовательской таблицы расписания считаются идущие подряд разрешенные тайм-слоты.

Данное условие проверяется для каждого трафика. В случае, если условие не выполняется для одного или нескольких трафиков, алгоритм завершается.

3.4.4.2. Проверка возможности построения расписания для бесконфликтных трафиков в выделенных тайм-слотах

Данная проверка представляет собой шаг 6.1 алгоритма Компонента 3 со следующими изменениями:

- Начальная расстановка трафиков. В оригинальном алгоритме время GenTime и SendTime устанавливается на начало первого тайм-слота, в 0 ms. В измененной версии начальная расстановка изменяется, GenTime и SendTime устанавливается на значения, полученные при выполнении проверки возможности передачи первых пачек трафиков (пункт).
- При смещении позиций пачек учитываются разрешенные и неразрешенные тайм-слоты. Если после смещения SendTime или EndTime пачки трафика попадает на неразрешенный тайм-слот, SendTime сдвигается на начало следующего разрешенного тайм-слота. Если разрешенных тайм-слотов больше нет, проверка завершается и пользователю выдается сообщение «Трафик(-и) невозможно передать за выделенное время. Рекомендуется увеличить количество разрешенных тайм-слотов», а также список трафиков, для которых данное условие не выполняется. Если после сдвига DeliveryTime превышает максимально допустимую задержку передачи пакета DeadlineTime, проверка завершается.

3.4.4.3. Проверка возможности построения расписания для конфликтных трафиков в выделенных тайм-слотах

Данная проверка представляет собой шаг 6.2 алгоритма Компонента 3 со следующими изменениями:

- Начальная расстановка изменяется также, как и в проверке возможности построения расписания для бесконфликтных трафиков в выделенных тайм-слотах.
- В оригинальном алгоритме для каждого трафика определяются две границы: WhitePosstart – номер тайм-слота, левее которого нельзя сдвигать первую пачку рассматриваемого трафика и WhitePosend – номер тайм-слота, правее которого нельзя сдвигать первую пачку рассматриваемого трафика. WhitePosstart

устанавливается на тайм-слот 0, а WhitePos_{end} – на тайм-слот, рассчитанный по формуле

$$\text{WhitePos}_{\text{end}} = \text{TimeSlot}(\text{Period}),$$

где Period – период генерации пакетов, TimeSlot(Period) – номер тайм-слота, время начала которого меньше значения в скобках, а время окончания – больше значения в скобках. Период генерации пакетов трафика может быть задан пользователем с помощью SANDS.

В измененном алгоритме WhitePos_{start} первой пачки устанавливается на первый разрешенный для данного трафика тайм-слот. Первым разрешенным для трафика тайм-слотом считается первый разрешенный тайм-слот для узла-отправителя данного трафика из таблицы расписания.

Для расчета WhitePos_{end} из строки таблицы расписания, выделенной для узла-отправителя текущего трафика, считывается первая позиция разрешенных тайм-слотов. Позицией пользовательской таблицы расписания считаются идущие подряд разрешенные тайм-слоты.

В случае, если рассчитанное значение WhitePos_{end} больше, чем последний тайм-слот считанной позиции, WhitePos_{end} устанавливается на последний тайм-слот считанной позиции, иначе остается прежним.

- Также, как и в проверке возможности построения расписания для бесконфликтных трафиков в выделенных тайм-слотах, в текущей проверке при смещении позиций пачек учитываются разрешенные и неразрешенные тайм-слоты.
- В случае, если с заданными границами, с учетом разрешенных и неразрешенных тайм-слотов, а также с учетом новой начальной расстановки не удалось разрешить конфликты, то проверка завершается.

В случае, если одна из проверок корректности таблицы расписания не будет пройдена, приложение завершит проверку и сгенерирует всплывающее окно, представленное на Рисунок 22.

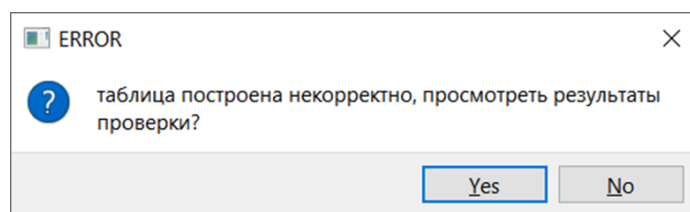


Рисунок 22 – Всплывающее окно, генерируемое в результате обнаружения ошибки при проверке корректности таблицы расписания

При нажатии в окне кнопки «Yes», открывается окно «Результаты проверки таблицы расписания СТП-ИСС-14», содержащее сообщение с текстом ошибки. Полный перечень

возможных сообщений представлен в таблице. Также данное окно может содержать списки трафиков, узлов и коммутаторов, в которых данная ошибка была обнаружена.

Если все проверки пройдены, значит расписание, заданное пользователем корректно. Приложение сгенерирует всплывающее окно, представленное на Рисунок 23.

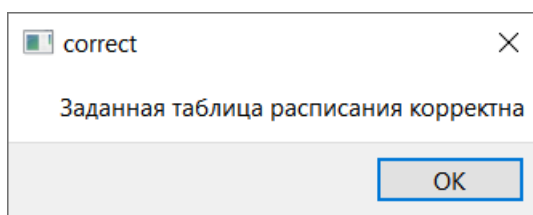


Рисунок 23 – Всплывающее окно, генерируемое в результате успешно пройденной проверки таблицы расписания.

3.5. Всплывающие окна приложения для проверки корректности таблицы расписания

В Таблица 1 содержится перечень сообщений, выдаваемых пользователю во всплывающих окнах. Данные сообщения выдаются окном «Проверка корректности таблицы расписания СТП-ИСС-14».

Таблица 1 – Всплывающие окна

<i>Текст сообщения</i>	<i>Вид всплывающего окна</i>	<i>Когда выдается</i>	<i>Действия пользователя</i>
Параметры не заполнены. Введите все параметры	Предупреждение	При нажатии кнопки «Сохранить параметры»	Пользователь должен заполнить все поля, предназначенные для задания параметров таблицы.
Длина эпохи должна лежать в диапазоне от 0 мс до 1024 мс	Предупреждение	При вводе значения в поле «Длина эпохи»	Пользователь должен задать параметр «Длина эпохи» в диапазоне от 0 мс до 1024 мс
Шаг между тайм-слотами должен лежать в диапазоне 0 мс до 1 мс	Предупреждение	При вводе значения в поле «Шаг между тайм-слотами»	Пользователь должен задать параметр «Шаг между тайм-слотами» в диапазоне от 0 мс до 1 мс

<i>Текст сообщения</i>	<i>Вид всплывающего окна</i>	<i>Когда выдается</i>	<i>Действия пользователя</i>
Загрузка каналов возможна от 1% до 100%	Предупреждение	При вводе значения в поле «Загрузка каналов»	Пользователь должен задать параметр «Загрузка каналов» в диапазоне от 1% до 100%
Количество тайм-слотов должно быть в диапазоне от 2 до 256	Предупреждение	При вводе значения в поле «Количество тайм-слотов»	Пользователь должен задать параметр «Количество тайм-слотов» в диапазоне от 2 до 256
Строка не заполнена	Ошибка	После нажатия на кнопку «Сохранить таблицу»	Пользователь должен заполнить таблицу расписания так, чтобы для каждого узла был разрешен хотя бы один тайм-слот
Минимальное число узлов - 2	Предупреждение	При нажатии кнопки «Сохранить параметры»	Таблицу расписания можно построить только для сети с двумя или более узлами. Пользователь должен изменить текущую схему или выбрать другую
	Предупреждение	При нажатии кнопки «Сохранить параметры»	Таблицу расписания можно построить только при условии, что в сети есть два или более трафиков СТП-ИСС-14 с включенным качеством сервиса «Планирование». Пользователь должен изменить текущую сеть или выбрать другую,

<i>Текст сообщения</i>	<i>Вид всплывающего окна</i>	<i>Когда выдается</i>	<i>Действия пользователя</i>
			соответствующую требованию.
Сеть построена некорректно. Просмотреть результаты проверки?	Ошибка	После нажатия на кнопку «Проверить»	Пользователь может просмотреть результаты проверки сети. Для этого необходимо нажать кнопку «Yes». При нажатии кнопки «No» всплывающее окно закрывается.
Для заданной сети невозможно построить корректное расписание. Просмотреть результаты проверки?	Ошибка	После нажатия на кнопку «Проверить»	Пользователь может просмотреть результаты проверки сети. Для этого необходимо нажать кнопку «Yes». При нажатии кнопки «No» всплывающее окно закрывается.
Таблица построена некорректно. Просмотреть результаты проверки?	Ошибка	После нажатия на кнопку «Проверить»	Пользователь может просмотреть результаты проверки таблицы. Для этого необходимо нажать кнопку «Yes». При нажатии кнопки «No» всплывающее окно закрывается.
Заданная таблица расписания корректна	Оповещение	После нажатия на кнопку «Проверить»	Пользователь должен закрыть окно сообщения нажатием кнопки «ОК». Пользователь может закончить работу программы нажатием кнопки «Выход» или приступить к

<i>Текст сообщения</i>	<i>Вид всплывающего окна</i>	<i>Когда выдается</i>	<i>Действия пользователя</i>
			заполнению новых параметров и таблицы для проведения новых проверок

3.6. Результаты проверок

В результате проведения всех проверок, приложение выдает пользователю информацию о корректности проверяемой таблицы в виде всплывающего окна. Описание всплывающих окон представлено в таблице. В случае, если построенная таблица не корректна, пользователь может перейти в окно «Результаты проверки корректности таблицы расписания СТП-ИСС-14». Примеры сообщений, выводимых в данном окне, представлены в Таблица 2.

Таблица 2 – Результаты проверок

<i>Текст сообщения</i>	<i>Группа проверок</i>	<i>Действия пользователя</i>
В топологии сети отсутствуют коммутаторы	Проверки корректности сети	Пользователь должен добавить в сеть коммутаторы
Для трафика(-ов) тип транспортного протокола на узле-отправителе и узле-получателя отличаются	Проверки корректности сети	Данное сообщение сопровождается списком трафиков. Пользователь должен внести изменения в трафики из списка.
Задана некорректная настройка коммутатора	Проверки корректности сети	Данное сообщение сопровождается списком коммутаторов. Пользователь должен внести изменения в коммутаторы из списка.
Мин. время передачи пакетов для трафика(-ов) не укладывается в макс. допустимое. Рекомендуется увеличить макс. допустимое время, уменьшить размер	Проверки возможности построения расписания для заданной сети	Данное сообщение сопровождается списком трафиков. Пользователь должен внести изменения в трафики из списка.

<i>Текст сообщения</i>	<i>Группа проверок</i>	<i>Действия пользователя</i>
пакета, либо изменить маршруты		
Пропускной способности сети недостаточно для передачи пакетов всех трафиков за эпоху. Рекомендуется уменьшить размеры пакетов, увеличить периоды отправки пакетов, либо изменить маршруты	Проверки возможности построения расписания для заданной сети	Данное сообщение сопровождается списком трафиков. Пользователь должен внести изменения в трафики из списка. Также пользователь может изменить длину эпохи.
Минимальное время доставки пакетов превышает период трафика(-ов). Рекомендуется увеличить периоды отправки пакетов, уменьшить размеры пакетов, изменить маршруты	Проверки возможности построения расписания для заданной сети	Данное сообщение сопровождается списком трафиков. Пользователь должен внести изменения в трафики из списка.
Для трафиков невозможно разрешить конфликты. Рекомендуется увеличить макс. допустимое время, уменьшить размеры пакетов, изменить маршруты	Проверки возможности построения расписания для заданной сети	Данное сообщение сопровождается списком трафиков. Пользователь должен внести изменения в трафики из списка.
Таблица расписания некорректна. Для узла(-ов) необходимо изменить расписание	Проверки корректности таблицы расписания	Данное сообщение сопровождается списком узлов. Пользователь должен внести изменения в заданную таблицу расписания.
Время передачи трафика(-ов) превышает допустимое. Рекомендуется изменить время генерации или увеличить максимально допустимую задержку передачи пакета	Проверки корректности таблицы расписания	Данное сообщение сопровождается списком трафиков. Пользователь должен внести изменения в заданную таблицу расписания.

<i>Текст сообщения</i>	<i>Группа проверок</i>	<i>Действия пользователя</i>
Трафик(-и) невозможно передать за выделенное время. Рекомендуется увеличить количество разрешенных тайм-слотов	Проверки корректности таблицы расписания	Данное сообщение сопровождается списком трафиков. Пользователь должен внести изменения в заданную таблицу расписания.

ВЫВОДЫ ПО РАЗДЕЛУ 3

Данный раздел посвящен описанию разработанного приложения для проверки корректности таблицы расписания для протокола СТП-ИСС-14. В ходе реализации данного приложения был разработан метод проверки таблицы расписания, включающий в себя ряд проверок, основанных на новом методе планирования канальных ресурсов. Разработанное приложение позволяет пользователю провести проверку корректности различных таблиц расписания для сетей, спроектированных в программном комплексе SANDS. Приложение взаимодействует с SANDS посредством xml-файла, генерируемого программным комплексом. Приведенное взаимодействие разработанного приложения с SANDS позволит в дальнейшем интегрировать приложение в программный комплекс, тем самым дополнить компонент 3, сделав его полностью функциональным и отвечающим требованиям и запросам пользователей.

4. ПРОВЕРКА КОРРЕКТНОСТИ ТАБЛИЦ РАСПИСАНИЯ

В данной главе рассмотрены примеры работы приложения для проверки корректности таблицы расписания. Будут рассмотрены примеры проверки корректных и некорректных таблиц расписания. В данных примерах описаны различные ситуации, учитывающие случайные ошибки пользователя, ошибки при проектировании сети, а также ошибки при построении таблиц расписания. В данной главе будут рассмотрены примеры построения таблиц расписания для двух сетей, описанных в разделах 4.1 и 4.2.

4.1. Проверка таблиц расписания для сети с конфликтными трафиками

4.1.1. Входная топология сети с конфликтными трафиками

С помощью программного комплекса SANDS была спроектирована сеть, состоящая из 5 узлов и 3 коммутаторов. Данная сеть представлена на Рисунок 24.

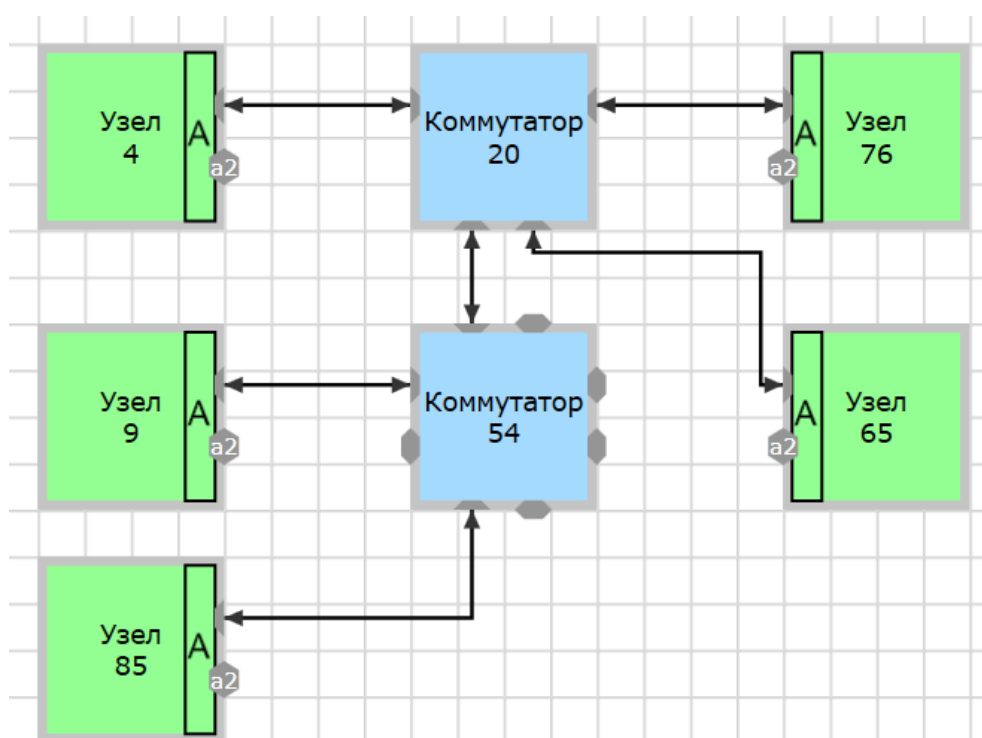


Рисунок 24 – Топология сети с конфликтными трафиками

Таблица 3 – Параметры трафиков в сети с конфликтными трафиками

ID трафика	Узел-отправитель	Узел-получатель	Длина поля данных	Число сообщений	Период отправки, мс	Макс. допустимая задержка, мс	Качество сервиса	Тип пакета	Планирование
26	9	65	1000	10	10	2	Негарантированная доставка данных	Срочное сообщение	Включено
27	4	65	1000	60	5	2	Негарантированная доставка данных	Срочное сообщение	Включено
28	85	76	1000	40	5	2	Негарантированная доставка данных	Срочное сообщение	Включено
29	9	76	100	10	5	5	Негарантированная доставка данных	Обычное сообщение	Включено

В сети осуществляется передача 4 трафиков, параметры которых представлены в Таблица 3. Трафики, передаваемые с узлов 4 и 9, при передаче по сети создают конфликты.

На Рисунок 25 стрелками отмечено направление передачи трафиков в сети. Начало стрелки указывает на узел-отправитель, а ее конец – на узел-получатель трафика. В Таблица 4 представлены маршруты трафиков.

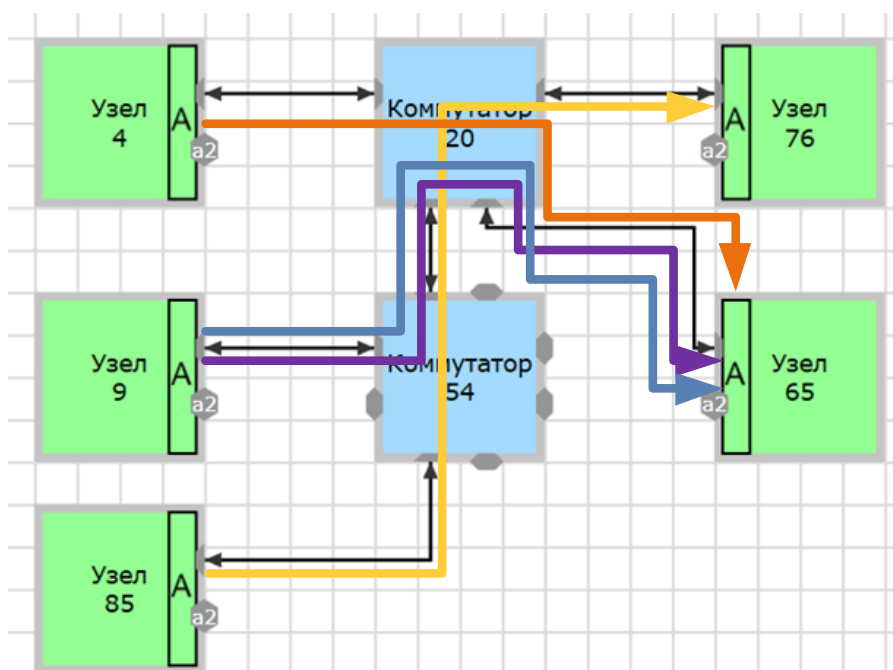






Рисунок 25 – Направления передачи трафиков в сети с конфликтными трафиками

Таблица 4 – Маршруты трафиков в сети с конфликтными трафиками

Номер трафика	Узел-отправитель	Узел-получатель	Цвет указателя
26	Узел 9	Узел 65	
27	Узел 4	Узел 65	
28	Узел 85	Узел 76	
29	Узел 9	Узел 65	

4.1.2. Сгенерированное расписание

С помощью Компонента 3 SANDS была сгенерирована таблица расписания для спроектированной сети. Данная таблица расписания представлена на Рисунок 26. Данное расписание было сгенерировано для следующих параметров:

- Количество тайм-слотов: 10;
- Длительность эпохи: 10 мс;
- Шаг времени между тайм-слотами: 0 мс;
- Максимальная загрузка каналов: 100%;
- Запас размера пакета: 0 байт.

ID узлов	0	1	2	3	4	5	6	7	8	9
4 Узел 4										
9 Узел 9										
85 Узел 85										
Начало временного интервала, мкс	0	1000	2000	3000	4000	5000	6000	7000	8000	9000

Рисунок 26 – Таблица расписания для сети с конфликтными трафиками, сгенерированная Компонентом 3

4.1.3. Примеры проверки таблиц расписания для сети с конфликтными трафиками

После построения сети при помощи SANDS будут проведены проверки различных таблиц расписания для данной сети.

4.1.3.1. Пример проверки таблицы расписания, сгенерированной Компонентом 3 для сети с конфликтными трафиками

В данном примере будет проверена таблица расписания, сгенерированная Компонентом 3 SANDS. После проведения проверки ожидается результат в виде сообщения-оповещения о корректно составленной таблице расписания.

На Рисунок 27 представлен результат проверки таблицы расписания. Ожидаемый результат не достигнут. В окне «Результаты проверки таблицы расписания СТП-ИСС-14», представленной на Рисунок 28, содержится текст ошибки, рекомендации пользователю, а

также список трафиков, подлежащих корректировке. Данный результат означает, что для трафика 26, отправляемого с узла 9, время передачи превышает допустимое.

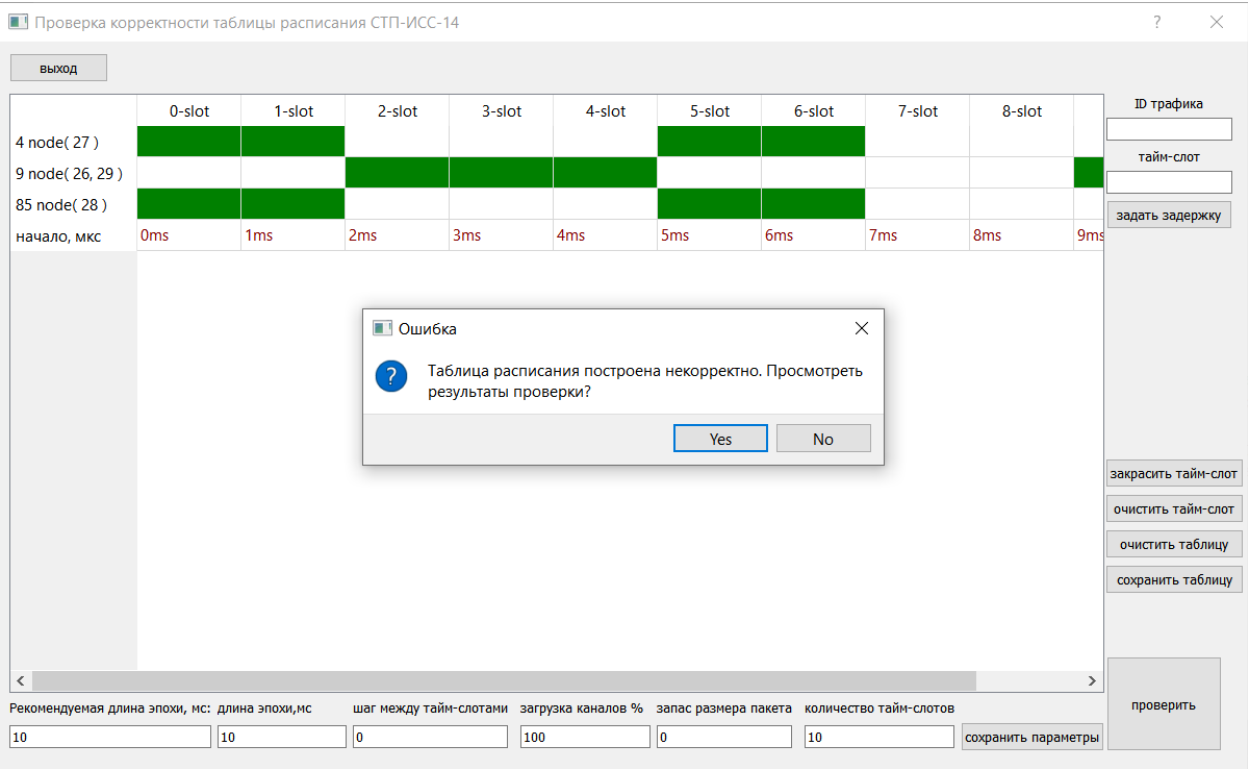


Рисунок 27 – Проверка таблицы расписания, сгенерированной Компонентом 3 для сети с конфликтными трафиками

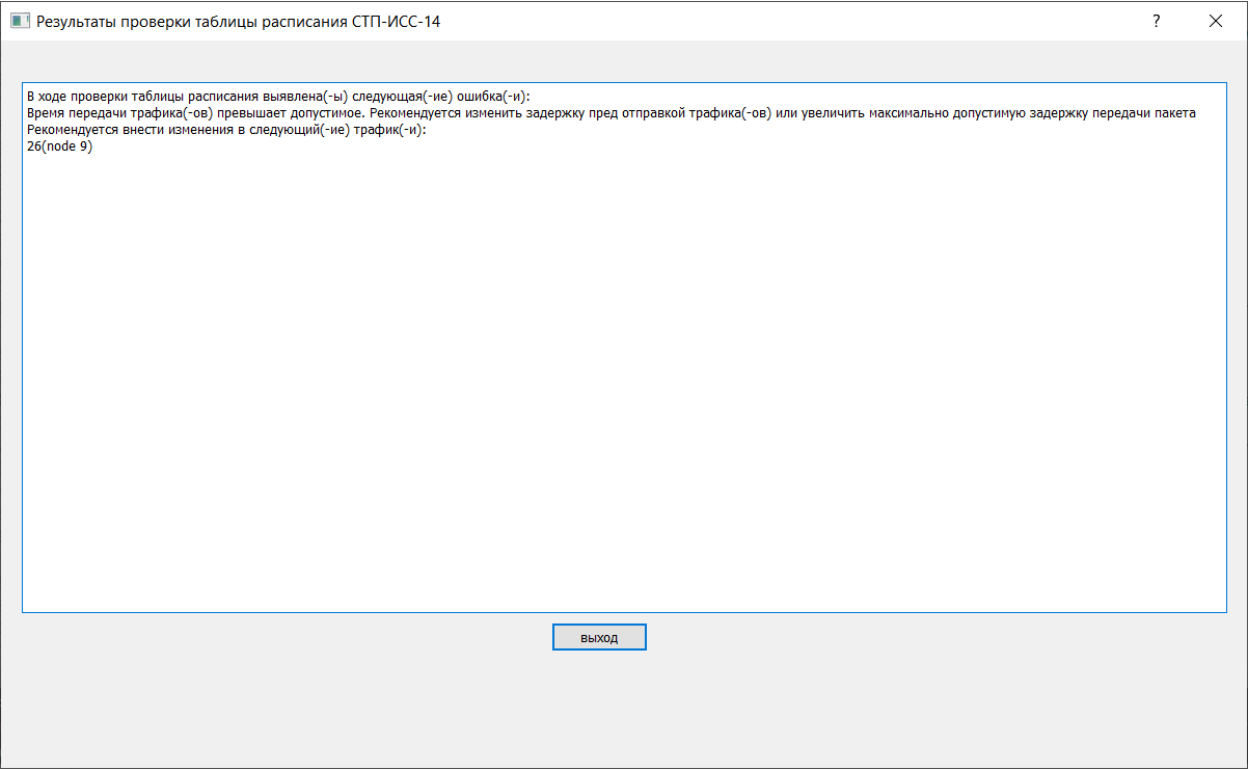


Рисунок 28 – Результат проверки таблицы расписания. Ошибка допустимой задержки трафика

Данный результат объясняется тем, что Компонент 3 при генерации таблицы расписания изменяет задержку перед отправкой трафиков, а в приложении для проверки корректности таблицы расписания изначально все задержки установлены на 0 мс. На Рисунок 28, в списке трафиков, подлежащих корректировке, указан один трафик – трафик 26, отправляемый узлом 9. Для получения корректной таблицы расписания необходимо изменить задержку перед отправкой для данного трафика.

По таблице расписания, представленной на Рисунок 27, можно однозначно установить, что первый трафик, отправляемый с узла 9, начинает передачу данных во втором тайм-слоте. Поэтому установим задержку для трафика 26 на тайм-слот 2 и проверим скорректированную таблицу. Результат проверки скорректированной таблицы представлен на Рисунок 29.

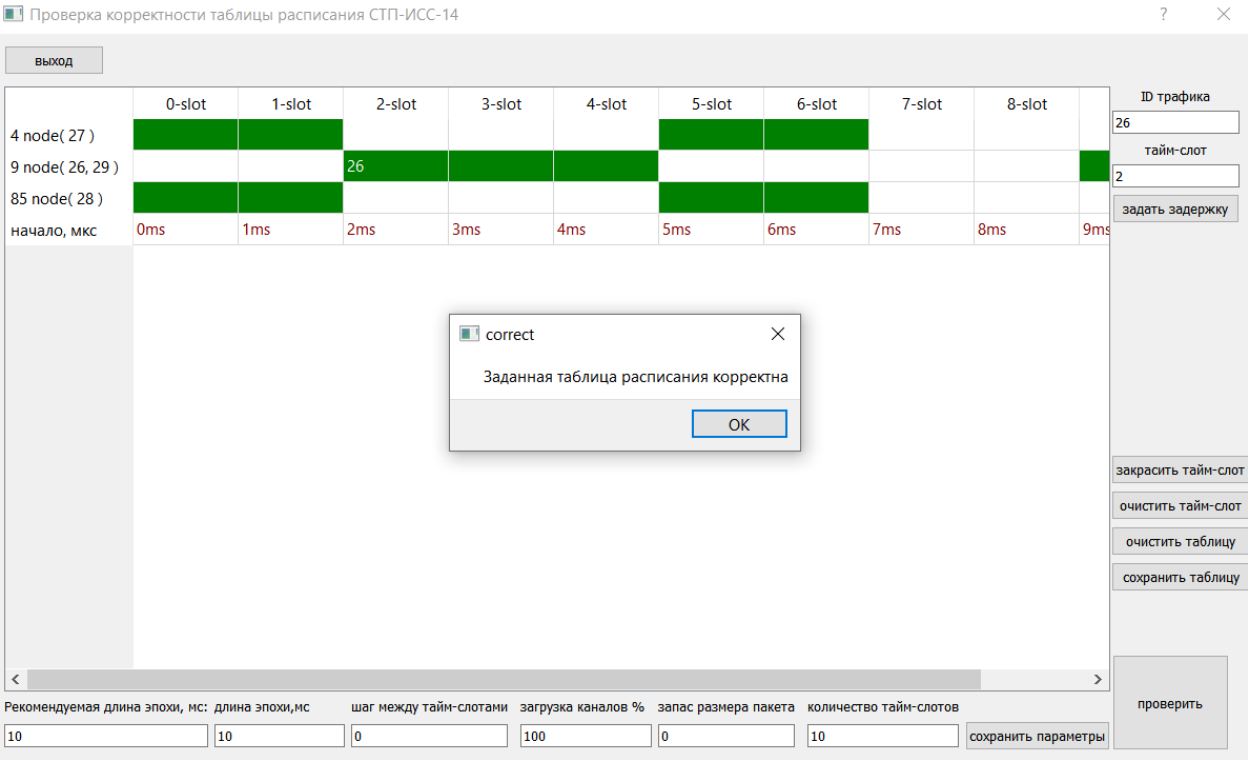


Рисунок 29 – Проверка скорректированной таблицы расписания

По результатам проверки скорректированная таблица расписания корректна. Результат в виде сообщения-оповещения о корректно составленной таблице расписания достигнут.

4.1.3.2. Пример проверки таблицы расписания с неиспользуемым разрешенным тайм-слотом

В данном примере в таблицу расписания, сгенерированную Компонентом 3, будет добавлен один дополнительный разрешенный тайм-слот для узла 4 – тайм-слот 7. После

проведения проверки ожидается результат в виде сообщения-оповещения о корректно составленной таблице расписания.

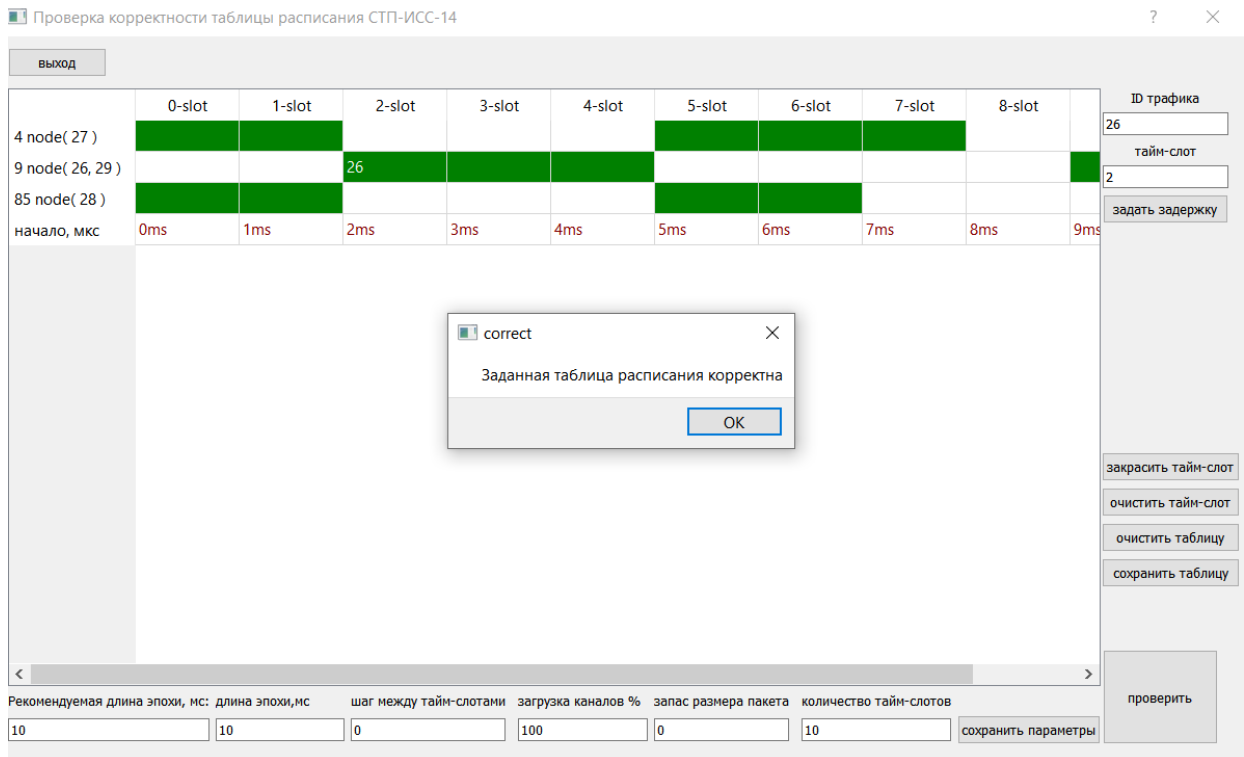


Рисунок 30 – Проверка таблицы расписания с неиспользуемым разрешенным тайм-слотом

На Рисунок 30 представлен результат проверки таблицы расписания. По результатам проверки таблица расписания с дополнительным разрешенным тайм-слотом корректна. Ожидаемый результат в виде сообщения-оповещения о корректно составленной таблице расписания достигнут.

Корректность таблицы расписания объясняется тем, что трафик 27, передаваемый с узла 4, для которого выделен дополнительный тайм-слот успеет передаться за предыдущие тайм-слоты, а также тем, что данный тайм-слот запрещен для других узлов. Следовательно, при разрешении данного тайм-слота не создастся новых конфликтных ситуаций в сети.

4.1.3.3. Пример проверки таблицы расписания с запретом одно тайм-слота, используемого узлом

В данном примере в таблице расписания, сгенерированной Компонентом 3, для узла 85 будет запрещен один тайм-слот, ранее разрешенный для этого узла – тайм-слот 1. После проведения проверки ожидается результат в виде сообщения-оповещения о некорректно составленной таблице расписания.

На Рисунок 31 представлен результат проверки таблицы расписания. По результатам проверки таблица расписания с запрещенным тайм-слотом некорректна. Ожидаемый результат в виде сообщения-оповещения о некорректно составленной таблице расписания

достигнут. В окне «Результаты проверки таблицы расписания СТП-ИСС-14», представленной на Рисунок 32, содержится текст ошибки, рекомендации пользователю, а также список трафиков, подлежащих корректировке. Согласно полученному результату, для трафика 28, отправляемого с узла 85, выделено недостаточно тайм-слотов.

Некорректность таблицы расписания объясняется тем, что для передачи трафика 28, отправляемого с узла 85, выделено времени меньше, чем необходимо для его полной передачи по сети.

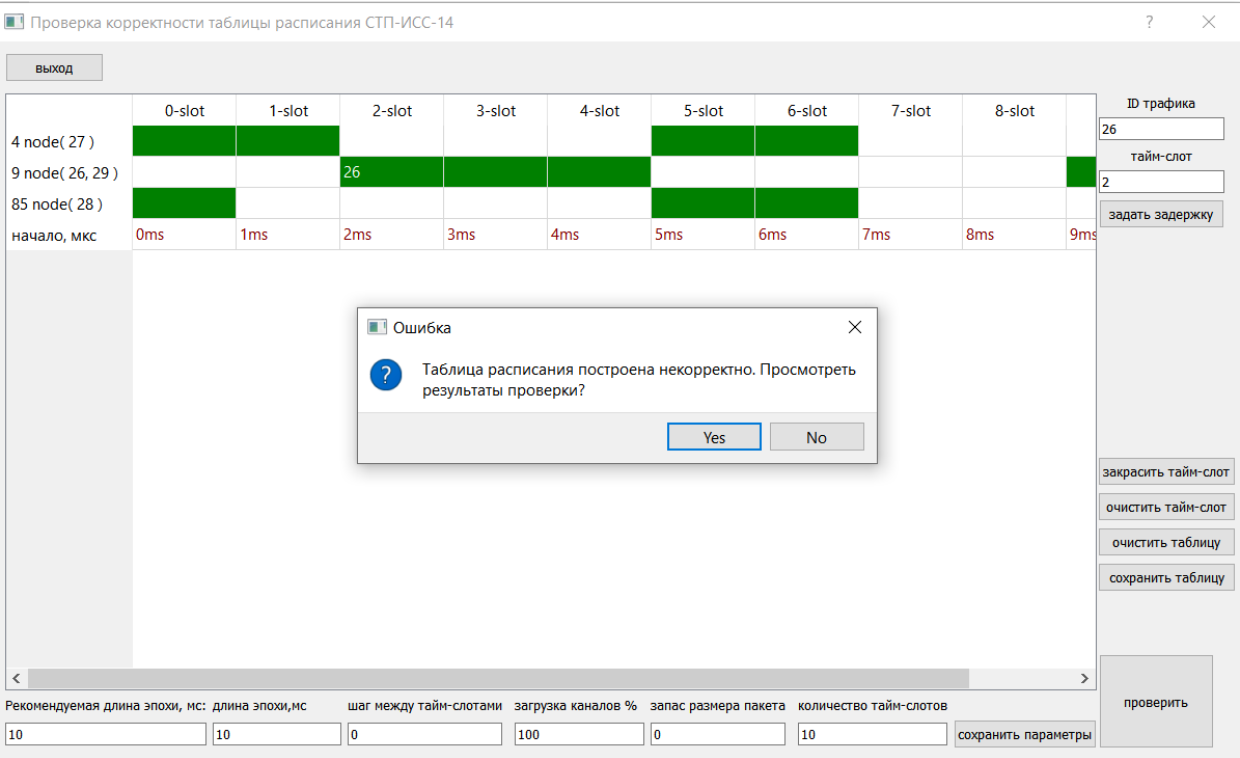


Рисунок 31 – Проверка таблицы расписания с запрещенным используемым тайм-слотом

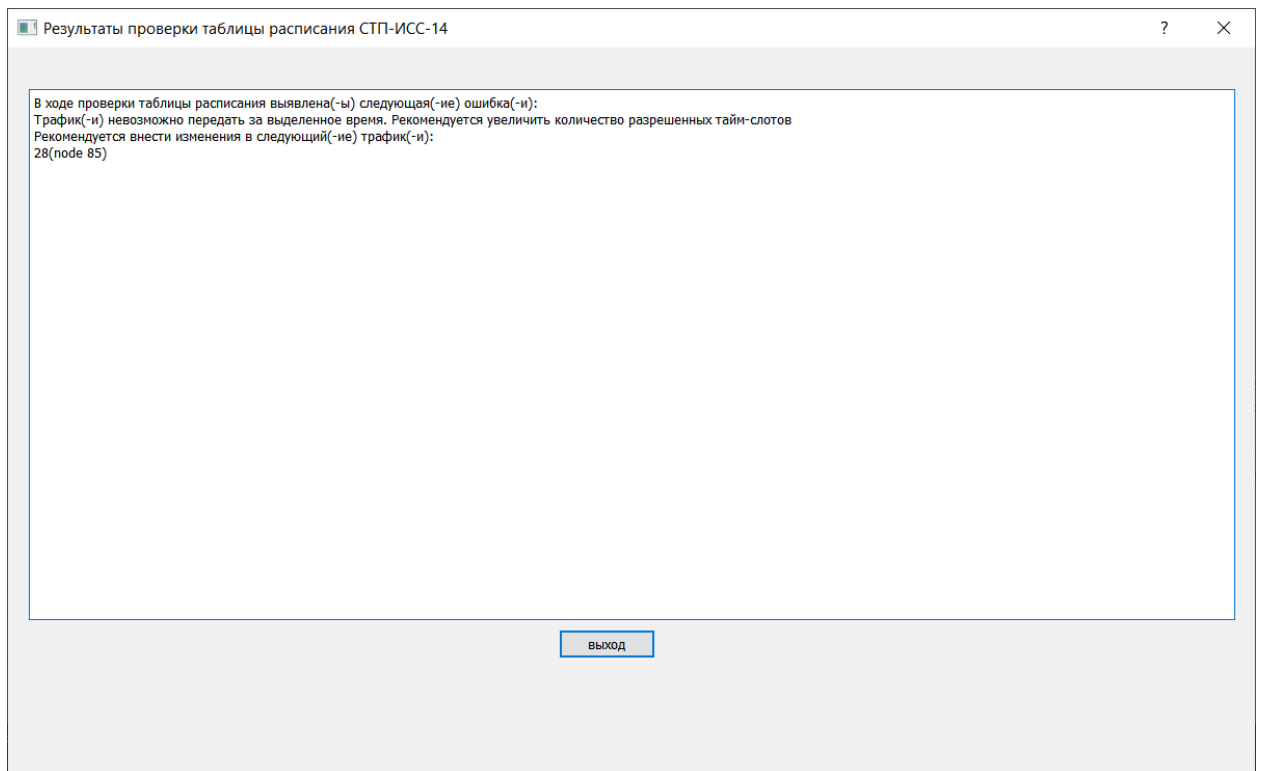


Рисунок 32 – Результат проверки таблицы расписания. Запрещенный тайм-слот

4.1.3.4. Пример проверки таблицы расписания со сдвигом влево расписания для одного узла

В данном примере в таблице расписания, сгенерированной Компонентом 3, для узла 9 расписание будет сдвинуто на два тайм-слота влево: для передачи будут разрешены тайм-слоты 0, 1, 2 и 7. После проведения проверки ожидается результат в виде сообщения-оповещения о некорректно составленной таблице расписания.

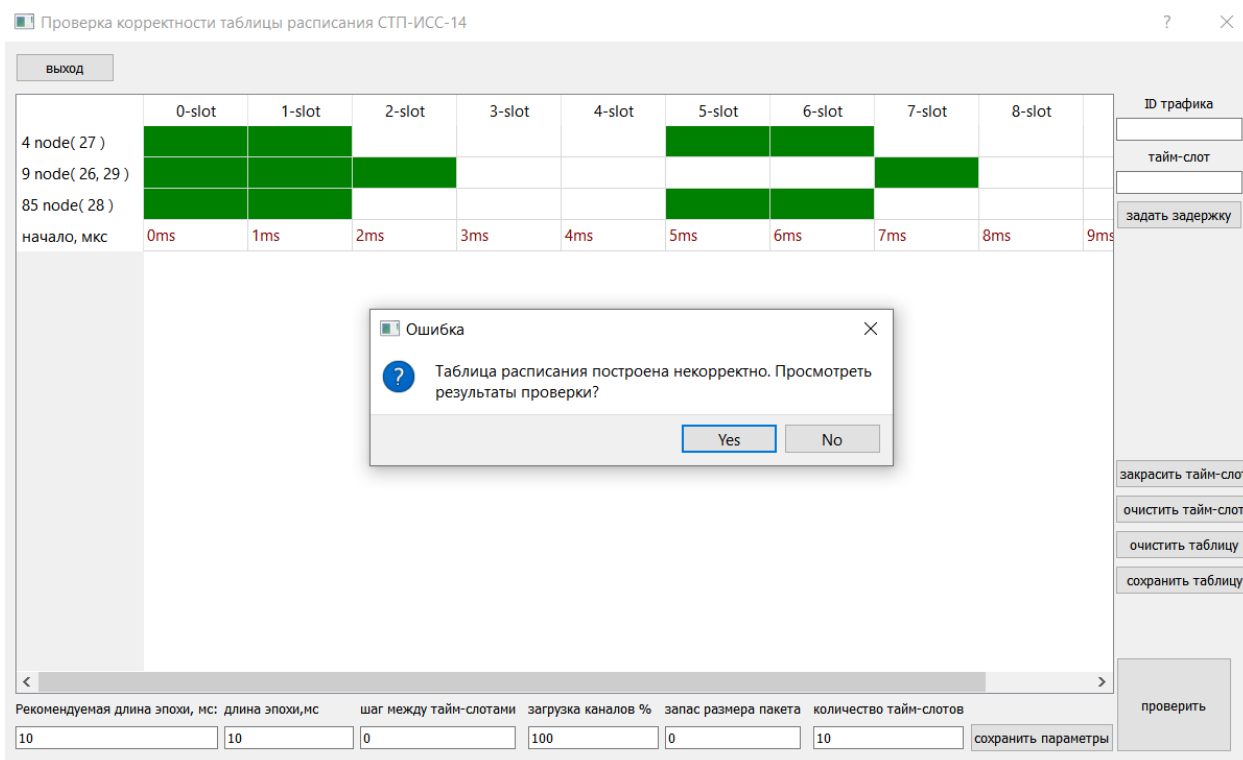


Рисунок 33 – Проверка таблицы расписания со сдвигом влево расписания для одного узла

На Рисунок 33 представлен результат проверки таблицы расписания. По результатам проверки таблица расписания со сдвигом расписания для узла 9 некорректна. Ожидаемый результат в виде сообщения-оповещения о некорректно составленной таблице расписания достигнут. В окне «Результаты проверки таблицы расписания СТП-ИСС-14», представленной на Рисунок 34, содержится текст ошибки, рекомендации пользователю, а также список узлов, подлежащих корректировке.

Некорректность таблицы расписания объясняется тем, что при сдвиге расписания для узла 9 трафики, отправляемые с узлов 4 и 9, передаются в одни и те же тайм-слоты (0 и 1), создавая конфликты. При этом выделенных тайм-слотов не хватит для того, чтобы разрешить конфликты и передать конфликтующие трафики.

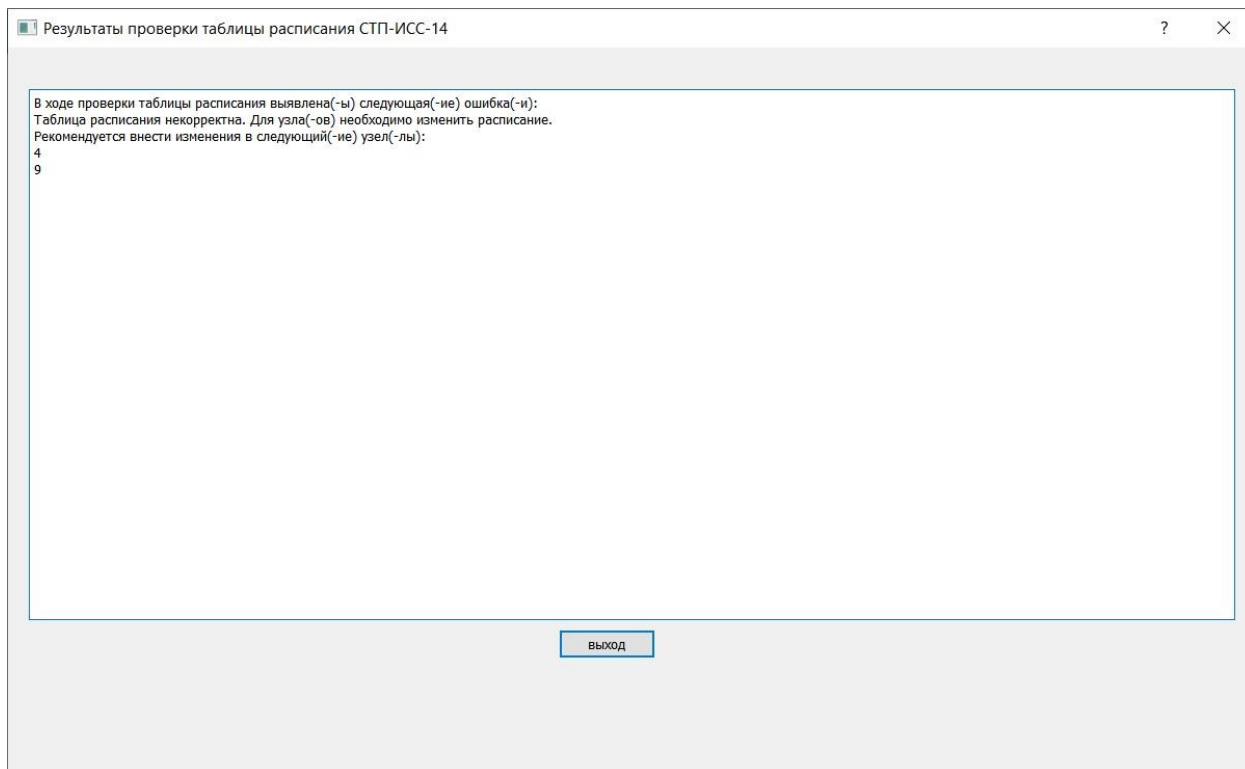


Рисунок 34 – Результат проверки таблицы расписания. Сдвиг расписания для одного узла

4.1.3.5. Пример проверки таблицы расписания со всеми разрешенными тайм-слотами

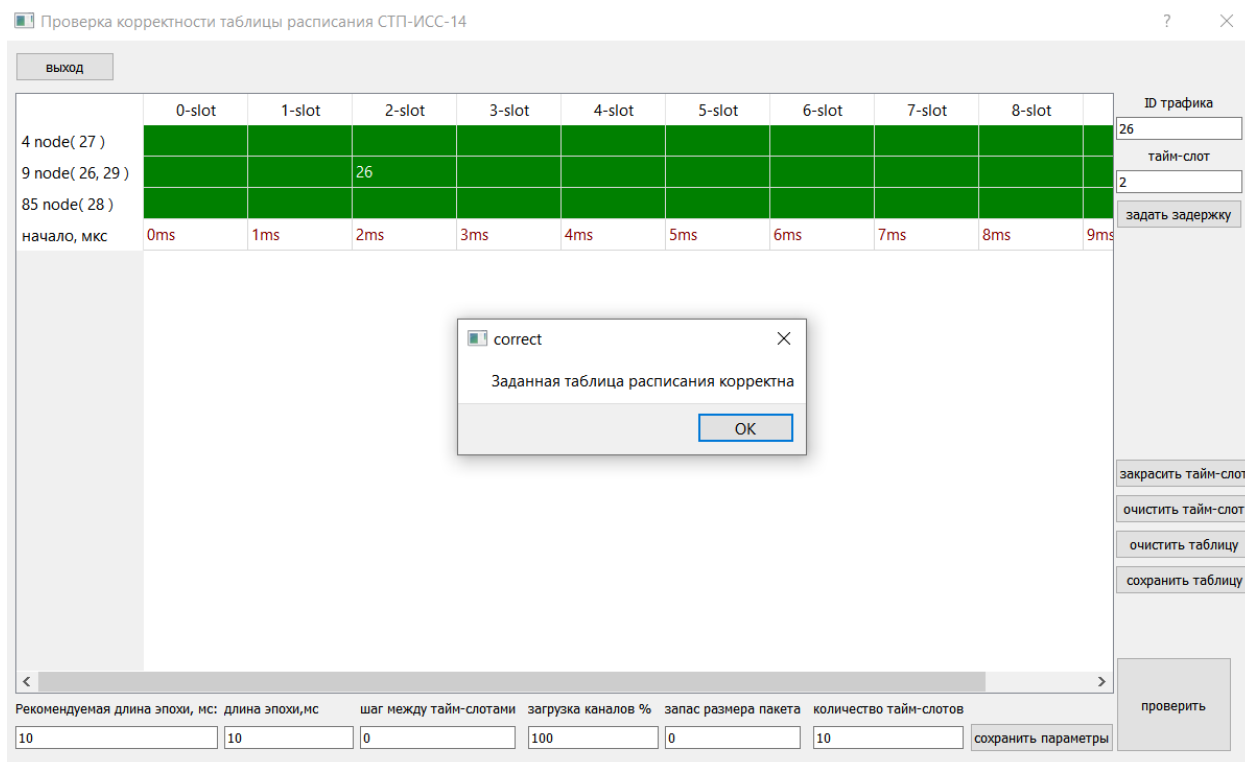


Рисунок 35 – Проверка таблицы расписания со всеми разрешенными тайм-слотами

В данном примере в таблице расписания для передачи разрешаются все тайм-слоты в таблице. После проведения проверки ожидается результат в виде сообщения-оповещения о корректно составленной таблице расписания.

На Рисунок 35 представлен результат проверки таблицы расписания. По результатам проверки таблица расписания со всеми разрешенными тайм-слотами является корректной. Ожидаемый результат в виде сообщения-оповещения о корректно составленной таблице расписания достигнут.

Корректность таблицы расписания объясняется тем, что для каждого узла были разрешены дополнительные тайм-слоты, в которые узлы не будут передавать данные. Новые конфликты не будут созданы, а времени, выделенного для передачи данных хватит для передачи всех трафиков в сети. Трафики будут отправляться в соответствии с расписанием, сгенерированным Компонентом 3.

4.1.3.6. Пример проверки таблицы расписания с увеличенным числом тайм-слотов

В данном примере будет проверена таблица расписания, сгенерированная Компонентом 3 SANDS. Однако, число тайм-слотов в данной таблице будет увеличено вдвое. Следовательно, и число разрешенных тайм-слотов для каждого узла будет увеличено в два раза. После проведения проверки ожидается результат в виде сообщения-оповещения о корректно составленной таблице расписания.

Полная таблица расписания, которая будет проверена в данном примере, представлена на Рисунок 36.

	0-slot	1-slot	2-slot	3-slot	4-slot	5-slot	6-slot	7-slot	8-slot	9-slot	10-slot	11-slot	12-slot	13-slot	14-slot	15-slot	16-slot	17-slot	18-slot	19-slot
4 node (27)																				
9 node (26, 29)																				
85 node (28)																				

Рисунок 36 – Таблица расписания для 20 тайм-слотов

На Рисунок 37 представлен результат проверки таблицы расписания. По результатам проверки таблица расписания со всеми разрешенными тайм-слотами является корректной. Ожидаемый результат в виде сообщения-оповещения о корректно составленной таблице расписания достигнут.

Корректность таблицы расписания объясняется тем, что данная таблица является масштабированной копией таблицы, построенной Компонентом 3 SANDS. При увеличении вдвое параметра «число тайм-слотов» размер тайм-слота пропорционально уменьшается вдвое, следовательно, для передачи всех пакетов данных требуется в два раза больше тайм-слотов. Проверяемая таблица соответствует этому требованию.

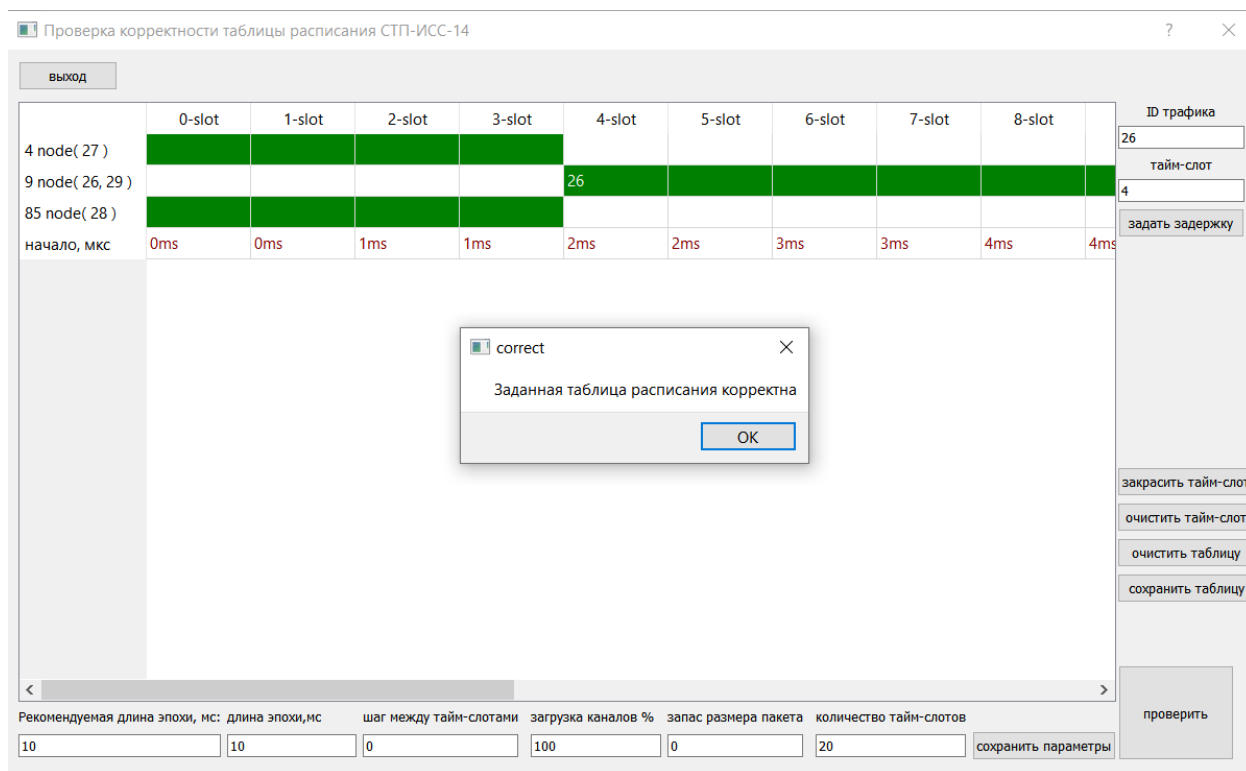


Рисунок 37 – Проверка таблицы расписания для 20 тайм-слотов

4.2. Проверка таблиц расписания для сети с неконфликтными трафиками

4.2.1. Входная топология сети с неконфликтными трафиками

С помощью SANDS была спроектирована сеть, состоящая из 3 узлов и 1 коммутатора. Спроектированная сеть представлена на Рисунок 38.

В сети осуществляется передача 2 трафиков, параметры которых представлены в Таблица 5. Трафики, передаваемые с узлов 283 и 288, при передаче по сети не создают конфликты.

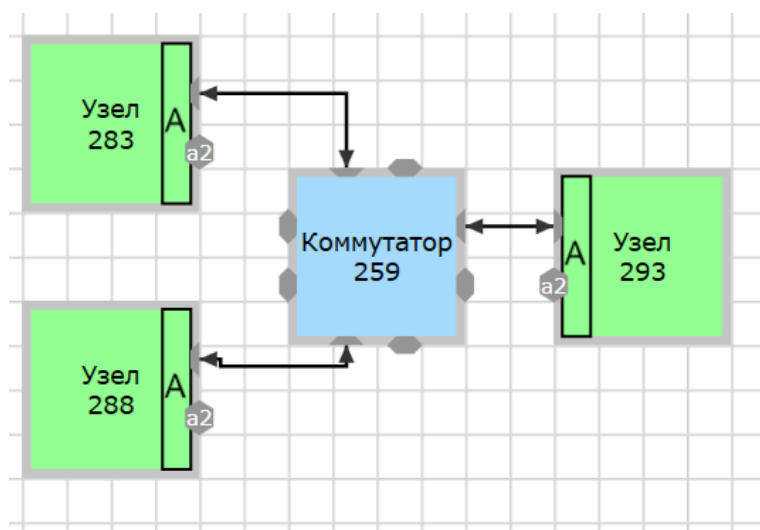


Рисунок 38 – Топология сети с неконфликтными трафиками

Таблица 5 – Параметры трафиков в сети с конфликтными трафиками

ID трафика	Узел-отправитель	Узел-получатель	Длина поля данных	Число сообщений	Период отправки, мс	Макс. допустимая задержка, мс	Качество сервиса	Тип пакета	Планирование
3	283	293	1000	60	4	3	Негарантированная доставка данных	Срочное сообщение	Включено
4	288	293	500	60	4	8	Негарантированная доставка данных	Срочное сообщение	Включено

На Рисунок 39 стрелками отмечено направление передачи трафиков в сети. Начало стрелки указывает на узел-отправитель, а ее конец – на узел-получатель трафика. В Таблица 6 представлены маршруты трафиков.

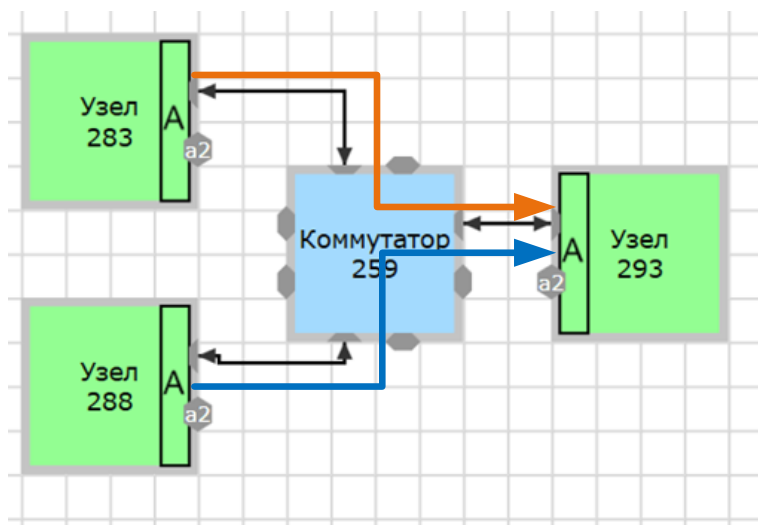




Рисунок 39 – Направления передачи трафиков в сети с неконфликтными трафиками

Таблица 6 – Маршруты трафиков в сети с неконфликтными трафиками

Номер траффика	Узел-отправитель	Узел-получатель	Цвет указателя
3	Узел 283	Узел 293	
4	Узел 288	Узел 293	

4.2.2. Сгенерированное расписание

С помощью Компонента 3 SANDS была сгенерирована таблица расписания для спроектированной сети. Данная таблица расписания представлена на Рисунок 40. Данное расписание было сгенерировано для следующих параметров:

- Количество тайм-слотов: 10;
- Длительность эпохи: 4 мс;
- Шаг времени между тайм-слотами: 0 мс;
- Максимальная загрузка каналов: 100%;
- Запас размера пакета: 0 байт.

ИД узлов	0	1	2	3	4	5	6	7	8	9
283 Узел 283										
288 Узел 288										
Начало временного интервала, мкс	0	400	800	1200	1600	2000	2400	2800	3200	3600

Рисунок 40 – Таблица расписания для сети с неконфликтными трафиками, сгенерированная Компонентом 3

4.2.3. Примеры проверки таблиц расписания для сети с неконфликтными трафиками

После построения сети при помощи SANDS будут проведены проверки различных таблиц расписания для данной сети.

4.2.3.1. Пример проверки таблицы расписания, сгенерированной Компонентом 3 для сети с неконфликтными трафиками

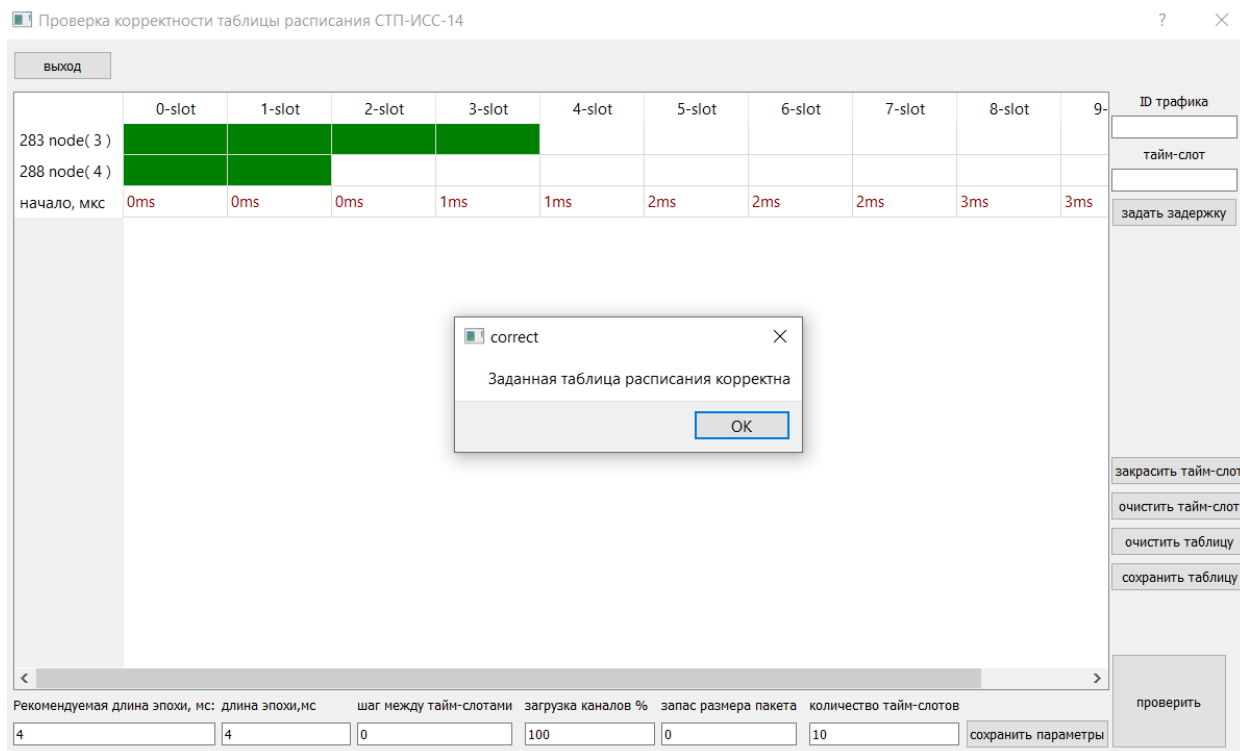


Рисунок 41 – Проверка таблицы расписания, сгенерированной Компонентом 3 для сети с неконфликтными трафиками

В данном примере была проверена таблица расписания, сгенерированная Компонентом 3 SANDS. После проведения проверки ожидается результат в виде сообщения-оповещения о корректно составленной таблице расписания.

На Рисунок 41 представлен результат проверки таблицы расписания. Ожидаемый результат в виде сообщения-оповещения о корректно составленной таблице расписания достигнут.

4.2.3.2. Пример проверки таблицы расписания со сдвигом вправо расписания для одного узла

В данном примере в таблице расписания, сгенерированной Компонентом 3, для узла 288 расписание будет сдвинуто на 7 тайм-слотов вправо: для передачи будут разрешены тайм-слоты 7 и 8. Задержка перед отправкой трафика 4, отправляемого с узла 288 будет установлена на тайм-слот 7. После проведения проверки ожидается результат в виде сообщения-оповещения о корректно составленной таблице расписания.

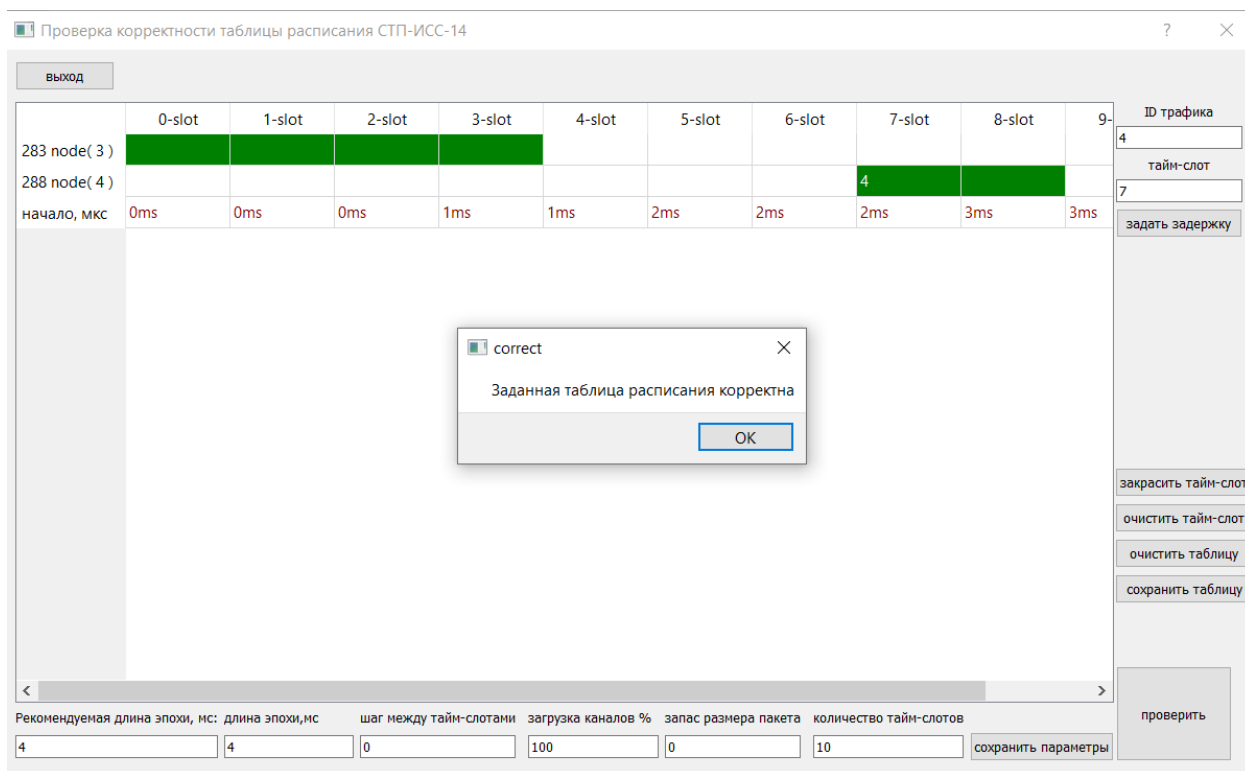


Рисунок 42 – Проверка таблицы расписания со сдвигом вправо расписания для одного узла

На Рисунок 42 представлен результат проверки таблицы расписания. По результатам проверки таблица расписания со сдвигом расписания для узла 9 корректна. Ожидаемый результат в виде сообщения-оповещения о корректно составленной таблице расписания достигнут.

Корректность таблицы расписания объясняется тем, что для полной передачи трафика 4, передаваемого с узла 288, выделено достаточно тайм-слотов. Так как трафики, передаваемые в сети, не конфликтуют, изменение положения разрешенных тайм-слотов в таблице не влияет на ее корректность.

ВЫВОДЫ ПО РАЗДЕЛУ 4

В данном разделе была проведена проверка корректности разработанного приложения посредством проверки заведомо корректных и некорректных входных данных. В ходе проведения данной проверки был сделан вывод о корректности разработанного приложения.

В ходе подтверждения корректности разработанного приложения были проведены проверки корректности различных таблиц расписания. Главным критерием корректности построенной таблицы расписания является предотвращение ситуации, при которой трафикам не хватает тайм-слотов для передачи всех данных или при которой конфликтные трафики осуществляют передачу в один и тот же разрешенный тайм-слот. В случае

обнаружения подобных ситуаций пользователь может увеличить количество разрешенных тайм-слотов, установить задержку, или поменять расположение разрешенных тайм-слотов. Данный раздел охватывает все перечисленные ситуации, а также показывает какие действия может совершить пользователь в зависимости от ситуации. Проверка разработанного приложения подтверждает его корректность.

ЗАКЛЮЧЕНИЕ

В данной выпускной квалификационной работе было разработано приложение для проверки корректности таблиц расписания. Данное приложение – это удобный и полезный инструмент, позволяющий пользователю, спроектировавшему сеть при помощи программного комплекса SANDS проверить корректность таблицы расписания для данной сети. Разработанное приложение может стать частью SANDS, так как взаимодействует с данным программным комплексом посредством входного xml-файла, генерируемого SANDS.

При разработке приложения были учтены различные критерии корректности таблиц, рассмотрены и учтены различные ситуации ввода некорректных входных параметров, а также возможные ошибки при проектировании сети.

При создании приложения был разработан метод проверки корректности таблиц в основу которого лег новый метод планирования, рассмотренный в разделе 2. Данный метод был выбран на основе исследования различных методов и средств управления канальными ресурсами, проведенного в разделе 1.

В разделе 3 было проведено подробное описание разработанного приложения, представлены все окна, текстовые поля, кнопки, а также сообщения, генерируемые приложением при возникновении различных ситуаций. Поля и кнопки обеспечивают удобство взаимодействия пользователя с приложением, сообщения позволяют пользователю не только узнать о возникших ошибках, но и получить рекомендации по исправлению некорректности таблиц или входных данных.

Разработанное приложение позволяет проверить корректность таблиц для протоколов СТП-ИСС-14, одна оно является достаточно гибким и может быть изменено для работы с таблицами других протоколов.

Корректность разработанного приложения подтверждена в разделе 4 посредством ряда проверок.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Таненбаум, Э. Уэзеролл, Д. Компьютерные сети /Э. Таненбаум, Д. Уэзеролл/ 5-е изд. — СПб.: Питер, 2012. 960 с.
2. Олифер, В. Г. Олифер, Н. А. Компьютерные сети /В. Г. Олифер, Н. А. Олифер/ 4-е изд. СПб.: Питер, 2010. 944 с.
3. Пятибратов, А. П. Вычислительные машины, сети и телекоммуникационные системы Учебно-методический комплекс / А. П. Пятибратов, Л. П. Гудыно, А. А. Кириченко. М.: Изд.центр ЕАОИ, 2009. 292 с
4. Шейнин, Ю. Солохина, Т. Петричкович, Я. Технология SpaceWire для параллельных и бортовых распределенных комплексов /Ю. Шейнин, Т. Солохина, Я. Петричкович// Электроника. Наука. Технология. Бизнес, 2006. с. 64-75
5. Кучерявый, Е.Ф. Управление трафиком и качество обслуживания в сети Интернет / Е.Ф. Кучерявый. - СПб.: Наука и Техника, 2004. - 336 с.:
6. Фалеева, М. Чумакова, Н. Сравнительный анализ подходов к реализации качества сервиса «планирование» в коммуникационных протоколах/ М. Фалеева, Н. Чумакова// 73-я международная студенческая научная конференция 2020. В печати
7. Фишман, Е.Б. Анализ обслуживания очередей в сетях с поддержкой «качества обслуживания» (QoS)/ Е.Б. Фишман// Качество Инновации Образование, 2006, №6, стр. 63-71
8. Gabrielyan, E. Hersch, R.D. «Efficient Liquid Schedule Search Strategies for Collective Communications», ReCALL, 2004, pp. 760-766
9. Разживин, Д.Б. Эффективное использование полосы пропускания в сетях с временным мультиплексированием // Сборник докладов научной сессии ГУАП: Сб. докл.: Ч.1 Технические науки, СПб.: - ГУАП, 2012, с. 109-111
10. Коробков И.Л. Метод планирования канальных ресурсов в бортовых сетях SpaceWire с технологией TDMA /// Радиопромышленность. 2019. Т. 29, № 4. С. 44–53
11. Ковалев, В.Н. Орлов, А.В. Управление заданиями в распределенной среде и протокол резервирования ресурсов// Москва, 2002
12. Малявко, А.А. Менжулин, С.А. Суперкомпьютеры и системы. Построение вычислительных кластеров: учебное пособие / А.А. Малявко, С.А. Менжулин// Новосибирск: Изд-во НГТУ, 2018. – 96 с.
13. Moab 9.0.1 Концептуальный обзор /Adaptive Computing EMEA// Перевод: ООО "Модуль- Проекты", 2016.

14. Шейнин, Ю. Е. Оленев, В.Л. Лавровская, И. Я. Дымов, Д. В. Кочура, С. Г. Разработка, анализ и проектирование транспортного протокола СТП-ИСС для бортовых космических сетей SpaceWire / Ю. Е. Шейнин, В.Л. Оленев, И. Я. Лавровская, Д. В. Дымов, С. Г. Кочура// Исследования наукограда, 2016, №1-2, С. 21-30
15. Оленев В. Л., Лавровская И. Я., Курбанов Л. И., Коробков И. Л., Шейнин Ю. Е. Система автоматизированного проектирования бортовых космических сетей //Вопросы радиоэлектроники. 2018. № 8. С. 145–153.
16. Официальный сайт компании АО «Информационные Спутниковые Системы». URL: <https://www.iss-reshetnev.ru>
17. Korobkov, I. L. Chumakova, N. Y. Scheduling-Table's Design for STP-ISS Transport Protocol // XXII международная научная конференция. Волновая электроника и инфокоммуникационные системы, 2019

ПРИЛОЖЕНИЕ А

Публикация по теме выпускной квалификационной работы

Фалеева, М. Чумакова, Н. Сравнительный анализ подходов к реализации качества сервиса «планирование» в коммуникационных протоколах/ М. Фалеева, Н. Чумакова// 73-я международная студенческая научная конференция 2020. В печати

ПРИЛОЖЕНИЕ Б

В данном приложении представлено описание текстовых полей и кнопок окна «Проверка корректности таблицы расписания СТП-ИСС-14».

Таблица Б.1 – Текстовые поля окна «Проверка корректности таблицы расписания СТП-ИСС-14»

Наименование	Область	Назначение	Разрешенные значения
Длина эпохи	№2	Данное поле предназначено для ввода длины эпохи в мс.	От 1 мс до 1024 мс
Шаг между тайм-слотами	№2	Данное поле предназначено для ввода значения шага между тайм-слотами в мс.	От 0 мс до 1 мс
Загрузка каналов	№2	Данное поле предназначено для ввода значения загрузки каналов в процентах	От 1% до 100%
Запас размера пакета	№2	Данное поле предназначено для ввода значения запаса размера пакета в байтах	От 0 байт
Количество тайм-слотов	№2	Данное поле предназначено для ввода значения количества тайм-слотов	От 2 до 256 тайм-слотов
ID трафика	№5	Данное поле предназначено для ввода ID трафика, для которого в будет задана задержка перед отправкой	В данное поле можно вписать ID трафика, присутствующего в текущей таблице расписания
Тайм-слот	№5	Данное поле предназначено для ввода номера тайм-слота, на начало которого будет установлена задержка перед отправкой трафика, ID которого введено в поле «ID трафика»	От 0 до значения поля «Количество тайм-слотов»

Таблица Б.2 – Кнопки окна «проверка корректности таблицы расписания СТП-ИСС-14»

Название	Область	Описание
Сохранить параметры	№3	При нажатии кнопки все параметры, введенные пользователем, сохраняются в памяти приложения. Также, после нажатия данной кнопки в области №2 генерируется пустая таблица расписания.
Закрасить тайм-слот	№4	При нажатии кнопки выбранный тайм-слот окрашивается в темно-зеленый цвет. Такой тайм-слот считается разрешенным для передачи данных по сети.
Очистить тайм-слот	№4	При нажатии кнопки выбранный тайм-слот окрашивается в белый цвет. Такой тайм-слот считается запрещенным для передачи данных по сети.
Очистить таблицу	№4	При нажатии кнопки все тайм-слоты окрашиваются в белый цвет и считаются запрещенными для передачи данных.
Сохранить таблицу	№4	При нажатии кнопки заполненная таблица сохраняется в памяти приложения.
Задать задержку	№5	При нажатии кнопки для трафика, id которого задано в поле «ID трафика» устанавливается задержка перед отправкой на начало тайм-слота, заданного в поле «Тайм-слот». После нажатия на кнопку в заданном тайм-слоте появится цифра, означающая id трафика, для которого была задана задержка
Проверить	№7	После нажатия кнопки начинается проверка заданной пользователем таблицы расписания на корректность
Выход	№8	При нажатии кнопки приложение закрывается

ПРИЛОЖЕНИЕ В

В данном приложении приведен листинг некоторых файлов разработанного приложения для проверки корректности таблицы расписания.

1. Листинг файла «oknotablitsa.h», содержащего описание класса oknotablitsa, основных функций, переменных и констант. Данный класс предназначен для реализации главного окна приложения для проверки корректности таблицы расписания

```
#ifndef OKNOTABLITSA_H
#define OKNOTABLITSA_H
#include <QTableWidgetItem>
#include <QDialog>
#include <QLineEdit>
#include <QDebug>
#include "xmlparser.h"
#include "scheduler.h"
#define MAX_COL 256
#define MAX_ROW 100
using namespace std;
using namespace scheduler_ns;
namespace Ui {
class oknotablitsa;
}

class oknotablitsa : public QDialog
{
    Q_OBJECT

public:
    explicit oknotablitsa(char *argv[],QWidget *parent = nullptr);
    ~oknotablitsa();
    uint mas [MAX_ROW][MAX_COL];
    void fillpos();
    _topologyAndNetwork topology;
    scheduler::epoch epochdata;
    uint epochlenght;//маx длина эпохи в таблице
    uint step; //шаг между врем. интервалами,мс
    uint zagruzka; //маx загрузка каналов %
    uint packetsize; //запас размера пакета в байтах
    uint tochno;//точность вычислений
    uint numbercolumn;
    QList <scheduler::scheduling_table_string> scheduling_table;
    QList <dl> time_slots_start; //ВРЕМЯ НАЧАЛА ТАЙМ-СЛОТОВ
    QList <scheduler::traffic> traffics_list; //СПИСОК ТРАФИКОВ
    QMap <QList <ull>, QList <scheduler::traffic_table_el>>
conflict_traffics_table;//конфликтные трафики
    QList <scheduler::group> traffic_groups;//группы трафиков
    QList <scheduler::traffic*> error_traffics_list;//список некорректных трафиков
    QList <scheduler::node> nodes_list;//список узлов всей сети
```

```

    QList <QList <scheduler::traffic*>> error_conflicts_list; //список некорректных
конфликтов
    QList <scheduler::group*> scheduling_groups; //группы планируемых трафиков
    QList <QList <scheduler::traffic*>> conflicts_list; //список корректных конфликтов
    QList <const _node*> error_nodes; //список некорректных узлов
    QList <const _switch*> error_switches; //список некорректных коммутаторов
    QList <scheduler::checktable> checktable_;
    QList <scheduler::TGEN> tgenl;
    scheduler sch;
protected slots:
void on_pushButton_2_clicked();
void on_pushButton_4_clicked();
void on_pushButton_5_clicked();
void on_pushButton_7_clicked();
void on_pushButton_3_clicked();
void on_pushButton_8_clicked();
private slots:
void on_pushButton_6_clicked();
private:
    uint numberrow_;
    Ui::oknotablitsa *ui;
};
#endif // OKNOTABLITSA_H

```

2. Листинг файла «oknotablitsa.cpp», содержащего реализацию главного окна графического интерфейса проверки корректности таблицы расписания

```

#include "oknotablitsa.h"
#include "ui_oknotablitsa.h"
#include "results.h"
#include "ui_rezults.h"
#include <QList>
#include <QWidget>
#include <QLabel>
#include <QTableWidget>
#include <QTableWidgetItem>
#include <QLineEdit>
#include <QLine>
#include <QMessageBox>
#include <QDebug>
#include <iostream>

using namespace scheduler_ns;
oknotablitsa::oknotablitsa(char *argv[], QWidget *parent) :
    QDialog(parent),
    ui(new Ui::oknotablitsa)
{
    ui->setupUi(this);
    setWindowTitle("Проверка корректности таблицы расписания СТП-ИСС-14");
    setFixedSize(1200,700); //размер окна

```

```

    int parserExitCode = 0;
    parserExitCode = xmlParser.doParsingTopology(argv, topology);
    if (parserExitCode == 4)
    {
        cout <<"incorrect input file" <<endl;
    }
    else
    { cout <<"correct input file" <<endl;
    }
    numberrow_=sch.get_sched_nodes_num(topology);
}

oknotablitsa::~oknotablitsa()
{
    delete ui;
}

void oknotablitsa::on_pushButton_2_clicked() //ЗАКРАСКА ЯЧЕЙКИ
{
    QTableWidgetItem *tbl = ui->tableWidget->currentItem();
    tbl->setForeground(Qt::darkGreen);
    tbl->setText(QString::number(1));
    tbl->setBackground(Qt::darkGreen); //при выделении цветов запись единицы в ячейку
    такого же цвета
}

void oknotablitsa::on_pushButton_4_clicked() //ОЧИСТИТЬ ТАБЛИЦУ
{
    int n=0;
    for(int i=0;i<ui->tableWidget->rowCount();i++){
        for( int j=0; j<ui->tableWidget->columnCount();j++){
            QTableWidgetItem *nol= new QTableWidgetItem(QString::number(n));
            nol->setForeground(Qt::white);
            ui->tableWidget->setItem(i,j,nol); //заполнить нулями
        }
    }
}

void oknotablitsa::on_pushButton_5_clicked()//КНОПКА "ПРОВЕРИТЬ"
{
    dl runtime = 0;
    const char *error_message = "";
    uint nodes_num;
    nodes_num = sch.get_sched_nodes_num(topology);

```

```

fillpos();
sch.fil_parameters(epochdata,zagruzka,packetsize,tochno);
cout<<"vse klevo"<<endl;
bool primary_checks_result =true;
bool minimal_delay_result =true;
bool epoch_result=true;
bool capacity_result=true;
bool necessty_result=true;
bool scheduling_result=true;
bool iscorrect= true;
QList <dl> runtime_mas;
runtime_mas.clear();
bool time_fail=true;
traffics_list.clear();
conflict_traffics_table.clear();
traffic_groups.clear();
error_message="";
error_traffics_list.clear();
error_nodes.clear();
error_switches.clear();
runtime=0;
nodes_list.clear();
error_conflicts_list.clear();
scheduling_groups.clear();
conflicts_list.clear();
checktable_.clear();
    if (iscorrect==true){
        traffics_list.clear();
        /*ИПОВЕРКА ШЛАГ 1*/
        primary_checks_result = sch.primary_checks(topology, traffics_list,
conflict_traffics_table, traffic_groups, error_message, error_traffics_list, error_nodes,
error_switches, runtime);
        if (!primary_checks_result) {

            iscorrect=false;
        }
        else{
            cout<<"step 1 correct"<<endl;
            iscorrect=true;
        }
    }

    if(iscorrect==true){
        /*ИПОВЕРКА ШЛАГ 2*/
        minimal_delay_result = sch.not_acceptable_traffics(topology, traffic_groups,
traffics_list, nodes_list, error_message, error_traffics_list, runtime);
        if (!minimal_delay_result) {

            iscorrect=false;
        }
    }
}

```

```

if(incorrect==true){

    /*ПРОВЕРКА ШАГ 3*/
    epoch_result = sch.get_epoch_duraton(traffic_groups, error_message, error_traffics_list,
runtime, epochdata.time_slot_num,epochdata);
    if (!epoch_result) {

        incorrect=false;
    }
    else{

        incorrect=true;
    }
}
epochdata.epoch_lcm.convert_unit(scheduler::ms);
uint lcm= epochdata.epoch_lcm.value;
if(incorrect==true){

    /*ПРОВЕРКА ШАГ 4*/
    capacity_result = sch.check_capacity(conflict_traffics_table, error_message,
error_traffics_list, error_nodes, error_switches, runtime);
    if (!capacity_result) {

        incorrect=false;
    }
    else{
        incorrect=true;
    }
}

if(incorrect==true){
    /*ПРОВЕРКА ШАГ 5*/
    error_traffics_list.clear();
    conflicts_list.clear();
    necessty_result = sch.check_scheduling_necessity(traffics_list, conflict_traffics_table,
error_message, traffic_groups, conflicts_list,
error_traffics_list, error_conflicts_list, scheduling_groups, runtime);
    if (!necessty_result) {

        incorrect=false;
    }
    else{
        incorrect=true;
    }
}
if(incorrect==true){
    uint strid=0;
    for(uint i=0;i<nodes_list.size();i++){
        if ((nodes_list.at(i).node_p->node_unit_list.at(0).transport_layer_list.at(0).name ==
"stp_iss_14") &&

```

```

        (nodes_list.at(i).node_p-
>node_unit_list.at(0).transport_layer_list.at(0).stp_iss_setting.m_scheduling_qos)){
            scheduler::checktable ch;
            ch.node_p= &nodes_list.at(i);
            ch.string_p= &scheduling_table.at(strid);
            strid++;
            checktable_.append(ch);
        }

    }

}

if(incorrect==true){
    /*ПРОВЕРКА ШАГ 6*/
    scheduling_result = sch.scheduling(traffics_list, nodes_list, conflicts_list,
error_message, scheduling_groups, error_traffics_list, runtime_mas,
time_fail,checktable_,tgenl);
    if (!scheduling_result) {

        incorrect=false;
    }
    else{
        incorrect=true;
    }
}

if(incorrect==false){
    QMessageBox::StandardButton reply = QMessageBox::question(this,"Ошибка","Таблица
расписания построена некорректно. Просмотреть результаты проверки?",
QMessageBox::Yes | QMessageBox::No);
    if(reply == QMessageBox::Yes){
        results window;
        window.setModal(true);
        window.showresults(error_message,error_traffics_list,error_nodes,error_switches);
        window.exec();

    }
    else{
    }
}
else{
    QMessageBox::about(this,"correct","Заданная таблица расписания корректна");
}

}

void oknotablitsa::on_pushButton_7_clicked() //СОХРАНЕНИЕ ПАРАМЕТРОВ
{

```

```

epochlenght=0;
step=0;
zagruzka=0;
packetsize=0;
tochno=1;
tgenl.clear();
if(ui->lineEdit->text().isEmpty() || /*ui->lineEdit_4->text().isEmpty() ||*/ ui->lineEdit_3-
>text().isEmpty() || ui->lineEdit_5->text().isEmpty() || ui->lineEdit_6->text().isEmpty())
{
    QMessageBox::warning(this,"предупреждение","введите все параметры");
}
else if (ui->lineEdit_3->text().toUInt()>1024 ||ui->lineEdit_3->text().toUInt()<=0 ) {
    QMessageBox::warning(this,"предупреждение","Длина эпохи должна лежать в
диапазоне от 0 мс до 1024 мс");

}
else if (ui->lineEdit_5->text().toUInt()>1) {

    QMessageBox::warning(this,"предупреждение","шаг между временными
интервалами может быть от 0 до 1");
}
else if (ui->lineEdit_6->text().toUInt()==0 ||ui->lineEdit_6->text().toUInt()>100 ) {
    QMessageBox::warning(this,"предупреждение","загрузка каналов возможна от 1%
до 100%");
}
else{

    epochlenght=ui->lineEdit_3->text().toUInt();//запись параметра максимальная длина
эпохи в переменную
    step=ui->lineEdit_5->text().toUInt(); //шаг между врем. интервалами,мс
    zagruzka=ui->lineEdit_6->text().toUInt(); //мах загрузка каналов %
    packetsize=ui->lineEdit->text().toUInt(); //запас размера пакета в байтах
    QList<unsigned long long> nodeId;
    QList < QList <unsigned long long>> trafId;
    uint tranum=0;
    for (int node_i = 0; node_i < topology.node_elements_list.size(); node_i++)
    {
        if
        ((topology.node_elements_list.at(node_i).node_unit_list.at(0).transport_layer_list.at(0).name
        == "stp_iss_14") &&

        (topology.node_elements_list.at(node_i).node_unit_list.at(0).transport_layer_list.at(0).stp_iss
        _setting.m_scheduling_qos)){

            nodeId.append(topology.node_elements_list.at(node_i).node_id);
        }
        QList <unsigned long long> tid;

        for(uint
        r=0;r<topology.node_elements_list.at(node_i).node_unit_list.at(0).aplication_layer_list.at(0).
        messages.stp_messages.size();r++){

```



```

tid.append(topology.node_elements_list.at(node_i).node_unit_list.at(0).aplication_layer_list.
at(0).messages.stp_messages.at(r).identificationNumber);

}

tr anum+=topology.node_elements_list.at(node_i).node_unit_list.at(0).aplication_layer_list.at
(0).messages.stp_messages.size();

if(topology.node_elements_list.at(node_i).node_unit_list.at(0).aplication_layer_list.at(0).mes
sages.stp_messages.size()>0){
    trafId.append(tid);
}
}
if(tr anum<2){
    QMessageBox::warning(this,"предупреждение","В сети задано менее 2 трафиков
с включенным качеством сервиса 'планирование");
}
epochdata.epoch_lcm.value=0;
epochdata.epoch_lcm.unit=scheduler::us;
    numbercolumn= ui->lineEdit_2->text().toUInt();
    scheduling_table.clear();
    epochdata.time_slot_num=0;
    epochdata.max_epoch_duration.value=0;
    epochdata.time_slot_duration.value = 0;
    epochdata.time_slot_duration.unit = scheduler::us;
    epochdata.delta_time_slot.value=0;
    epochdata.delta_time_slot.unit=scheduler::us;
    time_slots_start.clear();
    epochdata.time_slot_num=numbercolumn;
    epochdata.max_epoch_duration.value=epochlenght;
    epochdata.max_epoch_duration.unit=scheduler::ms;
    epochdata.max_epoch_duration.convert_unit(scheduler::us);
    /* Вычисление длительности тайм-слота */
    epochdata.time_slot_duration.value = epochdata.max_epoch_duration.value /
epochdata.time_slot_num;
    epochdata.time_slot_duration.unit = scheduler::us;
    epochdata.delta_time_slot.value=step;
    epochdata.delta_time_slot.unit=scheduler::ms;
    epochdata.delta_time_slot.convert_unit(scheduler::us);

    /* Начало каждого тайм-слота записываться в time_slots_start */
    for (ull ts = 0; ts <= epochdata.time_slot_num; ts++)
    {
        time_slots_start.append(ts * (epochdata.time_slot_duration.value +
epochdata.delta_time_slot.value));
        cout<<"tt"<<time_slots_start.at(ts)<<endl;
    }
    cout<<epochdata.time_slot_duration.value<<"duration"<<endl;
    cout<<epochdata.delta_time_slot.value<<"delta"<<endl;

    if (numberrow_>10) {

```

```

        QMessageBox::warning(this,"предупреждение","количество узлов не
должно превышать 10");
        return;
    }
    if (numberrow_<2){
        QMessageBox::warning(this,"предупреждение","Минимальное число узлов -
2");
        return;
    }
    if (numbercolumn>256 || numbercolumn<2){
        QMessageBox::warning(this,"предупреждение","Количество тайм-слотов
должно быть в диапазоне от 2 до 256");
        return;
    }

    else {
        ui->tableWidget->setRowCount(numberrow_+1);// количество строк в
таблице
        ui->tableWidget->setColumnCount(numbercolumn);//количество столбцов в
таблице

        for(int s=0;s<ui->tableWidget->rowCount()-1;s++){
            for(int c=0;c<ui->tableWidget->columnCount();c++){
                QTableWidgetItem *it = new QTableWidgetItem(QString::number(0));
                it->setForeground(Qt::white);
                ui->tableWidget->setItem(s,c,it);//заполнение таблицы белыми нулями
для записи и проверки
            }
        }

        for(int c=0;c<ui->tableWidget->columnCount();c++){
            int number=(int)(time_slots_start.at(c)/1000);

            QTableWidgetItem *it = new
QTableWidgetItem(QString::number(number)+"ms");
            it->setForeground(Qt::darkRed);
            ui->tableWidget->setItem(ui->tableWidget->rowCount()-1,c,it);
        }

        QStringList lst;
        QStringList lsb;
        for(uint i=1;i<=numberrow_;i++){

            QString s =QString::number(nodeId.at(i-1)) + " node( " ;
            for (uint f=0;f<trafId.at(i-1).size();f++) {
                if(f>0){
                    s+=", ";
                }
                s+=QString::number(trafId.at(i-1).at(f));
            }
        }
    }

```

```

    }
    s+=")";
    lst << s;
    ui->tableWidget->setVerticalHeaderLabels(lst); //именуем строки

}
QString s = "начало, мкс" ;
lst << s;
ui->tableWidget->setVerticalHeaderLabels(lst); //именуем строки
for(uint j=0;j<=numbercolumn;j++){
    QString k =QString::number(j) + "-slot" ;
    lsb << k;
    ui->tableWidget->setHorizontalHeaderLabels(lsb); //именуем столбцы
}
}
}

void oknotablitsa::fillpos(){
    scheduler::position pos;
    scheduler::scheduling_table_string str;//строка позиций
    bool posend=false; //конец позиции
    for (int f=0;f<ui->tableWidget->rowCount();f++) {
        str.string_id=f;//номер строки
        posend=false;
        QList<scheduler::position> poslist;
        for (int g=0;g<=ui->tableWidget->columnCount();g++) {
            if(posend==false&&mas[f][g]==1){
                pos.first_time_slot=g;
                posend=true;

            }
            else if (posend==true&&mas[f][g]==0) {
                pos.last_time_slot=g-1;
                posend=false;
                poslist.append(pos);//все позиции в строке
            }
        }
        str.positions_list= poslist;
        scheduling_table.append(str);

    }

}

void oknotablitsa::on_pushButton_6_clicked()
{
    QTableWidgetItem *tbl = ui->tableWidget->currentItem();
    tbl->setForeground(Qt::white);
    tbl->setText(QString::number(0));
    tbl->setBackground(Qt::white); //при выделении цветов запись единицы в ячейку
    такого же цвета
}

```

```

}

void oknotablitsa::on_pushButton_3_clicked()
{
    uint column;
    uint ID;
    ID=ui->lineEdit_7->text().toUInt();
    column=ui->lineEdit_8->text().toUInt();
    uint yzel;
    uint ni=0;
    QString eshe;
    for (int node_i = 0; node_i < topology.node_elements_list.size(); node_i++)
    {
        if
        ((topology.node_elements_list.at(node_i).node_unit_list.at(0).transport_layer_list.at(0).name
        == "stp_iss_14") &&

        (topology.node_elements_list.at(node_i).node_unit_list.at(0).transport_layer_list.at(0).stp_iss
        _setting.m_scheduling_qos)){

            for(uint
            r=0;r<topology.node_elements_list.at(node_i).node_unit_list.at(0).aplication_layer_list.at(0).
            messages.stp_messages.size();r++){

                if(topology.node_elements_list.at(node_i).node_unit_list.at(0).aplication_layer_list.at(0).mes
                sages.stp_messages.at(r).identificationNumber==ID){
                    yzel = ni ;

                    QTableWidgetItem *it = new QTableWidgetItem(QString::number(ID));
                    QBrush kistochka= ui->tableWidget->item(yzel,column)->background();
                    QBrush fintochka=ui->tableWidget->item(yzel,column)->foreground();
                    if(kistochka==Qt::darkGreen){
                        }
                    if(kistochka==Qt::darkGreen){
                        it->setForeground(Qt::white);
                        it->setBackground(kistochka);
                    }
                    else {
                        it->setForeground(Qt::darkMagenta);
                        it->setBackground(kistochka);
                    }

                    if(ui->tableWidget->item(yzel,column)->text().toUInt()==1 || ui-
                    >tableWidget->item(yzel,column)->text().toUInt()==0){
                        ui->tableWidget->setItem(yzel,column,it);
                    }
                    else{
                        eshe=ui->tableWidget->item(yzel,column)->text();
                        QTableWidgetItem *itit = new QTableWidgetItem(eshe+"
                        "+QString::number(ID));
                        if(kistochka==Qt::darkGreen){

```

```

        itit->setForeground(Qt::white);
        itit->setBackground(kistochka);
    }
    else {
        itit->setForeground(Qt::darkMagenta);
        itit->setBackground(kistochka);
    }
    ui->tableWidget->setItem(yzel,column,itit);
}
}
}
ni++;
}
}
bool find= false;
for(uint n=0;n<tgenl.size();n++){
    if(tgenl.at(n).trafid==ID){
        tgenl[n].timeslot =column;
        find= true;
    }
}

if(find==false){
    scheduler::TGEN tg;
    tg.trafid=ID;
    tg.timeslot=column;
    tgenl.append(tg);
}

}

void oknotablitsa::on_pushButton_8_clicked()//СОХРАНИТЬ ТАБЛИЦУ
{
    for(int c=0;c<100;c++){
        for(int v=0;v<100;v++){
            mas[c][v]=0;
        }
    }
    for(int i=0;i<ui->tableWidget->rowCount();i++){
        for( int j=0; j<ui->tableWidget->columnCount();j++){
            QString poz;
            QString zop;
            poz=ui->tableWidget->item(i,j)->text();
            zop=poz.at(0);
            mas [i][j]=zop.toUInt(); //запись таблицы в массив
        }
    }
    bool masempty = true;
    for (int f=0;f<ui->tableWidget->rowCount();f++) {
        for (int g=0;g<ui->tableWidget->columnCount();g++) {
            if(mas[f][g]==1){
                masempty=false;
            }
        }
    }
}

```

```

    }
}
}
if(masempty==true){
    QMessageBox::warning(this,"ошибка","Таблица расписания не заполнена");
    return;
}
for (int f=0;f<ui->tableWidget->rowCount()-1;f++) {
    bool stringempty=true;
    for (int g=0;g<ui->tableWidget->columnCount();g++) {
        if(mas[f][g]==1){
            stringempty=false;
            break;
        }
    }
    if(stringempty==true){
        QMessageBox::warning(this,"ошибка","Строка не заполнена");
        return;
    }
}
}
}

```

3. Листинг файла «results.h», содержащего класс results, основную функцию и переменные. Данный класс предназначен для реализации окна результатов проверки корректности таблицы расписания.

```

#ifndef RESULTS_H
#define RESULTS_H
#include "scheduler.h"
#include "oknotablitsa.h"
#include <QDialog>
#include <QTextEdit>
namespace Ui {
class results;
}

class results : public QDialog
{
    Q_OBJECT

public:
    explicit results(QWidget parent = nullptr);
    ~results();
    void showresults (const char message, QList <scheduler::traffic*> error_traffics_list, QList
<const _node*> error_nodes,QList <const _switch*> error_switches);
private:
    Ui::results *ui;
};

#endif // RESULTS_H

```

4. Листинг файла «results.cpp», содержащего реализацию окна вывода результатов проверки корректности таблицы расписания и рекомендаций пользователю

```
#include "results.h"
#include "ui_results.h"

results::results(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::results)
{
    setupUi(this);
    setWindowTitle("Результаты проверки таблицы расписания СТП-ИСС-14");
}

void results::showresults(const char *message, QList<scheduler::traffic *> error_traffics_list,
    QList<const _node *> error_nodes, QList<const _switch *> error_switches){

    textEdit->setText("В ходе проверки таблицы расписания выявлена(-ы) следующая(-ие) ошибка(-и):");
    textEdit->append(message);
    if(error_traffics_list.size()>0 && error_traffics_list.at(0)->source_node_p!=NULL){
        textEdit->append("Рекомендуется внести изменения в следующий(-ие) трафик(-и):");
        for (uint i=0;i<error_traffics_list.size();i++) {
            textEdit->append(QString::number(error_traffics_list.at(i)->traf_p-
                >identificationNumber)+"(node "+ QString::number(error_traffics_list.at(i)-
                >source_node_p->node_id) +")");
        }
    }
    if(error_nodes.size()>0){
        textEdit->append("Рекомендуется внести изменения в следующий(-ие) узел(-ы):");
        for (uint j=0;j<error_nodes.size();j++) {
            textEdit->append(QString::number(error_nodes.at(j)->node_id));
        }
    }
    if(error_switches.size()>0){
        textEdit->append("Рекомендуется внести изменения в следующий(-ие) коммутатор(-ы):");
        for(uint z=0;z<error_switches.size();z++){
            textEdit->append(QString::number(error_switches.at(z)->switch_id));
        }
    }
}
```