Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский авиационный институт (национальный исследовательский университет)»

На правах рукописи

Махалов Дмитрий Александрович

РАЗРАБОТКА КОМПЛЕКСА МОДЕЛЕЙ И МЕТОДИК АВТОМАТИЗИРОВАННОГО АНАЛИЗА ТЕЛЕМЕТРИЧЕСКОЙ ИНФОРМАЦИИ В РЕАЛЬНОМ МАСШТАБЕ ВРЕМЕНИ ДЛЯ ПИЛОТИРУЕМЫХ ОРБИТАЛЬНЫХ СТАНЦИЙ С ИСПОЛЬЗОВАНИЕМ СПЕЦИАЛИЗИРОВАННОГО ЯЗЫКА ПРОГРАММИРОВАНИЯ

Специальность 2.3.1. Системный анализ, управление и обработка информации, статистика (технические науки)

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
доктор технических наук
Матюшин Максим Михайлович

Москва – 2023

Оглавление

Введение

Актуальность темы диссертации. Космическое пространство является сферой национальных интересов и приоритетных разработок Российской Федерации, что находит отражение в концептуальных документах социально-экономического развития и обеспечения национальной безопасности [25, 28, 58, 59]. Стратегия развития космической деятельности России до 2030 года предполагает создание и запуск множества разнотипных космических аппаратов (КА) с различными телеметрическими системами. Начинаются разработки Российской орбитальной станции (РОС), которая продолжит пилотируемую программу освоения космоса после завершения эксплуатации Международной космической станции (МКС).

Расширение и развитие орбитального сегмента предполагает разработку и создание новых технологий построения секторов управления, обеспечивающих эффективное и надёжное управление КА. Интенсивные запуски новых КА накладывают повышенные требования на время разработки и эффективность эксплуатации средств обеспечения управления КА.

Телеметрический контроль состояния КА является неотъемлемой составляющей процесса управления полётом КА. В ходе проведения сеанса связи, который, как правило, длится от 6 до 20 минут (без задействования спутникового контура управления), группе анализа телеметрической информации (ТМИ) необходимо оценить состояние бортовых систем (БС) КА, его текущую ориентацию и режимы работы, результаты выполнения запланированных вне зоны радиовидимости операций, реализацию текущего сеанса связи, закладку программы полёта на следующие витки. Небольшая продолжительность сеанса связи в совокупности с возрастающим количеством контролируемых телеметрических параметров и динамических процессов на современных и перспективных КА создаёт объективные препятствия к проведению достоверного и всеобъемлющего анализа ТМИ КА в реальном масштабе времени.

Использование в системе управления спутников-ретрансляторов позволяет реализовывать программу полёта за существенно меньшие интервалы времени. В

частности, выведение транспортных кораблей «Союз МС» и «Прогресс МС» удалось сократить с 2 суток до 3 часов. В настоящее время прорабатывается возможность 1-виткового выведения (~2 часа). Данные обстоятельства существенно повышают требования к времени и достоверности выполнения анализа ТМИ в реальном времени. Работа с большими информационными потоками усиливает роль автоматизированного анализа информации [37].

Получение, обработка и анализ телеметрической информации с КА имеет важное значение на всех этапах подготовки, испытаний и эксплуатации космической техники. Телеметрическая информация посредством бортовой системы телеизмерений передается с различных систем и агрегатов КА, бортовых вычислительных систем, аппаратуры медико-биологического контроля состояния жизнедеятельности космонавтов, научных приборов и других источников и используется во всех основных функциональных составляющих процесса управления КА: информационно-телеметрического, баллистико-навигационного и командно-программного обеспечения [39].

Основу обработки ТМИ составляют алгоритмы предварительной, первичной и вторичной обработки, а также специализированные алгоритмы, которые разрабатываются для каждой бортовой информационно-телеметрической системы, бортовой системы управления, бортовой вычислительной системы, целевой аппаратуры – то есть, каждого источника специальной информации. При этом задача автоматизированного анализа ТМИ в большинстве существующих комплексов обработки ТМИ не решается вовсе, либо решается в ограниченном объёме. Методические основы для создания программных средств автоматизированного анализа ТМИ отсутствуют. Кроме того, в настоящее время активно развиваются технологии интеллектуального анализа информации, основанные на методах машинного обучения, таких как нейронные сети. Данные методы находят всё более широкое применение в работе средств автоматизированного анализа ТМИ и позволяют обеспечить решение задач, которые раньше не решались классическими методами или решались на ненадлежащем уровне. Таким образом, актуальной является задача разработки

общих методических подходов к реализации процесса автоматизированного анализа ТМИ КА в реальном времени.

Одной из ключевых составляющих эффективного анализа ТМИ КА является представление информации. При этом наиболее наглядные формы представления состояния бортовых систем КА на основе телеметрической информации обеспечивают мнемосхемы. В то же время методические основы по разработке наглядных и ёмких мнемосхем отображения состояния БС КА и контроля выполнения бортовых динамических процессов в настоящее время практически отсутствуют. Как правило, мнемосхемы реализуют либо путём непосредственного программирования, либо конструируют в простых графических редакторах из отдельных фигур, имеющих наборы состояний. Таким образом, существуют объективные препятствия к массовой разработке эффективных форм представления телеметрической информации, которые приводят при подготовке средств наземных комплексов управления к компромиссным решениям между количеством разрабатываемых мнемосхем, их сложностью и временем, необходимым на разработку.

Все вышеописанное определяет важность и **актуальность решаемой в диссертации научной задачи** – разработка методического аппарата автоматизированного анализа ТМИ в реальном масштабе времени.

Степень разработанности темы исследования

Вопросам разработки научно-методических и научно-технических подходов организации анализа телеметрической информации посвящены труды Микрина Е. А., Кравца В. Г., Соловьёва В. А., Соловьёва С. В., Любимского В. Е., Беляева М. Ю. (ПАО «РКК «Энергия»), Майдановича О. В., Мальцева В. Б., Охтилева М. Ю., Николаева А. Ю. (ВКА им. А.Ф. Можайского), Матюшина М. М., Титова А. М., Ронкина А. А., Валова Н. Н. (АО «ЦНИИмаш», г. Королёв), Некрасова М. В., Пакмана Д. Н., Талалаева А. А., Абрамова Н. С. (ОАО «ИСС им. М.Ф. Решетнёва»), Смирнова С. В., Ватутина В. М., Круглова А. В. (АО «РКС», г. Москва), Емельяновой Ю. Г. (ИПУ РАН), Тихомирова С. А. (РГРТУ), Скобцова В. Ю.,

Архипова В. И. (ОИПИ НАН Беларуси), Голубева И. Ю., Говорухиной Т. Н. и других [1, 6, 8, 10, 17, 18, 51, 54, 70, 71, 73, 78].

В работах Матюшина М. М., Тихомирова С. А., Абрамова Н. С., Донскова А. В., Соловьёва С. В., Охтилева М. Ю. [1, 15, 37, 38, 60, 72, 78] проведены исследования принципов проведения анализа ТМИ, контроля функционирования бортовых систем КА, разработаны методики и алгоритмы оценки и поддержки принятия решения о техническом состоянии КА по результатам автоматизированной обработки ТМИ.

В исследованиях Матюшина М. М., Титова А. М., Николаева А. Ю. [39, 76, 77] рассматриваются вопросы формирования функционально-логических схем обработки ТМИ из базовых алгоритмов.

Результаты анализа имеющихся научных трудов и исследований показали, что до настоящего времени недостаточно полно исследованы вопросы, связанные с построением универсальных систем автоматизированного анализа (САА) телеметрической информации в реальном масштабе времени. Не рассмотрены вопросы построения средств, обеспечивающих задание и исполнение алгоритмов автоматизированного анализа.

Мало исследована важнейшая задача наглядного представления (визуализации) результатов анализа ТМИ. Емельянова Ю. Г., Пономарёв И. А. рассматривают [18, 19, 64] различные абстрактные когнитивные образы для представления состояния КА, но не касаются вопросов реализации системы отображения этих образов.

Применение методов искусственного интеллекта, активно развивающихся в настоящее время, позволит найти решение для задач, не поддававшихся ранее автоматизации, или поднять решение существующих задач автоматизированного анализа ТМИ на новый качественный уровень. Исследованиям методов искусственного интеллекта для анализа ТМИ посвящены работы Соловьёва С. В., Скобцова В. Ю., Валова Н. Н., Скорнякова В. А. [6, 70, 72, 73]. В работах рассматриваются методы оперативного интеллектуального анализа отдельных значений ТМ-параметров или групп параметров. Вместе с тем вопросам анализа

динамических процессов на основе последовательных значений телеметрических параметров уделено недостаточно внимания.

Таким образом, разработка методического аппарата автоматизированного анализа ТМИ в реальном времени является актуальной темой исследований.

Целью диссертационной работы является повышение оперативности определения состояния бортовых систем контролируемого КА, своевременного выявления аномалий в их работе, автоматизация работы группы управления КА посредством разработки и реализации методических подходов к созданию программных средств автоматизированного анализа ТМИ в реальном времени. В частности, новые средства должны предоставлять возможности для управления большим количеством КА меньшим составом дежурных смен группы управления.

Объектом исследования является система автоматизированного анализа (САА) ТМИ, программные средства анализа ТМИ, а также телеметрический информационно-вычислительный комплекс (ТМИВК) ЦУП АО "ЦНИИмаш", использующийся для реализации информационно-телеметрического обеспечения полёта российского сегмента МКС, транспортных и пилотируемых космических кораблей, различных автоматических космических аппаратов, а также ракет-носителей и разгонных блоков, осуществляющих выведение указанных КА.

Предметом исследования являются модели, методики и алгоритмы автоматизированного анализа ТМИ, формы визуализации ТМИ, язык описания алгоритмов обработки, анализа и отображения ТМИ.

Методология и методы исследования. Для решения поставленных задач использовались общие методы системного анализа, теории алгоритмов, теории формальных грамматик, обработки информации, искусственного интеллекта, теории вероятностей и обработки экспериментальных данных, а также объектно-ориентированного проектирования и программирования.

Область исследования. Тема и содержание диссертации соответствует специальности 2.3.1. Системный анализ, управление и обработка информации, статистика (отрасль: технические науки).

Частными задачами исследования, определяющими его содержание и этапы, являются:

1. системный анализ существующих подходов реализации автоматизированного анализа ТМИ в реальном времени;
2. формализация задачи синтеза комплекса моделей и методик анализа ТМИ;
3. разработка лингвистической модели языка описания алгоритмов анализа ТМИ (язык анализа ТМИ) на основе современных языков программирования;
4. разработка различных, в том числе нейросетевых, алгоритмов автоматизированного анализа ТМИ, задаваемых на языке анализа ТМИ;
5. разработка методики формирования мнемосхем анализа ТМИ различных КА с использованием подпрограмм на языке анализа ТМИ, предназначенных для управления мнемосхемой на основе ТМИ;
6. оценка полученных результатов и работоспособности разработанного методического аппарата и программного обеспечения, обоснование рекомендаций по использованию языка анализа ТМИ и других результатов исследования в составе информационно-телеметрического обеспечения ЦУП.

Научная новизна

1. В результате системного анализа САА ТМИ сформирована система критериев и показателей, характеризующих эффективность выполнения анализа ТМИ КА в реальном времени, разработан новый частный показатель качества языка программирования: степень унификации языка программирования, позволяющий оценивать трудоёмкость изучения и применения языка программирования.

2. Формализована задача обработки и анализа ТМИ, разработана модель описания задач обработки телеметрической информации: лингвистическая модель языка описания алгоритмов анализа ТМИ (язык анализа ТМИ), отличающаяся от существующих возможностью в наглядной форме на высокоуровневом предметно-ориентированном языке описывать алгоритмы анализа ТМИ с использованием

базовых и специализированных алгоритмов, а также близостью по синтаксису к наиболее популярным языкам программирования.

3. Разработана методика визуализации и анализа телеметрической информации на основе компьютерных методов обработки информации с применением мнемосхем визуализации результатов анализа ТМИ, отличающаяся использованием исходных данных на языке анализа ТМИ, что позволяет формировать интерактивные динамические формы отображения в реальном времени.

4. Впервые разработана методика решения задачи обработки и автоматизированного анализа телеметрической информации, содержащей медицинские показания космонавтов, с использованием методов искусственного интеллекта, что в отличие от существующих методик позволяет в реальном времени проводить фильтрацию сбойных значений и адаптироваться к индивидуальным особенностям космонавта.

Теоретическая значимость работы заключается в развитии прикладных элементов системного анализа, теории формальных языков программирования и методов искусственного интеллекта применительно к задачам анализа ТМИ в реальном времени.

Практическая значимость полученных результатов состоит в следующем.

- Повышении оперативности, полноты охвата контролем и достоверности проведения анализа состояния бортовых систем КА.
- Сокращении времени и трудоёмкости подготовки средств информационно-телеметрического обеспечения к новым КА в части подготовки исходных данных на обработку, анализ и отображение ТМИ КА. В частности, разработанный язык анализа ТМИ позволяет готовить не только статические, но и динамические, интерактивные мнемосхемы состояния КА и контроля динамических процессов. Причём время подготовки мнемосхем уменьшено в 2,5÷4 раза в сравнении с существующими комплексами.

– Применимости полученных результатов для решения практических задач обработки и анализа ТМИ современных автоматических и пилотируемых КА, орбитальных станций и средств их выведения на существующих информационно-телеметрических комплексах.

На защиту выносятся следующие научные результаты диссертации, полученные автором лично, обладающие научной новизной, практической значимостью и отличные от результатов, полученных другими авторами.

1. Лингвистическая модель специализированного языка описания алгоритмов анализа ТМИ. Язык позволяет в наглядной и привычной большинству современных программистов форме задавать алгоритмы обработки и анализа ТМИ. Синтаксис языка построен на основе синтаксиса популярных в настоящее время языков программирования C++, C#, Java, но при этом имеет специализированные языковые конструкции, ориентированные на оперирование телеметрическими данными. Вводится новый показатель унификации языка программирования; показано, что разработанный язык обладает высокой степенью унификации.

2. Методика формирования мнемосхем визуализации результатов анализа ТМИ БС КА с использованием управляющих подпрограмм на языке анализа ТМИ, основанная на разработанной модели системы отображения мнемосхем, включающая в себя набор графических элементов, образующих макет мнемосхемы, транслятор подпрограмм на языке анализа ТМИ и интерпретатор подпрограмм анализа, исполняющий скрипт для управления поведением мнемосхемы. Такой подход позволяет сравнительно легко создавать динамические интерактивные мнемосхемы отображения состояния БС КА.

3. Методика нейросетевого анализа ТМИ, содержащей медицинские показания космонавтов, отличающаяся от существующих методик анализа медицинской информации процедурами учёта неравномерного поступления информации по нескольким каналам связи, наличия переменного уровня сигнала, наличия зашумлённости. Методика содержит процедуры точной привязки ТМ-кадров ко времени, коммутации единого сигнала из нескольких одновременно поступающих, определение пропущенных значений, устранение тренда сигнала

методом кусочно-линейной интерполяции. Далее применяется две нейросети, обученные на реальных показаниях космонавтов: первая определяет, содержит ли сигнал ЭКГ, а вторая выявляет в нём зубцы «R».

Степень достоверности и апробация результатов

Обоснованность разработанного методического аппарата обеспечивается корректной постановкой решаемой научной задачи, использованием методов исследований, не противоречащих основным положениям системного анализа и исследования операций, корректным использованием методов теории информации. Достоверность полученных результатов подтверждается результатами экспериментальных проверок и опытной отработки в составе действующих ЦУП КА.

Основные результаты диссертации прошли апробацию на семинаре кафедры «Системный анализ и управление» МАИ, в рамках докладов на научно-технических советах АО «ЦНИИмаш», а также на российских конференциях: V Научно-техническая конференция молодых учёных и специалистов ЦУП ФГУП ЦНИИмаш (2015 г.), «Лётные испытания, эксплуатация и целевое использование космических аппаратов и разгонных блоков» (2016 г.), VI Научно-техническая конференция молодых учёных и специалистов ЦУП ФГУП ЦНИИмаш (2016 г.), VIII Научно-техническая конференция молодых учёных и специалистов ЦУП (2018 г.), XXII Международная научно-практическая конференция, посвящённая памяти генерального конструктора ракетно-космических систем академика Михаила Фёдоровича Решетнёва (2018 г.), IX Научно-техническая конференция молодых учёных и специалистов ЦУП (2019 г.), XXIII Межведомственная научно-практическая конференция «Космические системы и комплексы: испытания, управление, применение» (2019 г.).

Основные научные результаты диссертации опубликованы в статьях, текстах докладов – всего в 12 трудах, в том числе: 6 статей в журналах рекомендованных ВАК РФ для публикации научных положений диссертационных работ [33, 34, 35, 44, 45, 49] («Космонавтика и ракетостроение» г. Королёв, «Сибирский журнал науки и технологий» г. Красноярск, «Пилотируемые полеты в космос» г. Звёздный

городок, «Космическая техника и технологии» г. Королёв), 4 сборниках тезисов и текстов докладов на научных конференциях и семинарах [41, 42, 43, 48]. Кроме того, результаты диссертации использованы при написании научно-технического отчета по НИР [66].

Личный вклад автора. В проведенном исследовании, совместных и личных публикациях лично автору принадлежат: результаты системного анализа существующих подходов реализации автоматизированного анализа ТМИ, лингвистическая модель языка описания алгоритмов анализа ТМИ, новый показатель оценки языка программирования – степень унификации, методика формирования мнемосхем визуализации результатов анализа ТМИ с использованием исходных данных на языке анализа ТМИ, методика автоматизированного анализа ТМИ, содержащей медицинские показания космонавтов, алгоритм нейросетевого анализа процесса отделения боковых блоков ракеты-носителя (РН) «Союз», а также практические исследования по оценке эффективности данных алгоритмов, практические исследования по оценке эффективности системы подготовки мнемосхем, по эффективности выполнения анализа ТМИ с использованием мнемосхем.

Реализация результатов исследования

Основные результаты, полученные при выполнении диссертационной работы, внедрены в:

– В АО «Российские космические системы» – при создании специального программного обеспечения обработки и анализа ТМИ в ходе выполнения СЧ ОКР «Создание наземного комплекса управления КА «Канопус-В» по договору № 07/19/2009 от 26.06.2009 между АО «Корпорация ВНИИЭМ» и ФГУП «РНИИ КП» и в ходе выполнения СЧ ОКР «Создание Белорусского наземного комплекса управления Белорусским космическим аппаратом» по договору между АО «Корпорация ВНИИЭМ» и ФГУП «РНИИ КП» (акт внедрения № 9-АК-79 от 02.02.2023 г.). Благодаря разработанным средствам в ЦУП КС «Канопус-В» удалось обеспечить надёжное управление группировкой из 6 КА минимальным количеством специалистов группы анализа.

– ПАО «РКК «Энергия» им. С. П. Королёва» – при модернизации телеметрического информационно-вычислительного комплекса ЦУП российского сегмента МКС в ходе выполнения составной части (СЧ) ОКР «МКС (Эксплуатация)» (Эксплуатация 4) по государственному контракту № 1922730301751217000241351/351-8647/19/175 от 23.10.2019 между Госкорпорацией «Роскосмос» и ПАО «РКК «Энергия» и СЧ ОКР «МКС» (МЛМ-У) по государственному контракту № 351-8517/15/361 от 30.12.15 г. между Госкорпорацией «Роскосмос» и ПАО «РКК «Энергия» (Акт внедрения № ГК-ВС/9 от 27.01.2023).

– В АО «ЦНИИмаш» – в составной части НИР «Прикладные исследования и проектирование ключевых элементов и технологий управления КА, бортовых комплексов управления…» (промежуточный, этап 2). Шифр СЧ НИР «Астролябия» (КА-2)», научно-технический отчет № 851-0226А/19/238-2-08201-612-2021. (Акт внедрения № 08103-59 от 27.02.2023).

– В АО «НИИ ТП» при создании единого ЦУП управления космической системой «Ресурс-П» в ходе выполнения СЧ ОКР «Доработка НКУ КА «Ресурс-П» для обеспечения управления КС «Ресурс-П». Участие в ЛИ КС «Ресурс-П» по договору № 171/19-14 от 01.07.2014 между АО «РКЦ «Прогресс» с АО «НИИ ТП» (акт внедрения № 19/64 от 01.03.2023).

Реализован ряд алгоритмов анализа и мнемосхем отображения состояния БС различных КА и контроля полётных операций, использованных как в штатных циклах управления КА, так и факультативно при выполнении сложных динамических процедур. Разработанные алгоритмы и мнемосхемы позволяют сократить количество формуляров ТМИ, необходимых для контроля полётных операций, количество необходимых рабочих мест анализа, повысить оперативность анализа и надёжность выявления аномалий.

Реализованы нейросетевые алгоритмы анализа ТМИ с использованием разработанных средств:

– алгоритм анализа динамического процесса отделения боковых блоков РН «Союз», позволяющий в реальном времени выявлять отклонения процесса отделения боковых блоков;

– методика анализа ЭКГ космонавтов в реальном времени, позволяющая в реальном времени проводить фильтрацию сбойных значений и адаптироваться к индивидуальным особенностям космонавтов.

Структура работы. Диссертация состоит из введения, четырех глав, заключения, списка сокращений и условных обозначений, списка литературы и приложений.

В первой главе проводится системный анализ современного состояния системы телеметрического обеспечения управления КА. Показано, что процесс обработки и анализа ТМИ в ЦУП удобно декомпозировать на 4 этапа и 23 подэтапа обработки. Определены проблемные аспекты выполнения автоматизированного анализа ТМИ КА в реальном времени. На каждом этапе выявлены типовые задачи, которые решаются универсальными, базовыми алгоритмами, и уникальные задачи, для которых требуется разработка специализированных алгоритмов анализа. Проведён подробный анализ особенностей передачи ТМИ, содержащей медицинские показания космонавтов. Далее проводится анализ методических подходов к реализации процесса анализа ТМИ в реальном времени. В завершение главы формируется система показателей и критериев качества системы автоматизированного анализа (САА) ТМИ и формулируется постановка задачи.

Во второй главе обоснован подход к совершенствованию САА ТМИ в реальном времени путём разработки набора инструментальных средств, в основу которых положен предметно-ориентированный, высокоуровневый язык описания алгоритмов автоматизированного анализа ТМИ (**язык анализа ТМИ**). Разработка языка выполнена на основе теории формальных грамматик, описание языка приводится с использованием расширенных форм Бэкуса-Наура. Разработана лингвистическая модель разработанного языка, его основные элементы, отличия от современных языков программирования с обоснованиями и примерами. В завершении главы приводится полная грамматика языка.

В третьей главе сформирован комплекс методик автоматизированного анализа ТМИ, основанных на использовании разработанного языка анализа ТМИ, включая методики трансляции и интерпретации (исполнения) исходных данных для автоматизированного анализа ТМИ КА, методику формирования мнемосхем отображения результатов анализа ТМИ БС КА с использование интерактивных динамических мнемосхем.

В качестве одного из направлений применения разработанных средств разработана методика нейросетевого анализа медицинских показаний космонавтов на участках орбитального полёта на транспортных пилотируемых кораблях (ТПК) «Союз МС», при выполнении внекорабельной деятельности в скафандрах «Орлан-МКС» и при прохождении различных медицинских обследований на МКС. Приводятся результаты практической отработки методики.

В четвёртой главе проводятся практические исследования по применению разработанных средств автоматизированного анализа ТМИ в составе программного информационно-телеметрического обеспечения ряда центров управления полётами, включая ЦУП российского сегмента МКС, ЦУП космической системы «Канопус-В», белорусского космического аппарата. Выполнена оценка результатов практического применения САА ТМИ на основе разработанного методического аппарата. Подтверждены преимущества выполнения анализа ТМИ с использованием мнемосхем, установлена высокая эффективность средств создания мнемосхем, построенных на основе разработанного методического аппарата. Разработан ряд мнемосхем, обеспечивающих автоматизированный анализ ТМИ КА в реальном времени. Разработан алгоритм нейросетевого анализа отделения боковых блоков РН типа «Союз». Алгоритм реализован на языке анализа ТМИ, выполнена оценка его эффективности.

В **заключении** сформулированы основные научные и прикладные результаты работы, а также предложения по их дальнейшему использованию

Структура и объем работы. Общий объем работы составляет 208 листов машинописного текста и содержит: рисунков 54, таблиц 27, список литературы включает 94 наименований на 9 листах, 1 приложение на 8 листах.

1. Анализ современного состояния системы телеметрического обеспечения управления КА. Постановка задачи исследования

Телеметрический контроль состояния КА является неотъемлемой составляющей процесса управления полётом КА. Обнаружить изменения технического состояния космического объекта можно только с помощью телеметрического контроля. Так как для управления КА все сведения о его техническом состоянии необходимо получить в реальном масштабе времени, то речь идёт об оперативном контроле [42, 44, 51]. В ходе проведения сеанса связи, который, как правило, длится от 6 до 20 минут, группе анализа ТМИ необходимо оценить состояние бортовых систем КА, его текущую ориентацию и режимы работы, реализацию работ, которые должны были быть выполнены непосредственно перед сеансом связи в «глухой» зоне, реализацию текущего сеанса связи, закладку программы работ на будущее. Небольшая продолжительность сеанса связи в совокупности с большим объёмом контролируемых телеметрических параметров и процессов создаёт объективные препятствия к проведению точного и всеобъемлющего анализа ТМИ КА в реальном масштабе времени.

В интересах обеспечения требуемой оперативности и достоверности получаемых оценок технического состояния бортовых систем КА в условиях воздействия помех, априорной неопределенности и возникающих нештатных ситуаций к обработке и анализу получаемой с борта КА измерительной информации предъявляется ряд требований [12, 40]:

– жесткие временные ограничения на оценку состояния КА и принятие решений в режиме, близком к режиму реального масштаба времени;

– высокие требования к уровню достоверности результатов обработки и анализа ТМИ;

– высокие требования к форме и качеству представления результатов обработки и анализа;

– требования по унификации, модульности и масштабируемости построения систем мониторинга;

– возможность интеграции данных о поведении бортовых систем и КА в целом, как объекта контроля.

Стратегия развития космической деятельности России до 2030 года предполагает создание и запуск множества разнотипных КА с различными телеметрическими системами. Разработка комплексов программ приёма, обработки и анализа телеметрической информации для каждого КА без использования задела увеличивает время и стоимость разработки и эксплуатации указанных комплексов. Таким образом, актуальной является задача разработки общих методических подходов к реализации процесса обработки и анализа ТМИ КА в реальном времени [56].

1.1.Этапы обработки телеметрической информации

Извлечение информации из телеметрических данных, математическое преобразование, анализ и представление результатов за минимальное время, обеспечивающее своевременное использование их в процессе управления КА – первая цель автоматизированной обработки телеметрической информации [51].

Рассмотрим наиболее общую этапность обработки телеметрической информации от пилотируемых и автоматических космических аппаратов, модулей российского сегмента международной космической стации, ракет-носителей и разгонных блоков [22, 54, 56, 60, 83, 86]. В зависимости от типа КА, его телеметрической системы и задач управления те или иные этапы могут отсутствовать [13, 80, 81]. Большинство задач обработки ТМИ решается базовыми алгоритмами обработки ТМИ [32, 39]. В общем случае обработку удобно разделить на следующие этапы, которые в отдельных комплексах могут быть объединены:

– этап регистрации и коммутации ТМИ;

– этап предварительной обработки;

– этап первичной обработки;

– этап вторичной обработки.

1.1.1. Этап регистрации и коммутации

I. **Оцифровка радиосигнала** выполняется на наземных приёмно-регистрирующих станциях (НПРС). При этом в телеметрической информации могут присутствовать ошибки, возникающие при неточном преобразовании физического процесса в электрические сигналы, излучении сигнала бортовым передатчиком, воздействии помех при передаче ТМИ по радиолинии или регистрации наземной станцией. Дальнейшая передача ТМИ идёт без искажений, но с возможными потерями отдельных пакетов данных.

II. При использовании методов помехозащищённого кодирования данных декодирование **свёрточных кодов** осуществляется, как правило, на НПРС. Данный вид помехозащищённого избыточного кодирования при помощи алгоритма Витерби [7, 36] позволяет сравнительно легко исправить подавляющее большинство сбоев, возникающих при передаче информации, однако он требует существенного сокращения информативности потока ТМИ (в 2 или больше раз) и поэтому применяется далеко не во всех телеметрических системах.

III. **Выделение телеметрических слов**. В общем случае поток телеметрической информации передаётся в ЦУП от НПРС в виде битового потока. Первой задачей обработки ТМИ в ЦУП является поиск отдельных слов. Решение данной задачи полностью зависит от типа телеметрической системы и опирается на особенности кодирования информации: размер слова (8, 10, 12 битов), наличие разрядов разметки, наличие разряда чётности и т. п. Как правило, один раз найдя позицию телеметрического слова, дальше можно отсчитывать заданное количество битов и получать следующие слова. Однако в процессе регистрации ТМИ с низким показателем сигнал/шум отдельные биты могут быть пропущены или вставлены лишние, что полностью нарушает синхронизацию на уровне слов, в результате чего требуется начинать сначала поиск битовых слов и определять точную позицию нарушения синхронизации.

IV. **Выделение телеметрических кадров**. Найденные слова собирают в телеметрические кадры (ТМ-кадры), используя такую служебную информацию, как маркер конца кадра, синхропосылку, идентификатор окраски борта, счётчик

кадров, позиции калибровочных уровней и др. Процедура идентификации кадра, особенно в зашумлённой информации, весьма трудоёмка. Поэтому, после того, как кадр найден, последующие кадры выделяются из битового потока (синхронизированного по границе слов) последовательно, и выполняется оценка полученных кадров на корректность.

V. **Первичная привязка ко времени**. Каждый ТМ-кадр привязывается ко времени регистрации на НПРС. Временем кадра считается время регистрации его первого слова. Корректная коммутация телеметрических потоков, принимаемых несколькими НПРС, может быть выполнена, если обеспечена точность привязки времени меньше трети интервала выдачи одного ТМ-кадра (обычно достигается точность в 0-3 мс).

VI. **Декодирование Рида-Соломона** [75, 87] выполняется для тех систем, для которых осуществляется соответствующее кодирование телеметрических блоков. На практике чаще всего применяются коды Рида-Соломона с характеристиками (255, 223), позволяющие исправлять до 16 байтов в блоке данных из 255 байтов, включающем 233 информационных и 32 проверочных байта. В зависимости от задач обработки декодирование Рида-Соломона может выполняться позже, на этапе предварительной обработки.

VII. **Точная привязка телеметрических кадров ко времени**. Для решения ряда задач требуется максимальная равномерность привязки ТМИ ко времени. Следует отметить, что ТМИ передаваемая от КА до различных НПРС, преодолевает различное расстояние. Хотя радиосигнал распространяется со скоростью света, разница в преодолеваемых расстояниях может быть настолько большой, что разница во времени регистрации одного и того же ТМ-кадра на двух НПРС может достигать от 1-2 мс до нескольких десятков и даже сотен миллисекунд (при использовании спутника-ретранслятора [31] или привязки ТМИ ко времени в ЦУП).

Таким образом, коммутация телеметрических потоков, поступающих от нескольких НПРС (описана далее), основанная только на времени регистрации ТМИ, невозможна: используя такое время, при переходе с одного НПРС на другой,

мы будем неизбежно терять часть телеметрических кадров или дублировать принятые ранее. Более того, задачи обработки высокочастотных измерений, таких, как виброизмерения, линейные ускорения, ударные нагрузки, медицинские показания космонавтов и т. п., требуют строгую монотонность следования времени, так как применяемые для обработки такой информации алгоритмы чувствительные к ошибкам даже в 1 миллисекунду.

Учитывая изложенное, на данном подэтапе выполняется точная привязка ко времени ТМ-кадров, обеспечивающая строгую монотонность следования времени. Время первого выделенного ТМ-кадра берётся за точку отсчёта t_0. Для каждого последующего кадра определяется порядковый номер кадра n после зафиксированного (с учётом возможных потерь), и время этого кадра пересчитывается относительно t_0:

$$t_n = t_0 + n \cdot \lambda ,$$

где λ – точная длительность передачи одного телеметрического кадра.

VIII. **Коммутация потоков ТМИ**. Как правило, ТМИ с одного КА принимается несколькими НПРС, расположенными по трассе полёта. Обработке подвергается единый, скоммутированный поток ТМИ КА. Существуют телеметрические комплексы, которые не реализуют подобный функционал и требуют от оператора вручную выбирать НПРС [54]. В большинстве же комплексов коммутация потоков ТМИ в единый поток заданного КА реализована, причём выполняется простейшим образом: с заданной периодичностью (например, 3 секунды) оценивается качество ТМИ каждого потока, и обработка переключается на поток с наивысшим качеством [54]. Зачастую в таких комплексах даже не выполняется описанная выше точная привязка времени ТМИ что приводит к коллизиям: при переходе с одной на другую НПРС время телеметрических кадров может перескакивать вперёд или назад, фрагменты ТМИ могут повторяться либо наоборот исчезать. Этот эффект приводит к потере цифровых пакетов, а также потере или двукратной регистрации срабатывания кратковременных событий.

Оптимальным алгоритмом коммутации, реализованном в ТМИВК, является следующий. Каждый поток привязывается к единой шкале времени по алгоритму, описанному выше. Затем выбор текущего потока осуществляется по каждому телеметрическому кадру по-отдельности, обеспечивая строгое следование кадров в результирующем потоке. Если в текущем потоке качество становится неудовлетворительным, а другой поток отстаёт по времени, алгоритм коммутации ожидает небольшое время в надежде получить требуемый кадр приемлемого для обработки качества. Для ряда телеметрических систем, использующих коды Рида-Соломона, разработаны алгоритмы, позволяющие восстанавливать достоверный телеметрический кадр (или кодовый блок) даже в том случае, когда кадры, принятые со всех НПРС, – сбойные [33].

1.1.2. Этап предварительной обработки

Входными данными для задачи предварительной обработки ТМИ являются телеметрические кадры, привязанные ко времени. Кроме того, наряду с телеметрическими кадрами передаются временные посылки, необходимые для отображения текущего времени потока и работы ряда алгоритмов обработки ТМИ. Оптимальной частотой следования временных посылок для большинства КА является 10 Гц.

IX. **Определение достоверности телеметрического кадра**. Если на более раннем подэтапе достоверность телеметрических кадров определялась для выбора наилучшего потока, то на данном подэтапе достоверность определяется для принятия решения о возможности обработки ТМ-кадра. Для телеметрических систем типа БР-9ЦУ, БИТС2, РТС выполняется контроль чётности телеметрических каналов и проверяется корректность служебных слов кадра, для систем типа ТКС и РТСЦМ корректность оценивается с помощью кодов Рида-Соломона и контрольной суммы. В обработку берутся не только кадры с абсолютно корректной чётностью. Допускается небольшой процент слов со сбойной чётностью (порядка 10 %), так как сбои в отдельных словах кадра будут устранены последующими алгоритмами.

X. **Привязка ко времени ТМИ режима воспроизведения (ВП)**. После оценки корректности кадра по служебной информации определяется режим передачи ТМИ. Если передаётся ТМИ, воспроизведённая из запоминающего устройства КА, необходимо выполнить привязку кадров, которые на данный момент сопровождаются временем регистрации, ко времени записи ТМИ на борту. На некоторых КА, например, КА типа «Канопус-В» с телеметрической системой ТКС, реализуется отсчёт бортового времени по декретному Московскому времени (ДМВ) [14, 46, 47], по которому работает ЦУП, и это время передаётся в ТМ-кадре. В этом случае привязка ТМИ режима ВП ко времени становится тривиальной.

На борту служебного модуля (СМ) РС МКС время также отсчитывается в соответствии с ДМВ, но бортовые часы идут существенно медленнее стандартного времени и за сутки равномерно отстают на 205,467 с. Каждую полночь время на бортовых часах сбрасывается. То есть, на бортовых часах время 00:00:00 идёт сразу после 23:56:35. Для корректного пересчёта бортового времени в ДМВ используется формула:

$$T_{\text{ДМВ}} = t_{\text{борт}} \cdot 1,00237809 .$$

На КА типа «Ресурс-П», многофункциональном лабораторном модуле РС МКС «Наука», а также ТПК «Союз МС» и ТГК «Прогресс МС» бортовое время отсчитывается от первого включения *i*-ого блока обработки информации (БОИ) на борту $t_0^{(i)}$. Таким образом, алгоритм привязки ТМИ режима ВП ко времени должен знать эту базу для вычисления поправки для каждого БОИ:

$$T_{ДМВ} = t_{борт} + t_0^{(i)}$$

XI. **Раскоммутация телеметрического кадра**. Содержимое телеметрического кадра определяется текущей программой опроса (сбора), номер которой указан в командно-служебном слове (КСС) или другом служебном параметре в начале кадра. Кадровая структура ТМИ предполагает, что в заданной программе опроса по конкретному каналу кадра передаётся один конкретный телеметрический параметр (или небольшая регулярная группа параметров). Таким

образом, раскоммутация телеметрического кадра, как правило, задаётся таблицами, описывающими состав программ опроса. Алгоритм раскоммутации последовательно извлекает значения каналов кадра и формирует соответствующие данным позициям параметры.

Значениям приписывается время самого ТМ-кадра для тех систем, в которых ТМ-кадр формируется однократным считыванием значений всех требуемых параметров из «зеркала» в момент формирования кадра (МБИТС, ТКС, …). Для кадров, формируемых последовательно (БР-9, БИТС2-12, …), время параметра корректируется относительности времени кадра на его позицию в кадре с учётом текущей информативности (скорости формирования кадра):

$$t_c = t_{\text{кадра}} + \frac{c}{L_w} \cdot \frac{L_f}{F_i},$$

Здесь с – номер канала в ТМ-кадре, L_w – размер кадра в словах, L_f – размер кадра в битах, F_i – информативность выдачи ТМИ. Например, для типового режима работы БИТС2-12 формула примет вид:

$$t_c = t_{\text{кадра}} + \frac{c}{512} \cdot \frac{5430}{256000} = t_{\text{кадра}} + c \cdot 0,0000414.$$

Параметры третьей ступени коммутации передаются группами по одному каналу, называемому субкоммутатором. Текущую позицию в субкоммутаторе указывает счётчик субкоммутаторов, в соответствии с которым выполняется раскоммутация.

XII. **Калибрование аналоговых параметров** осуществляется с учётом значений калибровочных уровней локальных коммутаторов, осуществляющих сбор данных параметров на борту. В любой телеметрической системе используется несколько локальных коммутаторов. Каждый локальный коммутатор питает телеметрические датчики вторичным напряжением (6,25-6,30 В и 0 В), оцифрованное значение которых передаётся по отдельным каналам телеметрического кадра, называемым калибровочными уровнями или

калибровками $k_{0\%}$ и $k_{100\%}$. Перевод значения аналогового параметра из двоичных единиц в проценты телеметрической шкалы осуществляется по формуле:

$$y_{\%} = \frac{x_{дв.ед.} - k_{0\%}}{k_{100\%} - k_{0\%}} \cdot 100\%$$

XIII. **Повышение достоверности** осуществляется путём отбраковки аномальных измерений [51] при помощи таких алгоритмов, как N-ε и M-N-ε. Ни в коем случае недопустимо на место отбракованных значений вставлять вымышленные значения, спрогнозированные при помощи тех или иных алгоритмов. Прогноз может оказаться неверным, а выводы, сделанные на основе таких вымышленных значений, могут привести к некорректным решениям.

XIV. **Сокращение избыточности** осуществляется рациональным исключением из последовательности значений каждого ТМ-параметра несущественных, близких друг к другу значений, без существенного снижения информационного содержания [79]. Повышение достоверности и сокращение избыточности решаются в большинстве случаев совместно [51].

XV. **Сборка цифровых регистров** выполняется с использованием алгоритмов повышения достоверности и сокращения избыточности. Цифровым регистром называют периодическую последовательность кодовых значений, передающихся по одному или нескольким каналам ТМ-кадра. Каждое слово регистра может повторяться фиксированное либо переменное число раз. Во втором случае регистр сопровождается маркерными битами, обозначающими смену значения.

При неполной синхронизации аппаратуры, формирующей регистр, и бортовой радиотелеметрической системы (БРТС) возникают т. н. краевые эффекты, когда часть битов регистра уже сменилась на следующее слово, а другая часть передаёт старое значение. Алгоритмы повышения достоверности, как правило, нивелируют этот эффект.

Если в словах регистра передаётся порядковый номер слова регистра, то эти слова можно обрабатывать независимо друг от друга, но в большинстве случаев регистр содержит маркер начала (конца) последовательности и её длину (если

длина не фиксированная). Для сборки таких слов в массивы, пригодные для дальнейшей обработки, разрабатывают специальные алгоритмы.

XVI. **Сборка цифровых массивов**, как правило, отличается от сборки регистров более сложной структурой, формируемой бортовой вычислительной системой. Цифровые массивы могут быть переменной длины, содержать контрольную сумму, сопровождаться собственными помехозащищёнными кодами. В СМ РС МКС цифровые массивы передаются фрагментами по 32 16-разрядных слова. Причём из-за небольшой разницы в скорости работы бортовой вычислительной системы (БВС) и БРТС один фрагмент с небольшой вероятностью может быть выдан повторно, что необходимо учитывать в алгоритме обработки. Маркер начала цифрового массива может случайно встречаться внутри массивов, что необходимо определять и обходить.

1.1.3. Этап первичной обработки

XVII. **Раскоммутация цифровых регистров** извлекает из полученных слов регистра значения отдельных параметров. Небольшие (короче длины слова) параметры передаются по несколько в одном слове, а большой параметр может передаваться в нескольких словах. В этом случае возможен эффект сборки двух частей от разных значений параметра в случае потерь фрагментов ТМИ, что можно нивелировать, только сравнивая время получения этих частей.

XVIII. **Раскоммутация цифровых массивов** похожа на раскоммутацию регистров, за исключением того, что массивы, как правило, формируются и передаются целиком. В отдельных случаях (например, в МКС) длинные массивы передаются адресным образом в виде фрагментов, которые можно как собирать в целый массив, так и раскоммутировать по частям. Для сборки таких массивов разрабатывают специализированные алгоритмы анализа ТМИ. Кроме того, в массивах возможна условная передача значений: в зависимости от значения управляющего параметра содержимое «хвостовой» части массива может быть переменным (например, для КА типа «Ресурс»).

XIX. **Декодирование значений**. Для кодирования вещественных значений используют различные типы кодов: нормальный, прямой, обратный,

дополнительные, инверсный, float, double и др. Задача этого подэтапа – выполнить обратное преобразование двоичного кода в вещественное значение и масштабировать его на заданный коэффициент.

XX. **Тарирование**. В результате предварительной обработки значения аналоговых параметров, как правило, формируются в процентах от телеметрической шкалы. Тарирование выполняется в два шага. Сначала проценты переводятся в Вольты или Омы (в зависимости от типа датчика) по формуле, общей для используемой телеметрической системы, а затем – в физические единицы измерений в соответствии с индивидуально заданной для каждого конкретного датчика тарировочной характеристикой.

XXI. **Формирование текстовых значений** выполняется для перевода абстрактных кодов в понятные текстовые сообщения. Значениям 0 и 1 многочисленных сигнальных параметров приписываются сообщения «Включено» / «Выключено», «Замкнуто» / «Разомкнуто», «Открыто» / «Закрыто», «Норма» / «Не норма» и т. п. Кодовым параметрам, которые могут принимать больше 2 значений, приписывают текстовые редакции для всех допустимых кодов. Если такой параметр примет неописанное значение, то оно может быть выведено на экран непосредственно либо проигнорировано.

1.1.4. Этап вторичной обработки

XXII. **Формирование обобщённых кодовых параметров** позволяет объединить значения нескольких кодовых параметров в один обобщённый и задать на него обобщённую текстовую тарировку, что упрощает анализ ТМИ и сокращает требуемое место на формулярах отображения.

XXIII. **Формирование обобщённых аналоговых параметров**, как правило, связано с формульными вычислениями различных вторичных показателей, выбора одного показания из нескольких идентичных, оценки полученных значений, необходимостью выбора преобразования в зависимости от текущего режима и т. п. В более сложных задачах применяют специальные алгоритмы анализа.

XXIV. **Агрегатные преобразования** выполняются над множеством значений одного параметра, поступающих во времени. Множество значений может

рассматриваться либо на протяжении всего сеанса обработки ТМИ, либо на фиксированных временных интервалах. К любым агрегатным преобразованиям может быть добавлен критерий. Тогда алгоритм рассматривает только значения, удовлетворяющие заданному критерию. Кроме того, можно задать интервал времени, который указывает алгоритму, как часто мы хотим получать его результат.

XXV. К задачам **анализа** телеметрической информации относят такие, которые не могут быть решены базовыми алгоритмами обработки ТМИ, в том числе ряд задач, перечисленных выше в данном перечне подэтапов обработки ТМИ. Это задачи сборки цифровых массивов, формирования текстовых протоколов, сложных вычислений, баллистических вычислений [35], восстановления сбойной информации, учёта наработки бортовых систем [46, 47], обработки сложных сигналов и т. п. Для решения таких задач дорабатывают программу обработки ТМИ либо пишут специальные алгоритмы на языке анализа ТМИ, чему посвящено данное диссертационное исследование.

1.1.5. Обработка медицинской ТМИ

Уникальной особенностью телеметрического обеспечения для пилотируемой программы полётов является задача обработки медицинских показаний космонавтов. К таким показаниям прежде всего относятся электрокардиограммы (ЭКГ) и пневмограммы (ПГ). Медицинские показания передаются в процессе выполнения следующих полётных операций:

1. выведение пилотируемого корабля (ТПК «Союз МС» или ПТК «Орёл») на ракете-носителе;
2. автономный полёт пилотируемого корабля, стыковка и расстыковка со станцией («Мир», МКС или РОС), посадка на Землю;
3. выполнение на станции медицинских обследований и медицинских экспериментов;
4. проведение внекорабельной деятельности (ВКД) космонавтов;

Контроль состояния космонавтов на всех стадиях полёта является критически важным, отклонения в сердечном ритме космонавта могут служить

сигналом к его отдыху (в процессе проведения внекорабельной деятельности), прекращению текущей полётной операции, изменению программы полёта, экстренному возвращению экипажа на Землю. Большой объём непрерывно поступающих данных медицинской телеметрической информации, подверженной зашумлению (рис. 1.1.1), пропаданиям фрагментов телеметрического сигнала (рис. 1.1.2), наводкам от мышечной активности (рис. 1.1.3), изменению уровня сигнала (рис. 1.1.4-1.1.6) создаёт объективные препятствия к проведению оперативного анализа медицинских показаний в реальном времени, и, как следствие, своевременному выявлению возможных отклонений в состоянии здоровья космонавтов и выдаче соответствующих рекомендаций.

Рассмотрим особенности передачи медицинской информации в ЦУП, знание которых необходимо для обработки сигнала. На космических кораблях ТПК «Союз МС» измерения ЭКГ и ПГ передаются непосредственно в составе телеметрического кадра на всех стадиях полёта с частотой 200 Гц. Ключевым требованием для обработки этих сигналов является предельно равномерная (не обязательно точная) привязка этих измерений ко времени и «бесшовная» коммутация ТМИ, принимаемой с различных НПРС (см. раздел 1.1 «Этапы обработки ТМИ», подэтапы VII и VIII). То есть на всех непрерывных участках приёма ТМИ интервал между измерениями ЭКГ должен быть строго 5 мс.

Самой сложной в настоящее время является схема передачи ТМИ со скафандров при выполнении внекорабельной деятельности (ВКД) (см. рис. 1.1.7). Каждый скафандр «Орлан МКС» является миниатюрным космическим аппаратом. Телеметрическая информация передаётся от измерительного комплекса скафандра на систему «Транзит» служебного модуля (СМ) РС МКС [24] по радиоканалу в структуре собственных телеметрических кадров. «Транзит» раскоммутирует эти кадры на ТМ-параметры и передаёт их в телеметрическую систему БИТС2-12 СМ РС МКС. В случае слабого уровня сигнала «Транзит» может некорректно определять телеметрические кадры, содержащие сбои.

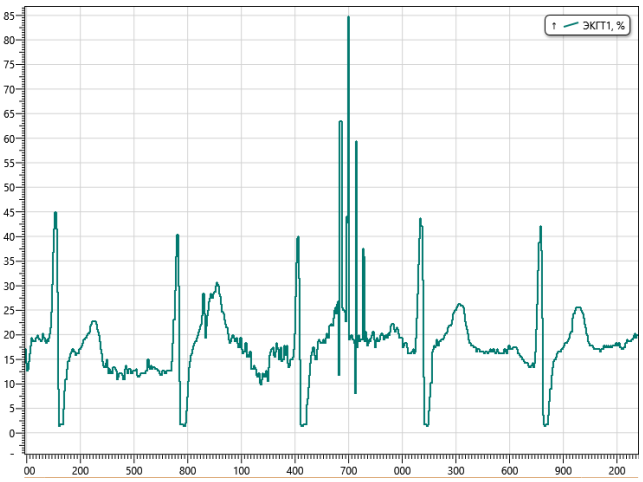


Рис. 1.1.1. Пример шумов

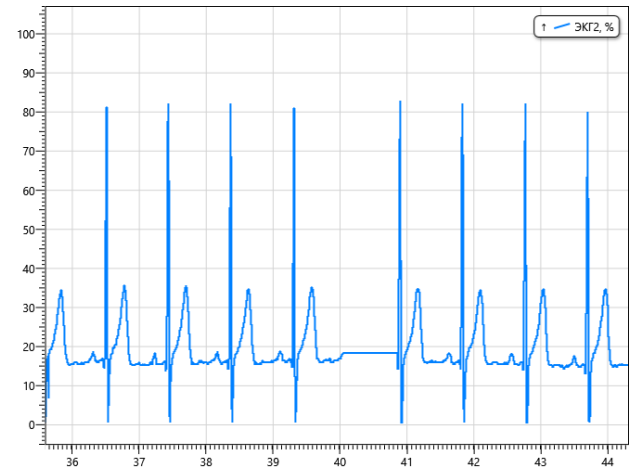


Рис. 1.1.2. Пример потери сигнала

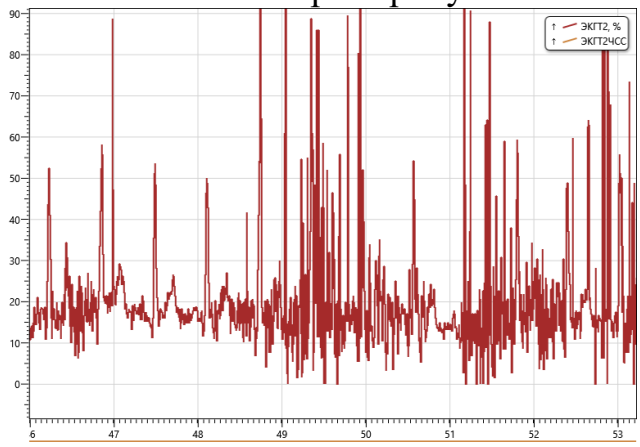


Рис. 1.1.3. Пример мышечных наводок

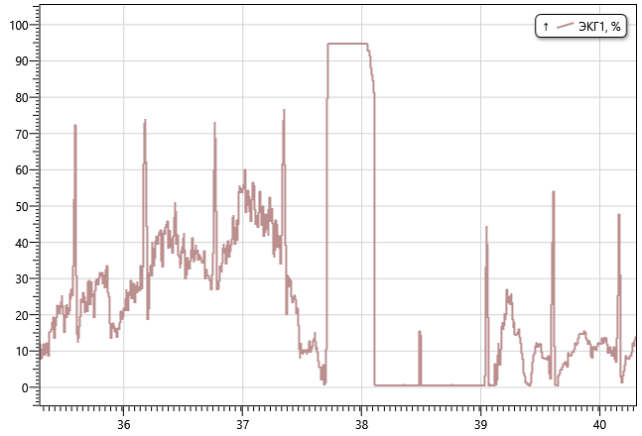


Рис. 1.1.4. Пример зашкаливания и падения уровня сигнала

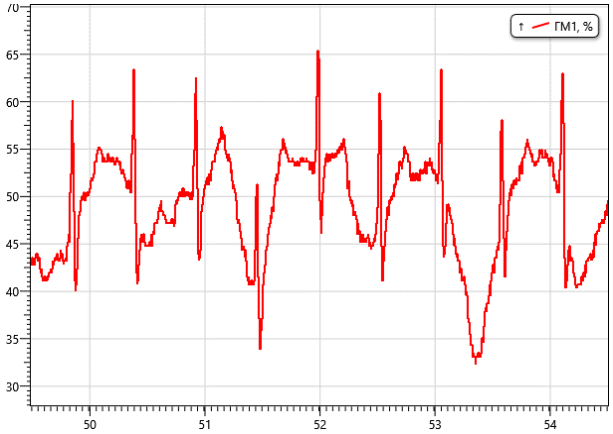


Рис. 1.1.5. Пример неустойчивого уровня сигнала

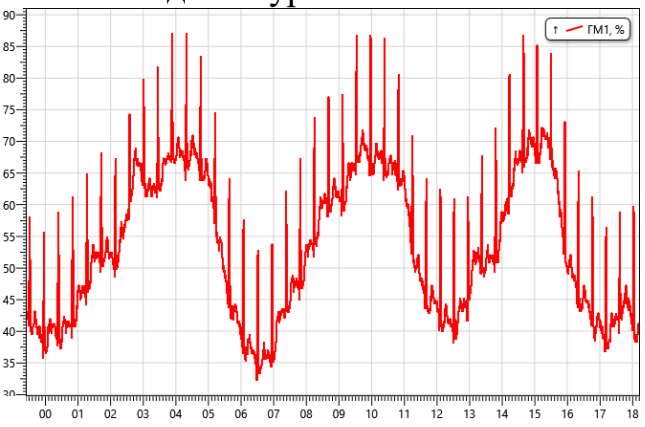


Рис. 1.1.6. Пример колебания уровня сигнала

БИТС2-12 передаёт измерения скафандров в составе отдельных каналов телеметрического кадра второго телеметрического потока (на рис. 1.1.7 обозначено «СМ2»). Поток СМ2 передаётся на российские наземные приёмно-регистрирующие станции (НПРС) в зоне их радиовидимости, либо через блок приёма информации низкой частоты (БПИ НЧ) на американский сегмент (АС) МКС, откуда поступает по каналам спутниковой системы передачи данных TDRSS

[93] в полосе частот Ku-диапазона в центр управления полётами ЦУП-Х (г. Хьюстон, штат Техас, США), и далее транзитом – в ТМИВК ЦУП АО «ЦНИИмаш».

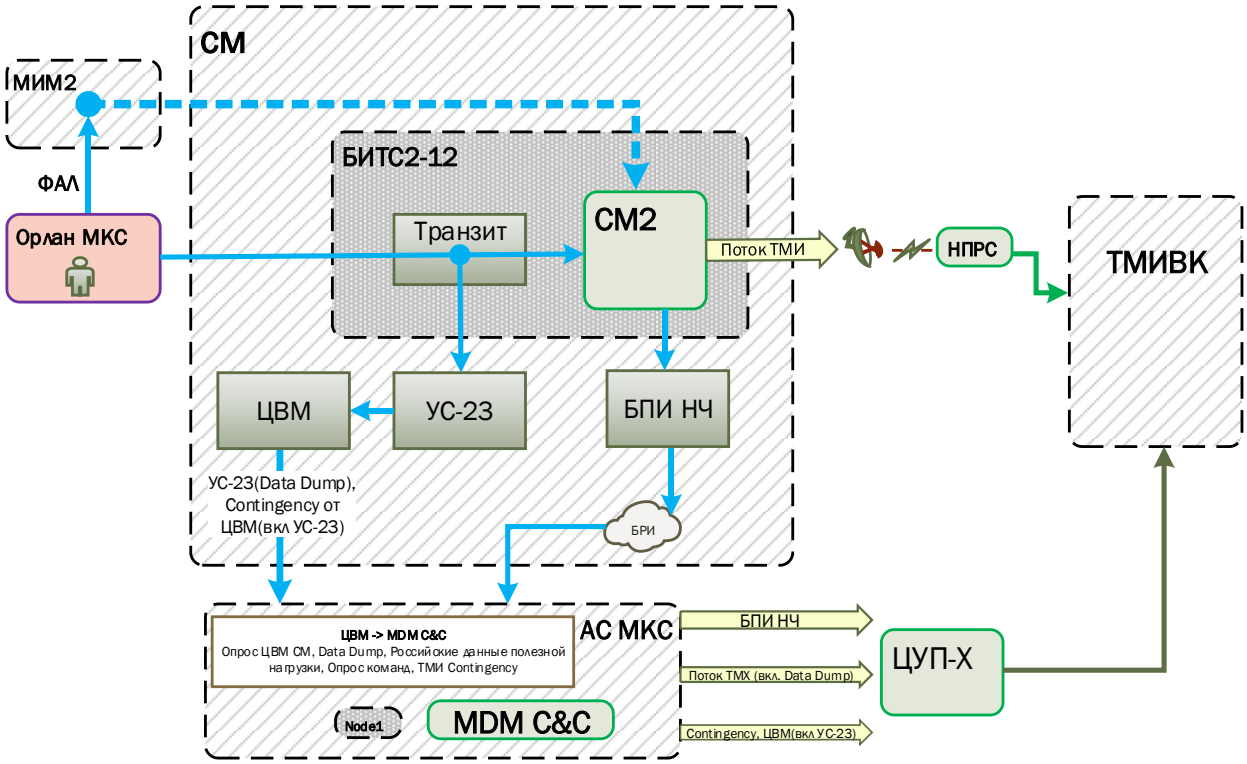


Рис. 1.1.7. Схема сбора ТМИ от скафандров «Орлан-МКС».

Третьим и основным путём передачи телеметрических данных со скафандров «Орлан МКС» является так называемый Data Dump. ТМИ скафандров упаковывается устройством согласования УС-23 в небольшие пакеты и через центральную вычислительную машину (ЦВМ) СМ РС МКС выдаётся в компьютер АС МКС. Там эти пакеты в составе американской ТМИ через TDRSS в полосе частот S-диапазона передаются в ЦУП-Х, откуда поступают в ТМИВК совместно с другой американской телеметрией. Этот поток телеметрии имеет наибольшую суммарную зону видимости и наиболее чистый сигнал.

Четвёртым является канал передачи цифровых массивов ЦВМ через американский канал TDRSS S-диапазона. Здесь ТМИ скафандров, включая ЭКГ и ПГ, существенно прореживается и может выступать лишь в качестве резерва.

Пятый канал передачи данных доступен только в период прямого и обратного шлюзования космонавтов – это передача ТМИ через электрофал, которым скафандр соединяется со шлюзовым отсеком МКС, расположенном в

модуле МИМ2 «Поиск». Так как данные передаются, минуя радиоканал скафандр-транзит, они менее подвержены искажениям, которые весьма распространены на этапе шлюзования по причине плохой радиовидимости, поэтому ТМИ с фала является более предпочтительной, когда она доступна.

Таким образом, актуальной является задача коммутации потоков ТМИ со скафандров, принятых по разным каналам, в единый поток, строго синхронизированный по времени, с последующим анализом медицинских показаний. Первичной задачей анализа является выявление QRS комплекса ЭКГ.

Используемый в ЦУП программно-аппаратный комплекс обработки медицинской информации [24] не выполняет такой коммутации и обрабатывает только один выбранный поток информации. Кроме того, его работа основана на процедурных методах, которые обрабатывают зашумлённые данные на ненадлежащем уровне и распознают небольшое число QRS комплексов.

1.2. Анализ методических подходов к реализации анализа ТМИ в реальном масштабе времени.

Систему обработки телеметрической информации можно условно разделить на следующие элементы: приём, обработка, выдача потребителям, отображение [51]. Для обработки каждого изделия необходимы индивидуальные исходные данные. Известно три основных подхода организации исходных данных на обработку и анализ телеметрической информации.

1. **База данных**. В базе данных (БД) задаются в табличной форме состав системы телеизмерений, структура телеметрического кадра и массивов цифровой информации, характеристики телеметрических параметров, указываются коэффициенты для алгоритмов повышения достоверности и сокращения избыточности и, в конечном итоге, тип и коэффициенты тарировочных характеристик. Данный подход является наиболее распространённым благодаря простоте реализации и эксплуатации. Использование базы данных как хранилища исходных данных при построении комплекса обработки и анализа ТМИ предлагается в работах С. А. Тихомирова [78], М. В. Некрасова [54], В. А. Чикурова [83]. Остальные подходы будем рассматривать в сравнении с ним. Вводом

исходных данных в базу данных могут заниматься люди средней квалификации, без обязательных навыков программирования. К недостаткам такого подхода можно отнести низкую гибкость системы обработки. При появлении новых бортовых систем, изменении требований к обработке и анализу ТМИ, реализации нового типа изделия, возникает необходимость модернизации модуля обработки, структуры базы данных, проведения полного цикла испытаний [30] и изменения эксплуатационной документации.

2. **Программная реализация**. Вся обработка полностью реализуется на языке программирования высокого уровня и поставляется в эксплуатирующую организацию в виде программного модуля. Достоинствами такого подхода являются высокая эффективность реализации, практически неограниченные возможности в применении различных алгоритмов и методов. Недостатком является высокая стоимость развития. Для изменения обработки необходимо привлекать программистов, проводить полный цикл испытаний и изменение эксплуатационной документации. К тому же, такой подход обладает наименьшей надёжностью в силу того, что ошибка в специализированных и, зачастую, достаточно сложных алгоритмах обработки может прервать весь процесс обработки, израсходовать всю доступную память или зациклиться. Такой подход, как правило, применяется в небольших комплексах обработки ТМИ малых КА. Кроме того, специализированные, сложные и наиболее требовательные к вычислительным ресурсам алгоритмы во всех комплексах, как правило, программируются непосредственно.

3. **Язык подготовки исходных данных**. Наиболее сложным в реализации, но при этом наиболее мощным и гибким в плане использования является подход, основанный на использовании эффективных базовых алгоритмов обработки [32, 39] и разработке высокоуровневого языка программирования, позволяющего описывать задание на обработку телеметрической информации. При таком подходе практически отсутствуют ограничения заранее определённого набора алгоритмов, заранее выбранной структуры базы данных. Написанные в системе подготовки исходных данных [52] на специализированном языке

алгоритмы исполняются интерпретатором в безопасном режиме. Недостатком данного подхода является высокая сложность первоначальной реализации (разработка языка, транслятора, интерпретатора), более высокие требования к специалистам по подготовке исходных данных в сравнении подходом, основанным на БД.

Кроме того, возможно применение подходов, комбинирующих в себе описанные выше.

Так как последний рассмотренный подход является наиболее мощным, гибким и перспективным, будем рассматривать задачу развития системы автоматизированного анализа ТМИ на его основе. Основы данного подхода заложены в телеметрическом информационно-вычислительном комплексе (ТМИВК) ЦУП.

Существующая в ТМИВК модель организации обработки и анализа ТМИ предполагает решение этих задач последовательно различными процессами (рис. 1.2.1, а). Такая организация процесса обработки, обусловленная историческим разделением решения этих задач по отдельным техническим средствам, в настоящее время потеряла актуальность. Более того, не всегда возможно построить процесс обработки так, чтобы задачи анализа выполнялись строго после задач основной обработки. В ряде случаев необходимо сначала выполнять специализированную обработку данных, например, извлечение и сборку цифровых массивов, разделение поступающих данных по источникам и т. п. В таких случаях в ТМИВК приходится запускать дополнительную задачу анализа ТМИ до задачи основной обработки (рис. 1.2.1, б), подавая ей на вход специальные исходные данные. Такой подход усложняет схему обработки ТМИ, понижает её надёжность и, при всём при этом, не обеспечивает решение всего спектра задач.

Ключевой задачей управления КА является контроль, диагностика и прогнозирование его состояния посредством анализа поступающей телеметрической информации [4, 15, 41, 42, 44, 48]. В [2] предлагается способ анализа ТМИ методом вейвлет-преобразования. В [6] предлагается метод

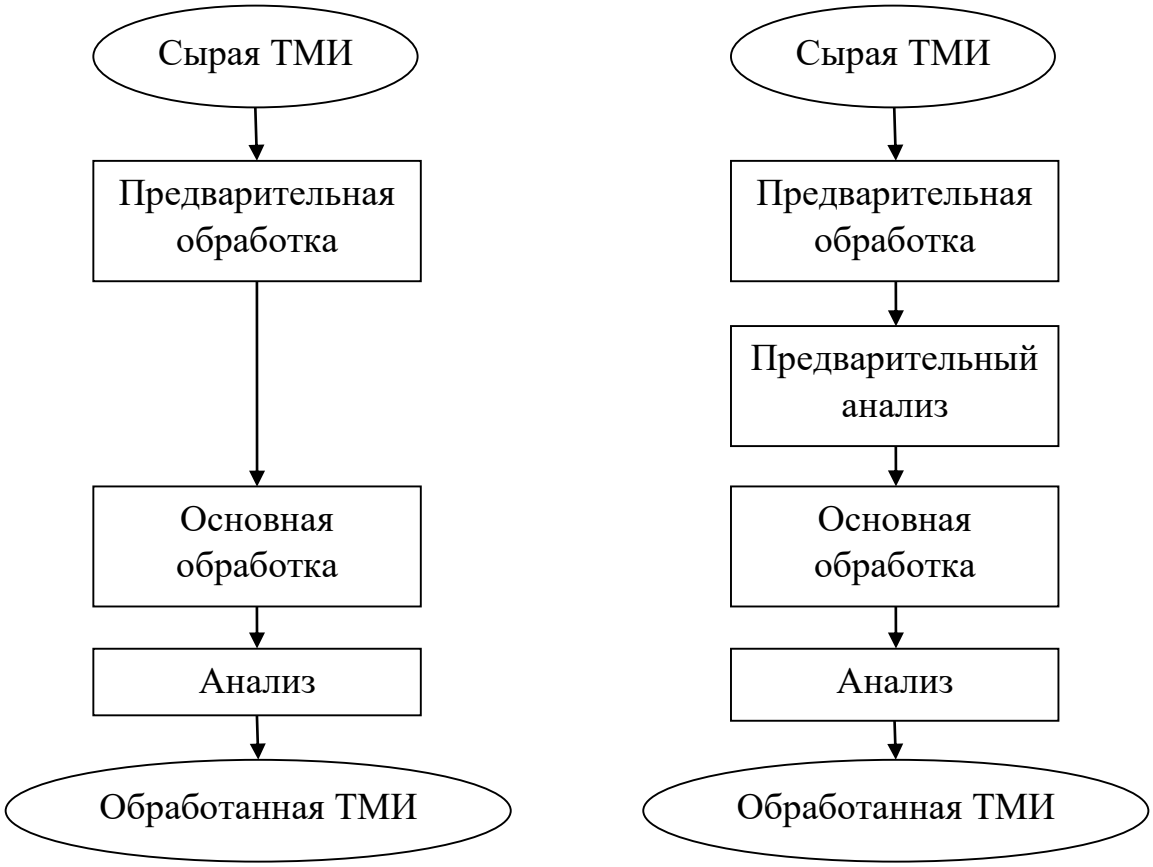


Рис. 1.2.1. Организация автоматизированного анализа ТМИ в ТМИВК

оперативного анализа состояния КА на основе принципов искусственного интеллекта с использованием нейронных сетей. При этом, в работе рассматривается лишь вопрос применения нейросетей, основанных на введённых вручную формулах допускового контроля, в то время как вопросы обучения нейросетей с использованием ранее принятой телеметрической информации опущены. Указанная задача рассмотрена в разделах 3.3 и 4.3.3 настоящей работы.

Одной из важнейших составляющих автоматизированного анализа телеметрической информации КА является её отображение. Человек зачастую не может эффективно справиться с большим объёмом поступающей информации [55] и эффективное представление этой информации непосредственно влияет на оперативность, качество, достоверность проведения анализа и, как следствие, надёжность управления КА. В настоящее время отсутствуют единые подходы представления результатов обработки и анализа ТМИ, обеспечивающие

представление больших объемов данных в ёмких и наглядных формах. На данный момент отсутствуют убедительные и завершённые результаты систематических исследований форм графического представления ТМИ, что препятствует их внедрению в рассматриваемой прикладной области [18].

Классическим способом отображения ТМИ являются табличное представление и графики. Ю. Г. Емельянова рассматривает [18] различные абстрактные когнитивные образы для представления состояния КА, но не касается вопросов реализации отображения этих образов.

Наиболее распространённым, информационно ёмким, наглядным и понятным человеку является способ представления информации в виде мнемосхем. В современных телеметрических комплексах отображение мнемосхем реализуется двумя способами.

В первом случае мнемосхемы непосредственно программируются в программном комплексе. Данный подход позволяет сравнительно быстро создать одну мнемосхему, но теряет свою эффективность при реализации набора мнемосхем. Любое изменение формы отображения требует изменения программного кода и перетрансляции программы отображения ТМИ с полным циклом испытаний (если она уже прошла испытания и находится в эксплуатации).

Во втором случае для описания мнемосхем используются исходные данные, формируемые вручную в текстовом виде или с использованием некоторой среды подготовки форм отображения. Мнемосхема строится из геометрических фигур и изображений, а также набора правил. Каждое правило в зависимости от значений телеметрических параметров указывает, какие графические элементы следует показывать, а какие скрывать (рис. 1.2.2). Таким образом реализуется отображение различных состояний КА.

К недостаткам последнего подхода можно отнести весьма ограниченные возможности по представлению информации. Оба подхода обладают достаточно высокой трудоёмкостью создания мнемосхем, что всегда является ограничивающим фактором при разработке телеметрического обеспечения полёта

КА. В частности, в 7 из 9 действующих в настоящее время ЦУП КА НСЭН мнемосхемы отображения состояния КА вообще не используются.

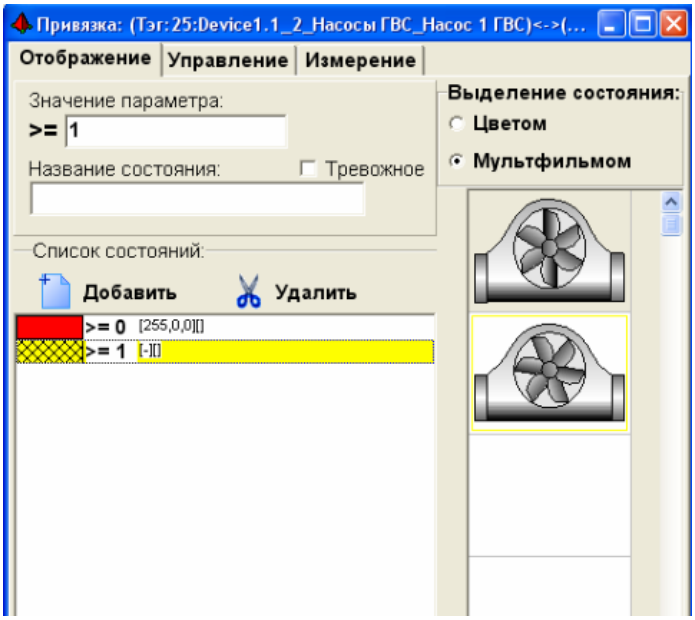


Рис. 1.2.2. Пример редактирования состояний элемента мнемосхемы

Современное программное обеспечение, предназначенное для отображения телеметрической информации в реальном времени, применяет статичные методы визуализации, сопровождающиеся гипертекстом, или же загромождают окно оператора сложными схемами контролируемых систем [18], что затрудняет проведения анализа состояния БС КА в реальном времени. Таким образом, существующий в настоящее время методический аппарат построения мнемосхем отображения состояния БС КА является недостаточно проработанным. В настоящей работе разработан методический аппарат, позволяющий сравнительно легко создавать динамические, интерактивные мнемосхемы отображения состояния БС КА с использованием языка анализа ТМИ.

1.3.Организация процесса обработки ТМИ в ТМИВК

В телеметрическом информационно-вычислительном комплексе (ТМИВК) ЦУП АО «ЦНИИмаш» обработка телеметрической информации организована в виде выстроенной в граф (не обязательно направленной) совокупности преобразований над значениями отдельных телеметрических (ТМ) параметров или групп ТМ-параметров. Каждому ТМ-параметру может быть назначена последовательность преобразований, реализуемых различными алгоритмами. Под

алгоритмом понимается конечное количество команд, для исполнения которых подаётся конечное число входных значений, и формирующих конечное число выходных значений за конечное число выполнений команд [3]. Подробно организация обработки ТМИ в ТМИВК с использованием базовых (универсальных) алгоритмов обработки описана в [39, 74, 76, 77]. При этом вопросы реализации специализированных алгоритмов автоматизированного анализа ТМИ практически не затронуты. Рассмотрим здесь кратко особенности, существенные для реализации автоматизированного анализа ТМИ на базе ТМИВК.

Преобразования ТМ-параметров задаются в исходных данных (ИД) на обработку ТМИ в виде текста на специализированном проблемно-ориентированном, декларативном языке, содержащем формализованные представления алгоритмов. Для подготовки исходных данных используется автоматизированная система подготовки исходных данных [52], включающая транслятор исходных данных. Транслятор проверяет синтаксис подготовленных ИД, преобразует их во внутренние структуры и таблицы, а затем сохраняет результаты трансляции в двоичные и текстовые (XML) файлы. Двоичные файлы ИД подаются на вход программе обработки ТМИ на сервере обработки и являются для неё заданием, а текстовые ИД используются программами отображения и документирования ТМИ (рис. 1.3.1).

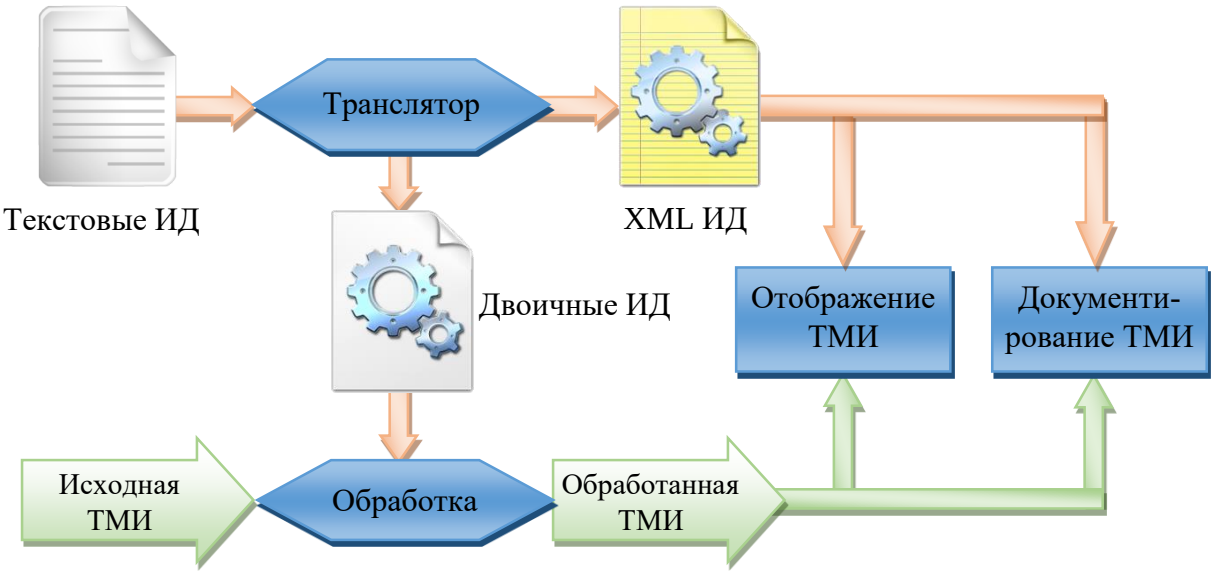


Рис. 1.3.1. Использование ИД для обработки ТМИ в ТМИВК

Взаимодействие алгоритмов друг с другом и с ТМ-параметрами можно изобразить при помощи простых схем (рис. 1.3.2), из которых видно, что алгоритм может: принимать на вход значения параметра, преобразовывать их и выдавать в этом же параметре (а), принимать на входе значения одного параметра, а формировать значения другого (б), на основе значений одного параметра формировать значения нескольких (в), останавливать обработку параметра (г), на основании значений нескольких входных параметров формировать (возможно, с задержкой) значения выходного параметра (д) и т. д.

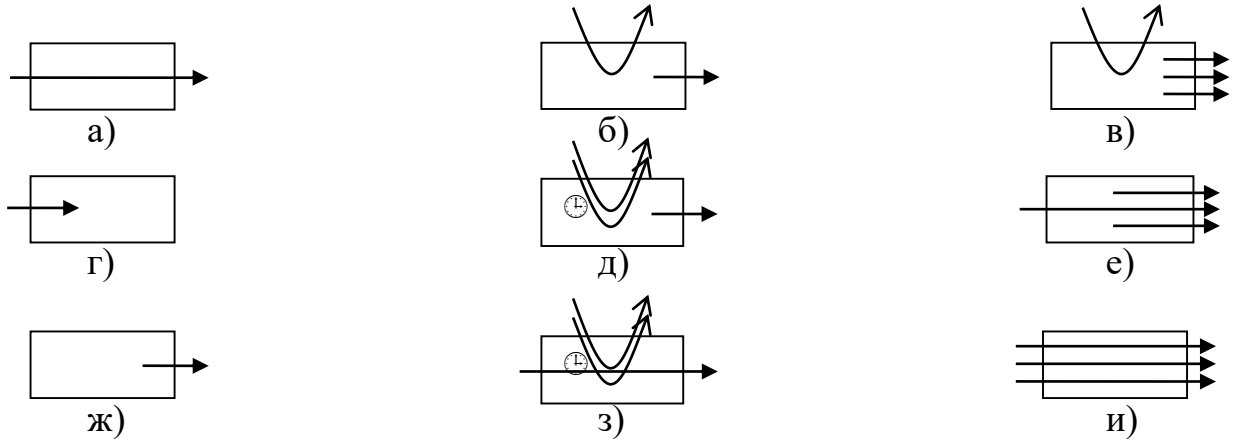


Рис. 1.3.2. Схемы алгоритмов обработки

Исходные данные на обработку ТМИ состоят из двух частей: файлы объявления параметров, которые в данной работе не рассматриваются, и файлы с описанием обработки. Файлы с описанием обработки состоят из **директив**, назначающих параметрам алгоритмы обработки. Здесь и далее при описании синтаксиса будем пользоваться расширенными формами Бэкуса-Наура (РБНФ) [65, 67]. Синтаксис директив следующий.

```
Директива = Список параметров, "=", Список алгоритмов, ";";
Список параметров = Имя параметра, { ",", Имя параметра };
Список алгоритмов = Алгоритм { ",", Алгоритм };
Алгоритм =
  Имя алгоритма, "(", Тело алгоритма, ")", [ "(", Вход, ")" ];
```

Здесь `Имя параметра` – это имя, объявленное в файлах объявления параметров, как правило, соответствует конкретному телеметрическому датчику, бортовой переменной или некоторому обобщённому параметру. `Имя алгоритма` – это имя базового алгоритма или объявленного ранее предварительного описания

алгоритма. `Тело алгоритма` – индивидуальное для каждого алгоритма описание его функциональных свойств.

Однострочный комментарий начинается с символа «@», а многострочный заключается в символы «@*» и «*@».

Приведём пример задания обработки.

```
ОДНа = ФОРМУЛА((ВХОД * (1 + 150.0 / 2215))),
       ФУНК(КГС/СМ2, 5.8:0, 21.7:3, 39.7:6, 57.9:9,
            75.9:12, 95.3:15);
```

Для параметра `ОДНа` задано последовательное преобразование двумя алгоритмами. Сначала алгоритм формульных преобразований преобразует входные значения по формуле:

$$f(x) = x \cdot \left(1 + \frac{150}{2215}\right).$$

Далее значение преобразуется с помощью кусочно-линейной функции, и результату приписывается размерность кгс/см^2. Принцип работы кусочно-линейной функции следующий. Задано множество узлов (x_i, y_i), сопоставляющих входным значениям выходные. Если $x_i \le x < x_{i+1}$, то результат вычисляется по формуле:

$$y = y_i + (x - x_i)\frac{x_{i+1} - x_i}{y_{i+1} - y_i}$$

График заданной в примере кусочно-линейной функции приведён на рис. 1.3.3.

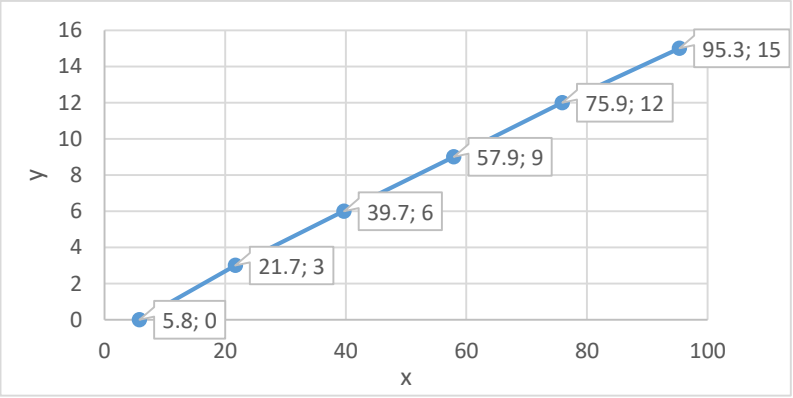


Рис. 1.3.3. График кусочно-линейной функции

В ТМИВК базового ЦУП Роскосмоса реализовано более 40 типовых (базовых) алгоритмов, с помощью которых решается большинство задач обработки ТМИ от пилотируемых и автоматических космических аппаратов, космических станций (ОК «Мир», МКС), разгонных блоков и ракет-носителей [32]. Специализированные задачи обработки реализуются алгоритмами анализа в отдельной программе автоматизированного анализа ТМИ или путём непосредственной реализации в программах предварительной и основной обработки ТМИ.

1.3.1. Идентификаторы параметров в исходных данных на обработку ТМИ

Идентификаторы параметров задаются в файлах объявления параметров на основе эксплуатационной документации на КА, исходных данных или на усмотрение специалиста по подготовке исходных данных. Традиционно в отечественной космонавтике используются следующие наименования параметров.

```
Имя параметра = { Символ идентификатора }- - { Цифра | Знак }-;
Символ идентификатора = Буква | Цифра | Знак;
Буква = "А"|"Б"|"В"|"Г"|"Д"|"Е"|"Ё"|"Ж"|"З"|"И"|"Й"|"К"|"Л"|"М"|
"Н"|"О"|"П"|"Р"|"С"|"Т"|"У"|"Ф"|"Х"|"Ц"|"Ч"|"Ш"|"Щ"|"Ъ"|"Ы"|"Ь"|"Э"|
"Ю"|"Я"|
"A"|"B"|"C"|"D"|"E"|"F"|"G"|"H"|"I"|"J"|"K"|"L"|"M"|"N"|"O"|"P"|"Q"|
"R"|"S"|"T"|"U"|"V"|"W"|"X"|"Y"|"Z";
Цифра = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9";
Знак = "-"|"+"|"/"|"!"|"%"|".";
```

Проще говоря, идентификатор ТМ-параметра состоит из заглавных букв латинского и русского алфавитов, цифр и некоторых знаков операций, в идентификаторе должна присутствовать по крайней мере одна буква, причём не обязательно стоять первой. Кроме того, длина идентификатора в большинстве КА ограничена 8 символами (реже – 9). Строчные и прописные буквы не различаются. Одинаковые по написанию буквы русского и латинского алфавитов считаются идентичными.

Указанные особенности накладывают определённые ограничения на использование синтаксиса современных языков программирования, так как допустимыми являются, к примеру, следующие идентификаторы: H2A%-B,

ШРС1+У, 2+Х, 1/СЕГ32, AL1.P!, ВКЛ.ПИТ. Такие идентификаторы будут трактоваться трансляторами обычных языков программирования как выражения. Кроме того, в отдельных бортах разрешены идентификаторы ТМ-параметров, содержащие любые печатные символы, включая пробелы.

Разрабатываемый язык анализа ТМИ должен поддерживать возможность оперирования ТМ-параметрами со сложными идентификаторами без чрезмерного усложнения синтаксиса.

1.3.2. Принцип работы программы обработки ТМИ в ТМИВК

Программа обработки ТМИ запускается сервером для обработки сеанса приёма ТМИ по конкретному борту. Она читает файл с двоичными исходными данными на обработку ТМИ, полученными в результате трансляции исходных данных, на основании которых формирует в памяти следующие структуры:

1. Коллекция алгоритмов (цепочек алгоритмов) обработки телеметрических (ТМ) параметров.
2. Таблица связей, указывающих, в какие алгоритмы на обработку подавать каждый ТМ-параметр.

В общем случае, у параметра p_i первым идёт собственная цепочка алгоритмов $A_1(p_i)$, которая преобразует его значения, а дальше могут указываться алгоритмы $A_j(p_i), j>1$, формирующие значения других параметров и принимающие значения параметра p_i только для чтения. Таким образом, формируется граф, узлами которого являются алгоритмы, а связями – параметры, передающиеся из одного алгоритма в другой. В общем случае граф не является направленным.

1.3.3. Программа автоматизированного анализа ТМИ

Для выполнения алгоритмов анализа ТМИ в ТМИВК используется отдельная программа (см. рис. 1.2.1), исходные данные для которой готовят на специализированном языке. ИД на этом языке транслируются в двоичное представление и подаются на вход программе автоматизированного анализа ТМИ

вместе с потоком ТМИ. Как было отмечено ранее, данная программа исполняется отдельно от программы обработки ТМИ, что препятствует их комбинированию. Кроме того, используемый там язык подготовки ИД для алгоритмов анализа ТМИ значительно отличается от современных языков программирования, что существенно усложняет его изучение и применение.

1.4.Система показателей и критериев качества системы автоматизированного анализа ТМИ в реальном масштабе времени.

При проведении исследований, связанных с выбором рациональных характеристик и параметров выполнения автоматизированного анализа состояния как всего КА в целом, так и его отдельных бортовых систем на основе телеметрической информации (ТМИ) в реальном времени, система автоматизированного анализа (САА) ТМИ рассматривается как организационно-техническая система, состоящая из ряда подсистем [62].

При исследованиях комплексных систем анализ их функционирования проводится преимущественно не по одному показателю и критерию, а на основе комплексного учета совокупности показателей и критериев [57]. Поэтому для оценки работы САА ТМИ необходимо сформировать систему показателей и критериев функционирования, максимально отражающих задачи системы [28].

Конечной целью САА ТМИ КА является обеспечение выполнения надёжного управления КА, нацеленного на обеспечение безопасности экипажа, поддержание живучести КА и выполнение целевых задач.

Для формализации выполнения автоматизированного анализа ТМИ КА, влияющего на надёжность управления КА, введём следующие обозначения:

R – множество телеметрических параметров (ТМ-параметров), определённых в обработке ТМИ КА;

$R^{контр} \subset R$ – множество ТМ-параметров, используемых для контроля полёта КА в реальном времени.

$R_{(i)}^{контр} \subset R^{контр}$ – множество ТМ-параметров, используемых для контроля i-й полётной операции;

$$R = R^{перв} \cup R^{втор} = R^{от} \cup R^{неот}$$

$$R^{контр} = \bigcup_i R^{контр}_{(i)} \subset R^{от}$$

Примечание. Не все ТМ-параметры КА выводятся на отображение для контроля состояния КА в реальном времени – часть параметров используется для послесеансного анализа, в том числе нештатных ситуаций, учёта наработки бортовых систем, оценки показаний других ТМ-параметров. Среди ТМ-параметров, выводимых на отображение, не все используются для контроля полётных операций – часть из них предназначена для оперативного анализа нештатных ситуаций, дублирования штатных параметров $R^{контр}$ в случае отказа части из них.

FO – множество полётных операций, выполняемых с КА.

Каждая полётная операция $fo_i \in FO$ характеризуется множеством параметров, которые необходимо контролировать во время её выполнения $R^{контр}_{(i)} = R^{контр}\left(fo_i \right)$ [73].

W – множество формуляров (форматов, окон) отображения, доступных для контроля состояния БС КА в реальном времени. Формуляры отображения можно условно разделить на табличные, содержащие идентификаторы и значения параметров, а также мнемосхемы, представляющие в графическом виде состояние бортовой системы, всего КА или процесса выполнения некоторой полётной операции. И табличные формуляры, и мнемосхемы могут содержать области вывода графиков значений параметров.

$W^{контр}_{(i)} = W\left(R^{контр}_{(i)} \right) = W\left(fo_i \right)$ – минимальный набор окон (формуляров), позволяющих контролировать ТМ-параметры $R^{контр}_{(i)}$ полётной операции fo_i.

$\left| W^{контр}_{(i)} \right|$ – количество формуляров, необходимых для контроля полётной операции fo_i.

Анализ

Анализ состояния КА в целом или его отдельной бортовой системы предполагает идентификацию этого состояния. В [37] рассматривается анализ состояния КА на основе значений одного параметра. Мы будем рассматривать множество параметров $R_{(i)}^{контр}$. Пусть КА имеет состояние s_k при выполнении полётной операции fo_i. Специалист группы анализа выполняет анализ $A_j\left(s_k \mid R_{(i)}^{контр}, W_{(i)}^{контр}\right)$ этой полётной операции на основе параметров $R_{(i)}^{контр}$ и формуляров $W_{(i)}^{контр}$. В результате анализа устанавливается состояние s_k^*. При этом анализ считается выполненным корректно, если удалось установить фактическое состояние КА $s_k^* = s_k$, и ошибочным в противном случае.

Обозначим $A_j^{оцен}$ – оценка результатов анализа, то есть корректно или нет выполнен анализ:

$$A_j^{оцен} = \begin{cases} 1, \text{при } s_k^* = s_k \\ 0, \text{при } s_k^* \neq s_k \end{cases} \tag{1.4.1}$$

Тогда, оценив результаты серии проведённых анализов $A = \{A_j\}$ состояния КА, выполненных на основе множества значений контролируемых параметров $R_{(i)}^{контр}$ и множества формуляров отображения $W_{(i)}^{контр}$ при выполнении полётной операции fo_i, получим вероятность успешного выполнения анализа состояния КА или достоверность анализа:

$$P(A) = P\left(s^* = s \mid R^{контр}, W^{контр}\right) = \frac{\sum_j A_j^{оцен}}{\|A\|}, \tag{1.4.2}$$

где $\|A\|$ - количество проведённых анализов.

Вероятность допустить ошибку в процессе проведения анализа обозначим:

$$\bar{P}(A) = 1 - P(A). \tag{1.4.3}$$

Основным критерием САА будет:

$$P(A) \to \max$$

или

$$\overline{P}(A) \to \min .$$

Через $A^{ЭКГ}$ обозначим анализ одного QRS комплекса ЭКГ космонавта, описанного в 1.1.5. Тогда $P\left(A^{ЭКГ}\right)$ – вероятность успешного выполнения анализа одного комплекса QRS. Задача разработки методики анализа ТМИ, содержащей медицинские показания космонавтов определяется критерием:

$$P\left(A^{ЭКГ}\right) \to \max \qquad (1.4.4)$$

Ещё одним важным параметром выполнения анализа состояния БС КА является время, затраченное на его выполнение. Обозначим $T\left(A_j\right)$ – время между получением значений параметров $R_{(i)}^{контр}$ на формулярах отображения $W_{(i)}^{контр}$ и формированием специалистом представления о состоянии КА s_k^*. Тогда дополнительным критерием выполнения анализа будет:

$$T(A) \to \min , \qquad (1.4.5)$$

что вычисляется также путём усреднения времени выполнения серии процедур анализа $T\left(A_j\right)$.

1.4.1. Показатели подсистемы подготовки исходных данных

Показатели подсистемы подготовки исходных данных на обработку, анализ и отображение ТМИ рассмотрим как показатели её составляющих [62]:

- показатели языка подготовки исходных данных для анализа ТМИ;
- показатели системы подготовки исходных данных отображения мнемосхем.

Можно сформулировать множество критериев оценки языка анализа ТМИ как системы программирования [21, 65, 88]. В таблице 1.4.1 приводятся результаты анализа таких критериев с оценкой важности их для выполнения настоящего исследования. Критерии оценивались прежде всего с точки зрения повышенных

требований к надёжности выполнения задач анализа ТМИ, ограниченному коллективу специалистов по разработке алгоритмов анализа ТМИ.

Таблица 1.4.1. Критерии оценки языка программирования

№	Критерий	Важность	Пояснения
1.	Ясность, лёгкость изучения, единообразие понятий	высокая	Язык должен быть близким к популярным языкам программирования, что упрощает подготовку специалистов, повышает понимание кода, снижает вероятность ошибок
2.	Скорость разработки/модификации кода на языке	высокая	создание алгоритмов анализа ТМИ выполняется небольшим числом специалистов за ограниченное время
3.	Выразительность (компактность) языка – сколько символов языка требуется для описания типовых задач.	средняя	повышает скорость создания и чтения кода и снижает вероятность ошибки
4.	Естественность, предметноориентированность – насколько язык отражает предметную область, для которой используется.	высокая	упрощает написание кода, повышает эффективность, снижает вероятность ошибок
5.	Выявление ошибок на этапе компиляции	высокая	повышает надёжность системы анализа ТМИ
6.	Скорость трансляции	низкая	трансляция выполняется редко, объёмы программ сравнительно небольшие
7.	Размер байткода	низкая	объёмы программ небольшие
8.	Эффективность исполнения (интерпретации)	средняя	потоки ТМИ низкоинформативные
9.	Объём занимаемой памяти в процессе исполнения	низкая	объёмы программ небольшие
10.	Совместимость с существующим комплексом обработки ТМИ	высокая	система анализа ТМИ должна быть внедрена в ТМИВК
11.	Сложность реализации (внедрения)	средняя	внедрение выполняется однократно и используется много лет

12.	Читабельность	высокая	позволяет легче выявлять ошибки
13.	Безопасность. Возможные ошибки в коде, написанном на языке, не должны нарушать другие алгоритмы обработки.	высокая	прерывание программы обработки ТМИ подвергает риску весь процесс управления КА
14.	Переносимость – возможность исполнять программы на языке на различных программно-технических средствах	низкая	современные ЦУП строятся на базе операционных систем типа Linux

Часть из приведённых показателей оценить невозможно численно либо их выполнение является обязательным, другие оцениваются только экспертным методом (например, читабельность), а для остальных введём систему формализации.

Для оценки **ясности, лёгкости изучения и единообразия понятий** в языке разработан новый показатель **унификации языка программирования**. Пусть:

E – множество всех элементов рассматриваемого языка программирования, включая синтаксис операторов, задания литеральных констант, вызова функций и т. п.

$E_u \subset E$ – множество унифицированных, заимствованных элементов языка, которые также присутствуют и в других популярных языках программирования, имеют такой же синтаксис и смысл и знакомы большинству программистов.

$E_n = E \setminus E_u$ – множество новых элементов языка, отсутствующих в других языках программирования.

Тогда степень унификации языка программирования определяется следующим образом:

$$U = \frac{|E_u|}{|E|}, \tag{1.4.6}$$

то есть равна отношению количества унифицированных элементов языка к общему количеству элементов этого языка. Чем больше используется унифицированных элементов, тем выше показатель унификации. Иначе этот показатель можно выразить так:

$$U = \frac{|E| - |E_n|}{|E|} = 1 - \frac{|E_n|}{|E|}.$$

Из этой формулы видно, что степень унификации тем выше, чем меньше новых, уникальных элементов привносится в язык.

Следует отметить, что степень унификации языка программирования может меняться со временем. Например, язык анализа ТМИ, использовавшийся ранее в ТМИВК, имеет много элементов, общих с языком программирования Эль-76, популярным в СССР в 1980-х годах. То есть, в те годы старый язык анализа ТМИ имел высокую унификацию, но в настоящее время она низкая, что будет показано в разделе 4.2.1.1.

При разработке языка программирования следует стремиться к более высоким показателям унификации [50], чтобы повысить надёжность и эффективность использования языка, а также скорость его освоения.

Длина кода. Разные языки позволяют один и тот же код записывать с помощью разных синтаксических конструкций, имеющих разную длину. Если в двух языках для кодирования одного фрагмента кода используются одинаковые операторы, то и длины их кода будут одинаковыми. Поэтому при оценке длины кода на языке анализа ТМИ следует рассматривать только те выражения, в которых применяются уникальные операторы языка.

Введём метрику $L(code)$ – показатель длины кода в существенных символах. Все символы операций и цифры считаются по количеству печатных символов. Пробельные символы не учитываются. Пользовательские идентификаторы независимо от длины считаются как один символ. Тогда критерием оценки языка анализа будет:

$$L(code) \to \min.$$

Ещё одним важным критерием подсистемы подготовки исходных данных является **время подготовки мнемосхемы** T_{cm} на основании заранее подготовленного задания [85]. Чем меньше время подготовки мнемосхем, тем

дешевле создание САА для нового КА, тем больше мнемосхем можно создать за отведённое время:

$$T_{cm} \to \min .$$

1.4.2. Показатели подсистемы отображения ТМИ

Эффективность подсистемы отображения ТМИ, выводящей мнемосхемы с результатами анализа ТМИ БС КА с использованием исходных данных на языке анализа ТМИ, предлагается оценивать по следующим показателям:

1. время, необходимое специалисту на анализ ТМИ с использованием мнемосхемы.
2. количество формуляров, необходимых для анализа состояния БС КА или динамического процесса.

В сеансах управления КА специалистам группы анализа приходится анализировать состояние всего КА, отдельных его бортовых систем или проводимых на КА динамических процессов. Для выполнения анализа состояния КА в процессе выполнения полётной операции fo_i оценивается поведение параметров $R_{(i)}^{контр}$, выводимых на формуляры отображения в виде таблиц, графиков, текстовых протоколов и мнемосхем. Грамотная организация таких формуляров может существенно сократить время, необходимое на анализ. В частности, параметры $R_{(i)}^{контр}$, изображённые на одной наглядной мнемосхеме анализировать существенно проще и быстрее, чем, в случае, когда они распределены по множеству табличных формуляров. Объем зрительного восприятия человека ограничен в среднем 5-9 несвязанных элементов [84], восприятие текстовой информации на меленьких маркерах является трудоемким и сложным [18], в то время, как анализаторам обычно приходится контролировать большое количество текстовых формуляров. Характеристики используемого графического образа определяют время, которое будет затрачено на восприятие отображаемой информации [64].

$W_{(i)}^{контр} = W\left(R_{(i)}^{контр}\right) = W\left(fo_i\right)$ – минимальный набор формуляров, необходимых для контроля ТМ-параметров $R_{(i)}^{контр}$. Одна хорошо спроектированная мнемосхема может заменить весь этот набор.

$\left|W_{(i)}^{контр}\right|$ – количество формуляров.

Грамотно организованное отображение ТМИ должно быть нацелено на минимизацию времени, затрачиваемого на анализ состояния КА $T\left(A_j\right)$ и повышение вероятности корректного выполнения анализа $P\left(A_j\right)$:

$$T\left(A_j\right) \to \min,$$

$$P\left(A_j\right) \to \max.$$

А система подготовки мнемосхем должна обеспечивать подготовку мнемосхем за минимальное время:

$$T_{cm}\left(R_{(i)}^{контр}\right) \to \min.$$

1.5.Постановка задачи синтеза рациональной системы автоматизированного анализа ТМИ в реальном масштабе времени

Реализация автоматизированного анализа ТМИ в реальном масштабе времени предполагает разработку комплекса моделей и методик, включая лингвистическую модель языка описания алгоритмов анализа *S*, методику формирования мнемосхем отображения состояния БС КА, алгоритмы анализа ТМИ КА, записанные на языке анализа ТМИ.

Язык анализа ТМИ должен позволять задавать всевозможные алгоритмы анализа ТМИ различных КА, включая автоматические и пилотируемые КА, орбитальные станции, ракеты-носители и разгонные блоки, а его синтаксис должен удовлетворять критериям максимальной степени унификации

$$U = \frac{\left|E_u\right|}{\left|E\right|} = \frac{\left|E\right| - \left|E_n\right|}{\left|E\right|} \to \max$$

и минимизации длины кода:

$$L(code) \to \min .$$

Система подготовки мнемосхем анализа ТМИ должна позволять создавать динамические, интерактивные мнемосхемы отображения состояния БС КА с использованием визуального конструктора и подпрограмм управления мнемосхемами, написанных на языке анализа ТМИ с минимальными трудозатратами, то есть за минимальное время:

$$T_{cm}\left(W_{(i)}, R_{(i)}^{контр}\right) \to \min$$

Методика анализа ТМИ, содержащей медицинские показания космонавтов должна обеспечивать распознавание сигнала ЭКГ с максимально высокой вероятностью:

$$p\left(A^{ЭКГ}\right) \to \max$$

Таким образом, задача исследования по совершенствованию средств автоматизированного анализа ТМИ для пилотируемых орбитальных станций сводится к разработке:

– языка описания алгоритмов анализа ТМИ в составе языка подготовки исходных данных на обработку ТМИ с высоким показателем унификации и компактным синтаксисом;

– методики подготовки и отображения мнемосхем анализа состояния БС КА, управление которыми осуществляется подпрограммами, записанными на языке анализа ТМИ;

– методика нейросетевого анализа ТМИ, содержащей медицинские показания космонавтов, более эффективного, чем существующие методики.

1.6.Выводы по первой главе

В главе 1 проведён анализ современного состояния системы телеметрического обеспечения управления КА. Определены проблемные аспекты выполнения автоматизированного анализа ТМИ КА в реальном времени. Показано, что процесс обработки и анализа ТМИ в ЦУП удобно декомпозировать на 4 этапа и 23 подэтапа обработки. На каждом этапе выявлены типовые задачи, которые решаются универсальными, базовыми алгоритмами, и уникальные задачи, для которых требуется разработка специализированных алгоритмов анализа. Проведён подробный анализ особенностей передачи и качества ТМИ, содержащей медицинские показания космонавтов, сформулирована задача по её коммутации и автоматизированному анализу.

Далее проводится анализ методических подходов к реализации процесса анализа ТМИ в реальном времени, включая вопросы организации исходных данных на обработку и автоматизированный анализ ТМИ. Обосновано проведение исследования на базе телеметрического информационно-вычислительного комплекса (ТМИВК) ЦУП АО «ЦНИИмаш».

Отдельно рассмотрены задачи по представлению ТМИ на мнемосхемах, как наиболее эффективному средству, обеспечивающему проведение анализа ТМИ специалистами группы управления. Выявлена недостаточная проработка существующего методического аппарата построения мнемосхем отображения состояния БС КА.

В завершение главы формируется система показателей и критериев качества системы автоматизированного анализа ТМИ и формулируется постановка задачи.

2. Разработка модели САА ТМИ в реальном масштабе времени на основе формальных грамматик

2.1. Методические основы разработки модели анализа ТМИ

2.1.1. Выбор метода решения задачи синтеза САА ТМИ в реальном масштабе времени

Существует два подхода организации обработки ТМИ: на сервере или на рабочих местах специалистов группы анализа. В [54] предлагается архитектура телеметрического комплекса, в котором задача обработки ТМИ решается на каждом рабочем месте анализа ТМИ параллельно. Данный подход имеет ряд ограничений в сравнении с обработкой, выполняемой на сервере:

1. В сложных алгоритмах обработки некоторые результаты вычислений могут сохраняться в памяти, а затем использоваться при обработке последующих сеансов данного КА, в частности, таким образом может вестись учёт наработки бортовых систем, что реализовано в ЦУП космической системы «Канопус-В» с использованием разработанных средств. В случае организации обработки ТМИ распределённым образом решение подобных задач невозможно.

2. Сервер может использовать результаты обработки ТМИ для организации телеметрических сеансов. Например, полученный из ТМИ вектор состояния КА может использоваться для прогнозирования его движения и расчёта зон радиовидимости с НПРС на последующих витках. Такая информация помогает специалистам ЦУП АО «ЦНИИмаш» планировать проведение сеансов в полуавтоматическом режиме. Зоны видимости, предварительно рассчитанные в телеметрическом комплексе, уточняются по заявкам от группы реализации, что частично автоматизирует действия персонала.

3. Результаты обработки ТМИ могут использоваться другими потребителями, например, комплексом моделирования и информационного обеспечения полёта, баллистическим комплексом, внешними потребителями, квитанционная информация может выдаваться в командно-программный комплекс

и т. п. В случае организации обработки ТМИ на рабочих местах специалистов организовать подобную выдачу невозможно.

4. Обработка ТМИ некоторых КА бывает весьма затратной в плане вычислений, она не может быть реализована на низкопроизводительных недорогих пользовательских персональных компьютерах, в то время как реализация обработки на сервере позволяет сэкономить на характеристиках рабочих мест ЦУП. Кроме этого, на сервере отсутствуют ограничения по используемой памяти для вычислительного процесса, которые весьма возможны на рабочих местах ЦУП.

Следует рассмотреть компромиссный вариант организации обработки: первичная обработка выполняется на сервере, а вторичные вычисления – в клиентских приложениях. Данный подход реализован в ЦУП американского сегмента МКС в космическом центре им. Л. Джонсона (г. Хьюстон, США). Он частично решает описанные выше ограничения, вызванные организацией полной обработки на рабочих местах, однако проблема передачи партнёрам всех необходимых телеметрических параметров сохраняется. Лишь спустя более, чем 10 лет полёта МКС в ЦУП г. Хьюстон был реализован сервис наземных вычислений, дублирующий полную обработку ТМИ и передающий внешним абонентам необходимый объём параметров.

Учитывая вышеизложенное, полная обработка ТМИ должна выполняться на сервере как это реализовано в ТМИВК ЦУП АО «ЦНИИмаш», а результаты обработки – выдаваться в реальном времени на рабочие места специалистов группы анализа ТМИ.

2.1.2. Методический подход к решению задачи

Методический подход к организации автоматизированного анализа ТМИ с использованием специализированного языка программирования в реальном масштабе времени включает в себя несколько этапов [5, 62].

1. Формализация задачи исследования
2. Разработка модели (грамматики) языка автоматизированного анализа ТМИ.

3. Разработка методики интерпретации задач анализа.

4. Разработка различных, в том числе нейросетевых алгоритмов автоматизированного анализа ТМИ.

5. Разработка методики формирования мнемосхем анализа ТМИ различных КА с использованием подпрограмм на языке анализа ТМИ, предназначенных для управления мнемосхемой на основе ТМИ.

6. Оценка полученных результатов и работоспособности разработанного методического аппарата и программного обеспечения, обоснование рекомендаций по использованию языка анализа ТМИ в составе информационно-телеметрического обеспечения ЦУП.

Разрабатываемый язык анализа будем строить на основе и рассматривать в сопоставлении с современными высокоуровневыми языками программирования общего назначения: C++, C#, Java. Именно эти языки являются основными при разработке программного обеспечения центров управления полётами различных КА. Для краткости далее будем называть их **основными языками программирования**.

2.1.3. Расширенная форма Бэкуса-Наура

При описании грамматики различных выражений в работе используются расширенные формы Бэкуса-Наура (РБНФ) [65, 67, 92].

В РБНФ применяются следующие обозначения:

- "a" – в кавычках задаются терминальные символы, обозначающие сами себя. В большинстве мест языка анализа ТМИ регистр символов не важен. Более того, буквы латиницы и кириллицы, имеющие в заглавном регистре одинаковое написание, считаются идентичными.
- A B – одним или несколькими словами без кавычек обозначаются нетерминальные символы, которые раскрываются через терминальные или другие нетерминальные символы выше или ниже по тексту.
- A | B – выбор альтернативы из двух элементов.
- (T) – группирование некоторой последовательности T.

- [T] – необязательный элемент, ноль или одно вхождение символа T.
- { T } – необязательная последовательность, ноль или более повторений символа T.
- { T }- – одно или более повторение символа T.
- A - B – обозначает всё то, что обозначает символ A, кроме того, что обозначает символ B (аналог операции разности множеств).
- a * B – обозначает ровно **a** повторений символа B.
- (* K *) – произвольный текстовый комментарий.

Для сокращения записи в перечислениях последовательных терминальных символов будем использовать символы «..». Например, вместо "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9" будем писать "0".."9"

2.2.Модель анализа ТМИ в реальном масштабе времени на основе формальных грамматик

В [54] предложена табличная организация исходных данных на обработку ТМИ. Такая структура является высоко детерминированной и не обладает достаточной гибкостью для решения широкого спектра задач. Наиболее естественным и гибким является способ описания с использованием текстовой формы. Кроме того, текстовая форма согласуется с существующим в настоящее время текстовым способом задания исходных данных на обработку ТМИ в ТМИВК. Предлагаемый язык описания алгоритмов анализа ТМИ строится на базе синтаксиса современных высокоуровневых языков программирования [49] C++, C#, Java, знакомых большинству современных программистов и широко применяющихся при разработке специального программного обеспечения ЦУП. Описание языка будем строить на основе формальных грамматик. Грамматика – это математическая система, определяющая язык [3]. Грамматика определяется как совокупность множества нетерминальных символов, множества терминальных символов, начального символа и множества правил [65].

Алгоритмы анализа ТМИ встраиваются в исходные данные на обработку ТМИ как отдельные алгоритмы (подпрограммы), наряду с базовыми алгоритмами обработки значений ТМ-параметров, что даёт возможность обычным алгоритмам

преобразований использовать результаты анализа ТМИ и наоборот. В каждой подпрограмме анализа задаются входные и выходные параметры, имена переменных, локальные функции и пользовательские типы. В результате трансляции исходных данных для каждой подпрограммы анализа транслятором исходных данных формируется внутреннее представление подпрограммы анализа в виде байт-кода, как это делается, например, в языках Java, Python и др. Исполнение подпрограмм анализа в реальном времени по мере поступления значений входных параметров осуществляется с помощью интерпретатора, который выполняет функции, аналогичные виртуальным машинам в современных языках программирования [45, 49].

Такой подход к обработке телеметрической информации позволяет создать программное обеспечение, реализующее транслятор и интерпретатор подпрограмм анализа, а всю логику вычислений, требуемых при анализе ТМИ конкретных КА, задать в виде исходных данных на высокоуровневом языке.

Модель языка анализа ТМИ (рис. 2.2.1) включает в себя следующие элементы [45]:

- входные и выходные ТМ-параметры;
- переменные;
- типы данных (скалярные, массивы, кортежи, текстовые, пользовательские);
- функции;
- инструкции (объявления локальных переменных, управляющие, исполнения операторов);
- операторы (арифметические, присваивания, вызова функций)
- модель вызова и исполнения алгоритма анализа.

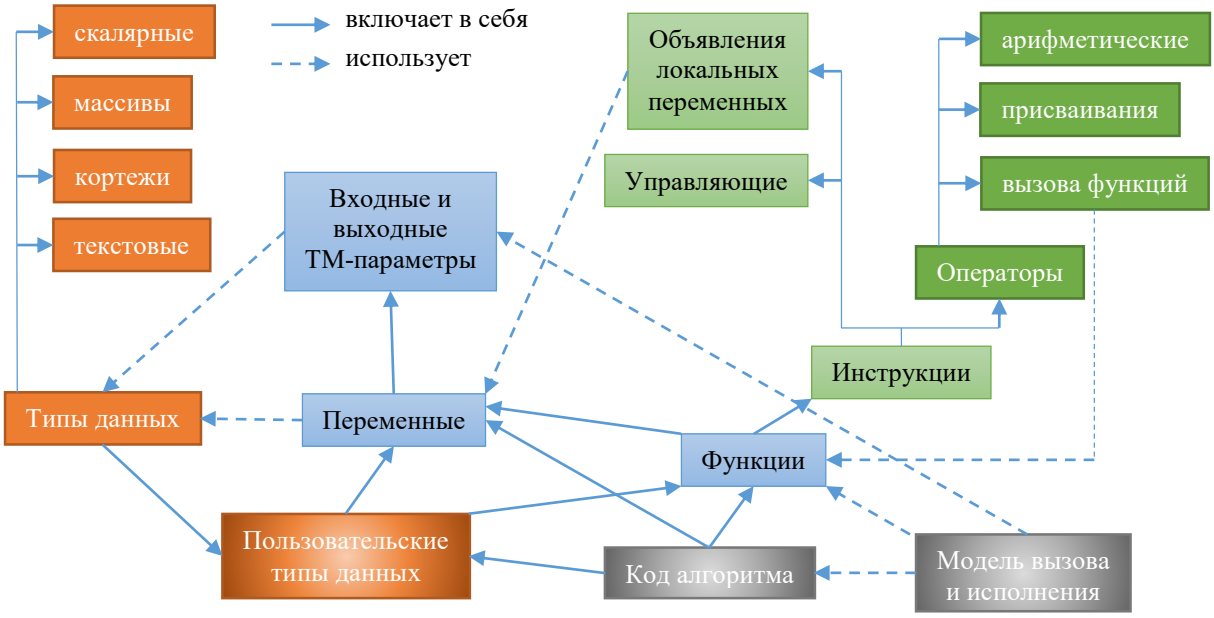


Рис. 2.2.1. Модель языка анализа ТМИ

Для задания подпрограмм анализа в языке подготовки исходных данных разработан алгоритм с именем «АНАЛИЗ». Подпрограммы имеют структуру, приведённую на рис. 2.2.2.

```
АНАЛИЗ(
     <Способ запуска и исполнения алгоритма>
     <Объявления входных и выходных параметров>
     <Объявления глобальных переменных>
     <Объявления пользовательских типов>
     <Описание функций>
{
     <Код главной функции алгоритма>
}
)
```

Рис. 2.2.2. Общая схема задания алгоритма анализа ТМИ

В каждом алгоритме анализа перечисляются входные и выходные ТМ-параметры с указанием типа их значений. Если выходной параметр в процессе исполнения алгоритма получит новое значение, то в конце работы алгоритма это значение ТМ-параметра будет сформировано и подано на дальнейшую обработку другим алгоритмам. При необходимости явного формирования значений ТМ-параметра, например, если за одну итерацию исполнения алгоритма требуется

сформировать несколько значений этого параметра, это можно сделать вызовом специального метода.

Рассмотрим различные элементы языка, подробнее останавливаясь на элементах, отсутствующих или отличающихся от основных языков программирования.

2.2.1. Типы данных

К базовым типам данных, доступным в алгоритмах анализа, относятся типы, основанные на типах языка C#:

bool – логический тип, принимающий значения **false** (ложь), **true** (истина);

byte – беззнаковое целое, занимающее 1 байт;

short, **ushort** – знаковое и беззнаковое целое, занимающее 2 байта;

int, **uint** – знаковое и беззнаковое целое, занимающее 4 байта;

long, **ulong** – знаковое и беззнаковое целое, занимающее 8 байтов;

float – вещественное, занимающее 4 байта;

double – вещественное, занимающее 8 байтов;

char – символьный тип;

string – строковый тип;

код – кодовый тип, занимающий 9 байтов, из них 8 байтов (64 бита) отводятся под значение кода и 1 байт под длину;

массив – совокупность элементов одного и того же типа;

список – совокупность элементов одного и того же типа переменной длины;

кортеж – совокупность элементов как одного и того же, так и разных типов;

ПАРАМ – динамический тип данных, основанный на значении некоторого ТМ-параметра, тип его содержимого зависит от текущего значения ТМ-параметра;

МАСАН – массив анализа, специальный тип для передачи цифровых массивов между алгоритмами анализа;

ТЕКСТАН – текст анализа, специальный тип для форматного вывода текстовых протоколов;

ASCII – тип ТМ-параметров, содержащих короткие текстовые значения.

На объекты базовых типов можно создать **указатель** и **ссылку**, содержащие адрес объекта.

При выполнении арифметических операций над аргументами двух разных типов тип результирующего выражения определяется по таблице 2.2.1. Числовые типы расположены в таком порядке, чтобы в результате выполнения над ними арифметических операций не происходило потери значащих разрядов и точности числа. Тип char поставлен как самый широкий для того, чтобы любое преобразование символа возвращало опять символ (обычно к символам применяются только операции сложения и вычитания).

Таблица 2.2.1. Порядок приведения типов для бинарных операций

	byte	short, ushort	int	uint	long	ulong	float, double	char
byte	int	int	int	uint	long	ulong	double	char
short, ushort	int	int	int	uint	long	ulong	double	char
int	int	int	int	uint	long	ulong	double	char
uint	uint	uint	uint	uint	long	ulong	double	char
long	long	long	long	long	long	ulong	double	char
ulong	ulong	ulong	ulong	ulong	ulong	ulong	double	char
float, double	double	double	double	double	double	double	double	char
char	char	char	char	char	char	char	char	char

2.2.2. Инструкции

Код алгоритма на языке анализа ТМИ состоит из инструкций. Тело любой функции, включая главную функцию алгоритма и метод пользовательского типа, представляет из себя последовательность инструкций, заключённую в фигурные скобки.

Инструкции бывают следующих видов:

- блок инструкций в фигурных скобках;
- пустая инструкция, состоящая из символа «;» или символов «{}»;
- объявление локальной переменной;
- управляющая конструкция (описаны в разделе 2.2.4);
- выражение, состоящее из совокупности операторов:

- присваивание;
- арифметическое выражение;
- вызов процедуры или алгоритма;
- выражение инкремента или декремента.

2.2.3. Операторы

Для оперирования числовыми выражениями используются операторы, большинство из которых аналогичны операторам языка C++. Операторы выполняют преобразования над выражениями, которые могут иметь следующие характеристики:

– **r-value выражение,** значения которого можно читать, но не писать, то есть можно использовать только справа от знака присваивания;

– **l-value выражение**, в которое можно присваивать, то есть, использовать слева от знака присваивания;

– **инициализированная переменная**, локальная переменная, в которую было присвоено значение;

– **инициализированный ТМ-параметр**, значение которого поступало в алгоритм или было присвоено внутри алгоритма;

– **инициализированное выражение**, которое имеет определённое значение, полученное путём выполнения операций с инициализированными переменными или ТМ-параметрами. Результат выполнения любой операции, в которой хотя бы один аргумент является **неинициализированным выражением**, также является неинициализированным выражением.

В таблице 2.2.2 приводятся операторы языка анализа ТМИ в порядке понижения приоритета.

Таблица 2.2.2. Приоритет операций языка анализа ТМИ

Оператор	Описание
`a = b`	Присваивание значения r-value выражения `b` в l-value выражение `a`. Тип выражения справа от знака присваивания должен быть приводим к типу слева от знака

`a += b`
`a -= b`
`a *= b`
`a /= b`
`a %= b`
`a >>= b`
`a <<= b`
`a &= b`
`a |= b`
`a ^= b`
`a $= b;`
	Присваивание в выражение `a` с выполнением указанной операции над аргументами `a` и `b`.

`a ? b : c`	Тернарный условный оператор. Если выражение `a` истинно (не равно 0), возвращает `b`, а выражение `c` не исполняется. Иначе возвращает выражение `c`, а выражение `b` не исполняется
`a || b`	Логическое «ИЛИ». Если выражение `a` не равно 0 (false), возвращает true, иначе проверяет выражение `b`. Если оно не равно 0 (false), возвращает true, иначе – false.
`a && b`	Логическое «И». Возвращает true, если оба выражения истины, в противном случае – false. Если первое выражение ложно (равно 0), второе выражение не проверяется.

`a == b`
`a != b`
`a > b`
`a >= b`
`a < b`
`a <= b`
	Сравнение двух числовых выражений на равенство, неравенство и отношение порядка. Возвращает true или false.

`a & b`
`a | b`
`a ^ b`
	Выполнение побитового «И», «ИЛИ», «Исключающего ИЛИ» над целочисленными выражениями.

`a + b` `a – b`	Сложение или вычитание двух числовых выражений. Тип результата определяется по таблице 2.2.1.

`a * b`
`a / b`
`a % b`
	Умножение, деление или поиск остатка от деления двух чисел. Тип результата определяется по таблице 2.2.1. Для целочисленных аргументов деление также будет целочисленным

`a >> b` `a << b`	Сдвиг целого (кодового) значения `a` на `b` бит вправо или влево. Эквивалентно $a \cdot 2^{-b}$ и $a \cdot 2^{b}$ соответственно.
`a $ b`	Склеивание двух битовых полей. `a` станет старшей частью результирующего кода, а `b` – младшей. Эквивалентно $a \cdot 2^{l} + b$, где l – длина кода в `b` битах. Для кодовых параметров длина известна, а для числовых определяется по типу данных. Результатом является значение типа код, содержащее информацию о длине получившегося поля.

`-a`
`!a`
`~a`
	Выполнение унарной операции: числовое отрицание, логическое отрицание, побитовое отрицание.

`a[b]`	Обращение к элементу с индексом `b` массива `a`. Индексирование массивов выполняется с 0. Если `a` является целочисленным типом, то это операция взятия бита с номером `b` (0-й бит является младшим).
`a[b:c]`	Взятие битового поля целочисленного выражения `a`. Здесь `b` – номер старшего бита поля, а `c` – длина поля (рис. 2.2.3). Результатом является значение типа код, содержащее информацию о длине получившегося поля.
`a . b` `a . f()`	Обращение к полю `b` объекта `a` или вызов метода `f`. Допустимо обращаться к полям как стандартных, так и от пользовательских типов.
`a(…)`	Вызов функции `a`.

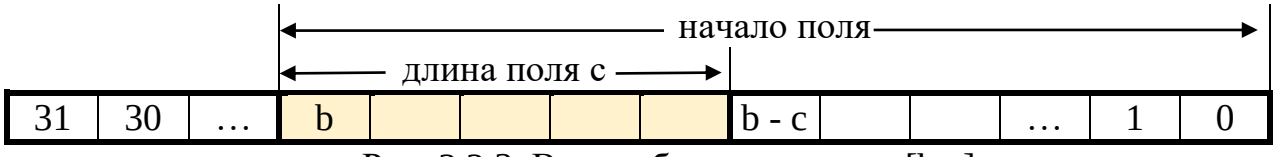


Рис. 2.2.3. Взятие битового поля a[b:c]

Традиции телеметрического обеспечения автоматических и пилотируемых КА, модулей орбитальных станций, ракет-носителей и разгонных блоков позволяют использовать в именах параметров большинство из служебных

символов, обозначающих операторы в языке. Для того, чтобы транслятор подпрограммы на языке анализа мог однозначно определять, где символ является частью идентификатора, а где обозначает операцию, программисту предлагается все подобные операции отделять от своих операндов пробелами.

Операции взятия битового поля и склеивания битовых полей отсутствуют в основных языках программирования и разработаны специально для работы с телеметрической информацией, поскольку являются достаточно распространёнными в такого рода задачах, а традиционный код для них чрезвычайно громоздкий. Предложенный синтаксис является более компактным, понятным и приближенным к документации, в которой описывается структура ТМИ. В разделе 1 показано, что в большинстве телеметрических систем значения ТМ-параметров и служебных полей плотно упакованы в двоичных структурах, и для их извлечения такие операции являются весьма востребованными. В разделе 4.2.1.2 проводится сравнительный анализ кода, написанного с использованием предложенных синтаксических конструкций с традиционным кодом.

2.2.4. Управляющие конструкции

К управляющим конструкциям языка относятся: **if**, **if** – **else**, **if** – **else if …**– **else**, **for, while, do – while**, **switch, codeswitch**, **break**, **continue**, **return**.

Данные операторы являются типовыми для основных языков программирования и в дополнительных пояснениях не нуждаются. В списке отсутствует распространённый в других языках оператор безусловного перехода **goto**, поскольку современный стиль программирования позволяет обходиться без него, а употребление этого оператора нарушает логику исполнения и ухудшает читаемость кода [89].

Рассмотрим подробнее оператор кодового выбора **codeswitch**, разработанный специально для задач обработки телеметрической информации, где часто встречаются задачи обработки пакетов цифровой информации и различных кодовых структур, содержащих плотно упакованную информации переменной структуры. При этом признаки, определяющие состав передающейся информации,

могут быть переменной длины или составными. Опишем синтаксис оператора **codeswitch** в расширенных формах Бэкуса-Наура [65, 67] (см. раздел 2.1.3).

```
Code switch = "codeswitch", Тип формата кода, "(", Выражение ")",
"{",
     Список кодовых элементов,
      [ Альтернатива ELSE ],
"}";
Тип формата кода = "Б" | (* битовый, двоичный *)
     "В" |                (* восьмеричный *)
     "Ш"                  (* шестнадцатеричный *)
Кодовый элемент = Кодовое значение, ":", Инструкция;
Кодовое значение = {Цифра Б}- | {Цифра В}- | {Цифра Ш}-;
Цифра Б = ["0"|"1"|"-"];  (* бинарные *)
Цифра В = ["0".."7"|"-"]; (* восьмеричные *)
Цифра Ш = ["0".."9"|"A".."F"|"-"]; (* шестнадцатеричные *)
Альтернатива ELSE = "else" : Инструкция;
```

В этой конструкции «Выражение» вырабатывает значение кодового типа. Кодовые значения задают возможные альтернативы, которые может принимать выражение. Эти альтернативы записываются в системе счисления, указанной Тип формата кода (если не указано, то десятичная). Особенность конструкции **codeswitch** заключается в возможности задать безразличное состояние каждой цифры с помощью символа «-» в двоичной, восьмиричной и шестнадцатеричной системах. Кроме того, такой подход позволяет задавать альтернативы в той системе счисления, которая используется в документации.

Пример 2.2.1

Рассмотрим структуру заголовка пакета ТМИ КА «Канопус-В» (рис. 2.2.4), размещаемую, начиная с 6 позиции в 8-битовом массиве типа **byte**.

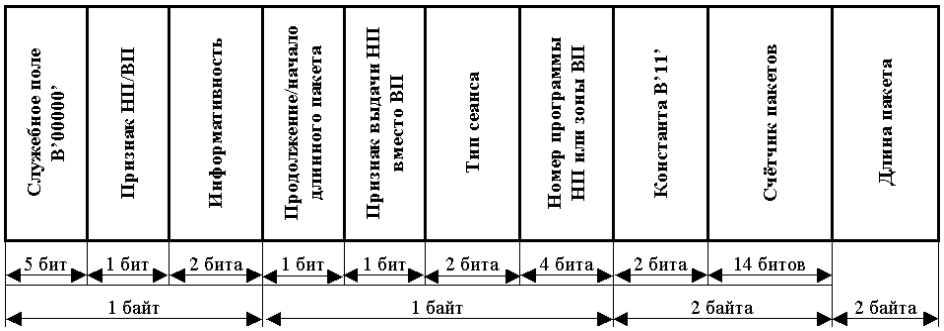


Рис. 2.2.4. Структура заголовка пакета ТМИ

Структуру пакета ТМИ определяют следующие поля заголовка пакета:

1) если «Признак НП/ВП» равен 1, «Продолжение/начало длинного пакета» равно 0, «Признак выдачи НП вместо ВП» равно 0, «Тип сеанса» равен B‘11’, то это пакет технологического режима НП;

2) иначе, если «Признак НП/ВП» равен 1, то это пакет штатного режима НП;

3) иначе, если «Признак НП/ВП» равен 0, «Признак выдачи НП вместо ВП» равен 0, то это пакет режима ВП;

4) другие состояния недопустимы.

Для выполнения подобных проверок потребуется следующий код:

```
@ Заголовок пакета ТМИ начинается с 6 байта
if (ТМИ-Q[6][7:5]==0 && ТМИ-Q[6][2:1]==1 &&
    ТМИ-Q[7][7:1]==0 && ТМИ-Q[7][6:1]==0 && ТМИ-Q[7][5:2]==3)
@ Пакет технологического режима НП
else if (ТМИ-Q[6][7:5] == 0 && ТМИ-Q[6][2:1] == 1)
@ Пакет штатного режима НП
else if (ТМИ-Q[6][7:5] == 0 && ТМИ-Q[6][2:1] == 0 &&
    ТМИ-Q[7][6:1] == 0)
@ Пакет режима ВП
else
@ Неизвестный пакет
```

С использованием конструкции codeswitch указанный пример можно записать существенно проще:

```
codeswitch Б, (ТМИ-Q[6] $ ТМИ-Q[7]) {
     000001--0011----: { … }  @ Пакет технологического режима НП
     000001----------: { … }  @ Пакет штатного режима НП
     000000---0------: { … }  @ Пакет режима ВП
     else: { … }              @ Неизвестный пакет
}
```

Два первых байта заголовка пакета ТМИ объединяются в одно 16-битовое значение (`ТМИ-Q[6] $ ТМИ-Q[7]`) и значение проверяется на совпадение с кодовыми значениями (двоичными константами). Разряды, значения которых несущественны для выбора, помечаются символом ‘-’.

Пример 2.2.2

Порядок задания альтернатив важен, поскольку значение выражения сравнивается с указанными альтернативами последовательно, от первой к последней. Благодаря безразличным состоянием одно и то же значение может удовлетворять нескольким различным альтернативам. При этом исполняться будет

только первая. В качестве примера рассмотрим выбор алгоритма обработки информации обратного канала (ИОК) целевой аппаратуры КА «Канопус-В-ИК»:

```
codeswitch Б, (ВОСПРБВС[0] $ ВОСПРБВС[1])
{
      --00------------: ВыдатьИок(19, 5, ВОСПРБВС,
         "ИОК ТМИ КПИ ЦА", 32, КИКПИЦА, ИКПИЦА-М);
      00--000000000000: ВыдатьИок(20, 5, ВОСПРБВС,
         "ИОК ТМИ Инф БУ", 32, КИИНФБУ, ИИНФБУ-М);
      else: ИОКТМИ . WRITELINE(4,
          "Error: Неизвестный вид ИОК ТМИ 35: {0:X8} ",
          ВОСПРБВС[0] $ ВОСПРБВС[1]);
}
```

В приведённом примере код 0 удовлетворяет и первой, и второй альтернативе, но попадёт он в первую. Код, не подошедший ни под одну альтернативу, попадает в часть `else` (при наличии). При необходимости исполнения нескольких инструкций, они заключаются в фигурные скобки.

Конструкция **codeswitch** может применяться и в тех случаях, когда безразличные состояния не требуются. Благодаря возможности выбрать систему счисления программист может задавать коды в удобной системе счисления, соответствующей документации. В следующем примере приводится код выбора источника управляющей команды, выданной в БРТС «Пирит» РБ «Персей».

```
КОДВЫБОР Ш, РЕГИЗМ[31:16]
{
     06E4: РЕГИЗМ-ПРОТ . Write(" (БЦВК -> РТС)");
     06E5: РЕГИЗМ-ПРОТ . Write(" (РТС осн.)");
     06E6: РЕГИЗМ-ПРОТ . Write(" (РТС рез.)");
}
```

2.2.5. Функции

Как и в других языках программирования, функция является именованным фрагментом кода, к которому можно обратиться из других мест программы. Объявление функций алгоритма анализа осуществляется до главной функции алгоритма, кроме того, каждая функция должна быть объявлена до её первого вызова. Это позволит выполнять трансляцию кода за один проход, что существенно упрощает транслятор. Синтаксис объявления функции следующий.

```
Объявление функции = [Тип возврата],
     Имя функции, "(", Аргумент, {"," Аргумент}, ")",
     Инструкция;
Тип возврата = Тип | Кортеж возврата;
```

```
Кортеж возврата = "(", Элемент кортежа, {"," Элемент кортежа},")";
Элемент кортежа = Тип, [Имя элемента];
Аргумент = Тип, ["&"], Имя аргумента, ["[]"];
```

Перед именем функции идёт необязательный тип возврата. Если он не указан, функция не возвращает значений. Кортежи подробно рассмотрены в разделе 2.2.6. Аргументами функции могут быть числовые значения, массивы (после имени аргумента указываются квадратные скобки), ТМ-параметры и объекты пользовательского типа. Числовые параметры можно передавать по ссылке. Для этого перед именем аргумента указывается амперсанд, как в языке C++. Функции допускают рекурсивные вызовы. Исполнение функции заканчивается на последней её инструкции. Для досрочного выхода из функции используется инструкция **return**. Если функция предполагает возврат результата, то после инструкции **return** должно идти выражение, вырабатывающее значение требуемого типа, или кортеж.

В языке предусмотрено две специальные функции:

– ИНИЦИАЛИЗАЦИЯ – исполняется после загрузки программы обработки ТМИ, используется для инициализации алгоритма, формирования некоторых управляющих значений, вывода статических текстовых сообщений (заголовков текстовых протоколов), загрузки результатов предыдущей работы и т. п.;

– ФИНАЛИЗАЦИЯ – исполняется в конце работы программы обработки ТМИ перед её завершением, используется для сохранения результатов работы, вычисления некоторых статистических и итоговых данных, формирования значений параметров, говорящих об окончании сеанса обработки ТМИ и т. п.

2.2.6. Кортежи

Кортеж – упорядоченный набор данных фиксированной длины. В отличие от массива, элементы кортежа могут быть разного типа. Тип и количество элементов кортежа определяется типом конкретного кортежа. В современных языках программирования кортежи только набирают популярность. Они есть в языке Python, в языке C# кортежи появились с версии 7.0 языка в 2017 году. В таких языках как C++ или Java кортежи реализуются при помощи классов, но на уровне языка не поддерживаются [45].

В большинстве случаев, когда функция должна вернуть упорядоченный набор данных, весь этот набор в целом имеет некоторый смысл, и под него объявляют отдельный тип данных. В задачах автоматизированного анализа ТМИ, как правило, решают не очень большие, конкретные задачи, в которых будет накладно создавать отдельные типы данных под каждый подобный набор данных, зато использование кортежей будет оптимальным.

Введём понятия трёх разных видов кортежей.

1. **Кортеж возврата** описывает тип возврата функции. Содержит типы элементов и необязательные имена. Имена служат трём целям: во-первых, подсказывают читающему код, какое значение на какой позиции передаётся, во-вторых, позволяют обратиться к одному конкретному элементу кортежа при необходимости, в-третьих, делают более осмысленными сообщения транслятора о возможных ошибках. Синтаксис кортежа возврата следующий:

```
Кортеж возврата = "(",
     Элемент возврата, {"," Элемент возврата},
     ")";
Элемент возврата = Тип, [ Имя элемента ];
```

2. **Кортеж выражения** представляет собой упорядоченный набор конкретных значений, вырабатываемых выражениями. Используется для возврата из функции или выполнения над ним некоторых операций (если все выражения l-value). Типы значений вырабатываемых выражений, должны быть совместимы по присваиванию с элементами кортежа, в который передаётся кортеж возврата, либо с аргументами функции, в которую передаётся кортеж. Синтаксис кортежа выражения следующий:

```
Кортеж выражения = "(", Выражение, {",", Выражение}, ")";
```

3. **Кортеж объявления** представляет из себя упорядоченный набор объявлений переменных или l-value выражений, в который присваивается другой кортеж, например, возвращённый функцией. Отметим, что любая позиция кортежа объявления может быть пустой. В этом случае данный элемент кортежа будет утерян. Синтаксис кортежа переменных следующий:

```
Кортеж объявления = "(", Объявление, {",", Объявление}, ")";
Объявление = [ Тип, Имя переменной |
               LValue выражение ];
```

Рассмотрим различные возможности по использованию кортежей на конкретных примерах.

Пример 2.2.3

```
01 (double q0, double q1, double q2, double q3)
ВекторВКватернион(double x, double y, double z) {
02   if (x == 0 && y == 0 && z == 0)
03        return (1, 0, 0, 0);
04   else
05        return (0, x, y, z);
06 }
```

В примере 2.2.3 приведена функция, преобразующая вектор в кватернион. Выражение `(double q0, double q1, double q2, double q3)` является кортежем возврата и описывает тип возвращаемых функцией значений – четырёхкомпонентный кортеж вещественных значений. После оператора `return` следует кортеж выражения, формирующей результат работы функции.

Пример 2.2.4

```
@ Функция нормирования вектора, параметры передаются по ссылке
Нормировать3(double &x, double &y, double &z) {
    double n = sqrt(x*x + y*y + z*z);@ n – локальная переменная
    if (n > 0)
      (x, y, z) /= n;                @ деление кортежа на скаляр
}
```

В примере 2.2.4 кортеж выражения `(x, y, z)` используется для выполнения групповой операции: деления каждой компоненты вектора на его норму. В данном случае использование кортежа сокращает запись и делает операцию визуально целостной. Приведённое выражение эквивалентно следующей записи:

```
x /= n;
y /= n;
z /= n;
```

Разница может заключаться лишь в том, что, если бы справа от знака присваивания использовалось вычисляемое выражение вместо одной переменной `n`, то в случае использовании кортежа оно вычислялось бы однократно, а в классической записи приходится либо вычислять выражение для каждого компонента, либо сохранять результаты вычислений в промежуточную локальную переменную. В следующем фрагменте кода это обстоятельство существенно:

```
(СВ0-512-1, СВ1-512-1, СВ2-512-1, БИТ-512-1) +=
```

```
СверкаСЭталоном(КАДР512[i][9:10], etalon1[i]);
```

Пример 2.2.5

```
(double q0, double q1, double q2, double q3)
ПроизведениеКватернионов(double q10, double q11, double q12, double
q13, double q20, double q21, double q22, double q23) {…}

(double q0, double q1, double q2, double q3)
СопряжённыйКватернион(double, double, double, double) {…}

(double q0, double q1, double q2, double q3)
ПроизведениеВектораНаКватернион(double x, double y, double z, double
q0, double q1, double q2, double q3)
{
     return ПроизведениеКватернионов(
          ПроизведениеКватернионов(q0, q1, q2, q3,
               ВекторВКватернион(x, y, z)),
          СопряжённыйКватернион(q0, q1, q2, q3));
}
```

В примере 2.2.5 демонстрируется передача кортежа в функцию. Функция `ВекторВКватернион` из примера 2.2.3 возвращает четырёхкомпонентный кортеж, который в виде четырёх отдельных значений передаётся в функцию `ПроизведениеКватернионов`. То есть, при вызове этой функции ей передаётся 4 вещественных числа и результат вызова функции, возвращающей 4-компонентный кортеж, но в саму функцию поступает 8 вещественных чисел. Аналогично при первом вызове функции `ПроизведениеКватернионов` ей передаётся 8 вещественных чисел, которые образуются из двух 4-компонентных кортежей.

Пример 2.2.6

```
(, x1, y1, z1) = ПроизведениеВектораНаКватернион(
  x1, y1, z1, LL0, -LL1, -LL2, -LL3);
```

В примере 2.2.6 демонстрируется использование результата вызова функции, возвращающей кватернион в форме кортежа. При этом, первый элемент возвращаемого кортежа, являющийся первой компонентой кватерниона, не используется, что задаётся пустым выражением между открывающей круглой скобкой и запятой в кортеже переменных. Остальные элементы кортежа присваиваются в переменные `x1, y1, z1`.

Пример 2.2.7

```
(ushort, ushort, uint, bool) РазобратьТМЦИ(ushort data[]){
   @ Определяются апид, №тмци, время
   @ Определяется контрольная сумма csArray, содержащаяся в ТМЦИ
```

```
    @ Вычисляется контрольная сумма cs ТМЦИ
    return (апид, №тмци, время, cs == csArray);
}

(апид, №тмци, время, csIsGood) = РазобратьТМЦИ(data);
```

В примере 2.2.7 демонстрируется кортеж разнотипных элементов на примере функции разбора заголовка телеметрического массива цифровой информации (ТМЦИ).

Пример 2.2.8

```
(number, data[1], info.name) = F1();
(тмци[i] . апид, тмци[i] . №тмци, тмци[i] . время) = F2();
тмци[i] . (апид, №тмци, время) = F2();
```

Пример демонстрирует, что в кортеже переменных могут использоваться наряду с переменными произвольные L-value выражения. В примере 2.2.8 заданы гипотетические выражения, в которых результаты вызова функций, возвращающих кортежи, присваиваются в различные выражения: переменные, элементы массива, поля объектов. Здесь третья строчка является более короткой записью второй.

Пример 2.2.9

В выражениях с кортежами допустимо многократно использовать одну и ту же переменную:

```
(index, array[index++], array[index++]) = F5(index);
```

При этом все выражения в кортеже вычисляются в порядке их описания до вызова функции и выполнения операции присваивания. Если в начале **index**=0, тогда кортеж образуется из ссылок на объекты **index**, **array[0]**, **array[1]**, а функция **F5()** будет вызвана с аргументом 2.

Пример 2.2.10

Кортежи могут использоваться в групповых операциях. Бинарная операция, у которой один аргумент является кортежем, а другой – скаляром, преобразуется в набор бинарных операций над каждым элементом кортежа и указанным скаляром.

```
(i, j, k, m[4]) = 0;
тмци[i].(апид, №тмци, время) = 0;
sum += (a, b, c);    @ эквивалентно sum += a; sum += b; sum += c;
(a, b) >>= 4;
(i, j)++;
(m1, m2)[i] = GetValue();  @ функция возвращает скаляр
```

```
(p, q, r) = i++;
```

Выполнение приведённых в примере операций осуществляется естественным образом. В последних двух строчках выражения справа от знака «=» вычисляются один раз и последовательно присваиваются каждому элементу кортежа слева от знака присваивания.

Кортежи применимы к следующим бинарным операторам: **=**, **+=**, **-=**, **/=**, ***=**, **%=**, **>>=**, **<<=**, **&=**, **|=**, **^=**, **$=**, **. , []**, и унарным операторам **++**, **--**. В общем случае применение бинарной операции к кортежу преобразуется транслятором в последовательное применение бинарной операции к каждому его элементу.

Бинарные операции присваивания допустимо применять к парам кортежей. При этом кортежи слева и справа от знака присваивания должны быть одного размера, а элементы на соответствующих позициях должны быть совместимы по присваиванию:

```
(a, b) += (c, d);
```

2.2.7. Работа с текстами

Одной из ключевых задач автоматизированного анализа ТМИ является формирование всевозможных текстовых протоколов. Передача текстовых данных в ТМИВК осуществляется в значениях ТМ-параметров. Каждая строка текста передаётся отдельным значением. Предусмотрено два текстовых типа данных:

– простое текстовое значение (до 16 символов), выводимое наряду с любыми другими значениями в позицию табличного формуляра отображения или в текстовый документ;

– тип текста анализа, который подразделяется на 3 основных вида по способу отображения:

- бегущая строка – построчно выводится в текстовую область формуляра отображения; дойдя до последней строки, продолжает вывод с первой после заголовка строки;
- протокол – последовательность текстовых строк, выводимая в текстовую область с возможностью прокрутки; благодаря

развитию средств отображения в настоящее время вытесняет бегущие строки;

- документ – вывод фрагментов текстовых сообщений осуществляется на текстовую область по заданным в каждом сообщении координатам; используется для формирования текстового документа определённой структуры и частичного обновления информации на нём в ходе обработки ТМИ.

В языке анализа ТМИ для объявления параметра с типом простых текстовых сообщений используется тип `ASCII`, а для текстов анализа – `ТЕКСТАН`. Тексты анализа перед началом использования необходимо инициализировать. Для этого, как правило в функции `ИНИЦИАЛИЗАЦИЯ` вызывается метод `ИНИТ` со следующим синтаксисом:

```
Инициализация ТЕКСТАН = Имя параметра, ". ИНИТ(",
     Номер кадра анализа, ",", (* целое число *)
     Ширина, "x", [Первая строка, "-"], Высота, ",",
     ("ПРОТОКОЛ" | "БЕГУЩАЯСТРОКА" | "ДОКУМЕНТ"),
     ")"
```

`Номер кадра анализа` указывает, на какой условный кадр анализа комплекса индивидуальных средств отображения ЦУП будет выводиться текст. На один кадр может осуществляться выдача нескольких параметров или наоборот – один параметр может выводить данные на разные кадры. Для этого значения параметра формируются в разных алгоритмах, либо номер кадра анализа изменяется в процессе работы. Когда вывод на конкретный кадр анализа не требуется, задают 0.

Ширина и высота задают размеры текстовой области, в которую будет выводиться протокол. Лишние символы обрезаются, а по достижении последней строки осуществляется переход на первую. Если `ПерваяСтрока` не задана, переход осуществляется на строку № 0.

В текстах анализа можно использовать управляющие последовательности для задания цвета и начертания шрифта [45]. Специфические свойства и методы текстов анализа приведены в таблице 2.2.3.

Таблица 2.2.3. Свойства и методы тектов анализа

№	Имя	Описание
1	ДЛИНА	Свойство возвращает текущую длину текста (текущей строки)
2	ВЫДАТЬ	Метод выдаёт текст анализа на отображение. После этого в переменной текст стирается (следующий текст будет формироваться с начала строки), осуществляется переход на следующую строку
3	ИНИТ	Инициализирует текст анализа (метод описан выше)
4	ЦВЕТ	Задаёт цвет текста текущей и последующих строк
5	КАДР	Задаёт номер кадра анализа для текущей и последующих строк, на котором будут отображаться строки
6	ПЕРЕНО-СИТЬ-СТРОКИ	Задаёт, следует ли переносить текст, если достигнут конец строки. Перенос осуществляется с учётом содержимого текста. Место для разрыва строки выбирается следующим образом: 1. ищется пробельный символ среди последних; 2. если пробелов нет, ищется знак пунктуации; 3. иначе перенос выполняется с заглавной буквы, которая следует за строчной.
7	Write	Дописать текст в текущую строку.
8	WriteLine	Дописать текст в текущую строку и выдать на дальнейшую обработку.

Базовый синтаксис использования методов `Write`, `WriteLine` позволяет вывести константную текстовую строчку.

Пример 2.2.11

```
01   ПРОТ . Write("     Контроль выдачи цифровых массивов ");
02   if (board == 1)
03        ПРОТ . WriteLine("МИМ1");
04   else
05        ПРОТ . WriteLine("МИМ2");
06   ПРОТ . WriteLine();
```

В строке 01 примера в ТМ-параметр `ПРОТ` печатается текстовая строка. Этот текст сохраняется в буфере параметра и никуда не выдаётся. Далее, в зависимости от значения переменной `board` в протокол дописывается либо текст «МИМ1», либо «МИМ2», и полученный результат выдаётся из алгоритма. В строке 06 в текстовый протокол `ПРОТ` добавляется пустая строка и сразу выдаётся.

Дописывание текста методами `Write`, `WriteLine`, а также отдельной функцией `ФорматСтроки` поддерживает средства форматного вывода, построенные на основе форматного вывода языка C# и платформы .NET [69]. После текстовой

строки в метод можно передать произвольное количество дополнительных аргументов. Чтобы вставить значение переданного аргумента в строку, к нему нужно обратиться по номеру, начиная с 0, указанному в фигурных скобках.

Пример 2.2.12

```
ИОК1 . WriteLine("В СТИ важное сообщение {0}", code & 0xff);
```

В примере значение выражения `code & 0xff` будет вставлено в текст на месте {0}.

В большинстве случаев требуется управлять форматом вывода числовых значений. Для этого в строке формата предусмотрено множество управляющих конструкций. Общий синтаксис выражения обращения к аргументу следующий:

```
Обращение к аргументу = "{",
     Номер аргумента, (* нумерация аргументов с 0 *)
     [ ",", ["-"], Ширина значения, ["-"] ],
     [ ":" (Код формата | Строка формата) ],
     "}";
Код формата =
     Кодовый формат |
     Вещественный формат |
     Временной формат;
Кодовый формат = ("X"|"H"|"x"|"h"|"O"|"o"|"B"|"b"|"T"),
     [Количество символов];
Вещественный формат = ("e"|"E"|"F"|"f"),
     [["-"|"+"] Количество значащих цифр];
Временной формат = "t" | "D" | "d";
Строка формата =
     ["-"|"+"], (* знак *)
     {"#"}, {"0"}, (* целая часть *)
     [[","|"."], {"0"}, {"#"}]; (* дробная часть *)
```

Приведённый синтаксис требует подробных пояснений. Выражение в фигурных скобках может состоять из 1, 2 или 3 частей. Сначала указывается порядковый номер аргумента, затем идут две необязательные части. После запятой можно указать ширину поля, в которое будет выводиться значение. Если перед шириной указан знак «-», то выравнивание значения осуществляется по правому краю поля. Если знаки «-» указаны и слева, и справа от ширины, то выравнивание осуществляется по центру поля.

После двоеточия указывается либо код формата, либо строка форматирования чисел. Коды форматов приведены в таблице 2.2.4.

Таблица 2.2.4. Коды форматов для текстов анализа

№	Код	Описание	Примеры
1	X H x h	Выводит целое число в шестнадцатеричном формате. Заглавная буква указывает на необходимость вывода заглавных шестнадцатеричных цифр (A-F), а строчная – строчных (a-f). После кода формата можно указать минимальное количество выводимых цифр.	"{0:X4}" – вывод значений как шестнадцатеричных с 4 знаками
2	O o	Выводит целое число в восьмеричном формате. После кода формата можно указать минимальное количество выводимых цифр.	" {0:O6}" – вывод значений в восьмеричном формате с 6 знаками
3	B b	Выводит целое число в двоичном (бинарном) формате. После кода формата можно указать минимальное количество выводимых цифр.	"{0:B16}" – вывод значений в двоичном формате с 16 знаками
4	t	Выводит время в формате чч:мм:сс. Если значение целое, ожидается, что оно содержит код времени как число миллисекунд с полуночи. Если значение вещественное – как число секунд. Перед отрицательным временем выводится знак «-».	. Write("{0:t}", 57632) Выводит 16:00:32
5	T	Выводит время в формате чч:мм:сс.ммм. Если значение целое, ожидается, что оно содержит код времени как число миллисекунд с полуночи. Если значение вещественное – как число секунд. После кода формата можно указать количество выводимых знаков после точки (долей секунд). По умолчанию – 3. Перед отрицательным временем выводится знак «-».	. Write("{0:T}", 4200.4) Выводит 01:10:00.400
6	E e	Выводит значение v в экспоненциальном формате $\pm м.ммм E \pm пп$, где $v = м.ммм \cdot 10^{пп}$, $1 \le м.ммм < 10$. После кода формата может быть указано количество выводимых значащих цифр мантиссы.	Write("{0:e3}", 12.88) Выводит 1.29e1
7	F f	Выводит значение как вещественное с фиксированным количеством значащих цифр. После кода формата указывается количество значащих цифр.	Write("{0:f3}", 12.03) Выводит 12.0
8	d D	Выводит дату, закодированную как число дней, прошедших с 06.01.1980 г. (принятый в большинстве КА формат) в формате дд.мм.гг	ФорматСтроки ("{0:d}", date + 7300) Выводит 01.01.2000

Для форматов, позволяющих указать количество значащих цифр, перед этим числом можно указать знак «+» или «-». Если указать знак «+», то он будет выводиться для положительных чисел; если «-», то перед положительными числами будет выводиться « », то есть будет зарезервировано место под знак. Это позволяет выводить числа в аккуратном табличном виде.

Строка формата позволяет гибко настроить формат вывода целых и вещественных чисел. Схема задания формата печати приведена на рис. 2.2.5.

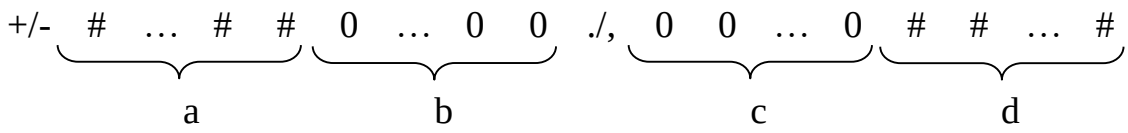


Рис. 2.2.5. Схема строки формата чисел

Здесь $a + b > 0$.

Символы 0 обозначают позиции, на которых обязательно должна быть выведена цифра, по крайней мере 0. Символы # обозначают позиции, отведённые под цифры значения, если они являются значимыми. В средней части строки формата можно указать символ «.» или «,» – этот символ будет использоваться в качестве разделителя дробной части. Если в начале указать символ «-», то всегда будет отводиться позиция под знак числа. Если указать знак «+», то положительные числа будут выводиться со знаком «+». Поясним сказанное на примерах в таблице 2.2.5. В примерах символом « ·» для наглядности отмечены пробельные символы.

Таблица 2.2.5. Примеры форматного вывода чисел

№	Код	Результат
1	`Write("{0:####0.0}", 480)`	··480.0
2	`Write("{0:###00.0}", 0)`	···00.0
3	`Write("{0:###0,0}", 1.125)`	···1,1
4	`Write("{0:###0.0#}", 1.125)`	···1.13
5	`Write("{0:###0.0#}", 1.3)`	···1.3·
6	`Write("{0:###0}", 16386)`	16386
7	`Write("{0:-###0}", 16386)`	·16386
8	`Write("{0:-###0}", -16386)`	-16386
9	`Write("{0:+###0.#}", 16386)`	+16386··
10	`Write("{0:0.0}", -0.04)`	0.0
11	`Write("{0:0.0##,6}", 100.045)`	100.05

В примере 10 в результате округления получается значение «-0.0», которое выглядит неестественно, поэтому оно заменяется на «0.0». В примере 11 ширина

поля, равная 6, имеет приоритет над необязательными позициями «#». Поэтому число, запись которого в указанном формате требует 7 позиций, должна быть сокращена. Целая часть сокращаться не может, поэтому отбрасываются последние цифры дробной части и результат округляется должным образом. Кроме того, в примере 11 можно видеть, что ширина поля задана после строки формата. Это допустимо, поскольку строка формата содержит символ «.», и программа понимает, что запятая отделяет длину поля. В примере 3 запятая трактуется как разделитель дробной части. Чтобы избежать неоднозначности, ширину поля рекомендуется указывать до строки формата.

2.2.8. Входные и выходные параметры

Основным средством взаимодействия алгоритмов анализа с другими алгоритмами обработки ТМИ и программой обработки ТМИ в целом являются значения ТМ-параметров. Каждому ТМ-параметру сопоставлено некоторое имя (идентификатор). Правила задания имён ТМ-параметров описаны в разделе 1.3.1.

Поскольку ТМ-параметры сами по себе не имеют определённого типа данных, более того, в процессе обработки они могут чередовать значения различных типов, а любая программа на алгоритмическом языке со строгой типизацией данных требует чёткого указания типа всех используемых параметров, при объявлении алгоритма анализа необходимо указывать тип всех входных и выходных ТМ-параметров. Единственным исключением являются выходные параметры целого или вещественного типа. Такой тип можно не указывать, он будет определён автоматически после присваивания в параметр значения. Для задания входных и выходных параметров используются следующие конструкции:

```
ВХПАР(СписокВходныхПараметров)
ВЫХПАР(СписокВыходныхПараметров)
ВХВЫХПАР(СписокВходныхИВыходныхПараметров)
```

Используя указанную информацию, транслятор сможет настроить подачу требуемых параметров в алгоритм, а их значения привести к типам, используемым в алгоритме анализа.

Общий синтаксис описания ТМ-параметра следующий:

```
Входной или выходной параметр =
   [ "АРГ" ],
   [ "СТРУКТ" | Тип ],
   ( Имя | "ПРИШЕДШИЙ" ) [ "[]" ],
   [ "AS", Имя2 ];
```

Центральной частью объявления является имя параметра. Такой параметр должен быть объявлен в борту. Вместо имени можно указать ключевое слово ПРИШЕДШИЙ в том случае, если это единственный параметр алгоритма. Тогда алгоритм можно будет приписывать нескольким схожим параметрам, не меняя имени внутри алгоритма, или использовать этот алгоритм в качестве предварительного описания.

```
A1, A2, A3 = АНАЛИЗ(ВХВЫХПАР(double ПРИШЕДШИЙ) { … });
```

Ключевое слово СТРУКТ перед именем параметра указывает, что этот параметр является структурным, то есть на уровне описания борта в составе этого параметра были указаны другие параметры. Включение такого параметра в состав входных или выходных параметров алгоритма включает и все дочерние параметры. Указанный параметр получает тип «массив параметров». К его дочерним параметрам можно обращаться как по имени, так и индексируя структурный параметр.

Пример 2.2.13

Например, параметр ИММАС содержит объявления всех цифровых массивов многофункционального лабораторного модуля (МЛМ) «Наука»:

```
ИММАС(ИМ00МЛМ , ИМ01МЛМ , ИМ02МЛМ , ИМ03МЛМ , ИМ04МЛМ , ИМ05МЛМ,
      ИМ06МЛМ , ИМ07МЛМ , ИМ08МЛМ , ИМ09МЛМ , ИМ10МЛМ , … )
```

В алгоритме обработки цифровых массивов этот параметр подключается как структура и используется как массив:

```
ВУКС = АНАЛИЗ(
     ВХПАР(ushort ВУКС[])
     ВЫХПАР(СТРУКТ ИММАС)
…
01   ПАРАМ им = ИММАС[nmas];
02   им . ИНИТ(ushort[len]);
03   им . ВРЕМЯ = времяМассива;
04   им . ФОРМАТ(Ш16);
05   КопироватьМассив(ВУКС, ind + 1, им, 0, len);
06   им . ВЫДАТЬ();
…
```

Дадим пояснения по коду. В строке 01 идёт обращение к параметру в составе структуры ИММАС на позиции nmas (например, ИМ09МЛМ для nmas=9). Ссылка на указанный параметр помещается в переменную им. В строке 02 параметр инициализируется как массив 16-разрядных элементов типа ushort и размера len. В строке 04 параметру приписывается время, а в строке 04 – формат значений. В строке 05 в выделенный массив копируются данные. В строке 06 значение сформированного ТМ-параметра выдаётся на дальнейшую обработку. Оно поступит в менеджер программы обработки ТМИ, который подаст его на вход алгоритма, который приписан для обработки значений данного параметра.

Указание квадратных скобок «[]» после имени входного или выходного параметра говорит, что в этом параметре передаются массивы значений. Допустимо использовать массивы только простейших типов: byte, ushort, int, uint, ulong, float, double. В предыдущем примере ВУКС является массивом 16-разрядных слов типа ushort.

Использование ключевого слова АРГ позволяет задавать алгоритм анализа в качестве предварительного описания алгоритма. Параметры, помеченные атрибутом АРГ необходимо будет задать при каждом использовании алгоритма, указав конкретные параметры. Если такой параметр в алгоритме один, то его можно не указывать.

Пример 2.2.14

```
ТХ-ДАТА-С-ВРЕМЕНЕМ = АНАЛИЗ(ВХПАР(АРГ uint код32)
     ВЫХПАР(АРГ ASCII код32 AS текст)
{
     текст = ФорматСтроки("{0:d}",
          (код32 + 3 * 3600 - GetLeapSeconds()) / 86400);
     текст . ВРЕМЯ =
          (код32 + 3 * 3600 - GetLeapSeconds()) % 86400 * 1000;
});
И-ВРЕМЯ = ТХ-ДАТА-С-ВРЕМЕНЕМ;
```

В приведённом примере алгоритм принимает параметр, содержащий код даты и времени в формате GPS (количество секунд, прошедшее с 00:00:00 06.01.1980 GMT). Значение переводится в декретное московское время (ДМВ) [14] путём прибавления 3 часов (разница часовых поясов) и вычитания секунд координации времени (leap seconds) [16]. В значение параметра записывается

строка текста, содержащая полученную дату (частное от деления на 86400 – количество секунд в сутках), время значения становится равным полученному времени в миллисекундах (остаток от деления на 86400). Параметр объявлен с типом `ASCII` (описан в разделе 2.2.7). Таким образом, дата и время будут выведены на формулярах индивидуальных средств отображения в формате <дд.мм.гг чч:мм:сс>.

Отдельного внимания заслуживает ключевое слово `AS`. В примере задан один входной и один выходной параметр: `код32` типа `uint` и `код32` типа `ASCII`. Это значит, что алгоритм оперирует одним ТМ-параметром (параметр `И-ВРЕМЯ` в примере), но он должен иметь разные типы входных и выходных значений. Чтобы этого добиться и строго различать эти значения в коде программы, второму вхождению ТМ-параметра назначено альтернативное имя `текст`, через которое к нему и обращаются. Допустимо также один параметр несколько раз объявить выходным с разными типами значений, используя ключевое слово `AS`.

2.2.9. Режимы запуска

Большое разнообразие задач обработки значений телеметрических параметров требует различных режимов запуска алгоритма анализа. Задать режим запуска можно при помощи специальной директивы в начале алгоритма со следующем синтаксисом.

```
Режим запуска = "ЗАПУСК(",
      ("КАЖДЫЙ" | "ВСЕ" | "ПАЧКА"), ",",
      ("НП" | "ВП" | "НП+ВП"),
      { ", ИНСТРУКЦИЙ:", Целое },
      { ", ПАРАМЕТРОВ:", Целое },
      ")";
```

Первый параметр директивы запуска указывает на порядок запуска алгоритма после получения значений входных параметров:

– **КАЖДЫЙ** – указывает на необходимость запуска алгоритма для каждого входного значения каждого параметра. Режим является самым распространённым и наиболее гибким, программист сам управляет порядком обработки значений. Ключевое слово **ПРИШЕДШИЙ** позволяет узнать, какой входной ТМ-параметр инициировал запуск алгоритма. Кроме того, можно

обращаться к свойству ПРИШЕДШИЙ, имеющемуся у всех входных ТМ-параметров – это свойство вернёт true, если данный параметр инициировал запуск алгоритма.

– **ВСЕ** – указывает на поведение, аналогичное базовым алгоритмам сборки, используемым в комплексе обработки ТМИ в ТМИВК, таким как ФОРМУЛА и СБК. А именно, алгоритм дожидается получения значений всех входных параметров, после чего запускает исполнение. Последующий запуск исполнения выполняется при получении новых значений параметров, но не каждого, а каждой группы значений. То есть, получив новое значение одного параметра, алгоритм запускает таймер на 200 миллисекунд, в течение которых ожидает возможные поступления других входных параметров. На рис. 2.2.6 иллюстрируется схема работы алгоритма; словом «зап.» обозначен запуск алгоритма. Такой режим используется для обработки значений параметров, поступающих группой в составе телеметрического кадра или цифрового массива (нескольких синхронно выдающихся массивов).

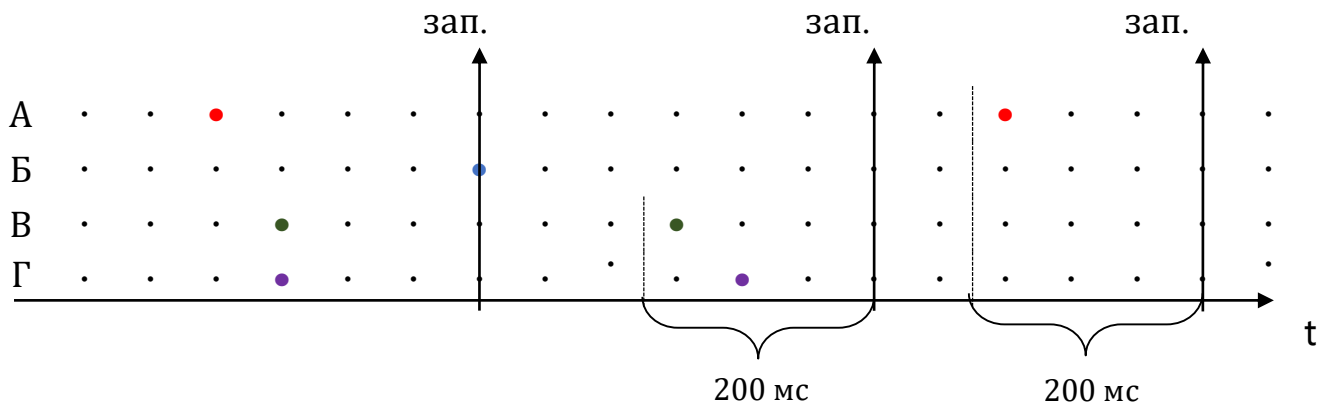


Рис. 2.2.6. Схема сбора значений нескольких параметров

– **ПАЧКА** – указывает на запуск алгоритма при получении значений всех входных параметров по одному разу. Режим аналогичен описанному выше, за исключением того, что проверяет, что все значения входных параметров имеют строго одно и то же время, то есть сформированы совместно, не ожидает поступления дополнительных значений после получения значения последнего параметра. Режим применяется для случав, когда параметры гарантированно выдаются «пачкой», без сжатия, когда недопустимо обрабатывать совместно

значения из разных пачек. Например, такой режим применяется для обработки компонентов вектора состояния КА.

Пример 2.2.15

```
АНАЛИЗ(
ЗАПУСК(ПАЧКА, НП+ВП)
ВХПАР(double 1KMB1M2, double 2KMB1M2, double 3KMB1M2, double 4KMB1M2,
double 5KMB1M2, double 6KMB1M2, double 7KMB1M2)
ВЫХПАР(ТЕКСТАН KMB1M2)
…
     KMB1M2 . WRITE("{0:0.000} {1:0.000} {2:0.000} {3:0.000} {4:0.000}
{5:0.000} {6:0.000}",
          1KMB1M2,  2KMB1M2,  3KMB1M2,  4KMB1M2,  5KMB1M2,  6KMB1M2,
7KMB1M2);
```

В примере приводится фрагмент алгоритма, который получает значения 7 температурных параметров Малого исследовательского модуля (МИМ2) «Поиск» и выводит их в табличном виде в текстовый протокол KMB1M2. Эти параметры поступают в составе одного массива, одновременно, с одним временем, следовательно, условие срабатывания режима «ПАЧКА» выполняется.

Параметры НП, ВП, НП+ВП указывают, в каких режимах передачи ТМИ будет запускаться алгоритм. Например, если указать режим НП, то значения параметров режима ВП на вход подаваться не будут.

После ключевого слова ИНСТРУКЦИЙ можно указать максимальное количество инструкций байт-кода, которое алгоритм может исполнить за один вызов. Указанный контроль позволяет защищать алгоритм от зацикливания. По умолчанию разрешается исполнять 1000000 инструкций за вызов. Для большинства алгоритмов этого достаточно. Параметр может быть увеличен для алгоритмов, выполняющих баллистические расчёты, алгоритмы искусственного интеллекта или накапливающих длительное время большие объёмы данных и обрабатывающих их за один вызов (например, в конце обработки).

После ключевого слова ПАРАМЕТРОВ можно указать максимальное количество параметров, которые алгоритм может сформировать за один запуск. Указанный контроль позволяет защищать алгоритм от зацикливания. По умолчанию разрешается формировать 1000 значений за вызов. Это число можно увеличить для алгоритмов, накапливающих большое количество данных, а затем

обрабатывающих их за один раз и, например, формирующих сразу большое количество строчек текстового протокола.

2.2.10. Пустые значения

Оперирование значениями телеметрических параметров требует уделять особое внимание пустым значениям или отсутствию значения. В языке анализа ТМИ любой ТМ-параметр, любая переменная, результат выполнения любого выражения, поле класса, элемент кортежа – могут быть пустыми. Переменная или параметр будут пустыми, пока в них ничего не присвоено. Входной параметр становится непустым, когда ему присваивается значение. Результат арифметического выражения с пустым аргументом также является пустым значением. Кроме того, пустым может оказаться результат вызова внешнего алгоритма.

Для работы с пустыми значениями у всех выражений доступно два метода:

Метод	Описание
ПУСТОЙ()	Метод возвращает true (истина), если значение выражения пустое. Иначе – false (ложь).
ОПУСТОШИТЬ()	Метод опустошает значение переменной. Применяется в алгоритмах анализа, например, для того, чтобы остановить дальнейшую обработку нежелательного значения параметра. Если выходной параметр имел значение, а потом его опустошили, на выход он подаваться не будет. Элемент числового массива опустошить нельзя.

Другим способом опустошить значение переменной или параметра может быть присвоение в них специального значения **null**. Это же значение можно вернуть из функции.

Особым образом пустые значения трактуются в условных операторах. Любое пустое значение считается эквивалентным **false**.

Предположим, что алгоритм анализа принимает на вход два ТМ-параметра *A* и *B*. Исполнение алгоритма запускается сразу же после получения значения первого параметра. Допустим, первым приходит значение параметра *A*, тогда параметр *B* пока содержит пустое значение, и следующий условный оператор выполнен не будет:

```
if (A > 0 && B > 0) { … }
```

Действительно, `(B > 0)` будет пустым. Тогда `(A > 0 && null)` также будет пустым. А пустое значение трактуется как **false**.

Пример 2.2.16

```
ВХВЫХПАР(ushort КАДР[])
…
if (КАДР . ДЛИНА < 256 || КАДР . АТРИБУТ == 8)
{
     КАДР . ОПУСТОШИТЬ();
     return;
}
```

В примере алгоритм выполняет предварительную оценку поступающих на вход обработки телеметрических кадров. Кадры, имеющие недостаточную длину или помеченные атрибутом 8 (плохое значение), опустошаются, и на дальнейшую обработку не идут.

2.2.11. Массивы

Массивы используются во многих задачах обработки телеметрической информации. В первую очередь – это задачи обработки цифровых массивов данных. Если входной параметр алгоритма объявлен как массив чисел, алгоритм будет ожидать поступления в этом параметре только значений указанного типа. Поступающие скалярные значения будут игнорироваться. Наиболее распространёнными являются массивы типа `byte[]`, `ushort[]`, `uint[]`. При исполнении любых операций с массивами осуществляется контроль выхода за границы массива. Для выполнения наиболее распространённых операций с массивами в языке поддерживаются следующие функции:

– **КопироватьМассив** – копирует содержимое одного массива (или его части) в другой массив данных того же типа;

– **СравнитьМассивы** – возвращает true, если длина и содержимое двух массивов совпадает.

– **ЗаполнитьМассив** – заполняет массив указанным значением. Обычно используется для обнуления массивов.

– **СортироватьМассив** – сортирует массив числовых элементов по возрастанию.

Для выделения необходимого места под новый массив существует метод ИНИТ:

```
ПАКЕТ . ИНИТ(byte[252 + 63]);
```

Отдельно стоит рассмотреть массивы параметров. Массив параметров можно объявить с помощью ключевого слова **СТРУКТ** (описано в разделе 2.2.8), если соответствующий параметр описан на уровне описания борта. Также в алгоритмах анализа можно создавать массивы параметров, которые будут ссылаться на входные, выходные и локальные параметры. Обращение к элементам таких массивов будет перенаправляться на обращение к конкретным параметрам и переменным.

Пример 2.2.17

```
01 АНАЛИЗ(
02 ВХПАР(ushort КАДР[])
03 ВЫХПАР(ushort ЛКТ71, ushort ЛКТ72, ushort ЛКТ81, ushort ЛКТ82)
04 ПАР(
05   uint времяМлТ,
06   int ССКмлТ = -1,
07   ushort млТ71, ushort млТ72, ushort млТ81, ushort млТ82,
08   ushort позиции[4] = { 454, 462, 452, 460 },
09   ПАРАМ млТМАС[4] = { млТ71, млТ72, млТ81, млТ82},
10   ПАРАМ тМАС[4] = { ЛКТ71, ЛКТ72, ЛКТ81, ЛКТ82}
11 )
…
12 тМАС[i] = КАДР[позиции[i] - 304][8:8] $ млТМАС[i];
```

В строке 03 примера описано 4 выходных параметра алгоритма, содержащих значения локального коммутатора температур четырёх боковых блоков РН «Союз-2». В строке 10 создаётся массив `тМАС`, собирающий эти параметры. Массив `млТМАС` из строки 09 собирает 4 переменных алгоритма, объявленных строкой ранее, в которых сохраняются младшие части локальных коммутаторов. В строке 12 значения последовательно собираются и выдаются в выходных параметрах через объявленный массив `тМАС`.

2.2.12. Реакция на события

Обычно исполнение алгоритма начинается с вызова главной функции в соответствии с режимами запуска, описанными в разделе 2.2.9. Однако в языке анализа ТМИ предусмотрены и другие варианты запуска, позволяющие реализовать более сложную логику поведения. Функция **Таймер** позволяет

подписаться на временные события таймера и вызывать заданную функцию с заданной периодичностью. Например, в следующем коде

```
01 Таймер(НаВремя, 500);
```

заказывается вызов функции `НаВремя` с периодичностью в 500 мс. Вызов будет выполняться с текущего момента плюс указанное время и до завершения сеанса обработки ТМИ или до отмены подписки, которая выполняется вызовом функции **СтопТаймер**:

```
01 СтопТаймер(НаВремя);
```

Обычно на таймер подписываются в функции ИНИЦИАЛИЗАЦИЯ в начале сеанса обработки ТМИ или при наступлении некоторого заданного события.

Для обработки временных событий функция-обработчик должна принимать один аргумент типа uint – текущее время. Следует отметить, что в программе обработки ТМИ источником временных событий являются временные посылки, передающиеся в потоке ТМИ. Эти временные посылки обычно идут не чаще 10 раз в секунду и могут быть неравномерными. Тем не менее, они соответствуют времени телеметрического потока и являются корректными даже при ускоренном прогоне записи ТМИ или переобработке телеметрического файла.

Если скрипт на языке анализа ТМИ выполняется в программе отображения ТМИ для отображения мнемосхем, то там для управления временными событиями используется локальное время компьютера, временные события могут вызываться с любой заданной частотой и являются равномерными, что позволяет с использованием этого механизма реализовывать плавную анимацию мнемосхем.

Для реализации интерактивных мнемосхем, в них язык анализа ТМИ позволяет подписываться на два события графического интерфейса пользователя: наведение курсора «мыши» на графический объект или нажатие на него. Обработчик этого события может требуемым образом менять отображение мнемосхемы, создавая интерактивность. Например, при наведении курсора на некоторый блок, выводить по нему более подробную информацию о состоянии прибора.

Итак, в разделе рассмотрены основные элементы языка анализа ТМИ, приведены примеры использования уникальных элементов, разработанных для анализа телеметрической информации, указаны задачи типичного применения разработанных элементов.

2.3.Формальная грамматика языка анализа ТМИ

В данном разделе приводится лингвистическая модель разработанного языка анализа ТМИ, заданная грамматикой в расширенной форме Бэкуса-Наура (РБНФ) [65, 67]. Для наглядности терминальные символы выделены коричневым цветом, нетерминальные – синим, комментарии – зелёным.

Подпрограммы на языке анализа ТМИ задаются в форме отдельного алгоритма обработки в составе языка подготовки исходных данных на обработку ТМИ телеметрического информационно-вычислительного комплекса ЦУП, и могут использоваться наряду с базовыми алгоритмами обработки при задании цепочек обработки значений ТМ-праметров.

```
Анализ = "АНАЛИЗ(",
  Заголовок,
  Объявление функций,
  Тело алгоритма,
")";
```

2.3.1. Базовые нетерминалы

Начнём описания с общих нетерминальных символов, используемых при объявлении всех элементов языка.

```
Имя параметра = Допустимый идентификатор – Расширенный тип;
Допустимый идентификатор = { Символ идентификатора }- |
  "'", { Символ - "'" | "''" }-, "'";
Символ идентификатора = Символ –
  ("""|"("|")"|","|":"|";"|"["|"]"|"{"|"}"|"<"|">"|"="|"_"|"*"|"@"|" ");
Цифра = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9";
```

Символ – это любой печатный символ. Отметим, что, в отличие от большинства языков программирования, язык анализа ТМИ поддерживает имена параметров, начинающиеся с цифр, включающие в себя арифметические операторы, буквы кириллицы, греческого и других алфавитов и пробелы. При это язык не чувствителен к регистру символов в любых идентификаторах, одинаковые

по написанию символы кириллицы и латиницы считаются эквивалентными. Для использования пробелов или пунктуационных символов имя параметра необходимо заключить в апострофы.

```
Строковый литерал =
  """, { Символ строки }, """;
Символ строки =
  Символ - ("""", "\") |
  "\\" | (* Символ \ *)
  "\"" | (* Символ " *)
  "\t" | (* Символ табуляции *)
  "\+b" | (* Вкл. режим полужирного текста *)
  "\-b" | (* Выкл. режим полужирного текста *)
  "\+i" | (* Вкл. режим курсивного текста *)
  "\-i" | (* Выкл. режим полужирного текста *)
  "\+u" | (* Вкл. режим подчёркивания *)
  "\-u" | (* Выкл. режим подчёркивания *)
  "\c", ("0".."9"|"a".."f"); (* Код цвета текста 0÷15 *)
```

Базовые типы данных языка, которые поддерживаются для ТМ-параметров:

```
Тип =
  "BYTE" |     (* беззнаковое 8-разрядное целое *)
  "USHORT" |   (* беззнаковое 16-разрядное целое *)
  "INT" |      (* 32-разрядное целое *)
  "UINT" |     (* беззнаковое 32-разрядное целое *)
  "LONG" |     (* 64-разрядное целое *)
  "ULONG" |    (* беззнаковое 64-разрядное целое *)
  "DOUBLE" |   (* 64-разрядное вещественное с плавающей точкой *)
  "МАСАН" |    (* массив анализа *)
  "ТЕКСТАН" |  (* текст анализа *)
  "ASCII" |    (* текст *)
  "КОД";       (* кодовое значение *)
```

Расширенные типы данных, которые могут использоваться для локальных переменных и управляемых объектов мнемосхем (использование языка анализа ТМИ для отображения мнемосхем описано в разделе 3.2):

```
Расширенный тип =
  Тип |
  "BOOL" |         (* логический *)
  "CHAR" |         (* текстовый символ *)
  "SHORT" |        (* 16-разрядное целое *)
  "FLOAT" |        (* 32-разрядное вещественное *)
  "ПАРАМ" |        (* ссылка на другой параметр *)
  "STRING" |       (* текстовая строка *)
  "XML" |          (* XML дерево или ссылка на его узел *)
  "LIST" |         (* коллекция элементов одного типа *)
  "ФИГУРА" |       (* любая фигура мнемосхемы *)
  "ПРЯМОУГОЛЬНИК" | (* фигура «прямоугольник» мнемосхемы *)
```

```
  "ЭЛЛИПС" |      (* фигура «эллипс» мнемосхемы *)
  "ОТРЕЗОК" |     (* фигура «отрезок» мнемосхемы *)
  "КАРТИНКА" |    (* фигура «картинка» мнемосхемы *)
  "ПОЛЕИЗОБРАЖЕНИЙ" | (* поле для отображения изображений *)
  "ТЕКСТ" |       (* фигура «текст» мнемосхемы *)
  "КОНТЕЙНЕР" |   (* контейнер для фигур на мнемосхеме *)
  "ВКЛАДКИ" |     (* элемент отображения вкладок на мнемосхеме *)
  "ВКЛАДКА" |     (* одна вкладка на мнемосхеме *)
  "LISTVIEW" |    (* элемент отображения табличной информации *)
  "LISTVIEWCOLUMN" | (* описания столбца ListView *)
  "LISTVIEWITEM"; (* описание строки ListView *)
```

2.3.2. Заголовок алгоритма

Заголовок алгоритма состоит из элементов, расположенных в произвольном порядке. Как правило, первым следует элемент «ЗАПУСК», который встречается однократно, за ним идут объявления входных и выходных параметров. Эти элементы могут повторяться. Завершают заголовок объявления типов данных и локальных параметров (переменных) алгоритма.

```
Заголовок = { Элемент заголовка };
Элемент заголовка =
  Заголовок запуска |
  Заголовок входные параметры |
  Заголовок выходные параметры |
  Заголовок изменяемые параметры |
  Заголовок локальные параметры |
  Заголовок объявление типа;
```

Подробное описание режимов запуска алгоритма приводится в разделе 2.2.9.

```
Заголовок запуска = "ЗАПУСК(",
  ("КАЖДЫЙ" | "ПАЧКА" | "ВСЕ"), ",",
  ("НП" | "ВП" | "ВСЕ"),
  [", ИНСТРУКЦИЙ" : Целое число ],
  [", ПАРАМЕТРОВ" : Целое число ],
")";
```

Подробное описание объявления входных и выходных параметров алгоритма приводится в разделе 2.2.8.

```
Заголовок входные параметры = "ВХПАР(",
  Входной параметр, { ",", Входной параметр },
  ")";
Входной параметр =
   [ "АРГ" ],
   ( "СТРУКТ" | Тип ),
   ( Имя | "ПРИШЕДШИЙ" ), [ "[]" ],
```

```
   [ "AS", Имя2 ],
   [ "=" Выражение ];

Заголовок выходные параметры = "ВЫХПАР(",
  Выходной параметр, { ",", Выходной параметр },
  ")";
Выходной параметр =
   [ "АРГ" ],
   [ Тип ],
   Имя, [ "[]" ],
   [ "AS" Имя2 ];

Заголовок изменяемые параметры = "ВХВЫХПАР(",
  Изменяемый параметр, { ",", Изменяемый параметр },
  ")";
Изменяемый параметр =
   [ "АРГ" ],
   Тип,
   ( Имя | "ПРИШЕДШИЙ" ), [ "[]" ],
   [ "AS", Имя2 ];

Заголовок локальные параметры = "ПАР(",
  Объявление параметра, { ",", Объявление параметра },
")";
Объявление параметра =
  Объявление скалярного параметра |
  Объявление массива;
Объявление скалярного параметра =
  ["CONST"],
  Расширенный тип, {"*"}, Имя,
  [ "=", Выражение ];
Объявление массива =
  Расширенный тип, {"*"}, Имя,
  (
     "[", [ Размер массива ], "]" |
     { "[]" }-
  )
  [ "= {" [ Выражение, { ",", Выражение } ] "}" ];
Размер массива = Целое;

Заголовок объявление типа =
  "ТИП", Имя нового типа,
  [ ":", Имя базового типа ],
  "{",
  [ Объявление члена типа, { ",", Объявление члена типа } ],
  "}";
Объявление члена типа =
  Объявление поля |
  Объявление метода;
```

```
Объявление поля =
  Расширенный Тип, Имя поля |
  Расширенный Тип, "*", Имя поля |
  Расширенный Тип, Имя поля, "[]";
Объявление метода =
  Расширенный Тип, Имя метода, "(",
  [ Аргумент метода, { ",", Аргумент метода } ],
  ")",
  "{", Тело метода, "}";
Аргумент метода =
  Расширенный Тип, Имя аргумента |
  Расширенный Тип, Имя аргумента, "[]";
Тело метода = Инструкция;

Имя = Имя параметра;
Имя2 = Имя параметра;
Имя поля = Имя параметра;
Имя аргумента = Имя параметра - "THIS";
Имя нового типа = Имя параметра - Расширенный тип;
```

2.3.3. Объявление функций

Практически любой алгоритм содержит ряд функций, описывающих основную логику алгоритма. Для того, чтобы можно было выполнять трансляцию алгоритма за один проход, каждая функция должна быть объявлена до её использования.

```
Объявление функции =
  [ Тип возврата ],
  Имя функции,
  Объявление аргументов функции,
  "{", Тело функции, "}";
Тип возврата =
  Расширенный тип |
  Расширенный тип, "[]" |
  Кортеж возврата;
Кортеж возврата = "(", Элемент кортежа, {",",Элемент кортежа},")";
Элемент кортежа = Расширенный тип, [Имя элемента];
Объявление аргументов функции =
  "()" |
  "(", Аргумент функции, {"," Аргумент функции}, ")";
Аргумент функции = Расширенный тип, ["&"], Имя аргумента, ["[]"];
Тело функции = Инструкция;
```

2.3.4. Тело алгоритма

Тело алгоритма представляет собой безымянную функцию, которая идёт после всех именованных функций.

```
Тело алгоритма = "{", { Инструкция }, "}";
Инструкция =
  "{", { Инструкция }, "}" | (* блок инструкций *)
  ";" | (* пустая инструкция *)
  Объявление локальной переменной |
  Инструкция return |
  "break;" |
  "continue;" |
  Инструкция if |
  Инструкция for |
  Инструкция switch |
  Инструкция codeswitch |
  Выражение, ";";
Объявление локальной переменной =
  Объявление массива |
  Объявление скаляра;
Объявление массива =
  Расширенный тип,
  { "*" }, (* указатель *)
  Имя переменной,
  "[", [ Выражение ], "];"; (* массив с указанием размера *)
Объявление скаляра =
  Расширенный тип,
  { "*" }, (* указатель *)
  Имя переменной,
  [ "=", Выражение ], ";";
Инструкция return =
  "return;" |
  "return", Выражение, ";";

Инструкция if =
  "if", "(", Выражение, ")", Инструкция,
  [ "else", Инструкция ];

Инструкция for =
  "for", "(",
  Инициализаторы for, ";",
  Условие продолжения for, ";",
  Итератор for, ")",
  Инструкция;
Инициализаторы for =
  Инициализатор for, { ",", Инициализатор for };
Инициализатор for =
  { Объявление локальной переменной |
  Выражение };
Условие продолжения for =
  Выражение; (* должно быть логического или числового типа *)
Итератор for =
  Выражение, { ",", Выражение };
```

```
Инструкция switch =
  "switch", Выражение, "{",
  { Вариант выбора },
  [ "else:", Инструкция ],
  "}";
Вариант выбора =
  Целое, (* управляющее выражение *)
  ":", Инструкция;

Инструкция codeswitch =
  "codeswitch", Основание, "(", Выражение, ")", "{",
  { Вариант кодового выбора },
  [ "else:", Инструкция ],
  "}";
Основание =
  "Б" | (* двоичная (бинарная) система счисления *)
  "В" | (* восьмеричная система счисления *)
  "Ш";  (* шестнадцатеричная система счисления *)
Вариант кодового выбора =
  Кодовое значение, ":", Инструкция;
Кодовое значение = {Цифра Б}- | {Цифра В}- | {Цифра Ш}-;
Цифра Б = "0"|"1"|"-";  (* бинарные *)
Цифра В = "0".."7"|"-"; (* восьмеричные *)
Цифра Ш = "0".."9"|"A".."F"|"-"; (* шестнадцатеричные *)

Выражение =
  Тернарное выражение |
  Тернарное выражение, Оператор присваивания, Выражение;
Оператор присваивания =
  "="|"+="|"-="|"*="|"/="|"%="|"&="|"|="|"^="|"$="|">>="|"<<=";
Тернарное выражение =
  Выражение логического ИЛИ |
  Выражение логического ИЛИ, "?",
  Тернарное выражение, ":",
  Тернарное выражение;
Выражение логического ИЛИ =
  Выражение логического И, { "||", Выражение логического И };
Выражение логического И =
  Выражение сравнения, { "&&", Выражение сравнения };
Выражение сравнения =
  Битовое выражение, { Оператор сравнения, Битовое выражение };
Оператор сравнения = "=="|"!="|">="|"<="|">"|"<";
Битовое выражение =
  Выражение суммы, { ("&"|"|"|"^"), Выражение суммы };
Выражение суммы =
  Выражение мультипликативное,
  { ("+"|"-"), Выражение мультипликативное };
Выражение мультипликативное =
  Выражение сдвига, { ("*"|"/"|"%"), Выражение сдвига };
```

```
Выражение сдвига =
  Выражение склеивания полей,
  { ("<<"|">>"), Выражение склеивания полей };
Выражение склеивания полей =
  Унарное выражение, { "$", Унарное выражение };
Унарное выражение =
  Унарный префикс, Унарное выражение |
  Унарное выражение, Унарный постфикс |
  Выражение с постфиксом
Унарный префикс = "++"|"--"|"!"|"~";
Унарный постфикс = "++"|"--";
Выражение с постфиксом =
  Элемент, { Обращение к полю | Индексирование };
Обращение к полю =
  ".", Имя свойства |
  ".", Имя метода,
  "(", [ Выражение, { ",", Выражение } ], ")" |
  ".", "ФОРМАТ(", Размерность, ")" |
  ".", "АТРИБУТ(", Код атрибута, ")" |
  ".", "РЕЖИМ(", Код режима, ")" |
  ".", Инициализация массива анализа |
  ".", Инициализация текста анализа |
  ".", Инициализация массива |
  ".", Форматный вывод;

Имя свойства = Стандартное свойство | Пользовательское свойство;
Стандартное свойство =
  "ВРЕМЯ",
  "ДАТА",
  "СУТКИ",
  "АТРИБУТ",
  "РЕЖИМ",
  "ДЛИНА",
  "ИНДЕКС",
  "НОМЕРМАССИВА",
  "ВРЕМЯМАССИВА",
  "РЕЗЕРВ",
  "ЦВЕТ",
  "КАДР",
  "ПЕРЕНОСИТЬСТРОКИ";
Пользовательское свойство = Имя параметра;
Код атрибута = ""|"НГРН"|"НГРВ"|"ВГРВ"|"ВГРН"|"ВД"|"ЗАШКАЛ";
Код режима =
  "НП"|"ВП"|"ВП33О"|"ВП33П"|"ВП9О"|"ВП9П"|"ВП257О"|"ВП256П";
Инициализация массива анализа =
  "ИНИТ(",
     ( "НЕТ"|"КОМ"|"АМ"|"ИМ" ), "," (* тип массива анализа *)
     ( "Ф8"|"Ф16"|"Ф32" ), "," (* размер элементов *)
     Выражение, (* длина массива *)
```

```
  ")";
Инициализация текста анализа =
  "ИНИТ(",
     Выражение, ",", (* Номер кадра анализа для отображения *)
     Целое, "x", Целое, [ "-", Целое ], ",",
     Тип кадра анализа,
Тип кадра анализа =
  "БЕГУЩАЯСТРОКА" |
  "ДОКУМЕНТ" |
  "ПРОТОКОЛ" |
  "ПРОТОКОЛСПРИОРИТЕТАМИ" |
  "ПРОТОКОЛСУНИКАЛЬНЫМИПРИОРИТЕТАМИ";
Инициализация массива анализа =
  "ИНИТ(",
     ( "НЕТ"|"КОМ"|"АМ"|"ИМ" ), "," (* тип массива анализа *)
     ( "Ф8"|"Ф16"|"Ф32" ), "," (* размер элементов *)
     Выражение, (* длина массива *)
  ")";
Инициализация массива =
  "ИНИТ(",
     Расширенный тип, { "*" },
     { "[]" },
     "[", { Выражение }, "]", (* длина массива *)
  ")",
  {
     "{", [
          Целое число, { ",", Целое число } |
          Вещественное число, { ",", Вещественное число } |
          Строка, { ",", Строка }
     ], "}",
  };

Форматный вывод =
  Частичный форматный вывод | Форматный вывод строки;
Частичный форматный вывод =
  "WRITE(",
     Строка формата,
     Аргументы формата
  ")";
Форматный вывод строки =
  "WRITELINE(",
     [
     Целое число | (* код цвета *)
     Целое число, ",", Целое число | (* Координаты *)
        (* Координаты и цвет *)
     Целое число, ",", Целое число, ",", Целое число |
     ],
     Строка формата,
     Аргументы формата
```

```
  ")";
Строка формата =
  """,
  {
     Символ строки - "{" |
     "{{" | (* Означает один символ “{” *)
     Обращение к аргументу
  },
  """,
Обращение к аргументу =
  "{",
     Целое число, (* Номер аргумента, начиная с 0 *)
     [
        Ширина поля |
        Ширина поля, Код формата,
        Код формата,
        Код формата, Ширина поля
     ]
  "}";
Ширина поля =
  ",",
  (
     Целое число | (* Выравнивание по левому краю поля *)
     "-", Целое число | (* Выравнивание по правому краю поля *)
     "-", Целое число, "-" (* Выравнивание по центру поля *)
  );
Код формата =
  ":", ("H"|"X"|"h"|"x"), [ Целое число ], (*Шестнадцатеричный*)
  ":", ("O"|"o"), [ Целое число ], (* Восьмеричный *)
  ":", ("B"|"b"), [ Целое число ], (* Двоичный *)
  ":", ("T"|"t"), [ "*" ], (* Формат времени *)
  ":", ("D"|"d"), (* Формат даты *)
  ":", ("E"|"e"), [ Целое число ], (* Экспоненциальный *)
  ":",              (* Форматный вывод целого или вещественного *)
     [ "+" | "-" ], (* Место под знак числа *)
     { "#" }, (* Позиции под необязательные старшие цифры *)
     { "0" }, (* Позиции под старшие цифры *)
     [
        ( "." | "," )
        { "0" }, (* Позиции под младшие цифры *)
        { "#" }, (* Позиции под необязательные младш. цифры *)
     };
Аргументы формата =
  { ",", Выражение };

Индексирование =
  Взятие битового поля |
  Обращение к элементу массива;
Взятие битового поля =
```

```
  "[", Выражение, "]" | (* Взятие 1 бита для целого аргумента *)
  "[", Выражение, ":", Выражение, "]";
Обращение к элементу массива =
  "[", Выражение, "]";
```

Элементом трансляции кода алгоритма нижнего уровня является «Элемент».

```
Элемент =
  Преобразование типа |
  Кортеж переменных |
  Кортеж значений |
  "(", Выражение, ")" |
  Строковый литерал |
  "NEW", Расширенный тип, "[", Выражение, "]" | (* Массив *)
  "NEW", Расширенный тип, (* Создание объекта пользователя *)
     "(", Выражение, { ",", Выражение, }, ")" |
  Вызов функции |
  Стандартная переменная или константа |
  Телеметрический параметр |
  Переменная |
  "'", Символ, "'" |
  Константа;

Преобразование типа =
  "(", Расширенный тип, ")", Унарное выражение;
Кортеж переменных =
  "(",
     [ Расширенный тип ], Имя переменной,
     { ",", [ Расширенный тип ], Имя переменной, }-
  ")";
Кортеж значений =
  "(",
     Выражение,
     { ",", Выражение }-,
  ")";
Вызов функции =
  Имя базового алгоритма, "(", Выражение, ")" |
  "MIN(", Выражение, ",", Выражение, ")" |
  "MAX(", Выражение, ",", Выражение, ")" |
  "ABS(", Выражение, ")" |
  "PI()" |
  "E()" |
  "SIN(", Выражение, ")" |
  "COS(", Выражение, ")" |
  "TAN(", Выражение, ")" |
  "ASIN(", Выражение, ")" |
  "ACOS(", Выражение, ")" |
  "ATAN(", Выражение, ")" |
  "ATAN2(", Выражение, ",", Выражение, ")" |
  "SQRT(", Выражение, ")" |
  "EXP(", Выражение, ")" |
```

```
  "LOG(", Выражение, ")" |
  "SIGN(", Выражение, ")" |
  "ЧИСЛОЕДИНИЧНЫХБИТ(", Выражение, ")" |
  "ЧИСЛОЕДИНИЧНЫЙБИТОВ(", Выражение, ")" |
  "ВЗЯТЬТЕКСТ(", Выражение, ",", Выражение, ",", Выражение ")" |
  "ТЕКУЩИЙВИТОК()" |
  "ТЕКУЩЕЕВРЕМЯ()" |
  "ТЕКУЩАЯДАТА()" |
  "ТЕКУЩИЕСУТКИ()" |
  "RANDOM(", Выражение, ")" |
  "GETLEAPSECONDS()" |
  "ФОРМАТСТРОКИ(", Строка формата, Аргументы формата, ")" |
  "ТАЙМЕР(", Выражение, ",", Выражение, ")" |
  "СТОПТАЙМЕР(", Выражение, ")" |
  "СЕЙЧАСВРЕМЕНИ()" |
  "СЕЙЧАСДАТА()" |
  "ИНИТУКАЗАТЕЛЬ(", Выражение, ",", Выражение, ")" |
  "КОПИРОВАТЬМАССИВ(", (
     Выражение, ",", Выражение, |
     Выражение, ",", Выражение, ",", Выражение, |
     Выражение, ",", Выражение, ",", Выражение, ",", Выражение,
     ",", Выражение
  ), ")" |
  "ЗАПОЛНИТЬМАССИВ(", (
     Выражение, ",", Выражение, ",", Выражение |
     Выражение, ",", Выражение, ",", Выражение, ",", Выражение
  ), ")" |
  "СРАВНИТЬМАССИВЫ(", (
     Выражение, ",", Выражение, |
     Выражение, ",", Выражение, ",", Выражение, |
     Выражение, ",", Выражение, ",", Выражение, ",", Выражение,
     ",", Выражение
  ), ")" |
  "СОРТИРОВАТЬМАССИВ("
     Выражение, |
     Выражение, ",", Выражение, |
     Выражение, ",", Выражение, ",", Выражение
  ), ")" |
  "ИМЯБОРТА(", , ")" |
  Имя переменной, "(", (* Вызов пользовательской функции *)
     [ Выражение, { ",", Выражение } ],
  ")";

Стандартная переменная или константа =
  "FALSE" |
  "ЛОЖЬ" |
  "TRUE" |
  "ИСТИНА" |
  "ПРИШЕДШИЙ" | (* Параметр, вызвавший обработку *)
```

```
  "SYSTEM . PROCESSNP" | (* Заказана ли обработка НП *)
  "SYSTEM . PROCESSVP" | (* Заказана ли обработка ВП *)
  "SYSTEM . ISCURRENT" | (* Обрабатывается ли текущий сеанс *)
  "NULL";

Телеметрический параметр =
  Идентификатор | (* Имя параметра *)
  Идентификатор, "_", Идентификатор (* Имя борта и параметра *)
Переменная =       (* Имя переменной, поля текущего типа *)
  Идентификатор;   (* или функции *)

Константа =
  Константа даты |
  Константа времени |
  Целая константа |
  Вещественная константа;
Константа даты =
  Цифра, Цифра, ".", Цифра, Цифра, ".", Цифра, Цифра,
  [ Цифра, Цифра ];
Константа времени =
  Цифра, Цифра, ":", Цифра, Цифра, ":", Цифра, Цифра,
  [ ".", Цифра, Цифра, Цифра ];
Целая константа =
  [ "-" | "+" ],
  Цифра, { Цифра };
Вещественная константа =
  [ "-" | "+" ],
  Цифра, { Цифра },
  [ ".", { Цифра }],
  [ "E", ["-" | "+"], Цифра, { Цифра }];
```

Приведенная грамматика языка анализа ТМИ содержит исчерпывающее описание синтаксиса языка и является основой для разработки транслятора подпрограмм анализа ТМИ, переводящего код подпрограммы в двоичное представление в составе результатов трансляции исходных данных на обработку ТМИ.

2.4.Выводы по второй главе

В главе 2 приведено подробное описание разработанной лингвистической модели языка анализа ТМИ. Модель языка учитывает особенности организации исходных данных на обработку ТМИ в телеметрическом информационно-вычислительном комплексе ЦУП РС МКС и позволяет встроить его в язык подготовки исходных данных в качестве одного из алгоритмов. Синтаксис языка основан на синтаксисе языков C++, C#, Java, знакомых большинству современных программистов. Рассмотрены основные элементы языка анализа ТМИ, приведены примеры использования уникальных элементов, разработанных для анализа телеметрической информации, указаны задачи типичного применения разработанных элементов.

В завершении главы приводится полная грамматика языка анализа ТМИ, которая должна лечь в основу разработки транслятора подпрограмм анализа ТМИ, переводящего код подпрограммы в двоичное представление в составе результатов трансляции исходных данных на обработку ТМИ. Вопросы реализации транслятора ввиду невысокой сложности в работе не рассматриваются.

3. Разработка методик синтеза САА ТМИ в реальном масштабе времени

3.1. Методика интерпретации исходных данных для автоматизированного анализа ТМИ КА

Транслятор переводит текстовое представление подпрограммы на языке анализа ТМИ в байт-код $B = \{b_i\}$.

Байт-код является последовательностью инструкций $I = \left\{c_i = \left(c_i^{code}, P_i\right)\right\}$, которые состоят из кода инструкции c_i^{code} и её параметров P_i. Инструкции делятся на следующие виды:

- Инструкции чтения значений;
- Вычислительные инструкции;
- Управляющие инструкции;
- Инструкции вызова функций и алгоритмов;
- Инструкции установки параметров исполнения.

Таким образом, каждая инструкция представлена одним или несколькими байтами байт-кода $Bytes(c_i) = \left\{b_{addr(c_i)}, ..., b_{addr(c_i)+len(c_i)}\right\}$, и каждый байт байт-кода соответствует какой-либо инструкции, то есть множество B образуется из байт-кода всех инструкций $B = \bigcup_i Bytes(c_i)$.

Интерпретатор, получая на вход значения телеметрических параметров и сигналов времени, исполняет байт-код конечным автоматом (finite-state machine) $FSM = \{B, ip, Var, VS\}$. Каждое значение входного параметра инициирует итерацию исполнения интерпретатора. Конечный автомат содержит байт-код B с адресом текущей инструкции ip, глобальную коллекцию переменных Var, вычислительный стек VS.

В начале исполнения стек VS пустой, адрес текущей инструкции указывает на начало байт-кода: $ip = 0$. Исполнение каждой инструкции изменяет адрес текущей инструкции:

для большинства инструкций, исполнение переходит к следующей инструкции: $ip \leftarrow ip + len(c_i)$;

для инструкций условного перехода и ряда управляющих инструкций адрес текущей инструкции может измениться на указанное положительное или отрицательное смещение $ip \leftarrow ip + offset$;

для инструкций безусловного перехода адрес текущей инструкции принимает указанное значение $ip \leftarrow addr$;

для инструкции возврата из функции адрес текущей инструкции становится равным адресу инструкции, вызвавшей исполняемую функцию $ip \leftarrow callAddr$.

Все инструкции байт-кода закодированы для выполнения в постфиксной нотации, когда код вычисления операндов предшествует коду самой операции, а результаты всех вычислений помещаются в стек вычислений *VS*. Стек вычислений также используется для передачи аргументов в функцию и возврата результата её работы, при этом кортеж занимет ячейки стека по числу своих элементов.

В случае возникновения ошибки исполнения подпрограммы, она помечается как недействительная, и её вызовы блокируются, защищая от сбоев остальные алгоритмы обработки и анализа ТМИ.

3.1.1. Переменные

Все данные интерпретатор хранит в переменных *Variable* и массивах переменных. Переменные представляют входные и выходные параметры, глобальные и локальные переменные, формальные аргументы функций и поля объектов пользовательских типов. Переменная представляет собой структуру, основное содержание которой приведено в таблице 3.1.1.

Таблица 3.1.1. Поля переменных

№	Поле	Описание
1.	Индекс	Индекс переменной в глобальной коллекции *Var*
2.	Тип	Тип хранимых значений
3.	Вид	Вид объекта, хранящегося в переменной: параметр, локальная переменная, поле пользовательского типа и т. п.
4.	Наличие значения	Содержит ли переменная значение, либо она не инициализирована

5.	Сформирована	Сформировано ли значение выходного параметра в ходе текущей итерации исполнения
6.	Имя переменной	Вспомогательное свойство, используется для отладки

Большинство переменных интерпретатор размещает в глобальной коллекции *Var*, и в поле Индекс содержится порядковый номер переменной в этой коллекции. В процессе загрузки алгоритма в коллекции размещаются входные и выходные параметры, глобальные переменные алгоритма, а в процессе исполнения в этой коллекции размещаются вводимые локальные переменные. Массив параметров будет занимать одну позицию в коллекции *Var* как представитель массива, и по одной позиции для каждого параметра (или ссылки на параметр).

3.1.2. Значения

Интерпретатор ориентирован на обработку телеметрических значений и в процессе вычислений оперирует специальными структурами *TmValue*, которые содержат тип хранимых данных и значение указанного типа. Типы хранимых в *TmValue* значений приведены в таблице 3.1.2. Такой подход позволяет на уровне построения кода полностью абстрагироваться от предмета проводимых вычислений и оперировать произвольными значениями посредством абстрактных арифметических операций.

Таблица 3.1.2. Типы значений, хранимых в TmValue

№	Поле	Описание типа значения
1.	None	Пустое значение
2.	Int	Целое 32-разрядное
3.	Uint	Беззнаковое целое 32-разрядное
4.	Long	Целое 64-разрядное
5.	Ulong	Беззнаковое целое 64-разрядное
6.	Double	Вещественное значение
7.	BoolRef	Ссылка на логическое значение
8.	ByteRef	Ссылка на целое значение типа byte
9.	UshortRef	Ссылка на целое значение типа ushort
10.	IntRef	Ссылка на целое значение типа int
11.	UintRef	Ссылка на целое значение типа uint
12.	FloatRef	Ссылка на вещественное значение типа float
13.	DoubleRef	Ссылка на вещественное значение типа double

14.	Code	Кодовое значение
15.	String	Текстовое значение
16.	StringRef	Ссылка на текстовое значение
17.	VariableRef	Ссылка на переменную
18.	ProcedureRef	Ссылка на процедуру или функцию
19.	UserObjectReference	Ссылка на объект пользовательского типа
20.	ManagedObjectReference	Ссылка на управляемый графический объект
21.	Array	Массив значений

Опытная реализация интерпретатора описана в разделе 4.1.2.

3.2.Методика формирования мнемосхем отображения результатов анализа ТМИ БС КА

Мнемосхемы являются одной из наиболее наглядных форм представления состояния бортовых систем КА на основе телеметрической информации [18]. Как правило, мнемосхемы реализуют либо путём непосредственного программирования, либо конструируют из отдельных фигур (групп фигур), для каждой из которых задают правила, по которым фигура будет менять своё состояние в зависимости от значений телеметрических параметров.

Первый подход даёт практически неограниченные возможности для разработки форм представления, но имеет высокую трудоёмкость при создании серии мнемосхем, а любые модификации мнемосхемы влекут доработку программы (модуля) отображения ТМИ с полным циклом испытаний. Трудоёмкость непосредственного программирования мнемосхем сопоставима с трудоёмкостью разработки специального программного обеспечения.

Второй подход позволяет специалистам средней или даже низкой квалификации относительно легко конструировать и модифицировать мнемосхемы по заданным исходным данным, минимально подвержен возможным ошибкам и практически не требует отладки, однако возможности таких мнемосхем по отображению состояний и учёту различных телеметрических параметров существенно ограничены.

В работе предлагается комбинированный подход для построения мнемосхем состояния КА. Мнемосхема строится из набора геометрических фигур, изображений и элементов управления, а подпрограмма на языке анализа ТМИ

управляет всеми элементами на основе поступающих значений телеметрических параметров. Предложенный подход обладает всеми достоинствами традиционных подходов и практически лишён их недостатков. К недостаткам можно отнести, разве что, сложную первоначальную реализацию предложенной схемы и ограниченные, хотя и очень широкие возможности по представлению информации.

3.2.1. Модель системы отображения мнемосхем состояния БС КА

Система отображения мнемосхем состояния БС КА состоит из следующих элементов (рис. 3.2.1):

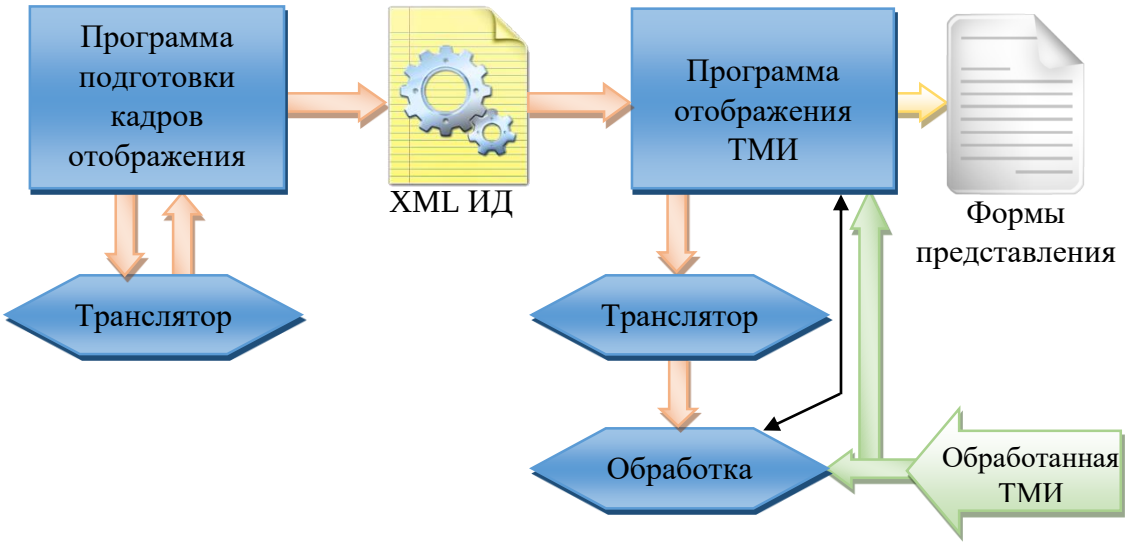


Рис. 3.2.1. Структурная схема системы отображения мнемосхем.

– Программа подготовки кадров отображения, предназначенная для создания макетов табличных формуляров и мнемосхем с использованием графического конструктора, а также создания скриптов управления поведением мнемосхем (подпрограмм на языке анализа).

– Программа отображения ТМИ, предназначенная для отображения в реальном времени обработанной ТМИ на различных формулярах отображения.

– Исходные данные, содержащие макеты мнемосхем и других кадров отображения:

- Каждая мнемосхема состоит из набора графических фигур, базовых элементов отображения ТМИ и скрипта на языке анализа ТМИ, управляющего мнемосхемой;

- Исходные данные для отображения ТМИ сохраняются в текстовом формате XML. Для поддержки этим форматом мнемосхем необходимо разработать структуры хранения фигур, составляющих мнемосхемы.

– Транслятор скриптов на языке анализа ТМИ, вызываемый программой подготовки кадров отображения и программой отображения ТМИ. На этапе подготовки транслятор используется для проверки создаваемых скриптов, а в программе отображения в результате работы транслятора формируется внутреннее представление скрипта в двоичном виде, удобном для интерпретации.

– Интерпретатор скриптов на языке анализа ТМИ (программа обработки ТМИ), получающий на входе результаты трансляции исходных данных, поток ТМИ и графические объекты мнемосхем, которыми он управляет (чёрная стрелка на рис. 3.2.1).

– Поток обработанной телеметрической информации, принимаемый программой отображения ТМИ в реальном времени. Телеметрические параметры из этого потока подаются интерпретатору на обработку.

Таким образом, скрипты должны сохраняться в файлах исходных данных в виде текста, и транслироваться в двоичное представление уже в системе отображения ТМИ. Такой подход допустим благодаря использованию текстового формата исходных данных и достаточно высокой скорости трансляции небольших скриптов. Достигаемым преимуществом является возможность транслировать скрипт под конкретную версию исходных данных, используемых в текущем сеансе. В противном случае пришлось бы решать проблему с возможными изменениями номеров параметров в текущем сеансе в сравнении с номерами, которые использовались в исходных данных, которые были текущими на момент первоначальной трансляции и записи исходных данных.

3.2.2. Фигуры мнемосхем

Рациональный перечень разработанных для мнемосхем фигур приведён на рис. 3.2.2 в виде иерархического дерева. Данный перечень не является

исчерпывающим и может дополняться по мере возникающих потребностей, но его достаточно для решения большинства задач анализа ТМИ. Кроме приведённых фигур для размещения на мнемосхемах доступны все элементы отображения, используемые на табличных (символьных) кадрах отображения, включая таблицы значений параметров, графики и текстовые протоколы. Однако данные элементы работают автоматически и не предназначены для управления ими из скрипта.

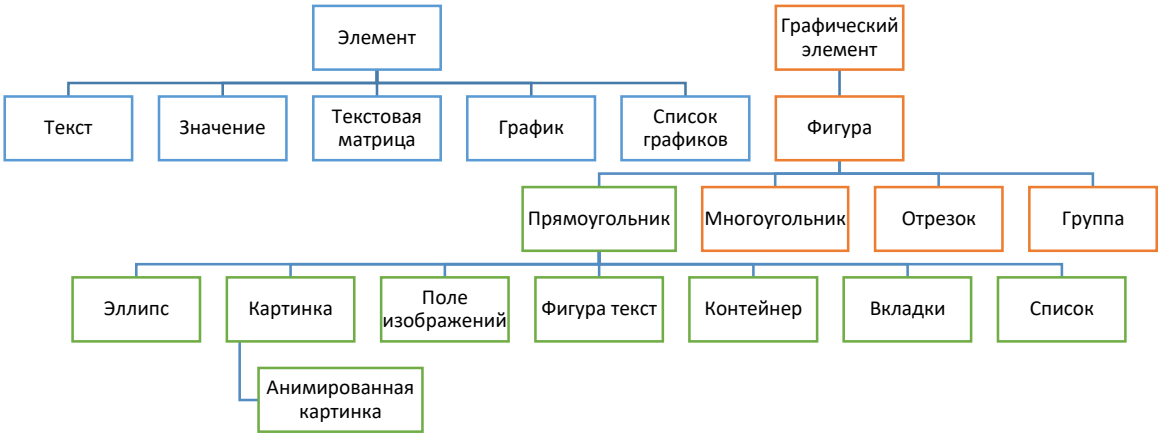


Рис. 3.2.2. Фигуры мнемосхем

В таблице 3.2.1 приведены основные свойства фигур, доступные для задания в системе подготовки кадров отображения и изменения из скрипта. Для каждой фигуры доступны все свойства, принадлежащие её базовому типу. Например, эллипсу доступны все свойства прямоугольника.

Таблица 3.2.1. Основные совйства фигур

№	Тип значений	Имя	Принад-лежность	Описание
1	double	толщинаГра-ницы	Фигура	Для большинства фигур, таких как прямоугольник, окружность, отрезок и т. п., задаёт толщину границ
2	Color	цветГраницы	Фигура	Цвет, которым рисуется граница ненулевой толщины
3	Color	цветФона	Фигура	Для фигур, имеющих внутреннюю область, таких как прямоугольник, окружность и. т. п., задаёт цвет заливки
4	string	ToolTip	Фигура	Для любого графического элемента можно задать подсказку, описывающую элемент. Это может быть расшифровка аббревиатуры, указание, какие ТМ-параметры отвечают за состояние элемента, подробности нештатного состояния объекта и т. п.

5	Обводка	обводка	Фигура	Задаёт начертание линии границы: сплошная, пунктирная, точечная, штрих-пунктирная и т. п.
6	bool	невидимая	Фигура	Указывает, является ли фигура в данный момент времени невидимой. Позволяет на динамических мнемосхемах показывать скрывать фигуры, относящиеся к конкретной ситуации.
7	double	x, y	Прямо-угольник	Координаты фигуры от левого-верхнего угла содержащего её контейнера
8	double	ширина, высота	Прямо-угольник	Размеры прямоугольника
9	double	радиусСкруг-ления	Прямо-угольник	Радиус скругления углов прямоугольника. Если равен 0, скругление не используется
10	double	уголНачала, уголКонца	Эллипс	Углы в градусах относительно верхней точки эллипса, задающие сектор этого эллипса. Например при уголНачала=90°, уголКонца=270° задаётся плоский сверху полукруг.
11	bool	внутренняя Граница	Эллипс	Указывает, необходимо ли рисовать радиусы для сектора эллипса
12	ТипСтрелки	типНачала, типКонца	Отрезок	Начертание начала и конца отрезка: простой, стрелка, круг
13	double	x1, y1, x2, y2	Отрезок	Координаты концов отрезка относительно верхнего-левого угла содержащего его контейнера
14	string	имяФайла	Картинка	Имя графического файла, который отображается в заданном прямоугольнике
15	bool	цикличность	Анимированная картинка	Для изображений типа gif задаёт, следует ли отображать анимацию в цикле бесконечно
16	double	скорость	Анимированная картинка	Для изображений типа gif задаёт коэффициент ускорения воспроизведения анимации относительно базовой
17	string	текст	Текст	Текстовая строка, которая будет выводиться. Может содержать escape-последовательности разметки: [BR] – перевод на новую строку, [B] – жирный текст, [I] – курсив, [U] – подчёркнутый
18	double	размер-Шрифта	Текст	Размер шрифта в пунктах
19	FontFamily	шрифт	Текст	Семейство шрифтов для отображения текста
20	Color	цветШрифта	Текст	Цвет, которым будет выводиться текст
21	bool	вспышка	Текст	Задаёт, будет ли смена текста сопровождаться вспышкой для привлечения внимания
22	Расположе-ние Элемента	положение ПоВертикали, положение ПоГоризон-тали	Текст	Расположение текста по горизонтали внутри заданного прямоугольника: слева, справа, сверху, снизу, по центру.

Для удобства работы некоторые специальные типы данных доступны в скрипте как базовые типы языка. Например, для типа FontFamily можно просто указать название шрифта (тип string), а цвет задаётся целым 32-разрядным числом, в котором старший байт отвечает за прозрачность, следующий – за красный, далее – за зелёный и последний – за синий. Так выглядит задание розового цвета текста:

```
АВД-т . цветШрифта = 0xFFFFA3C0;
```

Помимо обозначенных свойств разработчик мнемосхемы любой фигуре может назначить уникальное в пределах мнемосхемы имя, через которое сможет обращаться к фигуре из скрипта.

3.2.3. Структура мнемосхемы

Сама мнемосхема, а также фигуры Контейнер и Вкладка являются контейнерами фигур. Для них доступны следующие действия из скрипта:

- создать фигуру;
- добавить в контейнер ранее созданную фигуру;
- удалить из контейнера выбранную фигуру.

Эти действия позволяют создавать динамические мнемосхемы, состав фигур на которых заранее не определён. Примеры таких мнемосхем приводятся в разделе 4.3.

Фигура Контейнер, хотя и может иметь границу и цвет фона, являясь прямоугольником, основное своё предназначение она имеет в том, чтобы управлять сразу всеми фигурами, в ней располагающимися: перемещать, скрывать или поворачивать их все сразу.

3.2.4. Интеграция графических элементов с модулем обработки ТМИ

Как было сказано в разделе 2.2.8, в области переменных алгоритма «Анализ» на фиксированных позициях размещаются входные параметры (в системе отображения ТМИ выходные параметры не используются), а также глобальные переменные. Для скриптов мнемосхем в конец этой области помещаются переменные, соответствующие всем именованным фигурам мнемосхемы – управляемым объектам (рис. 3.2.3). Транслятор отводит под них позиции, а в

интерпретатор подаётся коллекция самих графических объектов, ссылки на которые помещаются в соответствующие переменные. Эти переменные получают особый тип: управляемый объект. Обращение к свойствам управляемых объектов на чтение заменяется на вызов универсального виртуального метода чтения, в который передаётся имя свойства, а метод возвращает нетипизированное значение, преобразуемое потом к типу, соответствующему свойству. Для обращения к свойству на запись вызывается метод записи, в который передаётся имя свойства и нетипизированное записываемое значение. Реализация этих методов выполняется индивидуально в каждом графическом объекте. Изменение любого графического свойства мгновенно изменяет соответствующим образом внешний вид фигуры.

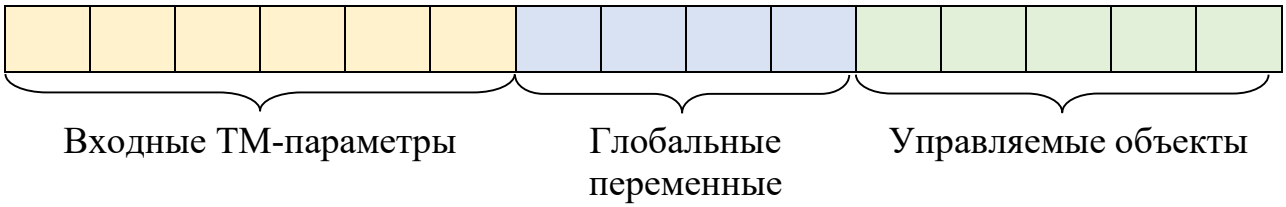


Рис. 3.2.3. Размещение управляемых объектов в списке переменных

Таким образом, предложенная модель системы отображения ТМИ позволяет с использованием разработанных средств автоматизированного анализа ТМИ реализовать отображение в реальном времени динамических мнемосхем отображения состояния БС КА. Мнемосхемы, созданные в данной системе, будут обладать куда большими возможностями, чем мнемосхемы, создаваемые в типовых конструкторах, а также их будет существенно легче создавать и модифицировать, чем в случае непосредственного программирования на языках высокого уровня. А возможность подписки на действия пользователя в скрипте позволяет создавать интерактивные мнемосхемы.

Рассчитать время создания мнемосхемы T_{cm} с использованием разработанных и существующих средств не представляется возможным. Оно зависит от сложности задачи, количества используемых фигур и параметров, квалификации разработчика, возможностей по отладке и т. п. В разделе 4.2.2 приводятся результаты практического исследования по сравнению времени создания мнемосхемы с использованием классических и разработанных средств.

3.2.5. Методика разработки мнемосхемы состояния БС КА

Одним из наиболее распространённых и востребованных мнемосхем является мнемосхема состояния отдельной бортовой системы или всего КА в целом. В работе предлагается методика разработки мнемосхем состояния КА. Процесс разработки новой мнемосхемы является творческим и, порой, весьма трудоёмким. Его можно разделить на следующие стадии:

1. проектирование;
2. реализация;
3. отладка;
4. лётные испытания.

3.2.5.1. Проектирование мнемосхемы состояния

Проектирование любой мнемосхемы анализа состояния КА начинается с анализа документации. Если в эксплуатационной документации на КА приводится функциональная схема требуемой бортовой системы КА, отталкиваться следует от неё. Мнемосхема, созданная по подобию схемы из документации, будет проще в освоении и эксплуатации группой анализа ТМИ. В случае отсутствия в документации подходящей схемы, её следует разрабатывать самостоятельно, исходя из функционального описания системы, при поддержке опытных специалистов группы управления и разработчиков бортовой системы.

Определяющим фактором при проектировании мнемосхем является конфигурация рассматриваемой бортовой системы и наличие необходимых телеметрических параметров. Прежде всего, необходимо установить, из каких элементов (блоков) состоит система, как они взаимосвязаны между собой, сколько комплектов каждого блока используется. По каждому блоку определяется состав доступных телеметрических параметров:

- комплект включён;
- комплект готов/активен;
- комплект неисправен;
- комплект является основным/резервным;
- наличие обменов/взаимодействия с другим блоком;

- режим работы комплекта;
- температура и потребляемый ток комплекта
- и т. п.

Исходя из этого состава необходимо определить, следует ли изображать блок на мнемосхеме. Элементы бортовой системы, не формирующие телеметрических параметров о своём состоянии, не несут информационной нагрузки и должны отображаться только при наличии свободного пространства для уточнения порядка взаимодействия других блоков.

3.2.5.2. Реализация мнемосхемы состояния

Простейшим способом изображения блока/комплекта является прямоугольник с надписью. Его состояние изображается цветом фона. Следующая цветовая схема является интуитивно понятной и хорошо различимой:

- незакрашенный/прозрачный – данные отсутствуют;
- серый – выключен;
- ярко-зелёный – работает;
- тёмно-зелёный – готов и выбран, но неактивен;
- красный – неисправность;
- жёлтый – предупреждение;
- ярко-фиолетовый – работает резервный комплект;
- тёмно-фиолетовый – готов и выбран резервный комплект, но неактивен.

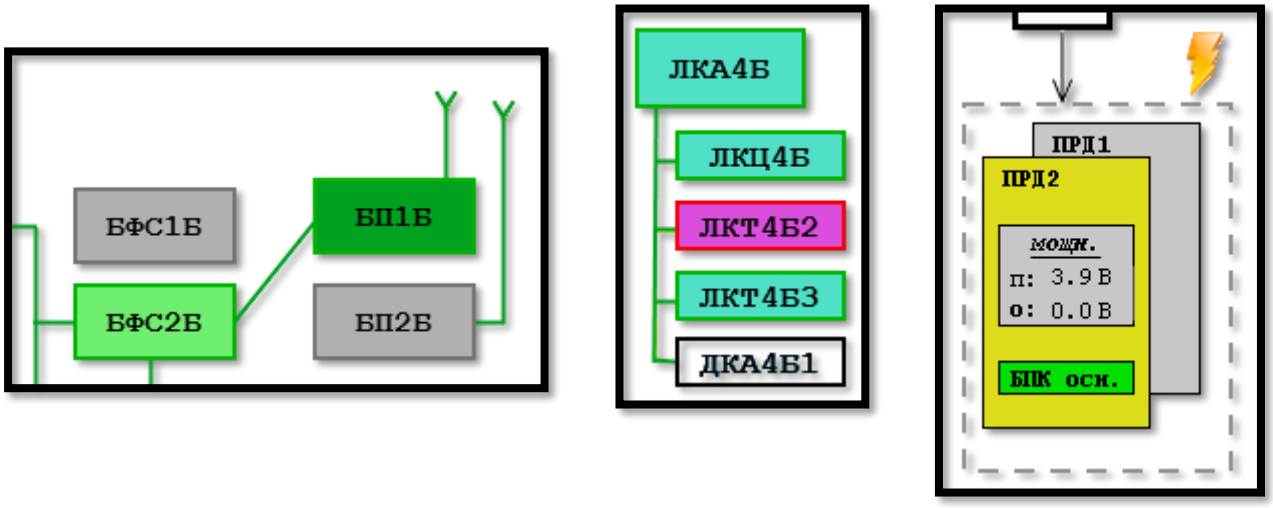
Если отдельно доступны ТМ-параметры, показывающие наличие электропитания блока, то поданное питание можно показывать жирной тёмно-зелёной границей прямоугольника или жирной зелёной линией, идущей к прямоугольнику, а отсутствие питания – тонкой линией.

Если у блока имеется несколько комплектов, работающих одновременно, следует изображать их все. Если контролируется только активный комплект, следует использовать один из трёх основных способов изображения.

1. Каждый комплект отображается отдельным прямоугольником. Активный комплект выделен ярким цветом, а неактивный – тёмным, если оба комплекта включены или оба выключены (при недопустимости подобного), они изображаются красным цветом. Стрелки/линии обменов от других блоков ведут только к активному (рис. 3.2.4, а).

2. Блок отображается одним прямоугольником, цвет которого говорит о состоянии и выбранном комплекте блока (рис. 3.2.4, б).

3. Основной и резервный комплекты отображаются один над другим с небольшим смещением. При этом от неактивного комплекта виден только край, цветом которого можно отображать его состояние (рис. 3.2.4, в).



а) б) в)

Рис. 3.2.4. Примеры отображения комплектов

В случае, когда состояние некоторого элемента контролируется несколькими одинаковыми параметрами, необходимо анализировать все эти параметры в совокупности и делать заключение по большинству сработавших. Допускается отказ отдельных ТМ-датчиков. Например, в корректирующей двигательной установке МЛМ-У «Наука» положение каждого клапана топливной системы контролируется тремя параметрами. Два срабатывают на замыкание, один – на размыкание. Тогда код отображения состояния клапана на языке анализа ТМИ может быть таким:

```
П166АМ:
П166БМ:
П166ВМ: ПокраситьКлапан(ЭПК166-пр,
          П166АМ + П166БМ + !П166ВМ >= 2);
```

Если приведённая сумма будет равна 2 или 3, клапан считается открытым.

Другим примером является контроль критически важных параметров, когда о несрабатывании одного из нескольких параметров необходимо сигнализировать для проведения детального анализа. Например, закрытое положение люка-лаза ТПК «Союз МС» контролируется тремя параметрами, и один несработавший параметр сигнализирует о неплотном закрытии люка. В этом случае код отображения состояния может быть таким:

```
ПЛЛ1:
ПЛЛ2:
ПЛЛ3:
      if (ПЛЛ1 + ПЛЛ2 + ПЛЛ3 == 3)
          событиеПЛЛ . Сработало(ПРИШЕДШИЙ . ВРЕМЯ);
      else if (ПЛЛ1 + ПЛЛ2 + ПЛЛ3 == 0)
          событиеПЛЛ . НеСработало(ПРИШЕДШИЙ . ВРЕМЯ);
      else
          событиеПЛЛ . СработалоЧастично(ПРИШЕДШИЙ . ВРЕМЯ);
```

3.3.Методика нейросетевого анализа ТМИ, содержащей медицинские показания космонавтов

Основным показателем, вычисляемым в реальном времени на основе ЭКГ, является частота сердечных сокращений (ЧСС) [82]. В существующем в ЦУП программно-аппаратном комплексе обработки медицинской информации [24] для вычисления ЧСС используется алгоритм, написанный с использованием процедурных механизмов. Данный алгоритм имеет низкую надёжность, плохо реагирует на сбои и потери в телеметрии, выдавая ошибочные значения.

Использовать для решения этой задачи методики, применяемые в современной медицине, не представляется возможным по следующим причинам:

1. частота опроса ЭКГ космонавтов существенно ниже общепринятой (190-200 Гц при 250 Гц минимально допустимых) [23, 91];
2. ЭКГ регистрируется только в отведении D-S [24], в то время как обычно предполагается съём ЭКГ пациента в 12 отведениях;
3. в связи с особенностями крепления датчиков регистрации ЭКГ космонавтов уровень измеряемого сигнала является непостоянным и может колебаться в достаточно широком диапазоне с частотой до 1 Гц;

4. принимаемый сигнал подвержен сбоям, наводкам и пропаданиям.

Проще говоря, общепринятые методики не рассчитаны на подобные сигналы.

Требуется разработать новую методику анализа ЭКГ космонавтов на всех стадиях полёта, включая вычисление ЧСС. Методика должна учитывать особенности передачи сигнала по космическому радиоканалу и быть надёжнее существующих. Реализовать методику необходимо с использованием разработанного языка анализа ТМИ.

Поскольку задача распознавания структуры ЭКГ не может быть строго формализована и надёжно решена классическими алгоритмами, будем использовать методы искусственного интеллекта на основе нейросетей. В [9] рассмотрены вопросы построения, обучения и применения нейронных сетей различного типа: от простейшего персептрона до нейронных сетей глубокого обучения. Для решения этой задачи должно быть достаточно нейросети типа многослойный персептрон.

Особенности передачи сигнала от скафандров «Орлан-МКС» в ЦУП изложены в разделе 1.1.5.

На рис. 3.3.1 приведена блок-схема методики нейросетевого анализа ТМИ, содержащей медицинские показания космонавтов. Далее по тексту подробно рассматривается каждый алгоритм методики.

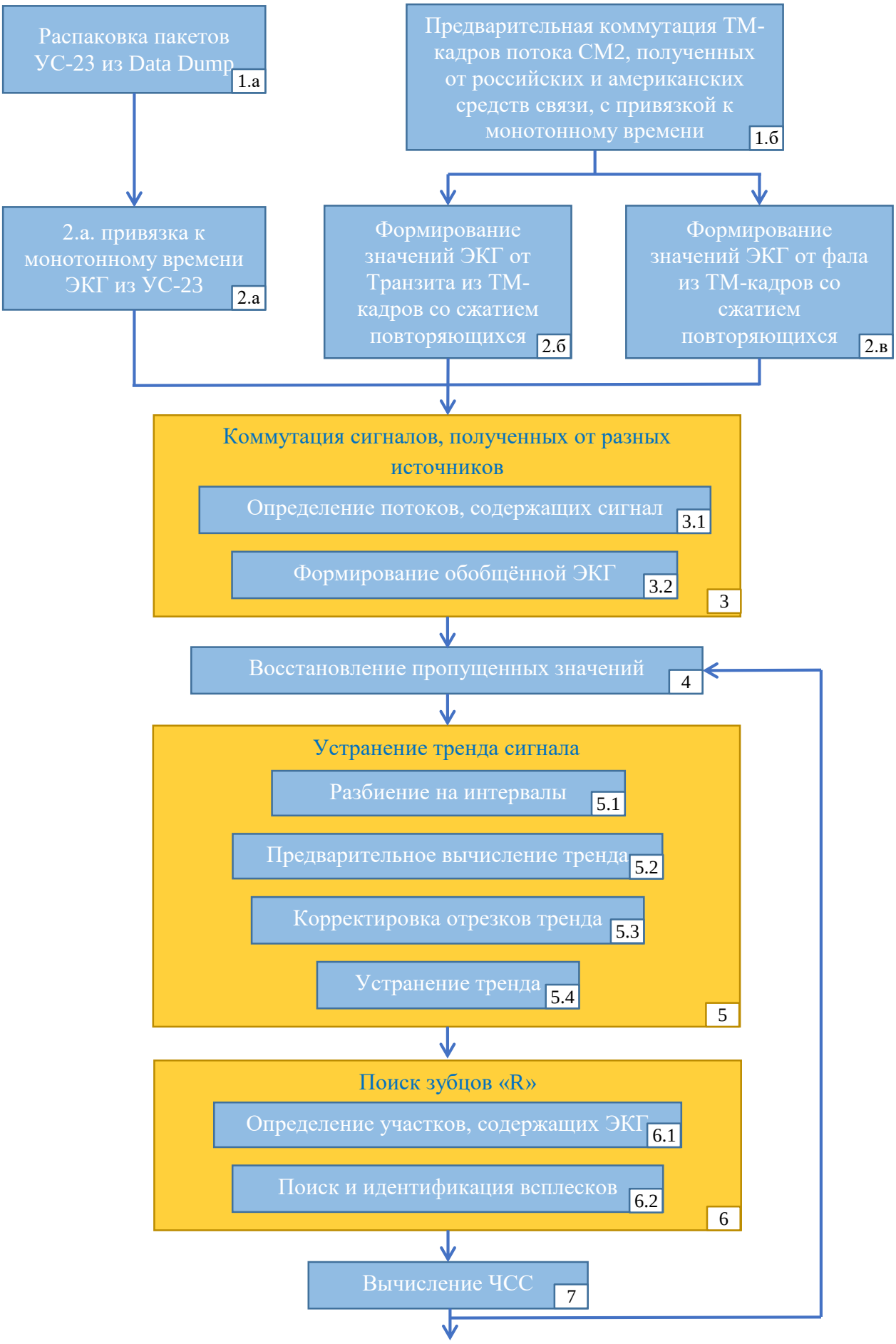


Рис. 3.3.1. Блок-схема методики нейросетевого анализа медицинской ТМИ

3.3.1. Обработка ТМ-кадров

Шаг (1.а) Массивы УС-23 распаковываются из пакетов Data Dump, передаваемых в потоке телеметрии из Хьюстона (ТМХ). В частности, выделяются массивы, содержащие непрерывные измерения ЭКГ за небольшой интервал времени.

Шаг (1.б) ТМ-кадры потока СМ2, принятые различными российскими НПРС, а также переданные по американским каналам связи, коммутируются в единый поток со строгой покадровой синхронизацией потоков. Далее выполняется подстройка времени ТМ-кадров с тем, чтобы обеспечить интервал чередования кадров, равный 21,21 мс [39]:

$$\Delta t_b = \frac{L_f}{F_i} = 0,02121 \text{ с}, \tag{3.3.1}$$

где $L_f = 5430$ бит – длина ТМ-кадра, а $F_i = 256000$ бит/с $= 256$ Кбит/с – информативность БРТС БИТС2.

3.3.2. Выделение сигналов ЭКГ из ТМИ

Шаг (2.а) Массивы УС-23 раскоммутируются на отдельные измерения ЭКГ η_i^{tmx} с частотой 200 Гц, этим измерениям устанавливается монотонное время с шагом в 5 мс, а значения калибруются по формуле:

$$\mathrm{E}_i^{tmx} = \frac{\mathrm{E}_i^{tmx} - 5}{250 - 5} \cdot 100\%$$

Шаг (2.б) Из ТМ-кадров потока СМ2 выделяются измерения ЭКГ, передаваемые непосредственно с аппаратуры «Транзит» $\eta_i^{тран}$. Так как каждое измерение передаётся в ТМ-кадре 4 раза, интервал следования между ними в соответствии с (3.3.1) будет:

$$\Delta t_b^{тран} = \frac{\Delta t_b}{4} = 0,005303 \text{ с} \tag{3.3.2}$$

а частота передачи составит:

$$f_b^{тран} = f_b \cdot 4 = 188,6 \text{ Гц} \tag{3.3.3}$$

Выполняется сокращение избыточности потока. Одинаковые измерения, следующие подряд, сжимаются, с тем, чтобы существенно сократить объём телеметрических файлов на участках, где «Транзит» не работает. То есть, из серии одинаковых измерений формируется только первое.

Шаг (2.в) Из ТМ-кадров потока СМ2 выделяются измерения ЭКГ, передаваемые со скафандров через электрофал $\eta_i^{фал}$. Эти измерения в соответствии с формулой (3.3.3) имеют также частоту 188,6 Гц.

В результате работы данного шага формируются отдельные измерения ЭКГ по каждому потоку.

3.3.3. Коммутация сигналов ЭКГ

Шаг (3) Коммутация сигналов η_i^{tmx}, $\eta_i^{тран}$, $\eta_i^{фал}$ в обобщённый сигнал η_i выполняется исходя из уверенности, что каждый сигнал хотя бы грубо привязан ко времени. Отметим сразу, что предпочтительным было бы синхронизировать поступающие потоки строго по времени, однако сделать это не представляется возможным, поскольку потоки поступают с разной частотой. На рис. 3.3.2 видно, что в целом по потокам η_i^{tmx} и $\eta_i^{тран}$ поступают приблизительно одинаковые значения, однако есть участки, где η_i^{tmx} (красная линия) опережает (временная метка 600 мс), а есть, где отстаёт (время 688 мс). Кроме того, из рисунка видны потери целых серий значений $\eta_i^{тран}$ (синяя линия). Например, отчётливо видно потерю плавного роста сигнала на время 664 мс. Эти потери являются следствием более медленной частоты передачи ЭКГ аппаратурой БИТС2-12 в сравнении с опросом сигналов аппаратурой «Транзит» на борту МКС. Таким образом, при коммутации сигнал от УС-23, передающий все измерения, будет предпочтительным.

Рис. 3.3.2. Один и тот же импульс, полученный через УС-23 и через БИТС2-12

Шаг (3.1) Для выполнения коммутации сперва определяется, по каким из трёх потоков идёт нестационарный, то есть, изменяющийся сигнал. Сигнал считается нестационарным, если

$$T - T\left(\eta_s^{тран}\right) < 0,7\,,$$

$$T - T\left(\eta_s^{tmx}\right) < 1,5\,,$$

$$T - T\left(\eta_s^{фал}\right) < 0,7$$

где T – текущее время в секундах, $T\left(\eta_i\right)$ – время значения η_i, а η_s – последнее существенное значение, такое, что:

$$\left|\eta_s - \eta_{s-1}\right| > 1,5\ \%.$$

На языке анализа ТМИ данная проверка выполняется следующим кодом:

```
НаВремя(uint time)
{
01 естьТМИТ = time - временаПоследнегоСущественногоЗнач[0] < 700;
02 естьТМИУ = time - временаПоследнегоСущественногоЗнач[1] < 1500;
03 естьТМИМ2 = time - временаПоследнегоСущественногоЗнач[2] < 700;
}
```

Шаг (3.2) Среди нестационарных сигналов выбирается сигнал с наивысшим приоритетом в соответствии с таблицей 3.3.1 и формируется на выходе алгоритма. Если тип сигнала не изменился, просто выдаётся очередное значение. Если тип

сигнала при очередной выдаче поменялся, выполняется процедура переключения на другой тип сигнала. Переключение может выполняться на сигнал с большим или с меньшим приоритетом.

Таблица 3.3.1. Приоритеты сигналов ЭКГ.

Приоритет	Источник	Пояснения
Высший	Электрофал (МИМ2), $\eta_i^{фал}$	Этот сигнал наименее подвержен радиошумам. Если он доступен, то есть это этап шлюзования, то следует брать его, так как другие сигналы на этом участке заметно зашумлены
Средний	УС-23 (ТМХ)	Этот сигнал в отличие от двух других передаётся с наивысшей частотой и имеет самые продолжительные зоны радиовидимости
Низкий	Транзит	Этот сигнал используется, когда остальные недоступны. Например, в зоне Российских НПРС, когда возникают перерывы в приёме ТМХ

Переключение на более высокий приоритет происходит в момент, когда лучший сигнал отсутствовал и начал поступать. В этом случае, как правило, наблюдается одновременное поступление двух сигналов, и алгоритм обеспечивает пропуск 1 секунды сигнала, чтобы два этих сигнала не слились в произвольном месте. Если это не делать, два разных сигнала соединятся в произвольном месте, и могут сформировать ложный фрагмент ЭКГ. Глядя на рис. 3.3.3, можно представить, какой получится результирующий сигнал, если η_i^{tmx} и $\eta_i^{фал}$ объединить в том месте, где начал поступать $\eta_i^{фал}$ (коричневая линия).

На рис. 3.3.4 приведён пример переключения сигнала. До времени 15:40:27 поступали сигналы от УС-23 и электрофала. В соответствии с таблицей 3.3.1 выбирался сигнал от электрофала (параметр ТЕКЭКГ1=2 на нижнем графике). В 15:40:51 после перерыва начал поступать сигнал от УС-23, который сразу попал в скоммутированный сигнал (параметр ЭКГ-СК1 на 3 графике). В 15:41:33 начал поступать сигнал от электрофала. Так как у него приоритет выше, этот сигнал был выбран текущим, а в результирующем сигнале была выдержана секундная пауза. Параметр ТЕКЭКГ1 указывает на номер потока, выбранного в качестве текущего.

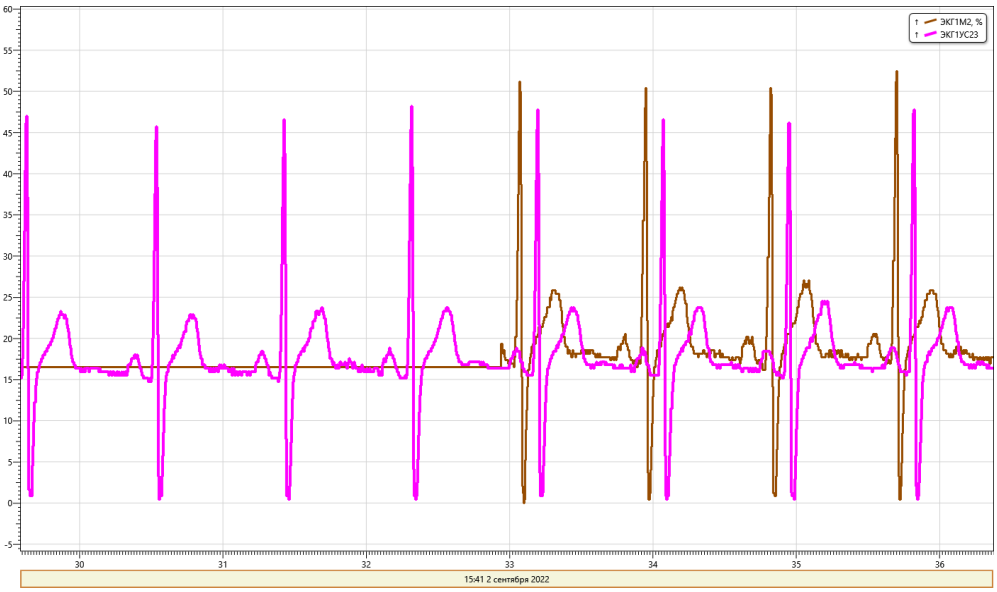


Рис. 3.3.3. ЭКГ с УС-23 и электрофала на одном полотне

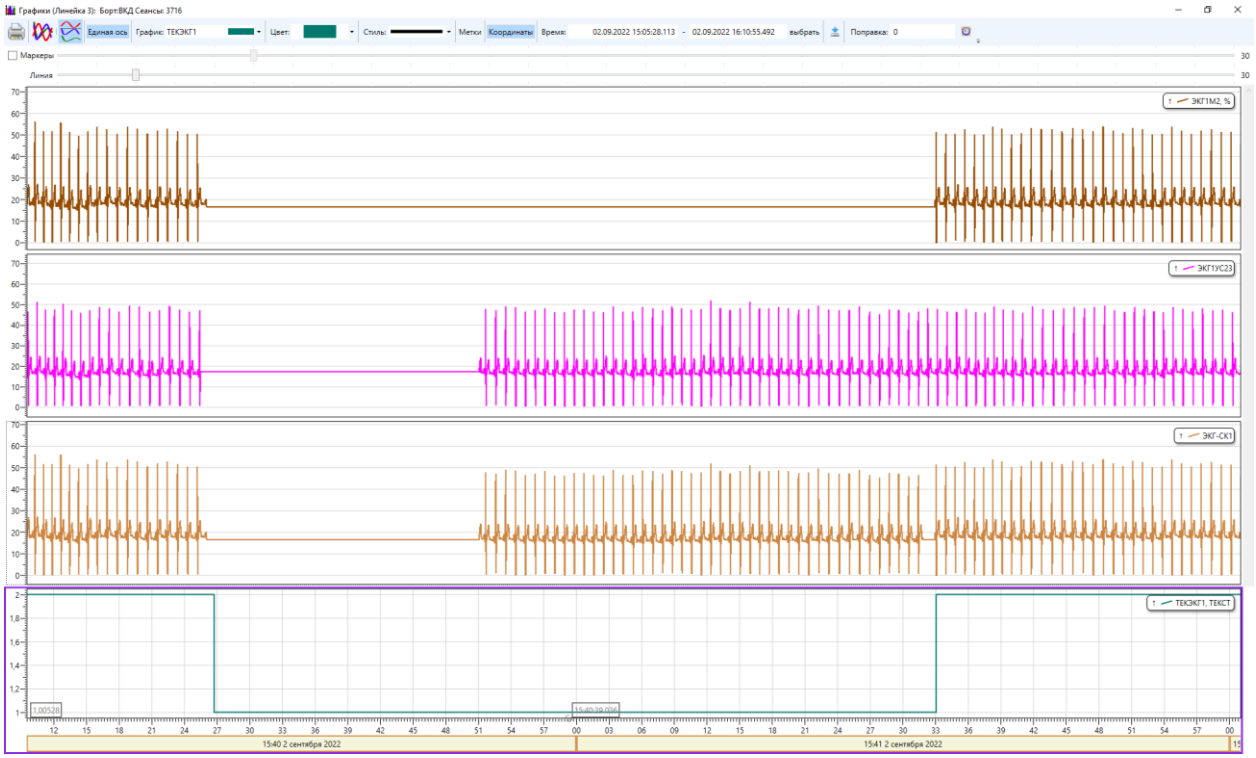


Рис. 3.3.4. Пример переключения сигнала в алгоритме коммутации.

Переключение на более низкий приоритет осуществляется, когда поступавший ранее сигнал некоторое время отсутствует. В этом случае также отсчитывается 1 секунда от последнего полученного значения, а затем выдаются значения нового сигнала, накапливаемые в небольшом буфере.

Приведённый метод позволяет получить единый скоммутированный поток ЭКГ по каждому скафандру, используя наиболее качественные данные.

Дальнейшая обработка ЭКГ и отображение сигнала выполняются на объединённом, скоммутированном потоке.

3.3.4. Восстановление пропущенных значений

Шаг (4.1) Обнаружение и восстановление пропущенных, отбракованных и сжатых значений осуществляется для обеспечения работы последующих алгоритмов обработки. На визуализацию сигнала пропуски отдельных значений абсолютно не влияют, но для обработки они существенны. Причинами таких пропусков могут быть:

- потеря/отбраковка одного или нескольких ТМ-кадров из-за шумов в радиолинии;
- потеря одного или нескольких пакетов УС-23 при передаче по американским каналам связи;
- потеря одного или нескольких транспортных пакетов ТМИ при передаче по наземным линиям связи;
- отбраковка сбойных измерений (по признаку чётности);
- сжатие повторяющихся значений.

В [1] предлагается метод восстановления пропущенных или искажённых значений телеметрических параметров на этапе предобработки. В общем случае представление пользователю таких "восстановленных" значений является крайне опасной процедурой. Ошибочно восстановленные данные могут привести к ошибочным заключениям и действиям группы управления. Максимальное доверие к телеметрическим данным требуется во время анализа аварийной ситуации специалистами госкомиссии, так как результатом работы госкомиссии являются выводы о причинах аварийной ситуации.

Единственным приемлемым применением методов восстановления пропущенных и искажённых телеметрических значений видится ситуация, когда для некоторого вторичного алгоритма обработки или анализа ТМИ требуется непрерывная последовательность значений ТМ-параметра. В этом случае интеллектуальное восстановление данных может работать лучше, чем простейшее

заполнение пропусков предыдущим достоверным значением. Например, указанное восстановление в будущем можно использовать для предварительной обработки значений электрокардиограммы (ЭКГ) космонавта перед подачей в алгоритм вычисления частоты сердечных сокращений (ЧСС). В данной задаче ошибочно восстановленные данные не приведут ни к каким негативным последствиям. В худшем случае одно из вычисленных значений ЧСС будет искажено. Однако в текущей реализации применяется повторение последнего полученного значения.

Алгоритм обеспечивает следование значений с шагом времени $\Delta T = T(\eta_i) - T(\eta_{i-1}) = 0,005$ с для УС-23 и $\Delta T = T(\eta_i) - T(\eta_{i-1}) = 0,005303$ с для «Транзита» и электрофала. Обнаружив два значения η_{i-1}, η_i, таких, что

$$0,007\,\text{с} < T(\eta_i) - T(\eta_{i-1}) < 0,2\,\text{с}$$

Алгоритм формирует новые ТМ-значения, равные η_{i-1} с заданным временным шагом.

Код на языке анализа ТМИ записывается следующим образом:

```
01   if ((long)(экгВремя - прошлоеВремя) > 7 &&
02       (long)(экгВремя - прошлоеВремя) < 200)
03   {
04        ulong время = прошлоеВремя;
05        for (int i = 1; i < 100; ++i)
06        {
07             прошлоеВремя = время + (uint)(i * ΔT);
08             if (прошлоеВремя >= экгВремя - 4)
09                  break;
10             ЭКГ = прошлоеЗначение;
11             ЭКГ . ВРЕМЯ = прошлоеВремя;
12             ЭКГ . ВЫДАТЬ();
13        }
14   }
15   прошлоеЗначение = ЭКГ;
16   прошлоеВремя = экгВремя;
```

3.3.5. Устранение тренда

Шаг (5) В [70] предлагаются различные алгоритмы для предобработки телеметрических значений перед подачей в нейронную сеть. Среди них: нормализация на стандартное отклонение, нормализация на максимальное абсолютное отклонение от показателя среднего значения. Там же предложен

алгоритм кластеризации состояний бортовой аппаратуры посредством обучения карты Кохонена. В рассматриваемой задаче в качестве алгоритмов предобработки телеметрических данных используется калибрование значений в диапазоне от 0% до 100% и устранение тренда сигнала.

Устранение тренда выполняется с целью стабилизировать уровень поступающего сигнала и сделать работу следующих алгоритмов более устойчивой. Отметим, что сигнал ЭКГ можно условно разбить на две составляющих: преобладающий сигнал низкого уровня, и кратковременные всплески сигнала – зубцы «R», которые могут иметь разный уровень. Для стабилизации сигнала будем определять тренд только сигнала низкого уровня, игнорируя зубцы. Тренд будем вычислять путём кусочно-линейной аппроксимации методом наименьших квадратов.

Шаг (5.1) Входной сигнал разбивается на интервалы $\eta_i \in I$ длиной $\Delta T^{tr} = 0,700\,\text{с}$, содержащих порядка $\|I\| = L = \dfrac{\Delta T^{tr}}{\Delta T} \approx 140$ точек, и все вычисления выполняются на этих интервалах, сохраняемых в массив data. Для того, чтобы разделить данные на сигнал низкого уровня $I^{низ}$ и пики $I^{пик}$ $\left(I = I^{низ} \cup I^{пик}, I^{низ} \cap I^{пик} = \varnothing\right)$ выполняется сортировка данных по возрастанию в отдельном массиве $\eta'_i \in \text{data2}$. Определим **верхнее предельное значение** сигнала на интервале как значение, находящееся на позиции:

$$i^{впз} = L \cdot 0,9\,,$$

$$\eta^{впз} = \text{data2}\left[i^{впз}\right]. \qquad (3.3.4)$$

Также запомним среднее (медианное) значение

$$\eta^{ср} = \text{data2}\left[\frac{L}{2}\right].$$

Тогда

$$\forall i_1 < i^{впз} : \eta_{i_1} < \eta^{впз} \Rightarrow \eta_{i_1} \in I^{низ}$$

$$\forall i_2 \geq \eta^{впз} : \eta_{i_2} \geq \eta^{впз} \Rightarrow \eta_{i_2} \in I^{пик}$$

То есть, верхнее предельное значение – это значение, которое больше 90% значений рассматриваемого интервала *I*.

Приведём фрагмент кода на языке анализа ТМИ, выполняющий указанные действия.

```
01 КопироватьМассив(data, data2, dataCount);
02 СортироватьМассив(data2, dataCount);
03 double верхнееПредельноеЗначение = data2[dataCount * 0.9];
04 double среднееЗначение = data2[dataCount / 2];
```

Шаг (5.2) Вычисление тренда. Формируем новый массив значений:

$$\text{data3} = \{\eta_i''\}, \quad \eta_i'' = \begin{cases} \eta_i, & \eta_i < \eta^{впз} \\ \eta^{ср}, & \eta_i \geq \eta^{впз} \end{cases},$$

в котором значения больше $\eta^{впз}$ заменены на средние $\eta^{ср}$. Для data3 будем определять тренд методом наименьших квадратов путём решения системы линейных уравнений (СЛАУ) [20, 29]. Значения $(\eta_i'', T(\eta_i''))$ аппроксимируются линейной функцией

$$f(t_i) = a_0 + a_1 \cdot t_i \tag{3.3.5}$$

с критерием

$$\sum_i \left(\eta_i'' - f\left(T(\eta_i'')\right)\right)^2 \to \min.$$

Здесь в качестве параметра времени используется время, прошедшее от начала интервала в секундах

$$t_i = T(\eta_i) - T(\eta_0). \tag{3.3.6}$$

Приведём пример кода на языке анализа ТМИ, решающего СЛАУ.

```
01 SolveSlau(double A[], double C[], double res[], int m)
02 {
03    int used[m];
04    int por[m];
05    for (int col = 0; col < m; ++col)
06    {
07       int line = -1;
08       for (int i = 0; i < m; ++i)
09          if (!used[i] && (line == -1 || Abs(A[i * m + col]) >
Abs(A[line * m + col])))
10             line = i;
11       used[line] = true;
```

```
12          por[col] = line;
13          for (int i = 0; i < m; ++i)
14             if (i != line)
15             {
16                double k = A[i * m + col] / A[line * m + col];
17                for (int j = col; j < m; ++j)
18                     A[i * m + j] -= k * A[line * m + j];
19                C[i] -= k * C[line];
20             }
21       }
22       for (int i = 0; i < m; ++i)
23          res[i] = C[por[i]] / A[por[i] * m + i];
24    }
```

Шаг (5.3) Если в качестве тренда использовать функцию (3.3.5), в общем случае линии тренда на границе двух соседних интервалов $I_{(j-1)}$ и $I_{(j)}$ не совпадут (рис. 3.3.5, оранжевая линия), и после устранения такого тренда, сигнал получится разрывным (нижний график), что является недопустимым. Для решения этой проблемы конец отрезка тренда интервала $I_{(j-1)}$ совмещается с началом отрезка тренда следующего интервала $I_{(j)}$ путём перемещения их в среднюю точку (рис. 3.3.6).

Пусть $a_{0,(j-1)}$, $a_{1,(j-1)}$ – коэффициенты тренда интервала $I_{(j-1)}$, а $a_{0,(j)}$, $a_{1,(j)}$ – коэффициенты тренда интервала $I_{(j)}$, измерения предыдущего и текущего интервалов соответственно: $\eta_{i,(j-1)} \in I_{(j-1)}$, $\eta_{i,(j)} \in I_{(j)}$. Тогда значение предыдущего тренда в первой точке текущего интервала согласно формулам (3.3.5), (3.3.6) будет:

$$y_1 = a_{0,(j-1)} + a_{1,(j-1)} \cdot \left(T\left(\eta_{0,(j)}\right) - T\left(\eta_{0,(j-1)}\right) \right).$$

Значение текущего тренда в первой точке интервала равно

$$y_2 = a_{0,(j)}.$$

Среднее значение равно

$$m = \frac{y_1 + y_2}{2}.$$

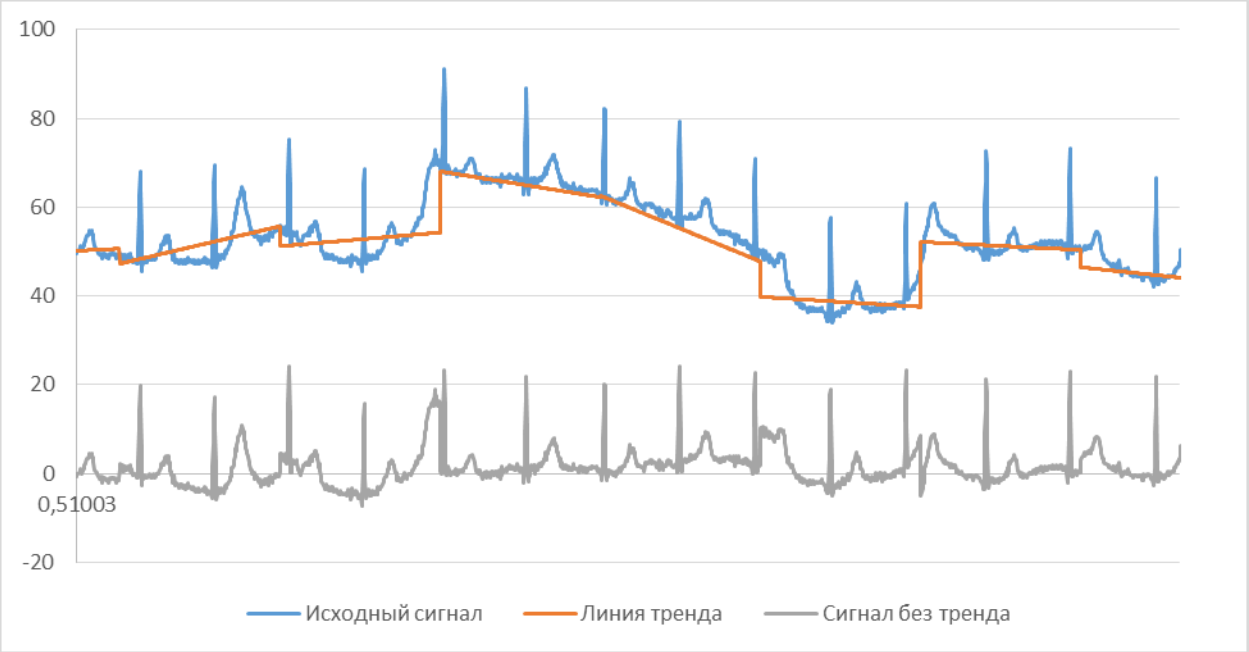


Рис. 3.3.5. Пример отрезков тренда с несовпадающими концами.

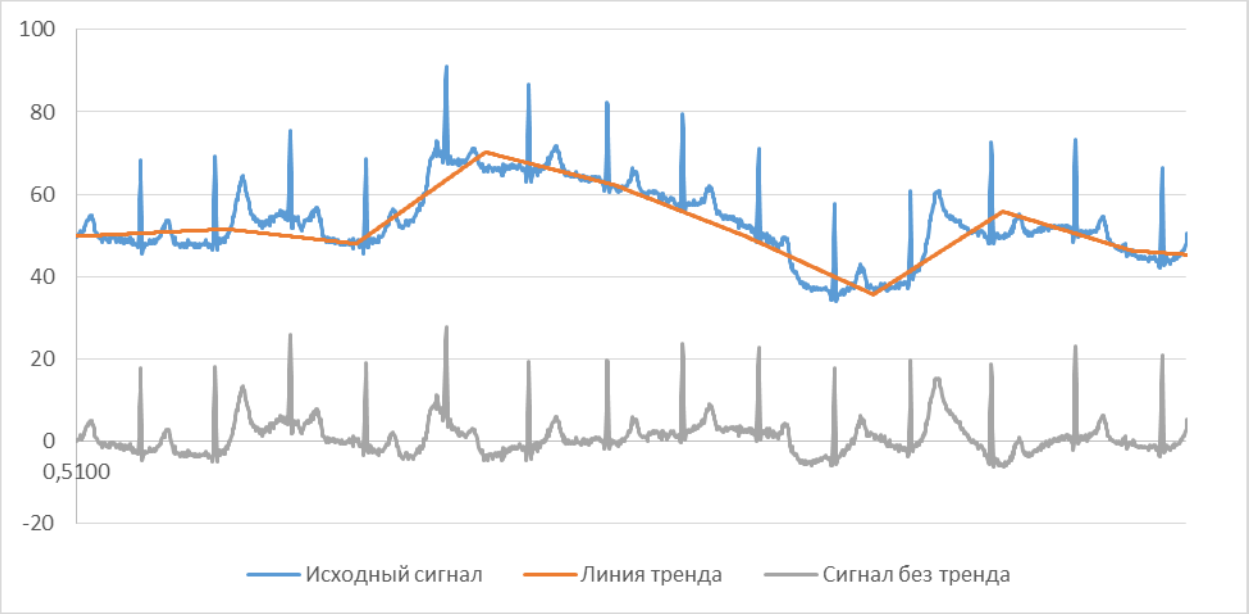


Рис. 3.3.6. Пример отрезков тренда с совмещёнными концами.

После совмещения концов отрезков трендов в средней точке коэффициенты будут равны:

$$a'_{0,(j-1)} = a_{0,(j-1)};$$

$$a'_{1,(j-1)} = \frac{m - a_{0,(j-1)}}{T\left(\eta_{0,(j)}\right) - T\left(\eta_{0,(j-1)}\right)};$$

$$a'_{0,(j)} = m;$$

$$a'_{1,(j)} = \frac{a_{0,(j)} + a_{1,(j)}\left(T\left(\eta_{L-1,(j)}\right) - T\left(\eta_{0,(j)}\right)\right) - m}{T\left(\eta_{L-1,(j)}\right) - T\left(\eta_{0,(j)}\right)}.$$

Здесь $\eta_{L-1,(j)} \in I_{(j)}$ – последнее измерение интервала.

Опытным путём установлено, что полученная описанным алгоритмом ломанная линия тренда достаточно хорошо повторяет сигнал ЭКГ низкого уровня (без зубцов «R») (рис. 3.3.7). Недостатком данного алгоритма является необходимость задержки выдачи данных на длину двух интервалов $2 \cdot \Delta T^{tr} = 1,400$ с, что может быть заметно специалистам группы медицинского обеспечения при проведении анализа ЭКГ в реальном времени.

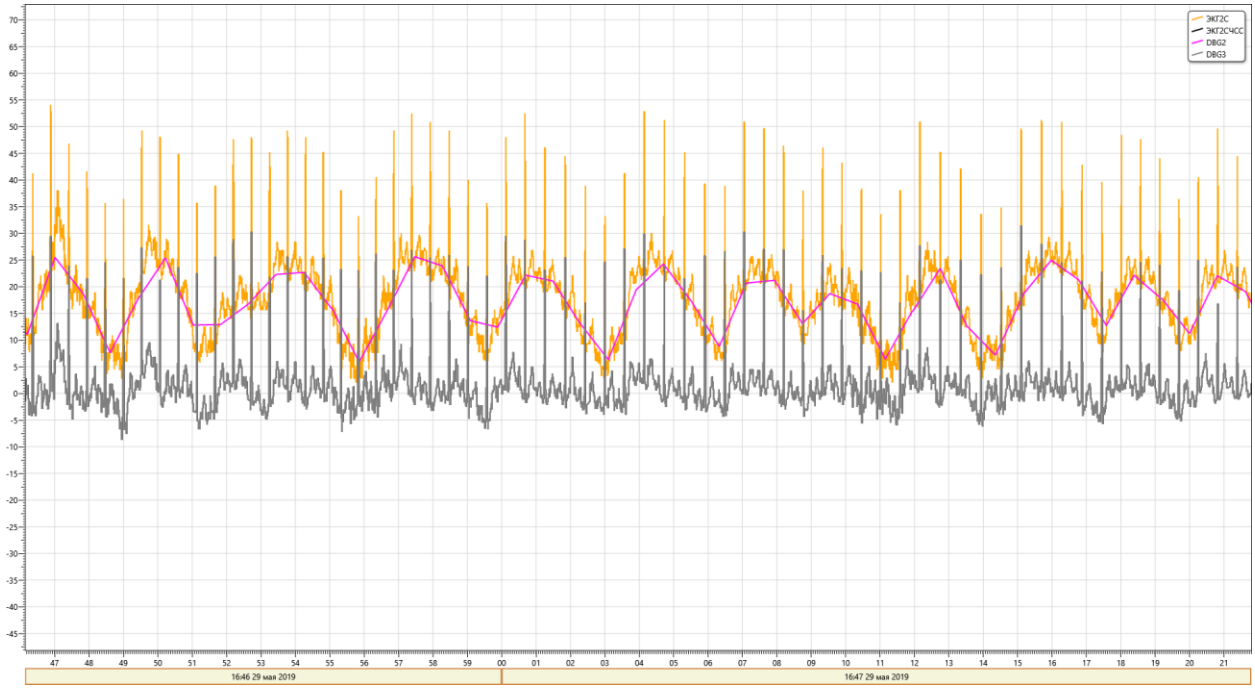


Рис. 3.3.7. ЭКГ, линия тренда и сигнал после устранения тренда.

Шаг (5.4) Итак, устранение тренда для измерений $\eta_i \in I$ выглядит следующим образом:

$$\eta'_i = \eta_i - \left(a_0 + a_1 \cdot \left(T\left(\eta_i\right) - T\left(\eta_0\right)\right)\right),$$

или на языке анализа ТМИ:

```
01 for (int i = 0; i < lastDataCount; ++i)
02 {
03     ЭКГ = lastData[i] -
          (lastCoefficients[0] +
          lastCoefficients[1] *
```

```
            (int)(lastTimes[i] - lastTimes[0]) / 1000.0);
04      ЭКГ . ВРЕМЯ = lastTimes[i];
05      ЭКГ . Выдать();
06  }
```

Пример сигнала до и после устранения тренда приведён на рис. 3.3.7. Сиреневой линий выведена вычисленная линия тренда.

3.3.6. Поиск зубцов «R».

Задача этого этапа – достоверно найти в сигнале ЭКГ зубцы «R», на основе которых вычисляется частота сердечных сокращений (ЧСС). В связи с тем, что в сигнале ЭКГ от скафандров встречаются помехи, содержащие много резких всплесков, похожих на зубцы «R», на первом шаге данного алгоритма необходимо разделить сигнал на содержащий ЭКГ и не содержащий.

Как и на предыдущих этапах, входной сигнал разбивается на интервалы $\eta_i \in I$ длиной $\Delta T^{tr} = 1{,}8$ с, содержащие порядка $\|I\| = L = \frac{\Delta T^{tr}}{\Delta T} \approx 340 \div 360$ точек. Длина интервала подобрана таким образом, чтобы он содержал по крайней мере один зубец «R». Кроме того, выполняется временна́я проверка. Если данные не поступают длительное время, а накоплена уже хотя бы половина интервала, то эти данные принудительно отправляются на обработку.

Шаг (6.1) Задачу определения участков, содержащих ЭКГ, невозможно строго формализовать и на приемлемом уровне реализовать алгоритмическими методами, поэтому для решения этой задачи будем использовать методы искусственного интеллекта, основанные на нейросетях типа многослойный персептрон [11, 68, 90]. Поскольку входные данные разделяются на интервалы произвольным образом, и ЧСС космонавтов является переменной величиной, подавать их непосредственно в нейросеть затруднительно, так как она не сможет сопоставить даже два очень похожих набора данных, сдвинутых друг относительно друга на несколько точек. Потребуется либо существенное усложнение архитектуры нейросети и увеличение объёма обучающих примеров, либо использование нейросетей другого типа. Учитывая особенности сигнала ЭКГ, предлагается входные данные подготовить особым образом перед подачей их в нейросеть, избавляясь от неопределённости нарезания входного сигнала на

интервалы. Ключевой особенностью ЭКГ с устранённым трендом является следующая: большинство измерений будут иметь значения в районе нуля, и лишь небольшая их часть (зубцы «R») будет иметь большие значения.

Отсортируем данные по возрастанию в отдельном массиве $\eta'_i \in \text{data2}$. Мы получим монотонный ряд медленноменяющихся данных. На рис. 3.3.8 приведены примеры нескольких таких рядов. Легко заметить, что у сигналов ЭКГ значения быстрее меняются в области младших и старших значений.

В соответствии с формулой (3.3.4) определим верхнее предельное значение $\eta^{впз}$, которое нам потребуется в дальнейшем.

Уменьшим размерность задачи, объединив каждые 5 соседних точек.

$$\eta''_j = \frac{\eta'_{5j} + \eta'_{5j+1} + \eta'_{5j+2} + \eta'_{5j+3} + \eta'_{5j+4}}{5}; \tag{3.3.7}$$

Чтобы уменьшить зависимость от вариативности амплитуды сигнала, будем работать с производной полученных данных $d\eta''_j \in \text{data/5}$ (рис. 3.3.9).

$$\mathrm{d}\eta''_j = \eta''_j - \eta''_{j-1}. \tag{3.3.8}$$

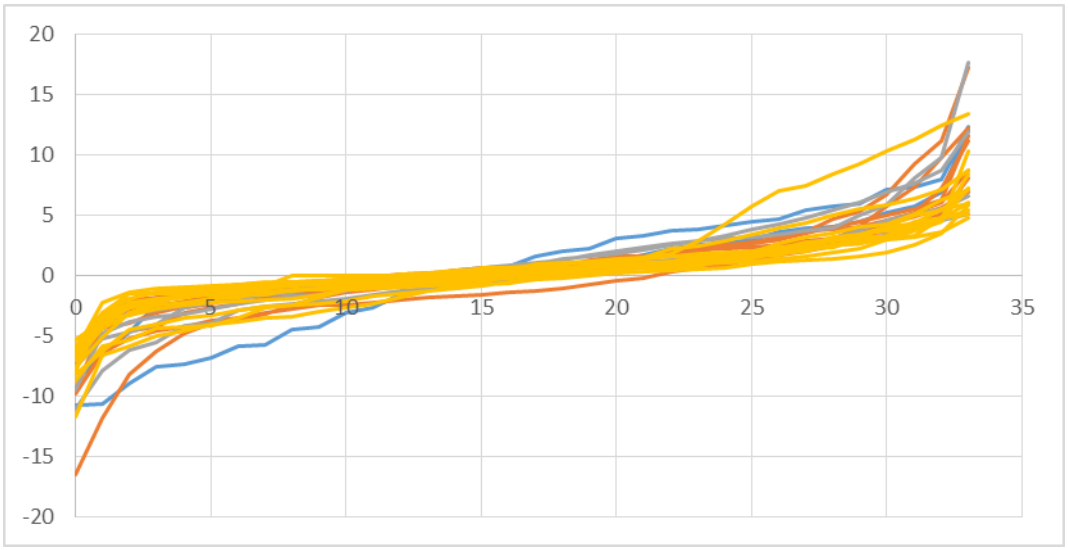


Рис. 3.3.8. Отсортированные значения фрагментов ЭКГ

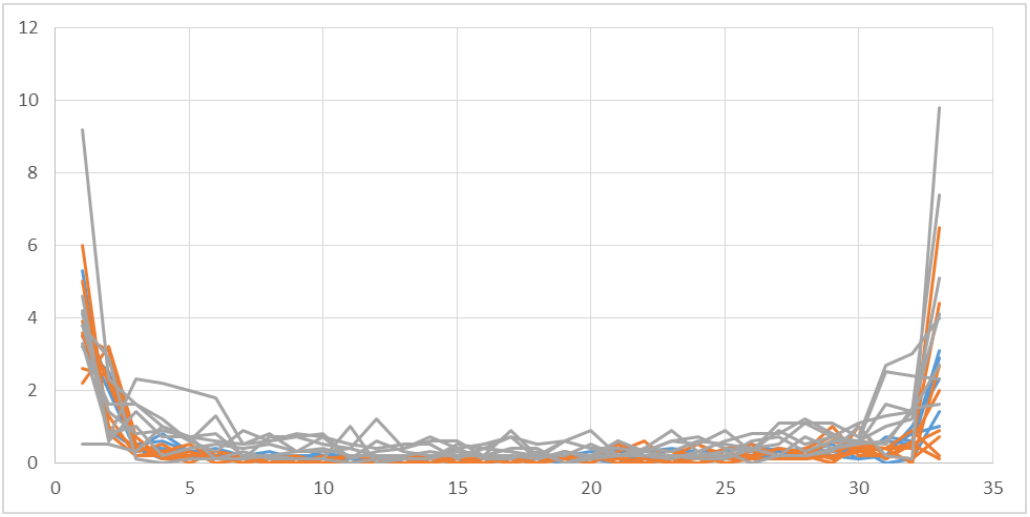


Рис. 3.3.9. Значения $d\eta_j''$ для фрагментов ЭКГ

Фрагмент кода на языке анализа ТМИ, выполняющий преобразования (3.3.7) в строках 3, 6 и (3.3.8) в строке 7, приведён далее.

```
01 КопироватьМассив(data, data2, dataCount);
02 СортироватьМассив(data2, dataCount);
03 double lastMidle = (data2[0] + data2[1] + data2[2] +
   data2[3] + data2[4]) / 5;
04 for (int i = 1; i < dataCount / 5; ++i)
05 {
06    double midle = (data2[i * 5] + data2[i * 5 + 1] +
      data2[i * 5 + 2] + data2[i * 5 + 3] + data2[i * 5 + 4]) / 5;
07    data/5[i - 1] = midle - lastMidle;
08    lastMidle = midle;
09 }
```

Для проверки данных data/5 была создана нейросеть nn_1 топологии (67, 30, 13, 1), где 67 – это размерность входного слоя, соответствующая размеру $\|\text{data/5}\| = 67$, 1 – это размерность выходного слоя. То есть, на выходе нейросети один нейрон, формирующий результат: соответствуют входные данные ЭКГ или нет. В качестве активационной функции использовалась логистическая

$$\varphi(x) = \frac{1}{1+\ell^{-x}}. \qquad (3.3.9)$$

Нейросеть обучена на множестве реальных показаний космонавтов, полученных в процессе автономного полёта на ТПК «Союз», проведения медицинских обследований на МКС и выполнения внекорабельной деятельности в скафандрах «Орлан». В качестве учебных примеров использовалось множество интервалов data/5, полученных описанным выше алгоритмом. Далее проводилась

экспериментальная отработка, в ходе которой оценивалась корректность работы нейросети. Участки данных, на которых нейросеть ошибалась или показывала слабую уверенность, добавлялись к учебным данным. В результате обучения нейросети получены матрицы весовых коэффициентов, которые включены в программу на языке анализа ТМИ в качестве параметров `n1-layer1-w`, `n1-layer2-w` и `n1-layer3-w`. Подробнее принцип работы с нейросетями описан в разделах 4.3.3.3-4.3.3.6. Здесь приведём код проверки data/5 с помощью нейросети.

```
01 double Logistic(double x)
02 {
03    return 1 / (1 + EXP(-x));
04 }

05 double ПроверитьИзмеренияУчастка(double input[])
06 {
07    double layer2[31];
08    double layer3[14];
09    double res[1];
10    input[67] = layer2[30] = layer3[13] = 1;

11    ПроизведениеМатриц(n1-layer1-w, input, layer2, 30, 68, 1);
12    for (int i = 0; i < 30; ++i)
13       layer2data[i] = Logistic(layer2[i]);

14    ПроизведениеМатриц(n1-layer2-w, layer2, layer3, 13, 31, 1);
15    for (int i = 0; i < 13; ++i)
16       layer3data[i] = Logistic(layer3[i]);

17    ПроизведениеМатриц(n1-layer3-w, layer3, res, 1, 14, 1);
18    return Logistic(res[0]);
19 }
```

Результатом работы нейросети является скалярное значение в диапазоне $nn_1(\text{data}/5) \in [0;1]$. Число, ближе к 1 означает, что проверяемые данные похожи на сигнал, содержащий ЭКГ, а ближе к 0 – что не похоже. Опытным путём выбран критерий. Если $nn_1(\text{data}/5) \geq 0,75$, то данные содержат ЭКГ, и их можно дальше обрабатывать.

Поиск зубцов «R» в интервалах ЭКГ выполняется двухэтапным методом. Сначала находятся всплески, похожие на зубец, а затем эти данные проверяются с помощью нейросети.

Шаг (6.2) Поиск и идентификация всплесков. Последовательно двигаясь по массиву, находим серию данных:

$$\exists j,k \in N : \begin{cases} k - j - 1 \geq 3 \\ \eta_{j-1} < \eta^{впз} \\ \eta_i \geq \eta^{впз},\ j \leq i \leq k \\ \eta_{k+1} < \eta^{впз} \end{cases}.$$

То есть, минимум 3 идущие подряд значения больше $\eta^{впз}$, окружённые маленькими значениями. Далее находим позицию максимального среди них значения:

$$\exists m \in N : \begin{cases} j \leq m \leq k \\ \eta_m \geq \eta_i, \quad j \leq i \leq k \end{cases}.$$

Если идёт подряд несколько одинаковых максимальных значений, выбираем из них среднее. Эту точку принимаем за середину всплеска, который будем проверять на соответствие зубцу «R» ЭКГ с помощью нейросети nn_2.

Для подачи в нейросеть формируем вектор:

$$\text{data}^{всплеск} = \begin{bmatrix} \text{data}[m-7] \\ \vdots \\ \text{data}[m] \\ \vdots \\ \text{data}[m+7] \\ \eta^{впз} \end{bmatrix}.$$

На 15 нейронов подаются значения вокруг найденного всплеска. Если значений недостаточно (например, $m - 7 < 0$), размножается крайнее доступное значение. На последний входной нейрон подаётся $\eta^{впз}$, которое поможет нейросети nn_2 понять, какой критерий для обнаружения всплеска мы использовали на данном интервале, какой величины значения здесь встречаются.

Мы нашли R_j зубец «R» на интервале *I*, если

$$nn_2\left(data^{всплеск}\right) > 0{,}85. \qquad (3.3.10)$$

Зубец «R» характеризуется величиной (которая данным алгоритмом не используется) и временем $T\left(R_j\right)$. Найденные зубцы выдаются на выход алгоритма,

а поиск зубцов на интервале I продолжается.

Приведём фрагмент кода на языке анализа ТМИ, выполняющего данную проверку и формирование результата.

```
01 int k = 0;
02 for (int j = pos - 7; j < 0; ++j)
03    dataПик[k ++] = data[0];
04 КопироватьМассив(data, MAX(0, pos - 7), dataПик, k,
      MIN(dataCount, pos + 8) - MAX(0, pos - 7));
05 k += MIN(dataCount, pos + 8) - MAX(0, pos - 7);
06 for (int j = dataCount; j < pos + 8; ++j)
07    dataПик[k] = data[dataCount - 1];
08 dataПик[15] = верхнееПредельноеЗначение;
09 double результатНейросети = ПроверитьИзмеренияПика(dataПик);
10 if (результатНейросети > 0.85)
11 {
12    ПИК = пиковоеЗначение;
13    ПИК . ВРЕМЯ = times[pos];
14    ПИК . ВЫДАТЬ();
15 }
```

В строках 02-07 в дополнительный массив `dataПик` копируются значения слева и справа от предполагаемого зубца «R». Если данных слева или справа не хватает, размножаются крайние значения. В строке 09 вызывается нейросеть для проверки сформированного массива. Код исполнения нейросети аналогичен коду исполнения $nn_1(\text{data}/5)$. В строках 12-14 выдаётся результат.

Если в конце интервала I содержатся большие значения, они могут принадлежать очередному зубцу «R», для идентификации которого данных пока недостаточно:

$$\exists k < L : \forall i \in [k, L-1]\ data[i] \geq \eta^{\text{впз}}.$$

Такие данные не отбрасываются вместе со всем интервалом I, а образуют начало следующего интервала.

3.3.7. Вычисление ЧСС

(7) Частота сердечных сокращений (ЧСС) – это физическая величина, получаемая в результате измерения числа сокращений сердца в единицу времени. Для вычисления ЧСС в реальном времени – мгновенного ЧСС – используются временные интервалы между зубцами «R», найденными на предыдущем шаге алгоритма:

$$\Delta t_j = T\left(R_j\right) - T\left(R_{j-1}\right). \qquad (3.3.11)$$

Опытным путём установлено, что в общем случае не удаётся найти все зубцы «R»:

1. В местах сильного зашумления или пропадания ТМИ зубец «R» может быть пропущен;
2. Если зубец «R» имел необычную форму (из-за индивидуальных особенностей сердечного ритма, крепления датчиков, или привнесённых помех), нейросеть nn_2 может не узнать его.
3. Наводки, вызванные мышечной активностью космонавта, также могут видоизменить зубец «R» до неузнаваемости.
4. Шумы и регистрация мышечной активности могут случайным образом сформировать всплески, которые будут приняты за зубец «R».

Таким образом, настоящий алгоритм должен быть устойчив к пропаданиям или появлениям лишних зубцов «R». В основу критерия, обеспечивающего устойчивость работы алгоритма, положено предположение о том, что ЧСС космонавта не может меняться резко, то есть соседние интервалы между зубцами «R» не могут отличаться более, чем на 20% (подобрано опытным путём):

$$\left|\frac{\Delta t_j - \Delta t_{j-1}}{\Delta t_{j-1}}\right| \le 0,2 \qquad (3.3.12)$$

Получая на вход зубец R_j, алгоритм хранит следующие данные:

dataR – множество из нескольких последних $T\left(R_j\right)$, причём $\mathrm{dataR}[0]$ – последний, полученный зубец, $\mathrm{dataR}[1]$ – предпоследний и т. д.

$\Delta t^{тек}$ – текущий интервал между зубцами «R», достоверно определённый алгоритмом;

Собрав 2 зубца «R», алгоритм проверяет интервал Δt_j на достоверность, при условии, что текущий интервал $\Delta t^{тек}$ определён. Новый интервал достоверный в соответствии с формулой (3.3.12), если

$$\left|\frac{\left(\Delta t_j - \Delta t^{тек}\right)}{\Delta t^{тек}}\right| < 0,2$$

Тогда Δt_j выдаём на выход алгоритма, и назначаем новый текущий интервал $\Delta t^{тек} = \Delta t_j$.

Если был пропущен один зубец «R», новый интервал будет вдвое больше корректного:

$$\left|\frac{\frac{\Delta t_j}{2} - \Delta t^{тек}}{\Delta t^{тек}}\right| < 0,2$$

Также обязательно проверяется, что предыдущий интервал не был двойным – маловероятно получить подряд два двойных интервала, скорее всего мы ошиблись с предыдущим. Если все условия выполняются, результатом работы алгоритма будет $\frac{\Delta t_j}{2}$, а новый текущий интервал назначается $\Delta t^{тек} = \frac{\Delta t_j}{2}$.

Если новый интервал Δt_j не был определён как корректный на основе текущего или текущий ещё не определён, будем оценивать два последних интервала:

$$\left|\frac{\Delta t_j - \Delta t_{j-1}}{\max\left(\Delta t_j, \Delta t_{j-1}\right)}\right| < 0,2,$$

$$0,3 < \Delta t_j < 2,0,$$

$$0,3 < \Delta t_{j-1} < 2,0.$$

Здесь числа выбраны исходя из допустимой ЧСС от 30 до 200 ударов в минуту.

Если указанные критерии выполняются, считаем, что оба интервала подтверждают друг друга и являются достоверными. Выдаём на выход из алгоритма Δt_{j-1}, Δt_j, а текущим назначаем $\Delta t^{тек} = \Delta t_j$.

Аналогичным образом проверяются случаи, когда интервал Δt_{j-1} является двойным:

$$\left| \frac{\Delta t_j - \frac{\Delta t_{j-1}}{2}}{\max\left(\frac{\Delta t_{j-1}}{2}, \Delta t_j \right)} \right| < 0,2$$

или интервал Δt_j является двойным:

$$\left| \frac{\frac{\Delta t_j}{2} - \Delta t_{j-1}}{\max\left(\Delta t_{j-1}, \frac{\Delta t_j}{2} \right)} \right| < 0,2$$

Последним в алгоритме идёт преобразование, переводящее найденные интервалы в ЧСС:

$$ЧСС_i = \frac{60}{\Delta t_i}.$$

На языке анализа ТМИ все описанные алгоритмы записываются отдельным подпрограммами, а обработка ЭКГ каждого космонавта задаётся припысыванием каждому параметру ЭКГ цепочки из предварительно описанных алгоритмов.

```
@ (4) Восстановление пропущенных значений
ВосстановлениеПропусков = АНАЛИЗ(
   ВХПАР(АРГ double ЭКГ)
   ВЫХПАР(АРГ double ЧСС)
   …
);
@ (5) Устранение тренда ЭКГ
ВычислениеТрендаЭКГ = АНАЛИЗ(
   ВХВЫХПАР(АРГ double ЭКГ)
   …
);
@ (6) Поиск зубцов «R»
ВычислениеПиковЭКГ = АНАЛИЗ(
   ЗАПУСК(КАЖДЫЙ, НП, ИНСТРУКЦИЙ: 10000000)
   ВХПАР(АРГ double ЭКГ)
   ВЫХПАР(АРГ double ПИК)
   …
);
@ (7) Вычисляем ЧСС по двум последовательным интервалам (по 3 точкам)
ВычислениеЧСС = АНАЛИЗ(
   ВХПАР(АРГ double ИМПУЛЬС)
   ВЫХПАР(АРГ int ЧСС)
   …
);
```

```
ВосстановлениеПропусков (ЭКГ-СК1, ЭКГ-СК1-ЧСС);
ЭКГ-СК1-ЧСС = ВычислениеТрендаЭКГ(ЭКГ-СК1-ЧСС),
   ВычислениеПиковЭКГ(ЭКГ-СК1-ЧСС, ЭКГ-СК1-ЧСС),
   ВычислениеЧСС(ЭКГ-СК1-ЧСС, ЭКГ-СК1-ЧСС),
   ФОРМУЛА(, (_ЕСЛИ ВХОД = 0 _ТО 0 _ИНАЧЕ 60.0 / ВХОД _ВСЕ));

ВосстановлениеПропусков (ЭКГ-СК2, ЭКГ-СК2-ЧСС);
ЭКГ-СК2-ЧСС = ВычислениеТрендаЭКГ(ЭКГ-СК2-ЧСС),
   ВычислениеПиковЭКГ(ЭКГ-СК2-ЧСС, ЭКГ-СК2-ЧСС),
   ВычислениеЧСС(ЭКГ-СК2-ЧСС, ЭКГ-СК2-ЧСС),
   ФОРМУЛА(, (_ЕСЛИ ВХОД = 0 _ТО 0 _ИНАЧЕ 60.0 / ВХОД _ВСЕ));
```

Результаты нейросетевого анализа ЭКГ космонавта приведены на рис. 3.3.10, 3.3.11.

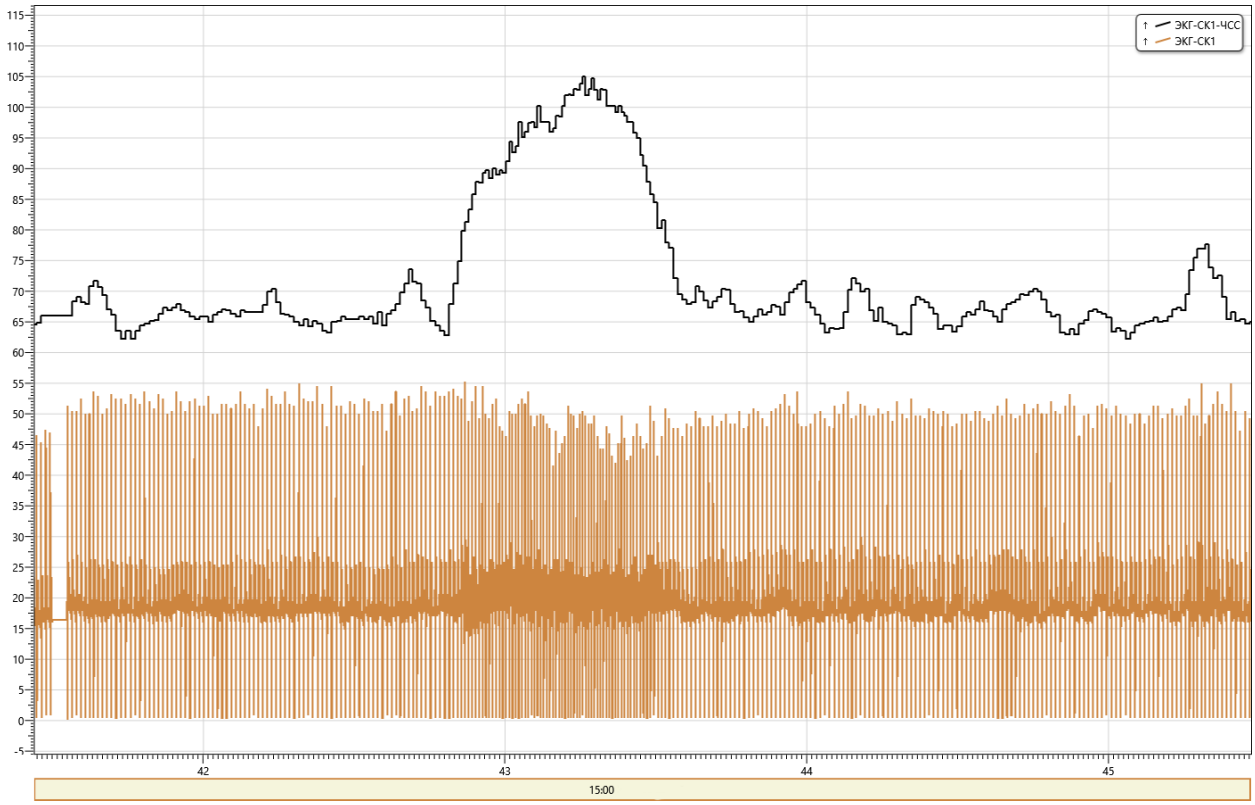


Рис. 3.3.10. Пример вычисленного ЧСС на длительном интервале времени

На рис. 3.3.12 приведён пример мнемосхемы контроля внекорабельной деятельности космонавтов. Здесь в реальном времени выводится ЭКГ, результаты вычисленной при помощи разработанной методики ЧСС, результаты расчёта суммарного уровня энерготрат и теплосъёма, а также значения основных параметров скафандра. Все расчёты выполняются алгоритмами, описанными в виде подпрограмм на разработанном языке анализа ТМИ. Сама мнемосхема реализована при помощи разработанных средств отображения мнемосхем, описанных в главе 3.2.

Рис. 3.3.11. Пример вычисленного ЧСС на коротком интервале времени

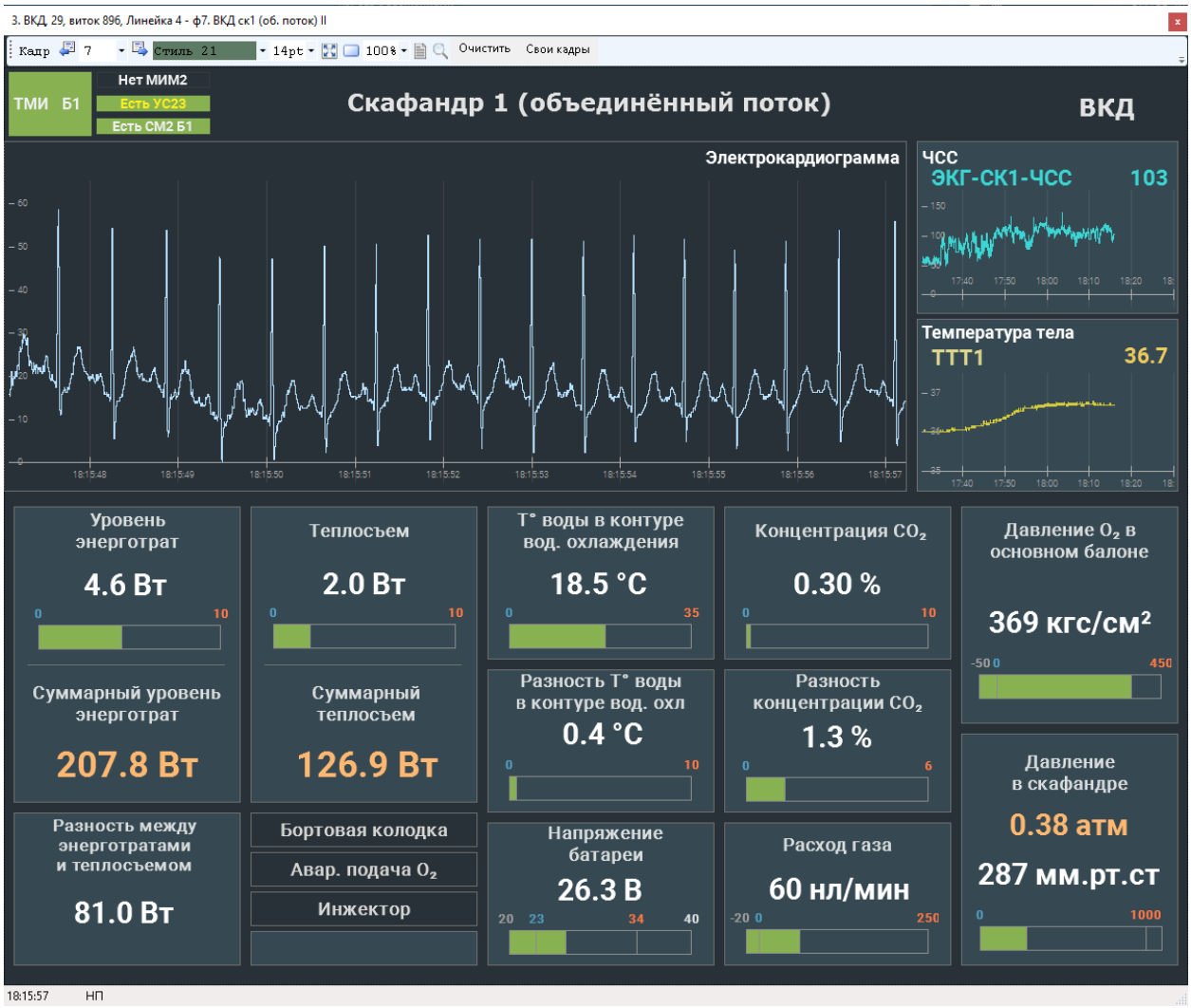


Рис. 3.3.12. Мнемосхема контроля ВКД.

3.3.8. Оценка методики

Результаты практической отработки методики на реальной ТМИ космонавтов в 2021-2022 гг. показали высокую надёжность и эффективность разработанной методики, устойчивость к пропаданиям и искажениям ТМИ, а также индивидуальным особенностям космонавтов. Специалисты группы медицинского обеспечения отмечают более стабильную работу разработанной методики в сравнении со штатным комплексом обработки ТМИ [26, 27]. В таблице 3.3.2 приведены результаты экспертной оценки разработанной нейросетевой методики анализа ТМИ, содержащей медицинские показания космонавтов. Оценка выполнена на реальных показаниях нескольких космонавтов.

Таблица 3.3.2. Результаты оценки нейросетевого алгоритма

Показатель	**Штатный комплекс**	**Нейросетевая методика**
Оцениваемый интервал	01:48:57	
Количество зубцов «R»	12036	
Распознано зубцов «R»	5583	11887
Ошибочные показания	862	28
Нераспознанные зубцы «R»	5442	149
Достоверность, $P\left(A^{ЭКГ}\right)$	**46,4 %**	**98,8 %**

Таким образом, установлено, что разработанная методика имеет крайне высокую эффективность (**98,8 %**) выполнения автоматизированного анализа QRS комплексов электрокардиограмм космонавтов, в том числе, подверженных различным шумам и наводкам, вызванным мышечной активностью, в то время, как эффективность штатного комплекса, работа которого основана на процедурных механизмах, составляет всего **46,4 %**.

3.4.Выводы по третьей главе

В главе 3 приведена методика интерпретации подпрограмм, написанных на языке анализа ТМИ и переведённых транслятором в байт-код. Отмечены особенности, отличающие предлагаемую методику интерпретации от интерпретаторов современных языков программирования и позволяющую оперировать потоком телеметрических значений.

Представлена методика формирования мнемосхем отображения результатов анализа ТМИ БС КА с использованием управляющих подпрограмм (скриптов) на языке анализа ТМИ, для чего разработана модель системы отображения мнемосхем, включающая в себя транслятор подпрограмм на языке анализа ТМИ, переводящий текст подпрограммы в двоичное представление, и интерпретатор подпрограмм анализа, исполняющий скрипт для управления поведением мнемосхемы. Графически мнемосхема состоит из геометрических фигур и других элементов отображения, свойствами которых управляет скрипт на основе поступающих значений телеметрических параметров. Такой подход позволяет сравнительно легко создавать динамические интерактивные мнемосхемы отображения состояния БС КА. В завершение раздела предложена методика разработки мнемосхемы состояния БС КА.

Далее подробно изложена методика нейросетевого анализа ТМИ, содержащей медицинские показания космонавтов, включающая процедуры точной привязки ТМ-кадров ко времени, коммутации единого сигнала из нескольких поступающих, определение пропущенных значений, устранение тренда сигнала методом кусочно-линейной интерполяции. Далее применяется нейросеть nn_1 топологии (67, 30, 13, 1), которая определяет, содержит ли сигнал ЭКГ, либо там шумы, пустой сигнал и т. п. Вторая нейросеть nn_2 топологии (16, 12, 8, 1) используется для поиска зубцов «R» ЭКГ. Обе нейросети обучены на реальных показаниях космонавтов и реализованы на языке анализа ТМИ. На основе найденных зубцов «R» вычисляется частота сердечных сокращений. Результаты практической отработки показали, что разработанная методика успешно обрабатывает **98,8%** сигнала, в то время как штатный алгоритм, основанный на процедурных механизмах, имеет вероятность лишь **46,4%**.

Разработанные методики обеспечивают решение поставленной задачи: совершенствование средств автоматизированного анализа ТМИ в реальном масштабе времени для пилотируемых орбитальных станций с использованием специализированного языка программирования.

4. Экспериментальная проверка и практическая отработка

Предложенные методические подходы были реализованы в составе программно-технических комплексов информационно-телеметрического обеспечения управления КА различного назначения. Программное обеспечение анализа ТМИ состоит из транслятора, осуществляющего перевод программного кода подпрограмм анализа ТМИ из текстового представления в байт-код, и интерпретатора, исполняющего байт-код в процессе обработки ТМИ. Байт-код включает в себя блок объявлений входных и выходных ТМ-параметров, глобальных переменных, пользовательских типов и используемых базовых алгоритмов, а также блок кода, состоящий функций, которые представляют собой последовательности инструкций.

Приводятся результаты экспериментальной оценки показателей унификации языка анализа ТМИ, компактности кода, скорости подготовки мнемосхем в системе, использующей язык анализа ТМИ, эффективности использования мнемосхем для анализа ТМИ КА.

Далее рассмотрены результаты внедрения разработанных методик автоматизированного анализа ТМИ в реальном масштабе времени с использованием специализированного языка программирования, более подробно описанные в [35, 41, 42, 44, 48].

4.1.Программная реализация транслятора и интерпретатора

4.1.1. Реализация транслятора исходных данных САА ТМИ КА

При разработке средств трансляции кода на языке анализа ТМИ использовались материалы монографии [65]. Транслятор и интерпретатор реализованы на языке программирования C# для работы в среде Microsoft .NET [69].

Транслятор (компилятор) выполняет грамматический разбор на основе метода рекурсивного спуска [65]. Сохраняются выражения в постфиксной записи, удобной для исполнения интерпретатором.

Компилятор имеет возможность обнаружить ошибки в программе на трёх этапах компиляции: во время лексического анализа, синтаксического анализа, при генерации кода [3]. В некоторых языках программирования компиляторы исправляют простые ошибки в коде, однако это уже давно считается опасной практикой. В основном, современные компиляторы, как и реализованный компилятор, позволяют небольшие допущения в коде, например, указывать лишнюю запятую в конце списка значений:

```
string сообщенияМассивов[] =
{
      "",
      "#Нарушение обм по МКО3",
      "ПрТМИ о работе пр А1",
      "Зав калибровки БИУС",
      "Зав участка оценки ухода БИУС",
}
```

Компилятор видит, что после последней запятой идёт закрывающая скобка, и понимает, что больше не должно быть элементов массива. Такое допущение упрощает генерацию или копирование кода.

4.1.2. Реализация интерпретатора исходных данных САА ТМИ КА

Интерпретатор подпрограмм на языке описания алгоритмов анализа ТМИ встраивается в программу обработки ТМИ ТМИВК как один из алгоритмов и реализует следующие элементы интерфейса взаимодействия:

- загрузка кода и параметров подпрограммы;
- исполнение кода при поступлении значений ТМ-параметров;
- подписку на события «начало сеанса», «конец сеанса» и исполнение кода при их наступлении;
- подписку на временные посылки и исполнение кода при наступлении событий;
- формирование новых значений ТМ-параметров и выдачу их на обработку;
- вызов базовых алгоритмов.

4.1.2.1. Загрузка кода и параметров подпрограммы анализа

В процессе загрузки подпрограммы читаются следующие параметры.

– Байт-код подпрограммы, сохранённый в виде единого массива байтов.

– Версия транслятора. При модернизации транслятора может поменяться структура кода. Данный параметр учитывается для возможности исполнения кода, транслированного более ранними версиями транслятора.

– Адрес функций инициализации и финализации в сегменте кода.

– Максимальное разрешённое количество исполненных инструкций и сформированных параметров за один вызов алгоритма. Данные параметры используются для контроля зацикливания кода конкретной подпрограммы.

– Объявленные в подпрограмме пользовательские типы данных, включая их имена, методы и свойства. Код методов будет располагаться в общем сегменте кода, а здесь указывается адрес первой инструкции каждого метода.

– Входные и выходные ТМ-параметры подпрограммы, представляемые переменными, и глобальные переменные. Для каждой переменной указывается тип данных и возможное инициализирующее значение. В том числе инициализировать можно массивы чисел, строк, объектов и ТМ-параметров, к которым позже можно обращаться по индексу.

– Базовые алгоритмы, к которым идёт обращение в коде подпрограммы анализа.

4.1.2.2. Исполнение кода

Исполнение кода осуществляется конечным автоматом $FSM = \{B, ip, Var, VS\}$ со следующими элементами.

– Байт-код функций `B`.

– Адрес текущей исполняемой инструкции `ip` (instruction pointer). В начале исполнения устанавливается на начало главной процедуры подпрограммы, либо на обработчик события, вызвавший запуск алгоритма.

– Коллекция переменных `Var`. В начале коллекции находятся глобальные переменные. Новые локальные переменные и фактические параметры, передаваемые в функции, добавляются в конец коллекции и удаляются из неё по завершению использования.

– Вычислительный стек VS, в который сохраняются результаты вычисления операций, результат вызова функции или базового алгоритма. Команды берут аргументы с вершины вычислительного стека и результат помещают на вершину стека.

4.1.2.3. Система команд

Байт-код алгоритма представляет из себя последовательность команд с аргументами. Команды могут быть унарными, бинарными, тернарными, более сложными, команды передачи управления, вспомогательные команды и т. п. Код команды занимает 1 байт, за которым следуют аргументы. Все многобайтовые значения записаны в байт-коде в прямом, сетевом порядке байтов (Big Endian) [63], принятом в структурах данных ТМИВК и большинстве современных бортовых телеметрических систем.

В качестве обзора рассмотрим некоторых представителей команд.

4.1.2.3.1. Прочитать байт

Структура команды чтения байта приведена в таблице 4.1.1. Команда считывает целочисленное значение из байт-кода алгоритма и помещает его на вершину вычислительного стека.

Таблица 4.1.1. Структура команды чтения байта.

№ байта	Содержимое	Описание
0	Код команды	9 – прочитать байт
1	Значение	Число от 0 до 255, которое необходимо поместить в вычислительный стек

4.1.2.3.2. Прочитать ссылку на переменную

Структура команды чтения ссылки на переменную приведена в таблице 4.1.2. Команда помещает в вычислительный стек ссылку на указанную переменную. Другие операции при обращении к значению в вычислительном стеке на чтение будут получать значение самой переменной, а при обращении на запись, будут записывать значение в переменную.

Таблица 4.1.2. Структура команды чтения ссылки на переменную

№ байта	Содержимое	Описание
0	Код команды	200 – прочитать ссылку на переменную
1-2	Значение	Индекс переменной в стеке переменных

Индекс переменной *i* преобразуется особым образом. Дело в том, что в байт-коде записан тот индекс, который был известен транслятору в процессе компиляции данной инструкции. Но в процессе исполнения программы в коллекцию переменных последовательно помещаются фактические параметры вызываемых функций и их локальные переменные. Поэтому фактический индекс переменной i_f будет зависеть от того, какие функции на данный момент были вызваны (рис. 4.1.1):

$$i_f = \begin{cases} i, & i < C_g \\ i - C_g + i_c, & i \geq C_g \end{cases}$$

Здесь C_g – количество глобальных переменных, позиции которых неизменны; i_c – текущий базовый индекс переменных исполняемой функции. В начале исполнения программы $i_c = C_g$, а при каждом вызове функции или метода i_c увеличивается на количество переменных, видимых в месте вызова, сверх количества глобальных переменных: количество аргументов вызывающей функции и количество локальных переменных, объявленных от начала вызывающей функции до места вызова.

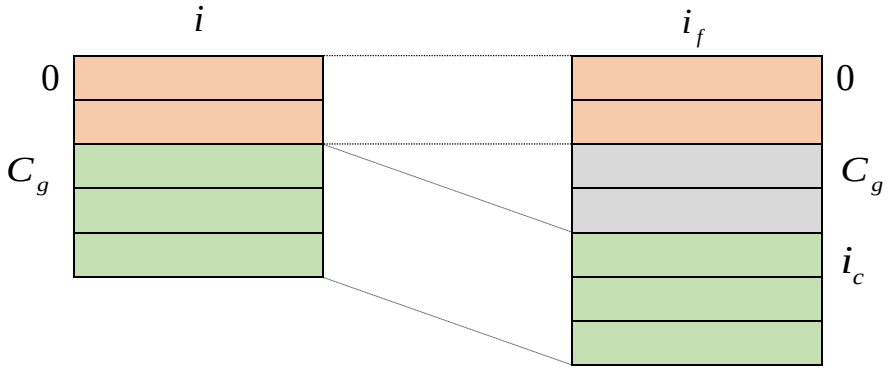


Рис. 4.1.1. Преобразование индекса переменной

Указанное преобразование обеспечивает, в частности, рекурсивный вызов функций, требуемый для реализации многих прикладных алгоритмов.

Поясним преобразование на конкретном примере.

```
01   АНАЛИЗ( ЗАПУСК(ВСЕ, НП+ВП)
02   ВХПАР(int A, int B)
03   ВЫХПАР(ТЕКСТАН C)
04   F1(double p) { … }
05   F2(int p, double q)
06   {
07      F1(0);
08      for (int i = 0; i < 10; ++i)
09         F1(i * p + q);
10   }
```

В алгоритме три глобальных переменных $C_g = 3$: два входных параметра A и B, один выходной параметр C. В момент вызова функции F1 в строке 09 $i_c = 6$, так как в этой точке дополнительно видно два параметра функции p и q, а также локальную переменную – счётчик цикла i. При вызове функции F1 из строки 07 $i_c = 5$.

4.1.2.3.3. Сложение

Команда извлекает с вершины вычислительного стека два аргумента и помещает в стек результат их сложения $c = a + b$. Если хоть один из аргументов является пустым, результат будет пустым. В противном случае аргументы приводятся к старшему общему типу по таблице 2.2.1.

4.1.2.3.4. Битовый сдвиг

Поддерживаются две команды битового сдвига: сдвиг вправо $c = a >> s$ и сдвиг влево $c = a << s$. Команда извлекает с вершины вычислительного стека два аргумента. Второй аргумент s преобразуется к целому значению – это количество разрядов, на которые должен быть сдвинут первый аргумент a. Тип результата совпадает с типом аргумента a, если он относится к целочисленным, иначе преобразуется к типу int. Если $a = f_1 = \{v_1, l_1\}$ – кодовый тип, где v_1 – значение кода, а l_1 – длина кода, то сдвиг вправо вычисляется по формуле:

$$f = f_1 >> s = \left\{ \left\lfloor \frac{v_1}{2^s} \right\rfloor, \max(0, l_1 - s) \right\}.$$

Для целочисленных аргументов сдвиг вправо обозначается как:

$$c = \left\lfloor \frac{a}{2^s} \right\rfloor$$

Сдвиг влево значения кодового типа вычисляется:

$$f = f_1 << s = \left\{ v_1 \cdot 2^s, l_1 + s \right\},$$

а целочисленного типа

$$c = a \cdot 2^s.$$

При этом результат вычисления не может выходить за пределы разрядной сетки аргумента. Например, если *a* – типа ushort – 16-разрядный целочисленный тип, то и результатом операции являются младшие 16 разрядов преобразования. Для кодовых типов ограничением является 64 разряда.

4.1.2.3.5. Склеивание битовых полей

Команда извлекает с вершины вычислительного стека два аргумента, преобразует их к битовым полям $f_1 = \{v_1, l_1\}$ и $f_2 = \{v_2, l_2\}$ и помещает в стек результат склеивания (конкатенации) битовых полей. Преобразование к битовому полю выполняется следующим образом. Если значение является битовым полем, то есть содержит целочисленное значение и длину в битах, то возвращается оно. Если значение является целым, то длина битового поля определяется по таблице 4.1.3.

Таблица 4.1.3. Длина битового поля для значений разных типов

№	Тип	Длина в битах
1.	byte	8
2.	short ushort	16
3.	int uint	32
4.	long ulong	64

Результат склеивания битовых полей вычисляется по формуле:

$$f = f_1 \$ f_2 = \left\{ v_1 \cdot 2^{l_2} + v_2, l_1 + l_2 \right\}.$$

4.1.2.3.6. Команда условного перехода

Команды условного перехода транслятор формирует для условных операторов, проверки условия циклов, и выполнения логических выражений. Команда смещает адрес текущей исполняемой инструкции на заданное смещение

(таблица 4.1.4), если на вершине стека находится ложное или истинное значение (в зависимости от команды). В противном случае исполнение продолжится со следующей инструкции. На рис. 4.1.2 приведена блок-схема команды условного перехода с проверками «если ложь».

Таблица 4.1.4. Коды команд условного перехода.

№ байта	Содер-жимое	Описание
0	Код команды	32 – если истина, прыгнуть вперёд на 1-байт. смещение 33 – если ложь, прыгнуть вперёд на 1-байтовое смещение 34 – если ложь, прыгнуть вперёд или назад на 2-байтовое смещение 35 – если ложь, прыгнуть вперёд или назад на 4-байтовое смещение
1	Значение	1-, 2- или 4-байтовое смещение *offset*, на которое будет сдвигаться указатель текущий инструкции при выполнении условия

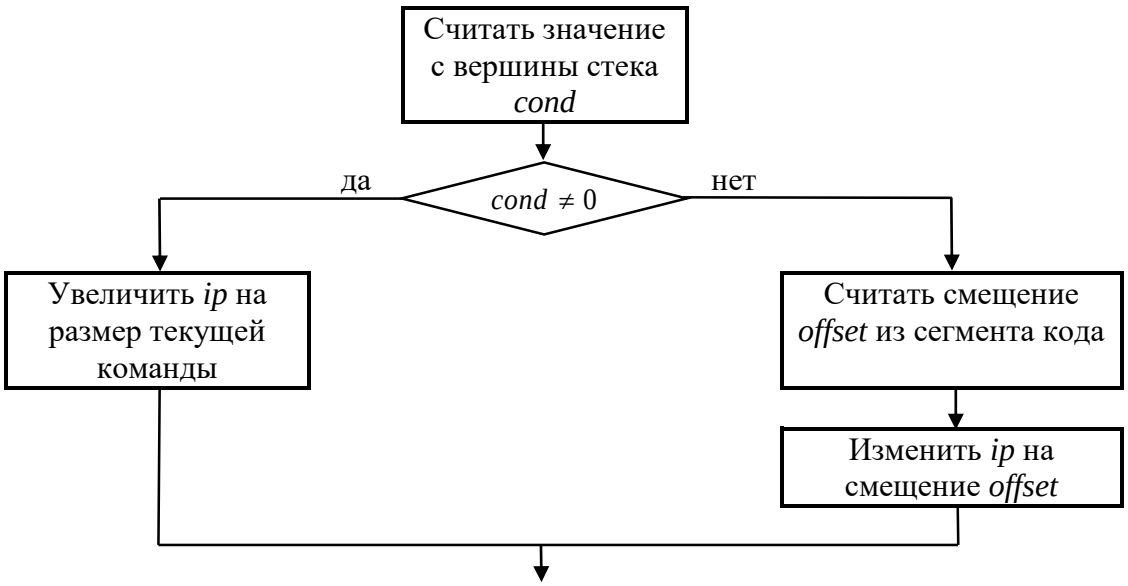


Рис. 4.1.2. Блок-схема работы команды условного перехода

Рассмотрим простейший пример использования команды условного перехода:

```
01  A = B && C;
```
(4.1.1)

Для исполнения данного выражения транслятор сформирует команды, приведённые в таблице 4.1.5. Заметим, что к моменту присваивания (шаг 5) на вершине стека будет либо значение `B` (если оно было ложным), либо значение `C`. На

шаге 3 выполняется условный переход на 4 байта – это суммарный размер команд шагов 4 и 5. Таким образом, исполнение указанного выражения обеспечивается последовательностью элементарных команд.

Таблица 4.1.5. Последовательность команд, сформированных для исполнения выражения (4.1.1)

№	Команда	Размер команды, б	Содержимое вычислительного стека
1.	Считать переменную A	3	A
2.	Считать переменную B	3	A B
3.	Если на вершине стека ложь, прыгнуть вперёд на 4	2	A B
4.	Вытолкнуть значение с вершины стека	1	A
5.	Считать переменную C	3	A C
6.	Выполнить присваивание	1	

Остальные команды в интерпретаторе реализованы подобным образом.

4.2.Проверка показателей эффективности разработанного методического аппарата

4.2.1. Показатели эффективности языка анализа ТМИ

4.2.1.1. Степень унификации языка

Для оценки степени унификации разработанного языка анализа ТМИ U_2 и использовавшегося ранее в ТМИВК языка анализа ТМИ U_1 были выписаны все элементы этих языков с пометкой, является ли каждый элемент унифицированным (то есть присутствует также в основных языках программирования) или нет. В таблице 4.2.1 приводится фрагмент этого списка для разработанного языка.

Таблица 4.2.1. Фрагмент списка элементов языка анализа ТМИ.

Категория	Элемент языка	Унификация
Литералы	Целочисленный литерал	1
	Восьмиричный литерал 0ЦЦЦ	1
	Шестнадцатеричный литерал 0хЦЦЦЦЦ	1
	Двоичный литерал 0bЦЦ	1
	Вещественный литерал	1
	Логический литерал	1
	Строковый литерал "x"	1
	Символьный литерал 'x'	1

	Литерал даты ЦЦ.ЦЦ.ЦЦЦЦ	0
	Литерал времени ЦЦ:ЦЦ:ЦЦ.ЦЦЦ	0
Управляющие операторы	if	1
	else	1
	for	1
	break	1
	continue	1
	return	1
	new	1
	null	1
	switch	1
	codeswitch	0
	…	

Степень унификации вычисляется по формуле (1.4.6). Результаты оценки приведены в таблице 4.2.2.

Таблица 4.2.2. Сравнение существующего и разработанного языка анализа.

Язык	**Количество элементов языка,** $\|E\|$	**Количество унифицированных элементов,** $\|E_u\|$	**Степень унификации,** U
Старый язык анализа ТМИ U_1	477	53	11,1 %
Разработанный язык анализа ТМИ U_2	151	107	71,8 %

Большое количество элементов старого языка анализа ТМИ обусловлено большим количеством возможных альтернативных способов записи одной и той же операции, а также специфическими, редко употребимыми командами.

Таким образом, разработанный язык анализа ТМИ имеет существенно более высокую степень унификации, чем использовавшийся ранее язык. Его проще изучать программисту, знающему основные языки программирования, проще использовать одновременно с написанием программ на основных языках, переносить алгоритмы с одного языка на другой, при этом его возможностей достаточно для написания подпрограмм, решающих практически все задачи автоматизированного анализа ТМИ КА.

4.2.1.2. Проверка компактности кода на языке анализа ТМИ

В ходе экспериментальной проверки проведём сравнительный анализ эффективности введённых в язык анализа ТМИ операторов взятия битового поля и склеивания битовых полей. В качестве метрики будем использовать количество символов, необходимых для написания требуемого кода. При этом идентификаторы считаются за 1 символ. В таблице 4.2.3 приведены примеры законченных выражений на языке C# – $code_1$, и на разработанном языке анализа ТМИ $code_2$ (примеры взяты из реальных алгоритмов обработки и анализа ТМИ КА). На языках C++ и Java синтаксис будет аналогичным, но расстановка скобок за счёт другого приоритета операций несколько отличается. При этом следует отметить, что большинство программистов, как правило, в подобных выражениях ставят избыточное количество скобок, чтобы исключить возможные ошибки в порядке выполнения операций и неверного прочтения кода другими программистами. В последней колонке вычисляется выигрыш Dl по количеству значащих символов у предложенного синтаксиса в сравнении с традиционным.

$$Dl\left(code_1, code_2\right) = \frac{L\left(code_1\right) - L\left(code_2\right)}{L\left(code_1\right)}$$

Таблица 4.2.3. Примеры кодовых выражений на языке анализа ТМИ

Традиционный код, $code_1$	Новый код, $code_2$	Выигрыш, Dl
`БВС-ПАК[7] >> 16 & 0x03FF`	`БВС-ПАК[7][25:10]`	`15-11 = 4` `4/15 = 27%`
`БВС-ПАК[15] >> 8 & 0x0003`	`БВС-ПАК[15][9:2]`	`15-10 = 5` `5/15 = 33%`
`(((ulong)БВС-ПАК[2] & 0x00FFFFFF) << 16) | (БВС-ПАК[3] >> 16)`	`(ulong)БВС-ПАК[2][23:24] $` `БВС-ПАК[3][31:16]`	`39-26 = 11` `11/37 = 30%`
`((ЦТМКАДР[i * 16 + 4] >> 12) & 0x00003FF80) | ЦТМКАДР[i * 16 + 5] & 0x0000003F`	`ЦТМКАДР[i * 16 + 4][16 + 15:12] $` `ЦТМКАДР[i * 16 + 5][30:7]`	`50-35 = 15` `15/50 = 30%`
`(((t1 << 24) | (t2 << 8) | (t3 >> 8)) + 3 * 3600 - GetLeapSeconds()) % 86400`	`(t1 $ t2 $ t3[15:8] + 3 * 3600 - GetLeapSeconds()) % 86400`	`41-30 = 11` `11/41 = 27%`
`(data[index + 6] << 24) |` `(data[index + 7] << 16) |`	`data[index + 6] $` `data[index + 7] $`	`44-27 = 17` `17/44 = 39%`

`(data[index + 8] << 8) | data[index + 9]`	`data[index + 8] $ data[index + 9]`	
`if ((data[index] & 0x0080) == 0)`	`if (data[index][7] == 0)`	`19-13 = 6` `6/19 = 32%`
`if ((КАДР[4] >> 1 & 0x3f) == ((lastSsk + 1) & 0x3f))`	`if (КАДР[4][6:6] == (lastSsk + 1) & 0x3f)`	`31-24 = 7` `7/31 = 23%`
`return (пакет[1] >> 13 & 3) == 3 && (пакет[5] & 0x00FF) == 32;`	`return пакет[1][15:2] == 3 && пакет[5][7:8] == 32;`	`35-29 = 6` `6/35 = 17%`
`(((ЦМ1КАДР[96] >> 8) & 0x000F) + ((ЦМ1КАДР[96] >> 4) & 0x000F) + (ЦМ1КАДР[96] & 0x000F) * 4`	`ЦМ1КАДР[96][11:4] + ЦМ1КАДР[96][7:4] + ЦМ1КАДР[96][3:4] * 4`	`58-36 = 22` `22/58 = 39%`
`(КАДР[10] & 0x3F) * 60000 + (КАДР[11] & 0x3F) * 1000 + (((КАДР[12] & 0x03) << 8) | КАДР[13])`	`КАДР[10][5:6] * 60000 + КАДР[11][5:6] * 1000 + КАДР[12][1:2] $ КАДР[13]`	`63-50 = 13` `13/63 = 21%`

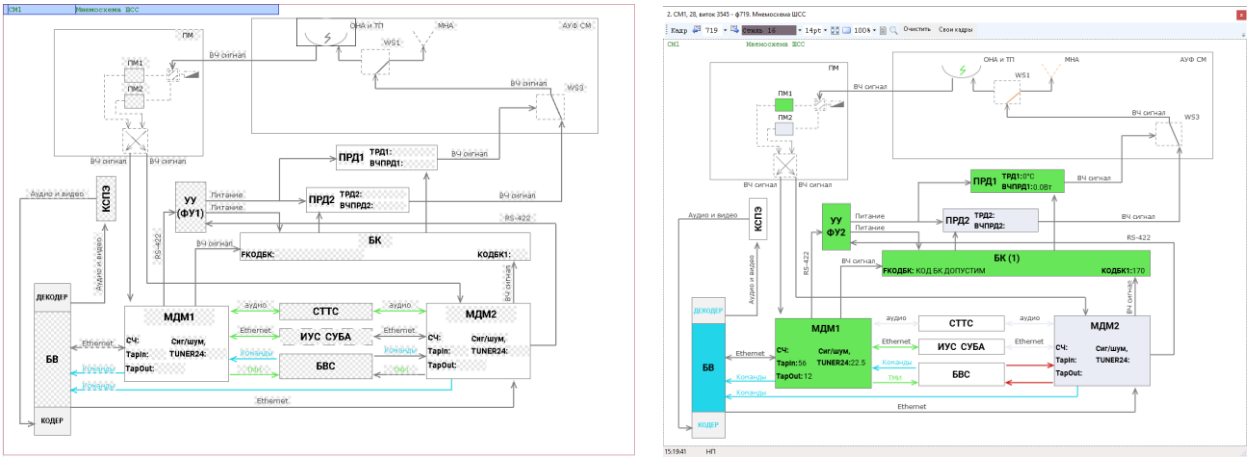
Таким образом, на примере представленных выражений, оперирующих с битовыми полями, показано, что новые синтаксические конструкции дают выигрыш в компактности, а, следовательно, и выразительности кода от 17% до 39% (в среднем 29%):

$$17\% \le Dl = \frac{L\left(code_1\right) - L\left(code_2\right)}{L\left(code_1\right)} \cdot 100\% \le 39\%$$

Данный код быстрее исполняется за счёт меньшего числа операций и меньше подвержен ошибкам при написании.

4.2.2. Проверка эффективности системы подготовки мнемосхем

Проведено практическое исследование по оценке скорости создания мнемосхем на основе заранее подготовленного задания заказчика. В трёх различных системах подготовки мнемосхем была создана достаточно простая мнемосхема анализа состояния бортовой аппаратуры широкополосной системы связи (ШСС) служебного модуля (СМ) «Звезда» МКС. Мнемосхему создавали специалисты, хорошо знакомые со своими системами подготовки. Внешний вид готовой мнемосхемы приведён на рис. 4.2.1 б). На рис. 4.2.1 а) приведён её макет.



а) б)

Рис. 4.2.1. Мнемосхема состояния ШСС СМ МКС

В ходе практического исследования замерялось время, затраченное специалистами на создание визуального макета мнемосхемы, описания логики её функционирования и отладку с использованием ТМИ. В таблице 4.2.4 приведены результаты исследования.

Таблица 4.2.4. Результаты оценки времени подготовки мнемосхемы

Система	**Время создания мнемосхемы** $T_{cm}\left(R_{(i)}^{контр}\right)$**, ч**
Система 1	20
Система 2	13
Разработанная система с использованием языка анализа ТМИ	5

Исследование подтвердило высокую эффективность разработанной системы, использующей визуальный конструктор макета мнемосхемы и скрипт на языке анализа ТМИ, управляющий поведением мнемосхемы. Достигнут выигрыш в скорости создания мнемосхем в сравнении с существующими системами в **2,5÷4** раза. Этот выигрыш позволяет создавать более сложные мнемосхемы или готовить больше мнемосхем для каждого КА, что в конечном итоге повышает качество анализа ТМИ и надёжность управления КА [71].

4.2.3. Проверка эффективности анализа с использованием мнемосхем

Проведено практическое исследование по оценке эффективности использования мнемосхем для проведения оперативного анализа состояния

бортовых систем КА в сравнении с использованием табличных формуляров. Исследование проводилось на примере мнемосхемы состояния комплекса двигательных установок (КсДУ) многофункционального лабораторного модуля (МЛМ) «Наука», запущенного к МКС 21.07.2021 г.

Анализ состояния КсДУ выполнялся на основе табличных формуляров как на рис. 4.2.2.

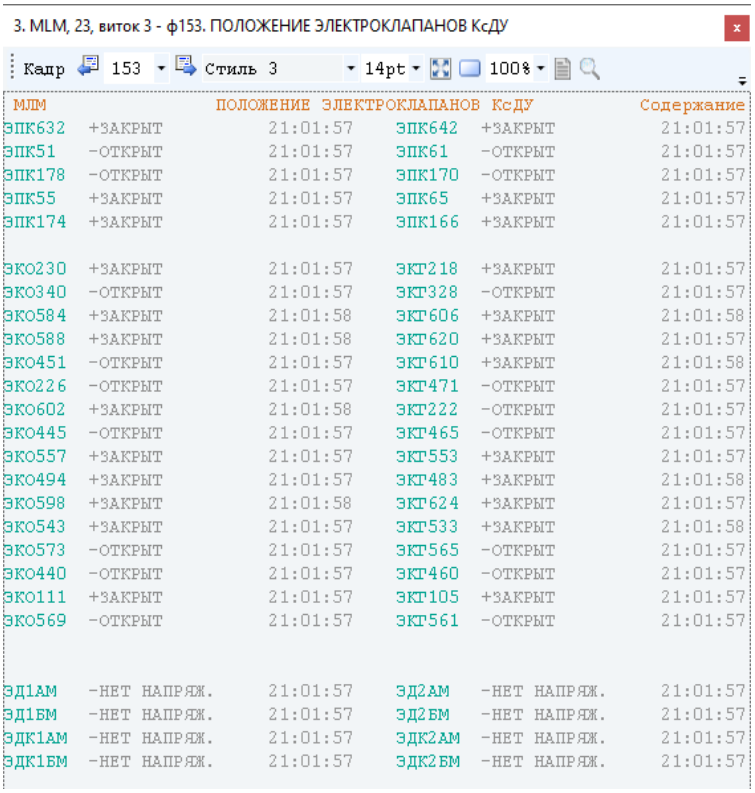


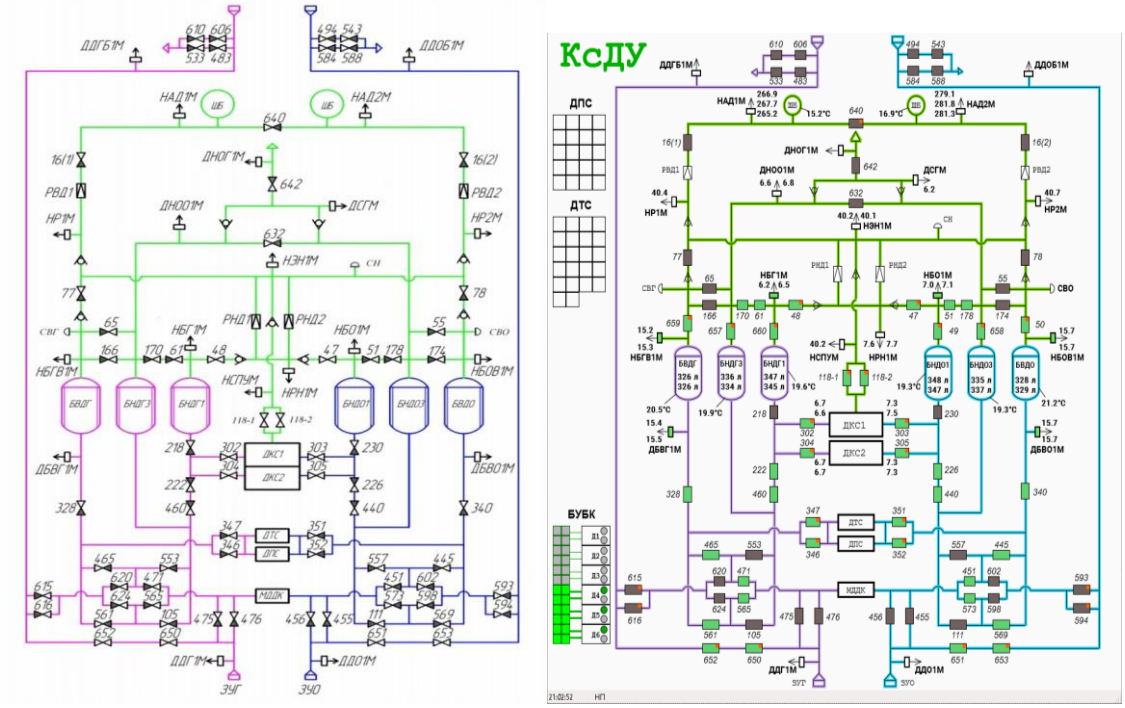
Рис. 4.2.2. Положение электроклапанов КсДУ МЛМ

Поскольку в первый день полёта МЛМ проявились нарушения в работе КсДУ, возникли повышенные требования к оперативности анализа ТМИ. За рекордно короткий срок в **2 дня** была создана мнемосхема анализа состояния КсДУ МЛМ (рис. 4.2.3 б). Мнемосхема имеет внешний вид, максимально приближенный к документации (рис. 4.2.3 а) и контролирует **280 ТМ-параметров**.

ПРИЛОЖЕНИЕ Ж

(обязательное)

Схема КсДУ пневмогидравлическая принципиальная



а) б)

Рис. 4.2.3. Схема (а) и мнемосхема (б) КсДУ МЛМ.

В рамках диссертационного исследования на примере данной мнемосхемы было проведено практическое исследование по оценке эффективности использования мнемосхем для оперативного анализа состояния КА и его бортовых систем в сравнении с табличными формулярами. Для оценки было выбрано четыре реальных состояния КсДУ, по которым были распечатаны табличные формуляры и мнемосхемы. Каждому участнику эксперимента предлагалось определить состояние по таблицам или мнемосхеме (выбиралось случайно). Время выполнения каждого варианта задания замерялось и заносилось в таблицу. Тестовое задание заключалось в том, чтобы определить, к какой топливной магистрали по линии окислителя и горючего подключена каждая группа двигателей:

– ДКС – двигатели коррекции и сближения;
– ДТС – двигатели точной стабилизации, ДПС – двигатели причаливания и стабилизации;

Возможно три варианта подключения двигателя по топливным магистралям:

– ОН – открыта магистраль низкого давления;

– ОВ – открыта магистраль высокого давления;

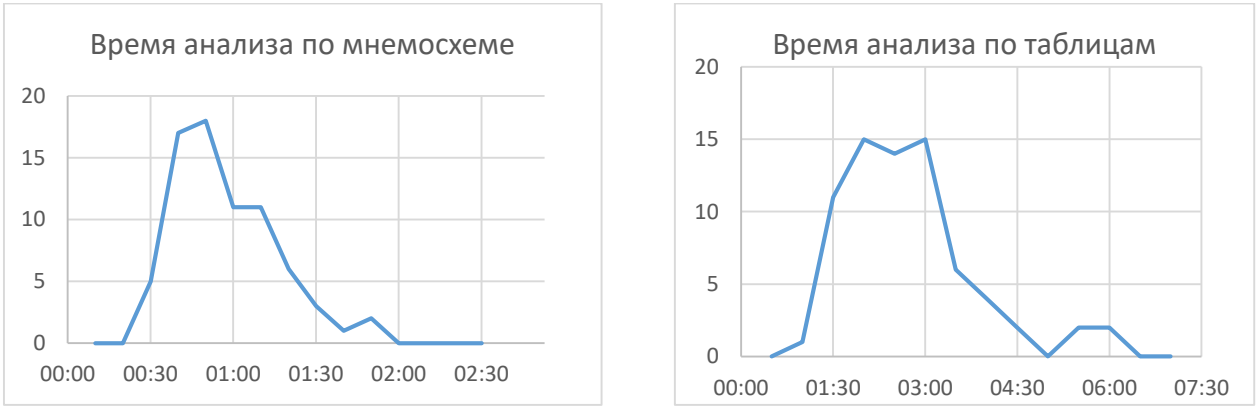
– З – магистрали закрыты.

В таблице 4.2.5 приведён фрагмент результатов опроса участников исследования. В столбце 1 указан номер анализируемого витка полёта, в 2-5 – результаты анализа, в столбце 6 – количество допущенных участником ошибок $A_j^{оцен}$ в соответствии с формулой (1.4.1), в столбце 7 – время, затраченное на анализ, $T\left(A_j\right)$. В последнем столбце буква «М» означает анализ, выполненный с использованием мнемосхемы, «Т» - с использованием табличных формуляров.

Таблица 4.2.5. Фрагмент результатов опроса участников исследования

Виток	**ДКС, Г**	**ДКС, О**	**ДТС, ДПС, Г**	**ДТС, ДПС, О**	**Ош.** $A_j^{оцен}$	**Время** $T\left(A_j\right)$	**Метод**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
17	З	З	ОН	ОН		0:00:48	М
33	ОН	ОН	ОН	ОН		0:00:45	М
1517	ОН	ОН	ОВ	ОВ		0:01:08	Т
3	ОН	ОН	ОВ	ОВ		0:01:58	Т
17	З	З	ОН	ОН		0:00:41	М
33	ОН	ОН	ОН	ОВ		0:01:15	М
1517	ОН	ОН	ОВ	ОВ		0:01:30	Т
3	ОН	ОН	ОН	ОВ	1	0:02:15	Т
3	ОН	ОН	ОВ	ОВ		0:01:15	М
1517	ОН	ОН	ОВ	ОВ		0:00:40	М
17	З	З	ОН	ОН		0:03:05	Т
33	ОН	ОН	ОН	ОН		0:02:58	Т
3	ОН	ОН	ОВ	ОВ		0:01:42	М
1517	ОН	ОН	ОВ	ОВ		0:00:52	М
33	ОН	ОН	ОН	ОН		0:04:11	Т
17	З	З	З	З	2	0:01:48	Т
3	ОН	ОН	ОВ	ОВ		0:01:18	М
1517	ОН	ОН	ОВ	ОВ		0:00:29	М
33	ОН	ОН	ОВ	ОВ	2	0:02:40	Т
17	З	З	ОН	ОН		0:01:57	Т
33	ОН	ОН	ОН	ОН		0:01:45	М
1517	ОН	ОН	ОВ	ОВ		0:01:10	М
17	З	З	ОН	ОН		0:06:05	Т
3	ОН	ОН	ОВ	ОВ		0:06:55	Т
17	З	З	ОН	ОН		0:01:05	М
33	ОН	ОН	ОН	ОН		0:00:50	М
1517	ОН	ОН	ОВ	ОВ		0:01:40	Т
3	ОН	ОН	ОВ	ОВ		0:01:35	Т
3	ОН	ОН	ОВ	ОН	1	0:01:00	М
1517	ОН	ОН	ОВ	ОВ		0:00:32	М

На рис. 4.2.4 приведены графики количества испытуемых, выполнивших анализ за каждый интервал времени на основе мнемосхем (а) и таблиц (б). Всего было проведено 148 тестов.



а) б)

Рис. 4.2.4. Количество испытуемых,
выполнивших анализ за каждый интервал времени

Результаты обобщения полученных экспериментальных данных приведены в таблице 4.2.6.

Таблица 4.2.6. Результаты сравнения выполнения анализа по таблицам и мнемосхемам

	Табличные формуляры	**Мнемосхема**	**Отношение**
Среднее время анализа $\overline{T\left(A_j\right)}$			
Минимальное время анализа $\min T\left(A_j\right)$		0	2,3
Максимальное время анализа $\max T\left(A_j\right)$			
Достоверность $P\left(A_j^{оцен}\right)$		2	
Ошибочность $1-P\left(A_j^{оцен}\right)$	5 5	7 9	

На основе полученных данных можно сделать следующие выводы.

1. Использование мнемосхем вместо (вместе) табличных формуляров существенно сокращает среднее время анализа состояния бортовых систем $\overline{T\left(A_j\right)}$ (в эксперименте – в 3,0 раз).

2. Минимальное время анализа $\min T\left(A_j\right)$, выполненного с использованием мнемосхем и табличных формуляров отличается не сильно, хотя и в пользу мнемосхем (в эксперименте – в 2,3 раза или на 26 секунд). Данный показатель говорит о том, что опытный специалист может выполнять анализ с использованием табличных формуляров практически так же быстро, как и с использованием таблиц.

3. Максимальное время анализа $\max T\left(A_j\right)$, выполненного с использованием мнемосхем и табличных формуляров, отличается в 4,7 раза (на 4 минуты 13 секунд) в пользу мнемосхем, что говорит о том, что мнемосхемы предъявляют существенно меньшие требования к специалисту, проводящему анализ, их использование более стабильно.

4. Вероятность допустить ошибку (формула (1.4.2)), выполняя анализ с использованием мнемосхем, в 1,2 раза (или на 20%) ниже, чем с использованием табличных формуляров, что критически важно для надёжности процесса управления КА.

Таким образом, проведённое исследование показало неоспоримое преимущество использования мнемосхем при создании систем автоматизированного анализа ТМИ КА в скорости и точности выполнения анализа.

4.3.Результаты практической отработки

4.3.1. ЦУП КС «Канопус-В» и БКА

Результаты проведённого исследования использовались при создании телеметрического комплекса центра управления полётами (ЦУП) космической системы (КС) «Канопус-В» в г. Королёв и идентичного ЦУП Белорусского космического аппарата (БКА) в г. Минск [46, 47]. Комплекс реализован на языках программирования C++ и C# [69]. В частности, было реализовано 15 мнемосхем отображения состояния БС КА, разработан ряд алгоритмов анализа ТМИ, включая алгоритмы учёта наработки БС.

На рис. 4.3.1 приведён пример разработанной мнемосхемы анализа состояния системы управления КА «Канопус-В».

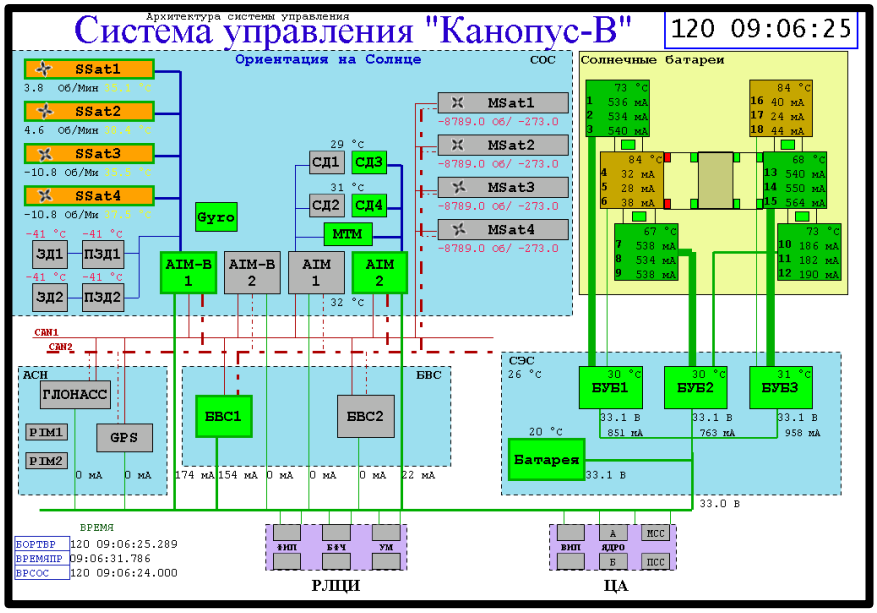


Рис. 4.3.1. Мнемосхема состояния системы управления КА «Канопус-В»

Данная мнемосхема анализирует $R_{(i)}^{контр} = 162$ телеметрических параметра, характеризующих состояние следующих подсистем, состояние которых в отсутствие мнемосхемы прошлось бы контролировать по $\left|W_{(i)}\right| = 10$ формулярам:

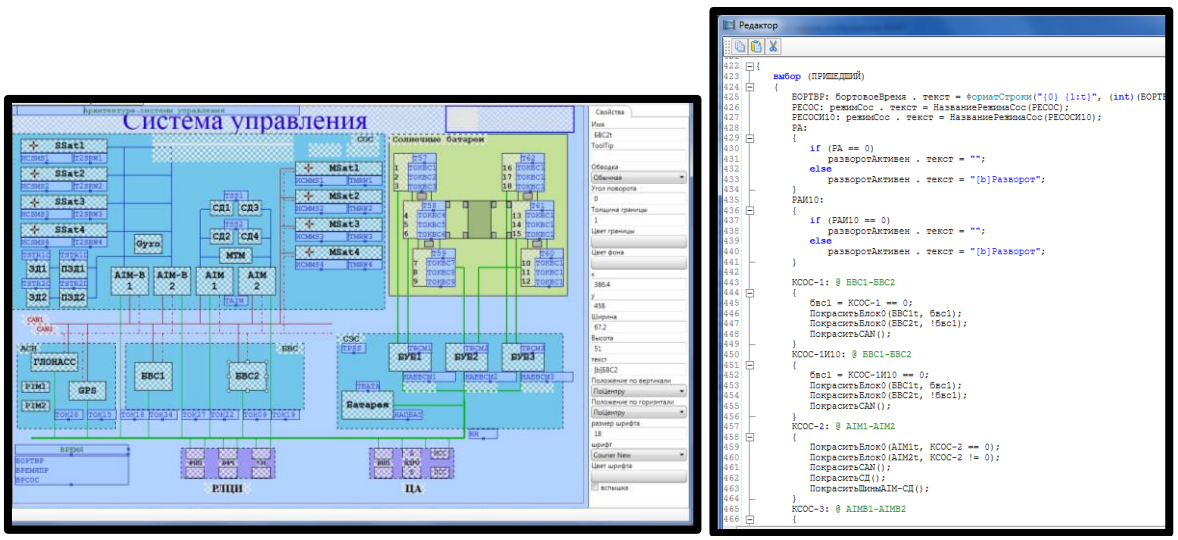
– бортовая вычислительная система (БВС);
– система ориентации и стабилизации (СОС);
– система электроснабжения (СЭС), включая состояние солнечных батарей;
– аппаратура спутниковой навигации (АСН);
– целевая аппаратура (ЦА);
– радиолиния передачи целевой информации (РЛЦИ).

На мнемосхеме неактивные элементы отображаются серым цветом, работающие штатно – зелёным, имеющие замечание – оранжевым, а находящиеся в некорректном состоянии – красным. Сила тока, подающегося от СЭС к различным блокам, обозначается толщиной зелёных линий. Специальный алгоритм анализа в конце каждого сеанса управления

запоминает состояние ключевых систем КА. Если в начале сеанса какая-то из систем оказывается в другом состоянии (например, произошёл переход на резервную БВС или КА перешёл в другой режим ориентации), вся мнемосхема на 2 секунды подсвечивается красным фоном, и воспроизводится предупреждающее звуковое оповещение. Таким образом, специалист группы анализа получает оповещение о важном событии в первую секунду сеанса.

Используя мнемосхему анализа состояния системы управления КА «Канопус-В», специалист группы анализа может мгновенно оценивать состояние КА. Таким образом, одна созданная мнемосхема заменяет для оперативного анализа 10 табличных формуляров, к которым достаточно обращаться лишь при необходимости детального анализа той или иной ситуации.

На рис. 4.3.2 а) изображён визуальный конструктор, в котором создан графический макет данной мнемосхемы, а на рис. 4.3.2 б) – фрагмент подпрограммы на языке анализа ТМИ (скрипта), управляющей поведением мнемосхемы. Мнемосхема проектировалась, создавалась и дорабатывалась несколько лет в течении разработки ЦУП и проведения лётных испытаний КА, поэтому оценить точное время её создания не представляется возможным.



а) б)

Рис. 4.3.2. Создание мнемосхемы системы управления КА «Канопус-В»

На рис. 4.3.3 приведён пример мнемосхемы общего контроля состояния КА «Канопус-В», а на рис. 4.3.4 – пример динамической мнемосхемы контроля

сброса информации из зон памяти КА. Здесь цветными линиями на временной диаграмме отображаются принимаемые в реальном времени с КА виды информации. В правой колонке выводится количество страниц памяти с невоспроизведённой информацией.

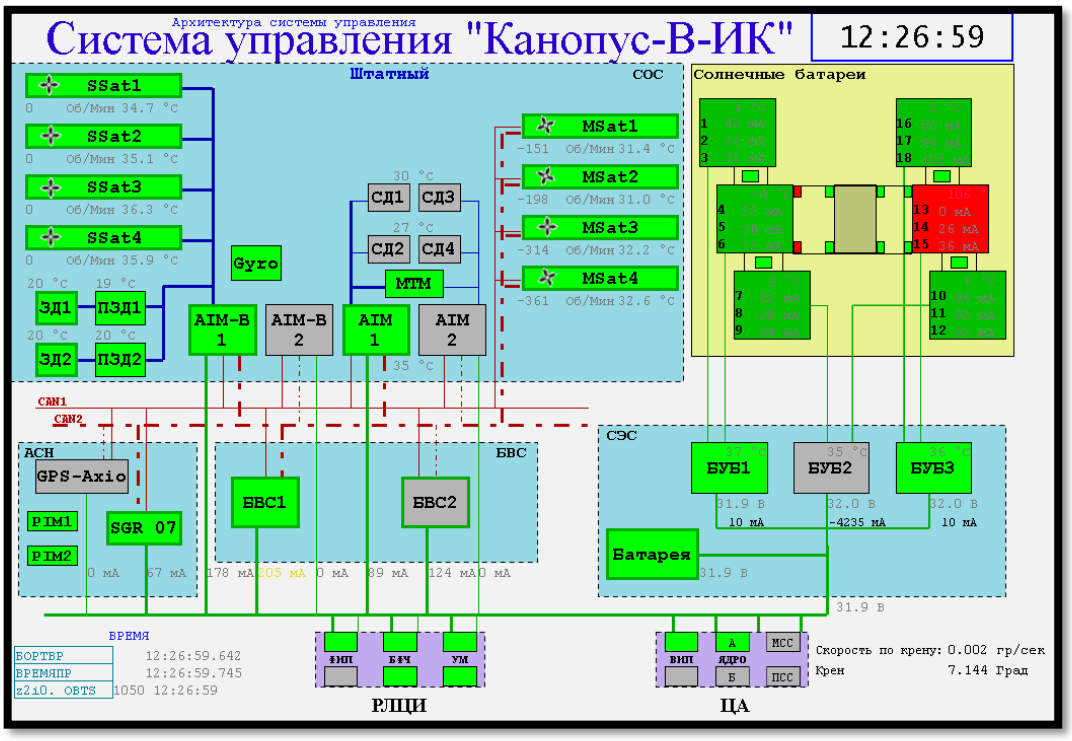


Рис. 4.3.3. Мнемосхема общего контроля состояния КА «Канопус-В»

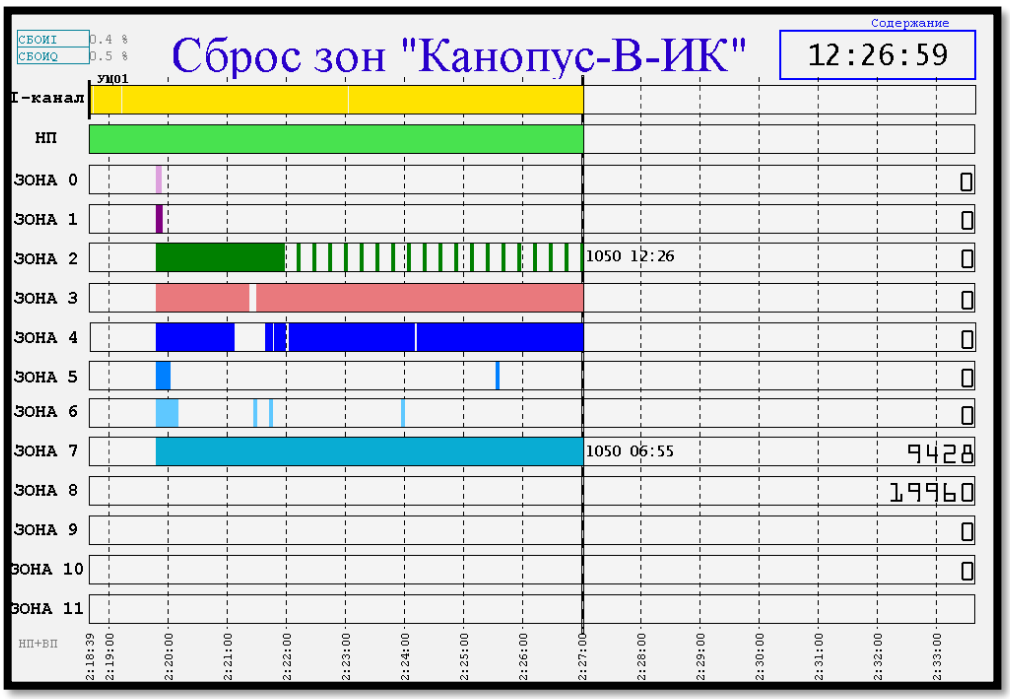


Рис. 4.3.4. Мнемосхема контроля сброса информации из зон памяти КА «Канопус-В».

4.3.2. Контроль выведения ТПК «Союз МС» на РН «Союз-ФГ»

Для контроля процесса выведения транспортных пилотируемых кораблей (ТПК) «Союз МС» на ракетах-носителях (РН) «Союз-ФГ» были разработаны две мнемосхемы. Первая динамическая мнемосхема впервые в реальном времени анализирует и отображает состояние всей ракеты космического назначения (РКН), как единого изделия, состоящего из РН и ТПК, а вторая выводит циклограмму основных полётных событий. Для этого выполняется совместный анализ двух потоков телеметрической информации от РН «Союз-ФГ» (один – от второй ступени РН, один – от третьей) и потока ТМИ от корабля. Кроме того, в реальном времени обрабатывается и выводится в отдельном окне видеоинформация от бортовой системы видеоконтроля (БСВК) РН. Процесс обработки и анализа ТМИ БСВК подробно описан в [33, 43].

Динамическая мнемосхема меняет отдельные области вывода в зависимости от стадии процесса выведения, которые отсчитываются по времени от момента контакта подъёма (КП).

– Всё время полёта контролируется работа телеметрических систем, значения калибровочных уровней, напряжение бортового питания. Текущая кажущаяся скорость РН сравнивается с расчётным показателем на данную секунду полёта. По ТМИ корабля контролируется работа его телеметрических систем, запуск УМРТС «Астра-06», телекамеры внешнего обзора и АСН.

– До контакта подъёма (КП) (а) в правой области выводится циклограмма набора стартовой готовности. При этом контролируется запуск необходимых систем, наддув топливных баков РН, запуск и набор тяги двигателей (рис. 4.3.5 а).

Рис. 4.3.5. Динамическая мнемосхема оперативного контроля выведения РН «Союз-ФГ» с ТПК «Союз МС»

– После КП в правой области выводятся графики осевой перегрузки по ТМИ РН и ТПК с подсвеченным среднеопытным поведением, графики отклонения ориентации РКН по тангажу, рысканию и вращению.

– В период отделения 1 ступени с 117 по 140 с вместо показателей боковых блоков (ББ) выводится подробная информация о срабатывании всех контактов отделения ББ, включая результаты нейросетевого анализа, описанные в разделе 4.3.3 (рис. 4.3.5 в).

– В период сброса головного обтекателя с 140 по 200 с выводится его чертёж с отмеченными контактами отделения (рис. 4.3.5 г).

– После отделения 2 ступени в левой нижней области мнемосхемы выводится подробная мнемосхема работы двигателя 3 ступени (рис. 4.3.6), которая контролирует температуры и давления топлива в баках, магистралях

и камерах сгорания, обороты турбонасосного агрегата, поворот рулевых сопел, расхождения в расходовании окислителя и горючего и пр.

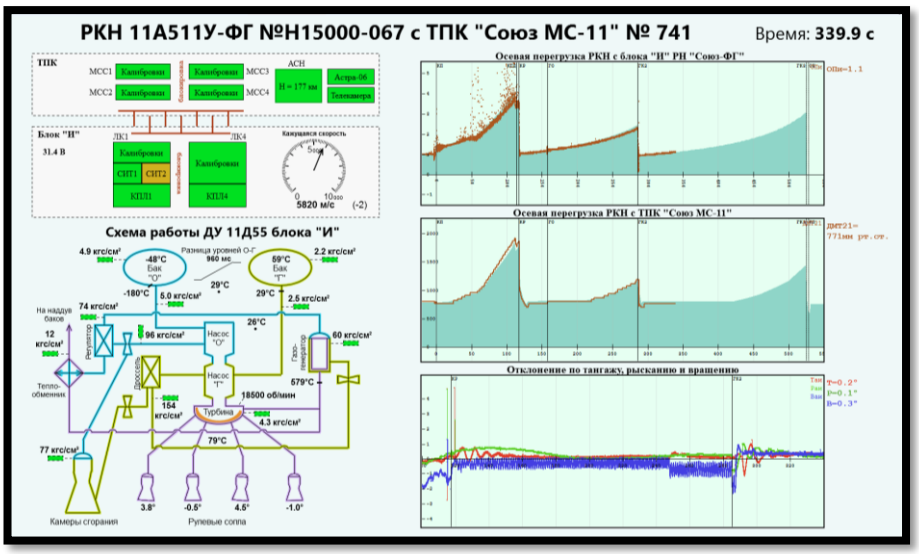


Рис. 4.3.6. Динамическая мнемосхема оперативного контроля выведения РН «Союз-ФГ» с ТПК «Союз МС»

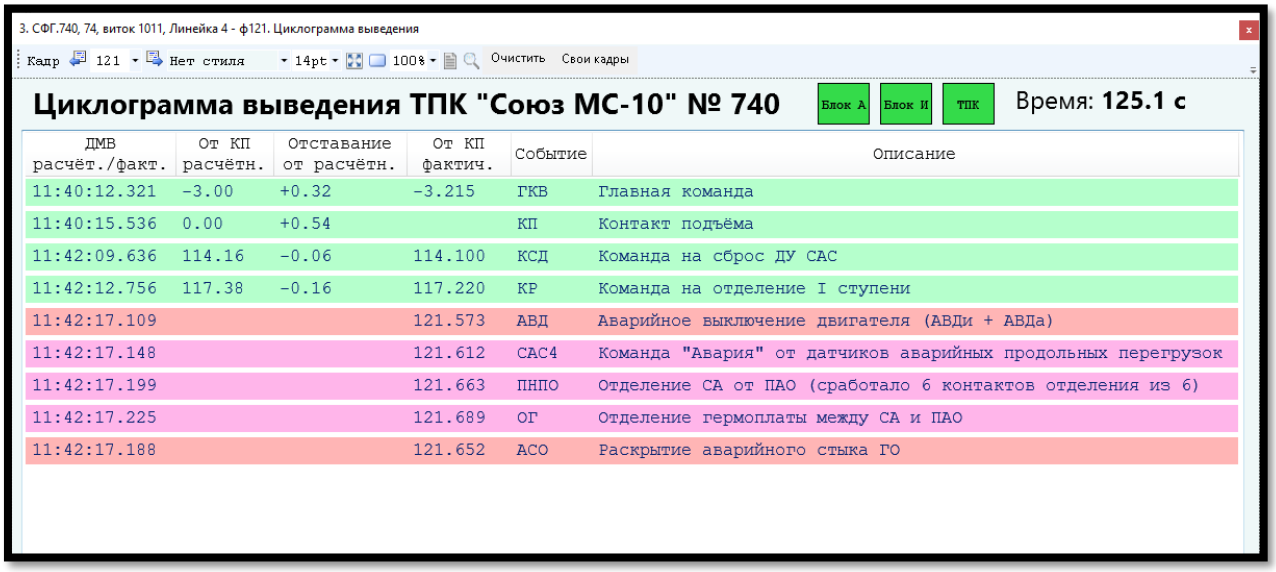
На циклограмме основных событий (рис. 4.3.7) процесса выведения выводятся события с указанием их расчётного и фактического времени срабатывания, вычисляется отклонение от расчёта, а сработавшие события

Циклограмма выведения ТПК "Союз МС-11" № 741 Блок А Блок И ТПК Время: **538.6 с**

ДМВ расчёт./факт.	От КП расчётн.	Отставание от расчётн.	От КП фактич.	Событие	Описание
14:31:49.413	-3.00	+0.41	-3.105	ГКВ	Главная команда
14:31:52.518	0.00	+0.52		КП	Контакт подъёма
14:33:46.682	114.16	+0.00	114.164	КСД	Команда на сброс ДУ САС
14:33:49.721	117.38	-0.18	117.203	КР	Команда на отделение I ступени
14:34:30.396	157.30	+0.57	157.878	СО	Сброс створок ГО
14:36:37.660	285.05	+0.09	285.142	ГК2	Команда на выключение ДУ II ступени (блок И)
14:36:37.742	285.05	+0.17	285.224	ГК2	Команда на выключение ДУ II ступени (ТПК, ДР)
14:36:49.700	297.14	+0.04	297.182	СП	Сброс панелей хвостового отсека
14:40:32.687	520.00	+0.17	520.169	ПО	Разрешение на отделение объекта (РН)
14:40:32.756	520.00	+0.24	520.238	ФШП	Разрешение на отделение объекта (ТПК)
14:40:37.651	524.96	+0.17	525.133	ГК3	Команда на выключение ДУ III ступени
14:40:40.922	528.43	-0.03	528.404	ОК	Отделение КА (РН, сработало 3 из 3 контакта отделения)
14:40:40.987	528.43	+0.04	528.469	КО	Отделение КА (ТПК, сработало 4 из 4 контакта отделения)
14:40:41.547	529.13	-0.10	529.029	КОС	Команда на откр. тормоз. сопла III ступени
14:40:49.252	536.43	+0.27	536.734	ПНА1	Команда на раскрытие антенн 1-й группы
14:40:52	539.73	-1.2		КРАС	Раскрытие антенны "Курс-НА": 42%
14:40:52	539.73	-1.2		КРАСФ	Раскрытие антенны "Курс-НА": 35%
14:40:55	543.47	-5.0		АР1	Раскрытие антенны МБИТС-ТКМ
14:40:55	543.47	-5.0		АЗ1	Раскрытие антенны УКВ системы "Рассвет-ЗБМ"
14:41:01	548.77	-10.3		РСБ	Раскрытие левой солнечной батареи
14:41:01	548.77	-10.3		РСБ	Раскрытие правой солнечной батареи

Рис. 4.3.7. Циклограмма процесса выведения ТПК «Союз МС».

подсвечиваются цветом: зелёным – зарегистрированные по ТМИ РН, голубым – по ТМИ ТПК. В случае возникновения аварийной ситуации все неслучившиеся события удаляются с экрана, а на их месте выводятся подробные события, характеризующие протекание аварийной ситуации (рис. 4.3.8).



ДМВ расчёт./факт.	От КП расчётн.	Отставание от расчётн.	От КП фактич.	Событие	Описание
11:40:12.321	-3.00	+0.32	-3.215	ГКВ	Главная команда
11:40:15.536	0.00	+0.54		КП	Контакт подъёма
11:42:09.636	114.16	-0.06	114.100	КСД	Команда на сброс ДУ САС
11:42:12.756	117.38	-0.16	117.220	КР	Команда на отделение I ступени
11:42:17.109			121.573	АВД	Аварийное выключение двигателя (АВДи + АВДа)
11:42:17.148			121.612	САС4	Команда "Авария" от датчиков аварийных продольных перегрузок
11:42:17.199			121.663	ПНПО	Отделение СА от ПАО (сработало 6 контактов отделения из 6)
11:42:17.225			121.689	ОГ	Отделение гермоплаты между СА и ПАО
11:42:17.188			121.652	АСО	Раскрытие аварийного стыка ГО

Рис. 4.3.8. Циклограмма аварийных событий.

Представленные мнемосхемы реализованы с помощью разработанного в настоящем диссертационном исследовании аппарата, код управления мнемосхемами написан на языке анализа ТМИ. Мнемосхемы оперативного контроля выведения ТПК «Союз МС» успешно отработаны в опытном режиме в ЦУП в процессе реальных запусков.

4.3.3. Применение нейросетей для анализа отделения ББ РН «Союз»

Рассмотрим пример использования языка анализа ТМИ для реализации нейросетевого алгоритма анализа отделения боковых блоков (ББ) ракет-носителей (РН) типа «Союз». Подробно данная работа изложена в [34].

4.3.3.1. Технические особенности контроля процесса отделения боковых блоков РН

Специалисты группы контроля запуска оценивают процесс отделения ББ РН по телеметрическим (ТМ) параметрам [53], которые характеризуют состояние РН, выдачу различных команд, срабатывание датчиков разделения.

Поток ТМ-параметров РН подвергается строгой обработке [39] и формализованному анализу в соответствии с эксплуатационной документацией на РН. Однако данный подход ограничивается анализом непосредственно отделения боковых блоков, но не включает рассмотрение динамического процесса отхода этих боковых блоков. Приведём краткое описание процедуры отделения боковых блоков РН типа «Союз» и используемых для её контроля ТМ-параметров.

Боковые блоки соединены с центральным блоком (ЦБ) РН нижними узлами связи посредством тяг и верхним узлом связи за счет фиксации верхней части ББ в шаровых опорах ЦБ [53]. После исполнения команды на разрыв нижних силовых связей и команды ВОД на останов основных двигательных установок боковых блоков ракета, двигаясь с ускорением, начинает опережать ББ, нижние части которых «веером» расходятся от ЦБ РН под силой последействия двигательных установок, расположенных под небольшим углом к продольной оси ББ. Верхняя часть ББ, выходя из «кармана» ЦБ, разжимает шток, который даёт команду на открытие сопла бака окислителя ББ. Повышенное давление в баке создаёт реактивную струю, которая разворачивает верхнюю часть ББ в направлении от центрального блока. Контроль процесса отхода бокового блока обеспечивает тросик, намотанный на катушку (см. рис. 4.3.9). В составе телеметрической информации передаются параметры ПВ (путь тросика), У1 и У2 (углы выхода тросика: α – в поперечной и β – в продольной плоскостях, соответственно), регистрирующие движение тросика в процессе отхода ББ. Анализ процесса отхода боковых блоков выполняется по значениям указанных параметров на участке непосредственно после отделения.

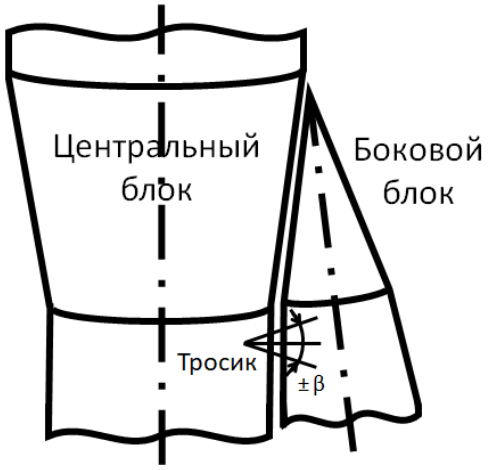
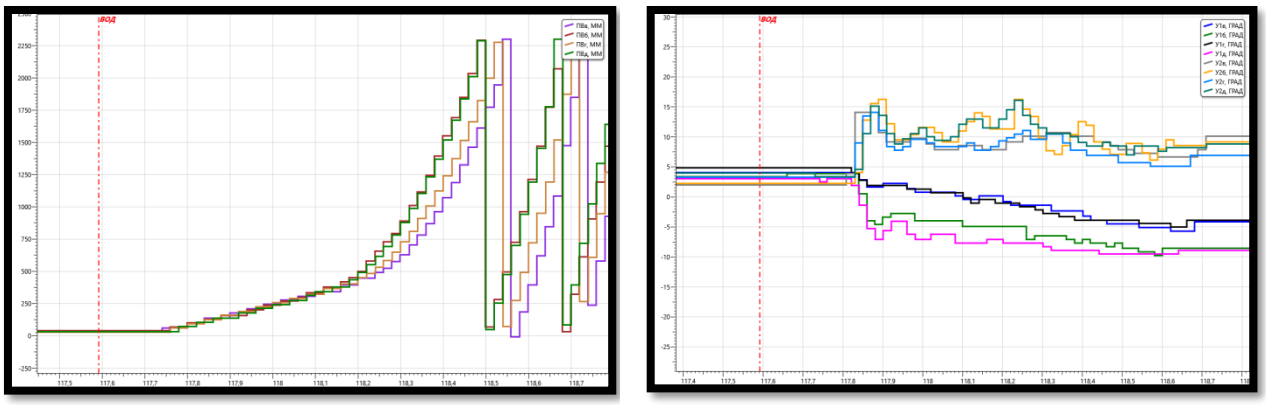


Рис. 4.3.9. Размещение тросика, связывающего ББ с ЦБ РН

На рис. 4.3.10 приведён пример значений параметров У1, У2, ПВ для одного запуска. Существующие подходы оценки этих параметров предполагают визуальное сравнение полученных графиков с расчётными.



а) Параметры ПВ б) Параметры У1, У2

Рис. 4.3.10. Графики параметров отхода ББ от ЦБ

Графики показывают небольшие отличия в протекании процессов отделения ББ от РН на одном пуске. При сравнении различных пусков могут наблюдаться более существенные отличия. Указанные вариации препятствуют объективному и безошибочному сравнению опытных значений ТМ-параметров с расчётными. Таким образом, неуклонно растущие требования к точности и оперативности анализа ТМИ требуют разработки новых подходов, которые обеспечат, в том числе, обнаружение предвестников аварийных ситуаций. В качестве одного из направлений развития методов анализа ТМИ рассматривается применение методов искусственного интеллекта, основанных на методах машинного обучения, а именно, на

нейросетевой обработке информации [19, 34]. Нейросетевые методы сегодня получили широкое распространение в задачах «распознавания образов», к которым сводится рассматриваемый процесс анализа телеметрических измерений процесса отделения боковых блоков РН.

4.3.3.2. Постановка задачи нейросетевого анализа

Даны измерения ТМ-параметров ПВ, У1 и У2, характеризующие процесс разматывания тросика при отходе бокового блока от центрального блока РН на интервале длительностью в 1 секунду (1000 миллисекунд). Измерения берутся начиная от команды ВОД (выключение основных двигателей). Требуется реализовать автоматизированный анализ процесса отделения боковых блоков путём создания нейросети, на вход которой подаются измерения ТМ-параметров, регистрируемые в течение 1 секунды после команды ВОД. Результатом работы нейросети должно быть заключение, насколько анализируемый процесс соответствует среднеопытному.

Набор обучающих данных для нейросети формируется из нескольких записей ТМИ, зарегистрированных на успешных пусках РН «Союз-ФГ». Тестовая выборка, используемая для оценки качества обучения нейросети, формируется из меньшего количества записей (например, 20% от общего количества). При этом расчётные данные из эксплуатационной документации на РН не используются, а сигнал оценивается на соответствие среднеопытному.

4.3.3.3. Общие сведения об используемой нейросетевой технологии

Нейросеть состоит из набора нейронов, разделённых на слои и связанных в виде направленного графа [11, 68, 90, 94] (см. рис. 4.3.11). Выбрана топология нейросети (4, 8, 25). Модель работы каждого нейрона описывается формулой:

$$y_i = f\left(\sum_{j=1}^{n} w_{ij} x_j + b_i\right), \qquad (4.3.1)$$

где x_j – значения, выданные нейронами предыдущего слоя, w_{ij} – весовые коэффициенты, b_i - коэффициенты смещения, а $f(x)$ – активационная функция.

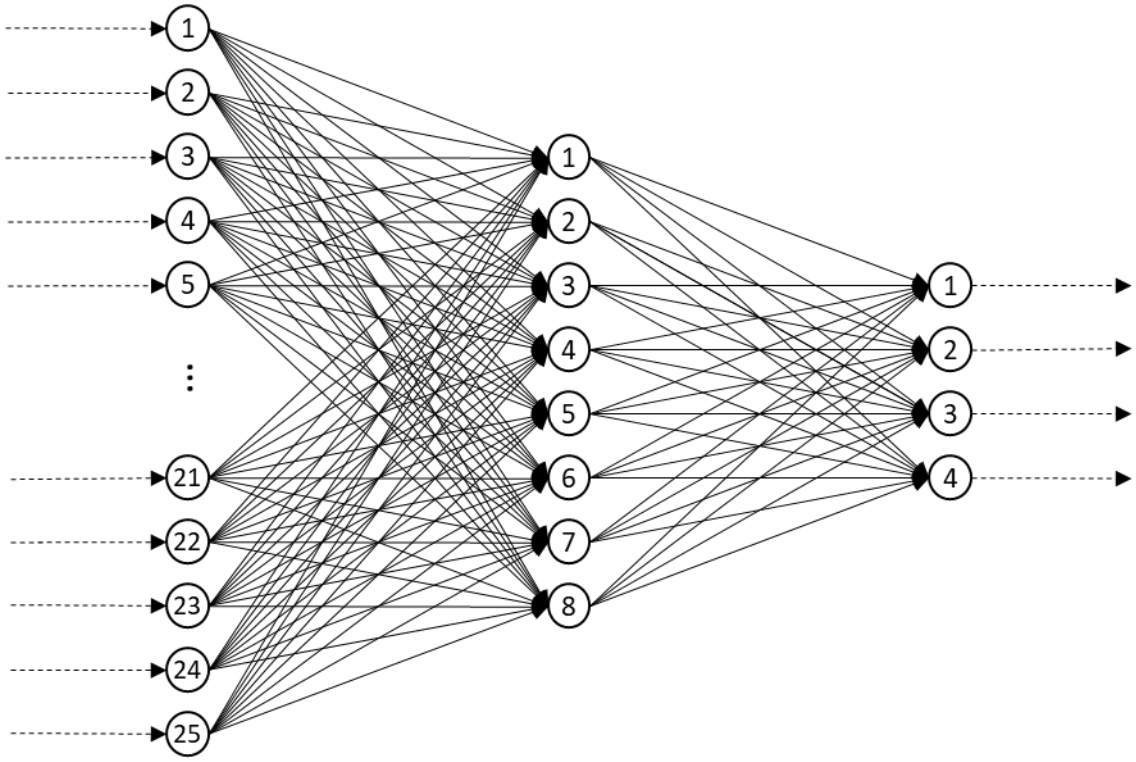


Рис. 4.3.11. Конфигурация нейросети

В процессе обучения нейросети решается задача минимизации отклонения опытных значений, формируемых нейросетью, от значений, заданных в примерах обучающей выборки [90]. Указанное отклонение называют ошибкой нейросети, а при обучении применяют метод обратного распространения ошибки.

4.3.3.4. Создание нейросети для анализа ТМИ

Для решения поставленной задачи возможно использование двух подходов построения нейросетей. В первом подходе создаётся три нейросети, обученных оценивать каждая сигнал своего параметра (У1, У2 или ПВ), формируя на выходном нейроне заключение: степень соответствия сигнала среднеопытному. Второй подход предполагает использование одной нейросети, на вход которой подаются значения любого из трёх параметров, а на выходе обеспечивается срабатывание того нейрона, который сопоставлен этому входному параметру. То есть, первый выходной нейрон срабатывает,

если на вход нейросети поступают штатные значения параметра ПВ. Второй выходной нейрон реагирует на значения параметра У1 и т. д.

Методика обучения нейросетей предполагает использование различных примеров учебных данных [68, 90]. Обучение идентификации нештатной ситуации (НШС) имеет определённые сложности при выполнении анализа реальных данных РН (КА), обусловленные низкой вероятностью возникновения НШС, что не позволяет сформировать достаточный набор обучающих и тестовых данных с использованием реальной ТМИ [17], а моделирование подобных нештатных ситуаций является крайне трудоёмкой задачей. При этом, обучая нейросеть только на хороших примерах, мы научим её всегда давать положительное заключение. Полноценное обучение нейросети возможно только при наличии в обучающей выборке данных из разных категорий. Исходя из сказанного, в данной задаче предпочтительным является второй способ построения нейросети, обученной распознавать три категории корректных сигналов.

В нейросеть вводится дополнительный выходной нейрон, формирующий «0», если входной сигнал корректный, и «1» в противном случае. Такой подход повышает устойчивость обучения нейросети и позволяет выявлять нештатные случаи. Нейросеть может использовать четвёртый нейрон для диагностики нештатного поведения, а не пытаться сопоставлять некорректный сигнал с одним из ожидаемых параметров.

Подготовка значений, поступающих в нейросеть заключается, прежде всего, в нормировании по диапазону изменений от 0,0 до 1,0 [68, 90, 94]. Далее, поскольку частота и скорость изменения выбранных параметров разная, необходимо их унифицировать. Для этого таблица размером 25 ячеек заполняется значениями параметров, выбираемых каждые 40 миллисекунд (подобрано опытным путём), что соответствует измерениям на интервале в 1 секунду. Такой подход даёт рациональную выборку, обеспечивающую разумный размер нейросети и приемлемую дискретизацию. На рис. 4.3.12 в

табличном виде изображены значения параметра ПВ, которые будут поданы на вход нейросети.

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
0,00	0,00	0,00	0,04	0,05	0,05	0,07	0,08	0,09	0,11	0,12	0,14	0,16	0,19	0,23	0,29	0,36	0,45	0,55	0,67	0,81	0,04	0,21	0,39	0,60

Рис. 4.3.12. Значения параметра ПВ, подготовленные для подачи в нейросеть

Рабочими, то есть выполняющими формирование результата, в нейросетях являются скрытые слои. Опытным путём подобрана конфигурация нейросети с одним скрытым слоем из восьми нейронов (см. рис. 4.3.11) [90]. Увеличение количества слоёв или нейронов качественно не влияет на решение поставленной задачи. Объединяя все слои, мы получаем нейросеть топологии (25, 8, 4).

В качестве активационной функции используется логистическая функция:

$$f\left(x\right)=\frac{1}{1+\ell^{-x}} \qquad (4.3.2)$$

4.3.3.5. Обучение нейросети для анализа ТМИ

Обучение нейросети проводилось на основе измерений с 6 запусков РН «Союз-ФГ», по 4 боковых блока на каждой. Для повышения устойчивости обучения на каждой эпохе выполнялось перемешивание обучающих примеров [68, 90, 94]. Кроме того, в обучающую выборку вносилось небольшое число случайно сформированных некорректных данных. В качестве случайных данных моделировались значения параметров ПВ, У1 и У2 на стационарных участках полёта, которые были предназначены для того, чтобы «устремить» нейросеть в сторону от правильных ответов при получении данных, которые не соответствуют корректному динамическому процессу.

В [17] предлагается выполнять инкрементное дообучение нейросетевых модулей в процессе функционирования с целью учёта изменений ТМИ со временем функционирования КА, вызванным дрейфом значений отдельных телеметрических параметров. В данной задаче также не составит труда выполнять дообучение на записях ТМИ с последующих запусков РН.

Следует отметить, что нейросети не формируют однозначных заключений, любой результат является вероятностным. Например, если мы учим нейросеть узнавать параметр У2, то просим от неё ответ вида [0,010 0,010 0,990 0,010] (значения 4 нейронов последнего слоя). Нейросеть в процессе обучения будет асимптотически стремиться к требуемому ответу, не достигая его в точности, и результат будет получаться, например, таким: [0,010 0,005 0,984 0,014]. Среднее квадратичное отклонение полученного ответа от требуемого составит **0,009**.

4.3.3.6. Применение нейросети

С целью применения созданной нейросети для анализа ТМИ в реальном масштабе времени она была реализована в матричной форме на языке анализа ТМИ в составе исходных данных на обработку ТМИ РН. Опуская этапы выборки и подготовки телеметрических данных из поступающих измерений, покажем, как лаконично и элегантно реализуется исполнение нейросети на языке анализа ТМИ. На рис. 4.3.13 приведён код исполнения нейросети. Функция Logistic реализует логистическую функцию (4.3.2).

В функцию `ПроверитьПараметр` передаётся массив входных значений `input` и массив для записи результата `res`. Коэффициенты b_i спрятаны в состав w_{ij} [90], которые в коде представлены матрицами `layer1_w`, `layer2_w` для первого и второго слоёв нейросети соответственно, для чего вектора входных значений x_j в строке 58 расширяются дополнительным элементом с постоянным значением 1. То есть, для удобства вычислений формула (4.3.1) заменяется на формулы:

$$Y = f\left(W \cdot X\right), \tag{4.3.3}$$

$$X = \begin{bmatrix} x_1 & \cdots & x_n & 1 \end{bmatrix}^T,$$

$$Y = \begin{bmatrix} y_1 & \cdots & y_m \end{bmatrix}^T,$$

$$W = \begin{bmatrix} w_{1,1} & \cdots & w_{n,1} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ w_{1,m} & \cdots & w_{n,m} & b_m \end{bmatrix}.$$

```
double Logistic(double x)
{
    return 1 / (1 + EXP(-x));
}

ПроверитьПараметр(double input[], double res[])
{
    double data2[9];
    double data3[4];
    input[25] = data2[8] = 1;

    ПроизведениеМатриц(layer1_w, input, data2, 8, 26, 1);
    for (int i = 0; i < 8; ++i)
        data2[i] = Logistic(data2[i]);

    ПроизведениеМатриц(layer2_w, data2, res, 4, 9, 1);
    for (int i = 0; i < 4; ++i)
        res[i] = Logistic(res[i]);
}
```

Рис. 4.3.13. Код исполнения нейросети

Компоненты матриц `layer1_w`, `layer2_w` получены в результате обучения и указываются в области данных алгоритма как константы.

Таким образом, разработанный язык анализа ТМИ позволяет компактно задавать и эффективно исполнять алгоритмы нейросетевого анализа ТМИ, в том числе в составе задачи обработки ТМИ в реальном масштабе времени.

4.3.3.7. Результаты апробации алгоритма нейросетевого анализа процесса отделения ББ

Для проверки эффективности разработанной нейросети были использованы записи ТМИ с последующих пусков РН, которые не участвовали в её обучении. В таблицах 4.3.1-4.3.3 приведены заключения, сформированные нейросетью при оценке параметров ПВ, У1, У2 РН «Союз-ФГ» при запуске 11.10.2018 г. транспортного пилотируемого корабля «Союз МС-10», в котором нештатное отделение блока Д привело к аварии ракеты-носителя.

Из приведённых данных видно, что разработанная нейросеть достоверно анализирует параметров отделения боковых блоков РН и указывает на соответствие поведения параметров блоков «Б», «В», «Г» среднеопытному и нештатное поведение параметров блока «Д».

Таблица 4.3.1. Оценка параметров ПВ 11.10.2018

	Соответствие ПВ	**Соответствие У1**	**Соответствие У2**	**Несоответствие**	**Ошибка**
ПВб	0,974	0,002	0,020	0,026	0,026
ПВв	0,976	0,002	0,019	0,026	0,024
ПВг	0,979	0,002	0,016	0,025	0,022
ПВд	0,562	0,004	0,031	0,351	0,548

Таблица 4.3.2. Оценка параметров У1 11.10.2018

	Соответствие ПВ	**Соответствие У1**	**Соответствие У2**	**Несоответствие**	**Ошибка**
У1б	0,003	0,976	0,011	0,030	0,025
У1в	0,003	0,977	0,011	0,030	0,025
У1г	0,003	0,976	0,010	0,040	0,034
У1д	0,005	0,918	0,010	0,132	0,142

Таблица 4.3.3. Оценка параметров У2 11.10.2018

	Соответствие ПВ	**Соответствие У1**	**Соответствие У2**	**Несоответствие**	**Ошибка**
У2б	0,017	0,002	0,982	0,017	0,014
У2в	0,017	0,002	0,984	0,017	0,013
У2г	0,016	0,002	0,982	0,018	0,015
У2д	0,977	0,001	0,035	0,014	1,359

Для пусков РН «Союз-ФГ», выполненных штатно, созданная нейросеть в реальном времени выполняет диагностику с уровнем ошибки 0,013 – 0,054. Таким образом, уровень ошибки, превышающий 0,100, можно расценивать как индикатор отклонений в процессе отделения ББ. Поскольку на РН «Союз-2» механизм отделения боковых блоков и структура ТМ-параметров такие же, как и на РН «Союз-ФГ» [53, 61], разработанную нейросеть можно применять для анализа процесса отделения боковых блоков и новых ракет.

Проведена оценка точности выполнения алгоритма нейросетевого анализа ТМИ $P\left(A^{нс}\right)$ в сравнении с оперативным анализом, выполняемым экспертами $P\left(A^{эксперт}\right)$. Людям и нейросети предлагалось для различных

запусков в условиях ограниченного времени по графикам параметров У1 и У2, имеющим сложное поведение, определить отклонения в протекании процесса отделения ББ. Подтверждение заключений выполнялось по графикам параметров ПВ или видеоинформации. В таблице 4.3.4 приведены результаты оценки. В столбцах (3), (6), (9), (12) указана интегральная оценка нейросетевого алгоритма; в столбцах (4), (5), (7), (8), (10), (11), (13), (14) – нейросетевые оценки по отдельным параметрам; в столбце (15) – количество ошибок, допущенных алгоритмом по данному запуску; в столбце (16) – количество ошибок, допущенных экспертами. В таблице 4.3.5 приведены итоги оценки нейросетевого алгоритма анализа отделения ББ РН «Союз» в сравнении с экспертными оценками.

Таблица 4.3.4. Результаты оценки алгоритма нейросетевого анализа отделения ББ

Дата	КА	Б	У1б	У2б	В	У1в	У2в	Г	У1г	У2г	Д	У1д	У2д	Ошибки нейросети	Ошибки экспертов
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)
01.12.2022	14Ф145 № 807	4,4%	1,3%	4,3%	3,4%	0,9%	3,4%	3,0%	1,4%	3,0%	2,3%	2,2%	2,0%		
28.11.2022	14Ф113	2,4%	1,1%	2,4%	2,1%	2,1%	1,5%	3,7%	3,7%	1,0%	2,1%	1,9%	1,2%		1
02.11.2022	14Ф142 №6	5,8%	0,9%	5,8%	1,8%	1,6%	1,8%	6,0%	1,5%	6,0%	9,5%	1,5%	1,2%		1
26.10.2022	Прогресс МС-21 № 451	1,6%	1,2%	1,6%	3,6%	3,5%	0,8%	2,3%	2,3%	0,8%	6,5%	2,6%	0,8%		
09.09.2022	Союз МС-22 № 751	68,2%	32,2%	1,7%	53,0%	6,2%	1,0%	46,1%	19,6%	1,3%	9,5%	7,5%	0,9%	2	4
09.08.2022	Проект 505	3,6%	2,9%	3,6%	4,1%	2,6%	2,0%	2,8%	2,6%	2,1%	4,0%	4,0%	1,8%		
03.06.2022	Прогресс МС-20 № 450	3,5%	1,3%	0,5%	50,4%	2,9%	1,6%	3,7%	2,2%	0,7%	137,5%	2,1%	137,5%	1	2
07.04.2022	14Ф145 № 806	10,0%	1,1%	10,0%	4,6%	1,0%	4,6%	88,1%	3,0%	88,1%	4,4%	4,4%	0,9%	1	0
18.03.2022	Союз МС-21 № 750	11,5%	1,6%	11,5%	16,4%	5,6%	3,8%	8,0%	1,4%	8,0%	22,6%	11,9%	1,6%	1	2
15.02.2022	Прогресс МС-19 № 449	1,6%	1,1%	0,8%	2,7%	2,3%	0,8%	4,2%	4,2%	0,9%	1,7%	1,5%	1,0%		
08.12.2021	Союз МС-20 № 752	16,8%	2,2%	1,5%	3,2%	1,7%	1,7%	3,4%	2,6%	1,6%	6,4%	2,0%	6,4%	1	1
25.11.2021	14Ф142 №5	2,8%	1,0%	2,8%	1,8%	1,2%	1,8%	11,7%	3,0%	11,7%	2,5%	2,5%	1,0%		2
28.10.2021	Прогресс МС-18 № 447	3,8%	0,9%	0,9%	35,2%	2,0%	1,0%	2,9%	2,9%	2,3%	4,4%	4,3%	0,8%		
05.10.2021	Союз МС-19 № 749	3,5%	2,9%	1,0%	2,2%	1,4%	2,2%	4,9%	3,6%	0,8%	11,7%	11,7%	0,8%	1	1
30.06.2021	Прогресс МС-17 № 446	2,6%	2,6%	1,3%	6,6%	3,4%	1,5%	3,4%	3,4%	1,1%	4,0%	4,0%	1,7%		
25.06.2021	14Ф139	3,9%	0,9%	3,9%	2,1%	2,1%	1,0%	1,8%	1,3%	1,8%	3,6%	3,6%	1,3%		1
09.04.2021	Союз МС-18 № 748	2,1%	1,0%	0,9%	3,0%	1,8%	0,8%	14,8%	4,3%	1,6%	4,3%	1,7%	1,1%		3
22.03.2021	CAS500-1	4,1%	4,1%	2,0%	2,8%	2,8%	1,9%	2,6%	2,6%	2,0%	3,3%	2,5%	1,9%		
28.02.2021	Арктика-М № 1	1,6%	1,6%	1,0%	5,9%	0,9%	5,9%	2,5%	2,5%	2,2%	1,5%	1,4%	1,3%		1
23.07.2020	Прогресс МС-15 № 444	4,2%	1,9%	1,1%	96,2%	1,1%	96,2%	4,8%	4,8%	0,6%	6,0%	6,0%	0,8%	1	1
25.04.2020	Прогресс МС-14 № 448	2,9%	2,6%	1,7%	2,7%	2,7%	2,1%	2,6%	2,6%	1,5%	3,2%	3,2%	1,7%		3
09.04.2020	Союз МС-16 № 745	3,6%	2,5%	2,4%	3,4%	2,5%	2,6%	3,7%	2,5%	2,4%	4,8%	3,2%	2,8%		1
20.02.2020	14Ф112 № 19	6,3%	2,7%	6,3%	3,3%	3,2%	3,3%	10,2%	2,7%	10,2%	12,9%	12,9%	1,8%		
30.07.2019	14Ф112 № 18	1,9%	1,9%	1,0%	6,3%	6,3%	1,7%	2,2%	1,3%	1,0%	135,9%	135,9%	1,0%	1	0
20.07.2019	Союз МС-13 № 746	1,4%	1,0%	1,1%	1,6%	0,9%	0,8%	1,4%	1,1%	1,3%	2,0%	2,0%	1,1%		
14.03.2019	Союз МС-12 № 742	1,8%	1,4%	1,1%	1,8%	1,3%	1,2%	2,6%	2,6%	1,0%	1,9%	0,8%	1,8%		

Таблица 4.3.5. Итоги оценки нейросетевого алгоритма

Характеристика	Показание
Запусков	$s = 26$
Процессов	$p = s \cdot 4 = 104$
Ошибок нейросети	$e^{нс} = 9$
Ошибочность нейросети	$\overline{P}\left(A^{нс}\right) = \frac{e^{нс}}{p} = 8,65\%$
Ошибок экспертов	$e^{эксперт} = 24$
Ошибочность экспертов	$\overline{P}\left(A^{эксперт}\right) = \frac{e^{эксперт}}{p} = 23,1\%$
Точность нейросети	$P\left(A^{нс}\right) = 1 - \overline{P}\left(A^{нс}\right) = 91,4\%$
Точность экспертов	$P\left(A^{эксперт}\right) = 1 - \overline{P}\left(A^{эксперт}\right) = 76,9\%$

Таким образом, разработан алгоритм нейросетевого анализа в реальном времени динамического процесса отделения боковых блоков ракет-носителей «Союз». Точность алгоритма составляет 91,4%, что на 14,5% выше, чем точность проведения оперативного анализа экспертами.

4.4.Выводы по четвёртой главе

В главе 4 рассмотрены результаты экспериментальной проверки и практической отработки разработанных моделей, методик и средств автоматизированного анализа ТМИ.

Установлено, что степень унификации разработанного языка анализа ТМИ составила **71,8 %**, в то время как унификация использовавшегося ранее языка анализа ТМИ составляет лишь **11,1 %**. Проверка компактности разработанного языка показала выигрыш в компактности, а, следовательно, и выразительности кодовых выражений на **29 %**.

Рассмотрены особенности реализации интерпретатора подпрограмм, написанных на языке анализа ТМИ, включая способ кодирования команд, внутреннего представления структур исполнения кода, проблемы адресации переменных, в том числе для обеспечения возможности рекурсивного вызова функций и т. п.

Выполнена оценка эффективности системы подготовки мнемосхем. Опытным путём установлено, что предложенная система подготовки мнемосхем в **2,5÷4** раза эффективнее других используемых в ЦУП систем. Кроме того, использование языка анализа ТМИ для управления поведением мнемосхем позволяет формировать динамические, интерактивные мнемосхемы.

Проведено практическое исследование по оценке эффективности применения мнемосхем для оперативного анализа ТМИ в сравнении с применением табличных формуляров. Установлено, что мнемосхемы дают выигрыш в **3 раза** по времени проведения анализа и на **20%** повышают достоверность анализа.

Приведены примеры применения разработанного методического и программного аппарата для реализации автоматизированного анализа ТМИ в реальном времени в различных ЦУП.

В завершение главы рассмотрен пример использования языка анализа ТМИ для реализации нейросетевого алгоритма анализа отделения боковых блоков ракет-носителей (РН) типа «Союз». Нейросеть обучена на ряде реальных запусков РН «Союз-ФГ» и показала высокую эффективность автоматизированного анализа процесса отделения боковых блоков РН «Союз-ФГ» и «Союз-2», успешно диагностируя штатные отделения боковых блоков и указывая на процессы, отличающиеся от среднеопытных. Точность работы нейросети составила **91,4 %** против точности оперативного анализа, выполняемого экспертами, – **76,9 %**.

5. Заключение

В диссертации поставлена и решена важная научно-практическая задача по разработке моделей и методик системы автоматизированного анализа (САА) телеметрической информации (ТМИ) в реальном масштабе времени для пилотируемых орбитальных станций, автоматических и пилотируемых КА, а также средств их выведения. Разработанные модели и методики основаны на использовании специального высокоуровневого предметно-ориентированного языка программирования, предназначенного для задания алгоритмов анализа ТМИ.

В ходе выполнения работы получены следующие основные новые и наиболее важные научные результаты:

1. В результате системного анализа САА ТМИ сформирована система критериев и показателей, характеризующих эффективность выполнения анализа ТМИ КА в реальном времени, разработан новый частный показатель качества языка программирования: степень унификации языка программирования, позволяющий оценивать трудоёмкость изучения и применения языка программирования.

2. Формализована задача обработки и анализа ТМИ, разработана модель описания задач обработки телеметрической информации: лингвистическая модель языка описания алгоритмов анализа ТМИ (язык анализа ТМИ), позволяющего на высокоуровневом предметно-ориентированном языке описывать алгоритмы анализа ТМИ. Синтаксис языка построен на базе современных высокоуровневых языков. Показана высокая степень унификации разработанного языка (**71,8 %**). Специально разработанные операторы и синтаксические конструкции позволяют компактно и наглядно описывать типовые для обработки и анализа ТМИ конструкции.

3. Разработана методика интерпретации (исполнения) исходных данных для автоматизированного анализа ТМИ в реальном времени, ориентированная на оперировании значениями телеметрических параметров,

предназначенная для функционирования в составе модуля обработки ТМИ совместно с базовыми (типовыми) алгоритмами обработки ТМИ.

4. Разработана методика визуализации и анализа телеметрической информации на основе компьютерных методов обработки информации с применением мнемосхем отображения результатов анализа ТМИ БС КА, отличающаяся от существующих использованием управляющей подпрограммы на языке анализа ТМИ, что позволяет формировать интерактивные динамические формы отображения в реальном времени. Показана высокая эффективность предложенной системы подготовки мнемосхем (в **2,5÷4** раза выше в сравнении существующими комплексами). Мнемосхемы, функционирующие под управлением подпрограмм на языке анализа ТМИ, предоставляют больше возможностей по наглядному представлению результатов анализа ТМИ, в том числе возможности по совместному анализу функционирования сложных изделий космической техники, таких как несколько модулей пилотируемой орбитальной станции и пристыкованных к ним космических кораблей, ракеты-носителя и выводимого ею космического корабля и т. п.

5. Разработана методика автоматизированного нейросетевого анализа ТМИ, содержащей медицинские показания космонавтов. Нейросети обучены на ТМИ, содержащей реальные показания космонавтов. Потоки ТМИ в реальном времени коммутируются в единый поток, включающий в себя наилучший по качеству сигнал из нескольких потоков. Методика показала высокую эффективность в сравнении с использовавшимися ранее, в том числе по обработке зашумлённых участков сигнала, адаптации к индивидуальным особенностям космонавтов и возможностям послесеансного анализа полученных данных. Методика реализована на разработанном языке анализа ТМИ и функционирует в опытном режиме в ЦУП российского сегмента МКС с 2020 года. Эффективность методики составляет **98,8 %**, в то время как штатный алгоритм анализа, основанный на процедурных механизмах и использующийся в ЦУП, распознаёт лишь **46,4 %** сигнала.

6. Разработан и реализован на языке анализа ТМИ алгоритм нейросетевого анализа процесса отделения боковых блоков РН «Союз». По поведению динамических параметров нейросеть оценивает, соответствует ли отделение каждого из четырёх боковых блоков РН среднеопытному. В ходе опытной эксплуатации в ЦУП на реальных данных подтверждена высокая точность работы обученной нейросети. Точность диагностики нейросетью штатного процесса отделения ББ составила **91,4 %** против точности оперативного анализа, выполняемого экспертами, – **76,9 %**. В частности, нейросеть диагностировала нештатное отделение блока «Д» при аварийном запуске ТПК «Союз МС-10».

7. Результаты диссертационного исследования внедрены в ЦУП российского сегмента МКС, ЦУП космической системы «Канопус-В», ЦУП Белорусского КА, единый ЦУП космической системы «Ресурс-П», что подтверждается актами, приведёнными в приложении.

В результате выполнения работы создан комплекс моделей и методик автоматизированного анализа телеметрической информации в реальном масштабе времени, использование которых при создании комплексов информационно-телеметрического обеспечения в центрах управления полётами пилотируемых орбитальных станций и автоматических космических аппаратов позволяет сократить время создания комплекса, время подготовки к новому КА, повысить эффективность выполнения анализа состояния КА.

Дальнейшие исследования планируется проводить в направлениях развития языка автоматизированного анализа ТМИ путём ввода в него дополнительных операций, облегчающих обработку и анализ ТМИ; исследовании методов оптимизации формируемого транслятором байт-кода; исследовании методов параллельных вычислений; расширении применения методов искусственного интеллекта для анализа ТМИ.

6. Список сокращений и условных обозначений

6.1.Список сокращений

АС МКС – американский сегмент МКС

АСН – аппаратура спутниковой навигации

ББ – боковой блок

БВС – бортовая вычислительная система

БД – база данных

БКА – белорусский космический аппарат

БОИ – блок обработки информации

БПИ НЧ – блок передачи информации низкой частоты

БРТС – бортовая радиотелеметрическая система

БС – бортовая система

БСВК – бортовая система видеоконтроля

ВКД – внекорабельная деятельность космонавтов

ВОД – команда на выключение основных двигателей РН «Союз»

ВП – режим воспроизведения ТМИ

ДМВ – декретное Московское время

ИД – исходные данные

КА – космический аппарат

КП – контакт подъёма РН

КС – космическая система

ЛИ – лётные испытания

МБИТС – малогабаритная телеметрическая система

МКС – международная космическая станция

МЛМ – многофункциональный лабораторный модуль МКС «Наука»

НИР – научно-исследовательская работа

НКУ – наземный комплекс управления

НП – режим непосредственной передачи ТМИ

НПРС – наземная приёмно-регистрирующая станция

НСЭН – научного и социально-экономического назначения
НШС – нештатная ситуация
ОКР – опытно-конструкторская работа
ПГ – пневмограмма
ПТК – пилотируемый транспортный корабль «Орёл»
РБ – разгонный блок
РБНФ – расширенная форма Бэкуса-Наура
РКН – ракета космического назначения
РН – ракета-носитель
РОС – Российская орбитальная станция
РС МКС – Российский сегмент МКС
САА – система автоматизированного анализа
СЛАУ – система линейных уравнений
СМ – служебный модуль «Звезда» МКС
СЧ – составная часть
ТГК – транспортный грузовой корабль «Прогресс МС»
ТКС – телекомандная система
ТМ – телеметрический
ТМИ – телеметрическая информация
ТМИВК – телеметрический информационно-вычислительный комплекс
ТМХ – телеметрия из г. Хьюстон (США, штат Техас)
ТМЦИ – телеметрический массив цифровой информации
ТПК – транспортный пилотируемый корабль «Союз МС»
УМРТС – универсальная малогабаритная телеметрическая система
ЦБ – центральный блок РН
ЦУП – центр управления полётами
ЧСС – частота сердечных сокращений
ЦВМ – центральная вычислительная машина
ЦУП – центр управления полётами
ЧСС – частота сердечных сокращений

ШСС	–	широкополосная система связи
ЭВМ	–	электронно-вычислительная машина
ЭКГ	–	электрокардиограмма
CCSDS	–	Consultative Committee for Space Data Systems (консультативный комитет по космическим системам передачи данных)
GMT	–	Greenwich Mean Time (Гринвичское время)
GPS	–	Global Positioning System (глобальная система позиционирования)
QRS комплекс	–	желудочковый комплекс электрокардиограммы
TDRSS	–	Tracking and data relay satellite system (Спутниковая система сопровождения объектов и передачи данных)

6.2.Список условных обозначений

Инструкция – наименьшая автономная часть языка программирования. Программа обычно представляет собой последовательность инструкций. Инструкция может объявлять переменную, выполнять присваивание, вызывать функцию, управлять вычислительным процессом. Инструкции объединяются в блоки инструкций.

ТМ-параметр – именованная сущность, характеризующая измерения состояния некоторого процесса на борту КА, например, температуры, давления, контакта, состояния бортовой вычислительной машины и т. п. Кроме того, ТМ-параметры используются для передачи массивов, числовых и текстовых значений, формируемых в процессе приёма и обработки ТМИ.

Значение ТМ-параметра – величина, измеренная в некоторый момент времени у конкретного ТМ-параметра. Также сопровождается атрибутом (норма – не норма), режимом передачи (непосредственная передача, воспроизведение), размерностью (физические единицы измерений).

Алгоритм обработки – программный модуль в модуле обработки ТМИ, реализующий определённые преобразования над значениями ТМ-параметров.

В исходных данных конкретные алгоритмы обработки приписываются конкретным ТМ-параметрам с заданными коэффициентами, режимом работы, признаками достоверности и др.

Исходные данные на обработку ТМИ – задание на обработку ТМИ КА, описываемое в ТМИВК в виде текста на специализированном языке программирования, содержащее объявление ТМ-параметров в структуре системы телеизмерений КА и алгоритмы, по которым необходимо обрабатывать параметры.

Обработка ТМИ – процесс дешифровки и преобразования исходного потока ТМИ, состоящего, как правило, из двоичных ТМ-кадров или пакетов. Результатом обработки ТМИ является последовательность достоверных и существенных значений ТМ-параметров в физических единицах измерений.

Основные языки программирования – языки программирования, являющиеся основными при разработке программного обеспечения ЦУП: C++, C#, Java.

Исходные данные на отображение ТМИ – описание формуляров отображения ТМИ БС КА, задаваемое в системе подготовки кадров отображения ТМИВК при помощи графического конструктора и используемое системой отображения ТМИ для формирования форм отображения результатов обработки ТМИ.

Метод – функция, применяемая к некоторому конкретному объекту. В большинстве языков программирования синтаксис вызова метода следующий: <Объект>.<Метод>(<Аргументы>).

Аномалия в работе бортовых систем (БС) — это локальное отклонение значений ТМ-параметров, находящееся в допустимых пределах, но не соответствующее номинальному среднестатистическому распределению величин ТМ-параметров, и это отклонение может иметь потенциальное развитие как нештатная ситуация (НШС).

7. Список литературы

1. Абрамов Н.С., Талалаев А.А., Фраленко В.П. Интеллектуальный анализ телеметрической информации для диагностики оборудования космического аппарата // Информационные технологии и вычислительные системы № 1, 2016. – С. 64-75.

2. Абанин О.И. Апробация автоматизированного алгоритма анализа телеметрических параметров состояния бортовых систем космического аппарата // Инженерный журнал: наука и инновации. №1(121), 2022. С. 1-16

3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. М.: Мир, 1978. – 1104 с.

4. Балухто А.Н., Романов А.А. Искусственный интеллект в космической технике: состояние, перспективы развития // Ракетно-космическое приборостроение и информационные системы. Том. 6, выпуск 1. 2019. – С. 65-75.

5. Болнокин В.Е., Чинаев П.И. Анализ и синтез автоматических систем управления на ЭВМ. Алгоритмы и программы: справочник. – М.: Радио и связь, 1991. – 256 с.

6. Валов Н.Н., Скорняков В.А. Принципы автоматизированного анализа состояния космического аппарата на основе нейромоделирования // Космонавтика и ракетостроение № 1 (58), 2010. – С. 177-182.

7. Витерби А.Д., Омура Дж. К. Принципы цифровой связи и кодирования. М.: Радио и связь, 1982. – 536 с.

8. Говорухина Т.Н. Модели, методы и алгоритмы управления и обработки информации адаптивными реконфигурируемыми модулями в телеметрических системах: автореф. дис. … канд. техн. наук: 05.13.01 / Говорухина Татьяна Николаевна. – Курск, 2013. – 23 с.

9. Головко В.А., Крощенко А.А. Персептроны и нейронные сети глубокого доверия: обучение и применение. Вестник Брестского государственного университета № 5, 2014. С. 2-12.

10. Голубев И.Ю. Модели и методы поддержки оптимального проектирования резервированных систем сбора и обработки информации кластерной архитектуры: автореф. дис. … канд. техн. наук: 05.13.12 / Голубев Иван Юрьевич. – СПб., 2013. – 19 с.

11. Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение, 2-е изд. – М.: ДМК Пресс, 2018. – 652 с.

12. Давыдов П.С. Техническая диагностика радиотехнических устройств и систем. – М.: Радио и связь, 1988. – 256 с.

13. Деев В.В., Чикуров В.А. Формирование и передача телеметрической информации в современных системах: учеб. пособие – СПб.: ВКА им. А.Ф. Можайского, 2013. – 89 с.

14. Декретное время. URL: https://ru.wikipedia.org/wiki/Декретное_время (дата обращения 12.12.2021).

15. Донсков А.В., Мишурова Н.В., Соловьев С.В. Автоматизированная система контроля состояния космического аппарата. Вестник Московского авиационного института, Том. 25, № 3, 2018. – С. 151–160.

16. Дополнительная секунда. URL: https://ru.wikipedia.org/wiki/Дополнительная_секунда (дата обращения 13.02.2021).

17. Дудкин А.А., Ганченко В.В., Марушенко Е.Е., Чарин С.Н. Контроль телеметрических параметров целевой аппаратуры космических аппаратов с использованием нейронных сетей // Вестник Брестского государственного университета № 5, 2014. – С. 21-25.

18. Емельянова Ю.Г. Алгоритмическое и программное обеспечение человеко-машинных интерфейсов с когнитивно-графическим отображением информации для систем космического назначения: автореф. дис. … канд. Техн. Наук: 05.13.11 / Емельянова Юлия Геннадиевна. – М., 2019. – 194 с.

19. Емельянова Ю. Г., Константинов К. А., Погодин С. В., Талалаев А. А., Тищенко И. П., Фраленко В. П., Хачумов В. М. Нейросетевая система контроля датчиков углов ориентации и дальности космического аппарата // Программные системы: теория и приложения, №1(1), 2010. – С. 45-59.

20. Жидков Н. П. Линейные аппроксимации функционалов. — М.: Изд-во МГУ, 1977. — 264 с.

21. Иванов С.О., Ильин Д.В., Большаков И.Ю. Сравнительное тестирование языков программирования // Вестник Чувашского университета. № 3, 2017. – С. 222-227.

22. Казаков В.В., Кравцов А.Н., Самойлов Е.Б. Методы обработки и анализа телеметрической информации при управлении космическими средствами. Курс лекций – СПб.: ВКА им А.Ф. Можайского, 2011. – 152 с.

23. Калиниченко А. Н., Юрьева О. Д. Влияние частоты дискретизации ЭКГ на точность вычисления спектральных параметров вариабельности сердечного ритма // Информационно-управляющие системы. 2., 2008. – С. 46-49.

24. Катунцев В.П., Осипов Ю.Ю., Филпенков С.Н., Тарасенков Г.Г., Краснов А.Н. Российский опыт медицинского обеспечения внекорабельной деятельности космонавтов, проведённой с борта Международной

космической станции, в 2001–2015 гг // Медицина экстремальных ситуаций, 2016. С. 8-18.

25. Киселев А.И., Медведев А.А., Меньшиков В.А. Космонавтика на рубеже тысячелетия. Итоги и перспективы. – М.: Машиностроение-Полет, 2001. – 672 с.

26. Котов О.В., Богомолов В.В., Гришин А.П., Почуев В.И., Алферова И.В., Хорошева Е.Г., Криволапов В.В., Шушунова Т.Г. Медицинские аспекты обеспечения безопасности полета экипажа МКС-64 (экспресс-анализ) // Пилотируемые полёты в космос. – Звёздный городок: ФГБУ «НИИ ЦПК им. Ю. А. Гагарина». – 2021. – № 3(40) – С. 29-42.

27. Котов О.В., Богомолов В.В., Гришин А.П., Почуев В.И., Савенко О.А., Хорошева Е.Г., Криволапов В.В., Шушунова Т.Г. Медицинские аспекты обеспечения безопасности полета экипажа МКС-65 (экспресс-анализ) // Пилотируемые полёты в космос. – Звёздный городок: ФГБУ «НИИ ЦПК им. Ю. А. Гагарина». – 2022. – № 1(42) – С. 31-49.

28. Куимов А. В. Комплексная методика параметрического синтеза адаптивной системы информационно-телеметрического обеспечения запусков перспективных ракет космического назначения: дис. … канд. техн. наук: 05.13.01 / Куимов Андрей Владимирович. – М., 2022. – 217 с.

29. Линник Ю. В. Метод наименьших квадратов и основы математико-статистической теории обработки наблюдений. — 2-е изд. — М., 1962. – 352 с.

30. Лисейкин В.А., Моисеев Н.Ф., Сайдов Г.Г., Фролов О.П. Основы теории испытаний. Экспериментальная отработка ракетно-космической техники; под ред. д-ра техн. наук В.К. Чванова. – М.: Машиностроение-Полет / Виарт Плюс, 2015. – 260 с.

31. Луч-5А. URL: https://ru.wikipedia.org/wiki/Луч-5А (дата обращения 04.12.2021)

32. Матюшин М. М., Кутоманов А. Ю., Иванов А. А., Котеля В. В. Анализ путей повышения эффективности управления космическими аппаратами различного целевого назначения за счет унификации и интеграции средств управления полетом // Инженерный журнал: наука и инновации. Вып. 11, 2021. – С. 1-16.

33. Матюшин М.М., Махалов Д.А. Автоматизированная обработка информации от бортовой системы видеоконтроля ракет-носителей // Пилотируемые полеты в космос. № 4(41), 2021. – С. 17-35.

34. Матюшин М. М., Махалов Д. А. Применение нейросетей к анализу отделения боковых блоков ракеты-носителя «Союз» // Пилотируемые полёты

в Космос. – Звёздный городок: ФГБУ «НИИ ЦПК им. Ю. А. Гагарина». – 2020. – № 4(37). – С. 42-56.

35. Матюшин М. М., Махалов Д. А., Титов А. М., Анализ параметров движения транспортных кораблей по результатам обработки телеметрической информации // Космонавтика и ракетостроение. – Королёв: АО «ЦНИИмаш». – 2020. – № 6(117) – с. 19-36.

36. Матюшин М. М., Махалов Д. А., Титов А. М. Применение свёрточного кодирования для исправления ошибок при передаче телеметрической информации // Космонавтика и ракетостроение. – 2022. – № 5(128). – с. 67-81.

37. Матюшин М. М., Саркисян Х. В. Построение оценочной функции для поддержки принятия оперативных решений при контроле параметров состояния космического аппарата // Наука и образование, вып. № 4, 2011. – С. 1-15.

38. Матюшин М.М. Системный анализ, онтологический синтез и технологические средства обработки информации в процессах принятия решений при оперативном управлении полетом объектов космической техники с Земли: автореф. дис. … д-ра техн. наук: 05.13.01 / Матюшин Максим Михайлович. – М. – 2013. – 40 с.

39. Матюшин М.М., Титов А.М. Теоретические основы обработки телеметрической информации. – М.: Машиностроение-Полет, 2018. – 508 с.

40. Майданович О. В., Каргин В. А., Мышко В. В., Охтилев М. Ю., Соколов Б. В.. Теория и практика построения автоматизированных систем мониторинга технического состояния космических средств. – СПб.: ВКА им. А.Ф. Можайского, 2011. – 219 с.

41. Махалов Д. А., Никитина М. П., Манойло А. В. Реализация обработки телеметрической информации, передаваемой через МКСР «Луч» от разгонного блока «Фрегат» при запуске КА «Канопус-В» № 5 и № 6 с космодрома «Восточный» // Сборник статей IX научно-технической конференции молодых учёных и специалистов центра управления полётами – Королёв: АО «ЦНИИмаш». – 2019. – с. 238-248.

42. Махалов Д. А., Никитина М. П. Программные средства оперативного контроля полёта ракеты-носителя по ТМИ, передаваемой через МКСР «Луч» // Материалы XXII Международной научно-практической конференции, посвященной памяти генерального конструктора ракетно-космических систем академика М. Ф. Решетнева. Часть 2. – Красноярск: СибГУ им. М. Ф. Решетнева. – 2018. – с. 135-137.

43. Махалов Д. А., Никитина М. П., Стариков Д. В. Обработка и отображение информации от БСВК РН // Сборник статей VIII научно-

технической конференции молодых учёных и специалистов центра управления полётами – Королёв: ФГУП ЦНИИмаш. – 2018, с. 257-266.

44. Махалов Д. А., Никитина М. П., Усиков С. Б., Манойло А. В. Телеметрическое обеспечение оперативного контроля полёта ракет и разгонных блоков с использованием спутникового контура управления // Сибирский журнал науки и технологий. – Красноярск: СибГУ им. М. Ф. Решетнева. – 2019. – Т. 20, № 3. – с. 344-355.

45. Махалов Д. А., Разработка языка анализа телеметрической информации // Космическая техника и технологии. – Королёв: АО «ЦНИИмаш». – 2023. – № 3(42)/2023. – с. 112-123.

46. Махалов Д.А. Создание программного комплекса телеметрического обеспечения полётов КА «Канопус-В» и БКА // Новые материалы и технологии в ракетно-космической и авиационной технике. Сборник материалов VIII международной конференции молодых специалистов организаций ракетно-космической, авиационной и металлургической промышленности России, – Королёв: ФГУП ЦНИИмаш. – 2010. – С. 48-54.

47. Махалов Д. А., Тачёнов С. А., Кондаков А. Ю., Небосенко С. С. Организация обработки телеметрической информации КА «Канопус-В» и БКА // Сборник статей III научно-технической конференции молодых учёных и специалистов центра управления полётами. – Королёв: ФГУП ЦНИИмаш. – 2013. – С. 169-177.

48. Махалов Д. А., Тачёнов С.А., Никитина М. П., Манойло А. В. Обработка телеметрической информации, передаваемой через многофункциональную космическую систему ретрансляции «Луч» от ракет-носителей и разгонных блоков, при пусках с космодрома «Восточный» // Сборник статей VIII научно-технической конференции молодых учёных и специалистов центра управления полётами – Королёв: ФГУП ЦНИИмаш. – 2018. – с. 244-257.

49. Махалов Д.А., Титов А.М. Автоматизированный анализ телеметрической информации // Космонавтика и ракетостроение. № 2 (95), 2017. – С. 146-155.

50. Медведев А. А. Инновационные подходы при создании ракетно-космической техники. Унификация как проектный параметр управления эффективностью. Монография. – 2-е изд. – М.: Издательство «Доброе слово и Ко», 2020. – 400 стр.

51. Назаров А. В., Козырев Г. И., Шитов И. В., Обрученков В. П., Древин А. В., Краскин В. Б., Кудряков С. Г., Петров А. И., Соколов С. М., Якимов В. Л., Лоскутов А. И. Современная телеметрия в теории и на практике. Учебный курс. – СПб.: Наука и Техника, 2007. – 672 с.

52. Небосенко С. С., Махалов Д. А., Манжурин Ф. Ф. Разработка автоматизированной системы подготовки исходных данных для задания обработки и отображения ТМИ БВС // Сборник статей VI научно-технической конференции молодых учёных и специалистов центра управления полётами – Королёв: ФГУП ЦНИИмаш. – 2016. – с. 282-291.

53. Некоторые подробности о разделении первой и второй ступеней советских/российских ракет пакетной схемы. URL: https://kik-sssr.ru/IP_4_Turatam_old_Razdel_1.htm (дата обращения 21.02.2020).

54. Некрасов М. В. Автоматизированная система многопоточного приёма, обработки и анализа телеметрической информации: дис. … канд. техн. наук: 05.13.06 / Некрасов Михаил Викторович. – Красноярск, 2014. – 164 с.

55. Некрасов М. В., Пакман Д. Н., Антамошкин А. Н. Методы унификации современных средств обработки телеметрической информации в центрах управления полётами космических аппаратов // Вестник СибГАУ № 1(53), 2014. – С. 48-53.

56. Николаев Д. А. Модель и алгоритмы оперативной структурно-параметрической обработки телеметрической информации космических средств: автореф. дис. … канд. техн. наук: 05.13.01 / Николаев Дмитрий Андреевич. – СПб., 2017. – 17 с.

57. Ногин В.Д. Принятие решений в многокритериальной среде. – М.: Физматлит, 2002. – 176 с.

58. Основные положения Основ государственной политики Российской Федерации в области космической деятельности на период до 2030 года и дальнейшую перспективу [Электронный ресурс]: утв. Президентом РФ от 19.04.2013 N Пр-906. Доступ из справ.-правовой системы «Гарант».

59. Основы государственной политики в области использования результатов космической деятельности в интересах модернизации экономики Российской Федерации и развития ее регионов на период до 2030 года [Электронный ресурс]: утв. Президентом РФ 14 января 2014 г. N Пр-51. Доступ из справ.-правовой системы «Гарант».

60. Охтилев М.Ю. Теоретические основы и методы автоматизированного анализа измерительной информации в реальном времени и их приложение к задачам мониторинга состояний объектов ракетно-космического вооружения: дис. … док. техн. наук: 05.13.11 / Охтилев Михаил Юрьевич. – СПб., 2000. - 445 с.

61. Пачин А.С., Сергеев С.А. Сравнительный анализ параметров траектории ракет космического назначения «Союз-ФГ» и «Союз-2» этапа 1а на основе данных внешнетраекторных измерений // Инженерный журнал: наука и инновации, вып. 3, 2021. – С. 1-13.

62. Перегудов Ф.И., Тарасенко Ф.П. Введение в системный анализ. – М.: Высшая школа, 1989. – 360 с.

63. Порядок байтов. URL: https://ru.wikipedia.org/wiki/Порядок_байтов (дата обращения 08.03.2022).

64. Пономарев И. А. Разработка моделей и алгоритмов для многокритериальной оценки качества графического пользовательского интерфейса: диссертация канд. техн. наук. Москва, 2006. – 185 с.

65. Пратт Т., Зелковиц М. Языки программирования: разработка и реализация. 4-е издание. – СПб.: Питер, 2002. – 688 с.

66. Прикладные исследования и проектирование ключевых элементов и технологий управления КА, бортовых комплексов управления по теме: «Обоснование путей повышения надежности управления группировками КА различного назначения. Исследование методологии планирования каналов связи спутников-ретрансляторов. Разработка математической модели задачи планирования.» (Шифр СЧ НИР «Астролябия» (КА-2)): отчет о составной части НИР (промежуточный, этап 2) // Д.А. Орлов, Д.А. Махалов, А.В. Манойло, А.В. Куимов и др. – Королев: АО «ЦНИИмаш», 2021. – 251 с.

67. Расширенная форма Бэкуса-Наура URL: https://ru.wikipedia.org/wiki/Расширенная_форма_Бэкуса—Наура (дата обращения 07.01.2022)

68. Рашид Т. Создаём нейронную сеть. – СПб.: ООО «Альфа-книга», 2017. – 272 с.

69. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. — СПб.: Питер, 2013. — 896 с.

70. Скобцов В.Ю., Архипов В.И. Нейросетевой анализ данных телеметрической информации бортовой аппаратуры космических аппаратов. Космическая техника и технология № 3(34), 2021. – С. 111-124.

71. Соловьёв В.А., Лысенко Л.Н., Любинский В.Е. Управление космическими полётами. Часть 1. Учебное пособие – М.: Изд-во. МГТУ им. Н.Э. Баумана., 2009. – 478 с.

72. Соловьёв С.В. Принципы и методы разработки интеллектуализированных систем телеметрического контроля космических аппаратов и орбитальных комплексов.: дис. … док. техн. наук: 05.13.01 / Соловьев Сергей Владимирович, Москва, 2021. – 303 с.

73. Соловьёв С.В. Формирование требований к автоматизированной системе контроля состояния современных космических аппаратов. Современная наука: актуальные проблемы теории и практики. Серия «Естественные и технические науки». №1, 2021. – С. 115-120.

74. Т1. Титов. А.М. Комплекс телеметрического обеспечения управления полетом орбитального корабля «Буран». – Космонавтика и ракетостроение, вып. 1(74), 2014. – С. 190-200.

75. Титов А. М. Применение кодов Рида-Соломона для передачи данных от бортовых телеметрических систем и вычислительных комплексов. Космонавтика и ракетостроение. Вып. 5 (116), 2020. – С. 114-129.

76. Титов А.М. Реализация преобразований над значениями телеметрических параметров. Часть 1. Космонавтика и ракетостроение, вып. 8(93), 2016. – С. 77-86.

77. Титов А.М. Реализация преобразований над значениями телеметрических параметров. Часть 2. Космонавтика и ракетостроение, вып. 1 (94), 2017. – С. 72-85

78. Тихомиров С.А. Алгоритмы анализа телеметрической информации и поддержки принятия решений в системах автоматизации испытаний космических ракет-носителей: автореф. дис. … канд. техн. наук: 05.13.18 / Тихомиров Сергей Александрович. Рязань, 2014. – 18 с.

79. Федяев А.Ю. Алгоритмы предварительной обработки для задач сжатия данных в информационно-измерительных системах: автореф. дис. … канд. техн. наук: 05.11.16/ Федяев Александр Юрьевич. Хабаровск, 2013. – 17 с.

80. Феофилактов В.Т., Даньков В.С., Исев М.В., Шивырталов А.В., Шельпяков С.М., Вылегжанин К.И. Телеметрические системы межорбитального буксира «Фрегат» // Вестник НПО им С.А. Лавочкина № 1(22), 2014. – стр.78-83.

81. Фремке А.В. Адаптивные телеизмерительные системы – Л.: Энергоиздат, 1981. – 246 с.

82. Частота сердечных сокращений. URL: https://ru.wikipedia.org/wiki/Частота_сердечных_сокращений (дата обращения 12.05.2022)

83. Чикуров В. А., Шмелев В. В., Зиновьев В. Г. [и др.] Автоматизированная обработка телеметрической информации. Учебник. – СПб.: ВКА им. А.Ф. Можайского, 2014. – 473 с.

84. Шупейко И. Г. Теория и практика инженерно-психологического проектирования экспертизы. Учебно-методическое пособие. Минск: БГУИР, 2010. – 120 с.

85. Янченко, А. А. Анализ факторов, влияющих на длительность процесса идентификации модели КА при обработке ТМИ в ЦУП // Успехи современной радиоэлектроники. – М.: Радиотехника, 2013. – С. 88–92.

86. Янченко, А. А. Обзор современных методов построения информационно-телеметрического обеспечения управления космическими

аппаратами // Успехи современной радиоэлектроники. – М.: Радиотехника, 2013. – С. 49–59.

87. The Consultative Committee for Space Data Systems. TM synchronization and channel coding. Recommendation for Space Data System Standards // CCSDS 131.0-B-3. Blue Book. Issue 3. Washington, D.C.: CCSDS, September 2017.

88. Derdemyansky. Выбор языка программирования. URL: https://habr.com/ru/post/143556/ (дата обращения 30.11.2021).

89. Dijkstra E. Go To Statement Considered Harmful. Communications of the ACM Vol. 11 № 3 (March 1968), pp. 147-148.

90. Haykin S., Neural Networks and Learning Machines. 3rd ed. – Pearson Education, New Jersey, 2009, 906 p.

91. Heart Rate Variability. Standards of Measurement, Physiological Interpretation, and Clinical Use, Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology // Circulation. N 93, 1996. – P. 1043–1065.

92. ISO/IEC 14977 Information technology – Syntactic metalanguage – Extended BNF. 1996.

93. TDRSS Space Ground Link Terminal Forestalls Obsolescence with Component Replacement and Upgrades. SpaceOps 2012 Conference, 2012, Stockholm, Sweden.

94. Winderton. Машинное обучение. Создание нейронной сети. URL: https://www.youtube.com/watch?v=CtlHxItrvbk (дата обращения 15.08.2020).

Приложение

СВИДЕТЕЛЬСТВА НА ПРОГРАММЫ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

RU2019660783

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2019660783
Дата регистрации: **13.08.2019**
Номер и дата поступления заявки:
2019619296 26.07.2019
Дата публикации и номер бюллетеня:
13.08.2019 Бюл. № 8
Контактные реквизиты:
patent@roscosmos.ru

Автор(ы):
Махалов Дмитрий Александрович (RU),
Небосенко Сергей Сергеевич (RU),
Никитина Мария Павловна (RU),
Манжурин Федор Федорович (RU)
Правообладатель(и):
Российская Федерация, от имени которой выступает Государственная корпорация по космической деятельности «Роскосмос» (RU)

Название программы для ЭВМ:
Программный комплекс центральной системы комплекса базовых средств телеметрического обеспечения

Реферат:
Назначение программы: обработка, анализ и оценка качества ТМИ, представление результатов в графическом и текстовом видах, передача результатов на базовые средства сбора и хранения. Область применения программы: телеметрическое обеспечение управления полетами космических аппаратов. Функциональные возможности программы: обработка ТМИ по базовым и специальным алгоритмам; подготовка и проведение сеанса обработки ТМИ; запуск специальных вычислительных процессов; исполнение задач анализа ТМИ; предоставление пользователям и передача на хранение результатов обработки ТМИ; просмотр и оценка качества ТМИ, полученной на электронных носителях.

Язык программирования: C++; Java

Объем программы для ЭВМ: 2,56 Мб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

RU 2021615865

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

(12) **ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ**

Номер регистрации (свидетельства):
2021615865

Дата регистрации: **13.04.2021**

Номер и дата поступления заявки:
2021614980 09.04.2021

Дата публикации: 13.04.2021

Контактные реквизиты:
нет

Авторы:
Махалов Дмитрий Александрович (RU),
Небосенко Сергей Сергеевич (RU)

Правообладатель:
Российская Федерация, от имени которой выступает Государственная корпорация по космической деятельности «Роскосмос» (RU)

Название программы для ЭВМ:
Программа автоматизированной подготовки исходных данных

Реферат:
Программа предназначена для формирования ИД, необходимых для использования при обработке и анализе ТМИ КА типа "Канопус-В" и КА "Канопус-В-ИК", коррекции введенных данных и для предоставления на рабочем месте пользователя результатов ввода и коррекции ИД. Программа обеспечивает автоматизированный ввод на языке пользователя ИД, проверку ИД на ошибки, трансляцию ИД и перевод их во внутренние структуры данных, хранение и администрирование базы ИД, настройку алгоритмов вторичной обработки, создание и редактирование формуляров отображения. Тип ЭВМ: IBM PC-совмест. ПК; ОС: Windows с установленным пакетом Microsoft.NET Framework версии 4.5 (или выше).

Язык программирования: C#

Объем программы для ЭВМ: 3 МБ

Извещения об изменениях сведений о зарегистрированной программе для ЭВМ

Другие изменения

Изменения в поле: Правообладатель:
Российская Федерация, от имени которой выступает Государственная корпорация по космической деятельности «Роскосмос» (RU)

Дата внесения записи в Реестр: **25.03.2022**

Дата публикации и номер бюллетеня: 25.03.2022 **Бюл. №4**

УТВЕРЖДАЮ
Генеральный конструктор
ПАО «РКК «Энергия»
_____ В. А. Соловьёв
«__» _____ 2023 г.

Акт №ГК-ВС/8 от 27.01.23
об использовании (внедрении) результатов диссертационной работы Махалова Дмитрия Александровича на тему «Разработка комплекса моделей и методик автоматизированного анализа телеметрической информации в реальном масштабе времени для пилотируемых орбитальных станций с использованием специализированного языка программирования»

Комиссия ПАО «РКК «Энергия» им. С. П. Королёва» в составе председателя комиссии Калашникова Д. А. и членов комиссии: Бондаря А. А., Колганова В. М., Воробьёва В. А., Дёмина О. В., рассмотрев материалы диссертационной работы Махалова Д. А., начальника отдела АО «ЦНИИмаш», пришла к заключению, что результаты работы Махалова Д. А. были использованы при выполнении СЧ ОКР «Управление полётом российского сегмента МКС и космических кораблей. Проведение космических экспериментов. Создание модернизированного скафандра для внекорабельной деятельности. Медицинское и медико-биологическое обеспечение полётов, проведение работ по медицинскому отбору кандидатов в космонавты. Проведение работ по расширению ассортимента продуктов питания для экипажей» (Шифр СЧ ОКР: «МКС (Эксплуатация)» (Эксплуатация 4)) по государственному контракту № 1922730301751217000241351/351-8647/19/175 от 23.10.2019 между Госкорпорацией «Роскосмос» и ПАО «РКК «Энергия» и СЧ ОКР «Обеспечение подготовки к управлению и управления автономным полётом многоцелевого лабораторного модуля с улучшенными эксплуатационными характеристиками (МЛМ-У) из Центра управления полётами РС МКС» (Шифр: СЧ ОКР «МКС» (МЛМ-У)) по государственному контракту № 351-8517/15/361 от 30.12.15 г. между Госкорпорацией «Роскосмос» и ПАО «РКК «Энергия».

В указанных СЧ ОКР были использованы следующие результаты работы Махалова Д. А.:

– лингвистическая модель специализированного языка описания алгоритмов анализа ТМИ;

– методика интерпретации алгоритмов автоматизированного анализа ТМИ в реальном времени в составе программ обработки ТМИ и отображения ТМИ;

– методика формирования мнемосхем визуализации результатов анализа ТМИ с использованием программ на языке анализа ТМИ.

Использование при проведении сеансов управления орбитальными модулями российского сегмента (РС) МКС, пилотируемыми и грузовыми кораблями разработанного автором методического аппарата позволило:

– повысить качество и оперативность проведения анализа состояния РС МКС, пилотируемых и грузовых кораблей;

– автоматизировать анализ служебной ТМИ от сервера полезной нагрузки ТВМ1-Н служебного модуля РС МКС и работающей под его управлением бортовой научной аппаратуры (НА) «Икарус», НА «Терминатор-Лимб», НА «Терминатор-Надир», НА «СОКП» и аппаратуры РСПИ;

– автоматизировать анализ ТМИ многофункционального лабораторного модуля «Наука» на участке автономного полёта и при передаче ТМИ через служебный модуль РС МКС;

– автоматизировать анализ ТМИ от европейского роботизированного манипулятора ERA, установленного на многофункциональном лабораторном модуле «Наука» и передаваемой через широкополосную систему связи;

– автоматизировать учёт количества собираемой влаги в системе обеспечения жизнедеятельности служебного модуля РС МКС.

Председатель комиссии:
Калашников Д. А.,
руководитель центра 11Ц

Члены комиссии:
Бондарь А. А.,
начальник отдела

Колганов В. М.,
зам. начальника отдела

Воробьёв А. В.,
зам. начальника отдела

Дёмин О. В.,
ведущий инженер-программист

НИИТП
РОССИЙСКИЕ КОСМИЧЕСКИЕ СИСТЕМЫ

Акционерное общество
«Научно-исследовательский институт точных приборов»
(АО «НИИ ТП»)

Декабристов ул., вл. 51, Москва, 127490
Почтовый адрес: Декабристов ул., вл. 51, Москва, 127490
тел.: + 7 495 231-38-22, факс: + 7 499 204-79-66
e-mail: info@niitp.ru, http://www.niitp.ru
ОКПО 11482462, ОГРН 1097746735481
ИНН/КПП 7715784155/771501001

_____№ _____

На №_____ от _____

УТВЕРЖДАЮ
Главный конструктор по АПК
космических систем и средств –
заместитель генерального директора
АО «НИИ ТП»

_____ Г.Б. Голубин
«28» _____ 2023 г.

Акт № 19/64 от 01.03.2023

об использовании (внедрении) результатов диссертационной работы
Махалова Дмитрия Александровича

на тему «Разработка комплекса моделей и методик автоматизированного анализа телеметрической информации в реальном масштабе времени для пилотируемых орбитальных станций с использованием специализированного языка программирования»

Комиссия АО «НИИ ТП» в составе председателя комиссии - начальника отделения 19 Голубева Е.А., членов комиссии: начальника отдела 195 Грибова В.И., ведущего инженера-программиста отдела 195, к.т.н. Бондарева Н.Н., рассмотрев материалы диссертационной работы Махалова Д. А., начальника отдела АО «ЦНИИмаш», пришла к заключению, что результаты работы Махалова Д.А. были использованы при выполнении СЧ ОКР «Доработка НКУ КА «Ресурс-П» для обеспечения управления КС «Ресурс-П». Участие в ЛИ КС «Ресурс-П» (Шифр: «НКУ «Ресурс-П») по договору № 171/19-14 от 01.17.2014 АО «РКЦ «Прогресс» с АО «НИИ ТП».

В указанной СЧ ОКР были использованы следующие результаты работы Махалова Д.А.:

- лингвистическая модель специализированного языка описания алгоритмов анализа ТМИ;
- методика интерпретации алгоритмов автоматизированного анализа ТМИ в реальном времени в составе программ обработки ТМИ и отображения ТМИ;
- методика формирования мнемосхем визуализации результатов анализа ТМИ с использованием программ на языке анализа ТМИ.

Использование при создании Единого центра управления полётом (ЕЦУП) космической системы «Ресурс-П» разработанного автором методического аппарата и программного обеспечения позволило повысить качество и оперативность проведения анализа состояния космических аппаратов, как в реальном времени, так и в послесеансном режиме.

Председатель комиссии: Е.А. Голубев
Члены комиссии: В.И. Грибов
Н.Н. Бондарев

УТВЕРЖДАЮ

Заместитель генерального директора по науке

АО «Российские космические системы»

_____ А.В. Круглов

«__» ____01____ 2023 г.

Акт N 9-АК-79 от 02.02.2023 г.

об использовании (внедрении) результатов диссертационной работы Махалова Дмитрия Александровича

на тему «Разработка комплекса моделей и методик автоматизированного анализа телеметрической информации в реальном масштабе времени для пилотируемых орбитальных станций с использованием специализированного языка программирования»

Комиссия АО «Российские космические системы» в составе председателя комиссии Фролова М.А. и членов комиссии: Никишина Г.С., Жидковой С.К., Янченко А.А., рассмотрев материалы диссертационной работы Махалова Д. А., начальника отдела АО «ЦНИИмаш», пришла к заключению, что результаты работы Махалова Д. А. были использованы при выполнении СЧ ОКР «Создание наземного комплекса управления КА «Канопус-В» (Шифр: «НКУ «Канопус-В») по договору № 07/19/2009 от 26.06.2009 между АО «Корпорация ВНИИЭМ» и ФГУП «РНИИ КП» и СЧ ОКР «Создание Белорусского наземного комплекса управления Белорусским космическим аппаратом» (Шифр: СЧ ОКР «БНКУ») по договору № 07-20/2007 от 16.11.2007 между АО «Корпорация ВНИИЭМ» и ФГУП «РНИИ КП».

В указанных СЧ ОКР были использованы следующие результаты работы Махалова Д. А.:

– лингвистическая модель специализированного языка описания алгоритмов анализа ТМИ;

– методика интерпретации алгоритмов автоматизированного анализа ТМИ в реальном времени в составе программ обработки ТМИ и отображения ТМИ;

– методика формирования мнемосхем визуализации результатов анализа ТМИ с использованием программ на языке анализа ТМИ.

Использование при создании Центра управления полётом (ЦУП) космической системы «Канопус-В» и ЦУП Белорусского космического аппарата (БКА) разработанного автором методического аппарата и программного обеспечения позволило:

– повысить качество и оперативность проведения анализа состояния космических аппаратов, как в реальном времени, так и в послесеансном режиме;

– автоматизировать учёт наработки бортовых систем;

– проводить совместный анализ состояния управляемого КА и используемой для управления наземной станции командно-измерительной системы «Клён».

Председатель комиссии:
Фролов М.А.,
начальник центра ц68, к.т.н.

Члены комиссии:
Никишин Г.С.,
начальник отделения

Жидкова С.К.,
начальник отдела

Янченко А.А.,
ведущий научный сотрудник, к.т.н.

ЦНИИМАШ
TSNIIMASH

Государственная корпорация
по космической деятельности «Роскосмос»

Акционерное общество
«Центральный научно-исследовательский институт
машиностроения» (АО «ЦНИИмаш»)

ул. Пионерская, д. 4, корп. 22
г.о. Королёв,
Московская область, 141070

Тел.: +7 (495) 513 5951
Факс: +7 (495) 512 2100

e-mail: corp@tsniimash.ru
http://www.tsniimash.ru

ОГРН 1195081054310
ИНН / КПП 5018200994 / 501801001

_____ исх. № _____
исх. №_____ от _____

УТВЕРЖДАЮ
Первый заместитель генерального директора –
начальник ЦУП, д.т.н.
_____ М. М. Матюшин
«27» февраля 2023 г.

Акт №08103-59 от 27.02.2023
о внедрении результатов диссертационной работы
Махалова Дмитрия Александровича

В соответствии с распоряжением первого заместителя генерального директора – начальника ЦУП № 08-33рп от 21.02.23 комиссия в составе председателя комиссии А. Ю. Кутоманова и членов комиссии: А. В. Донскова, В. С. Паненко, А. М. Титова рассмотрела материалы диссертационной работы начальника отдела Д. А. Махалова «Разработка комплекса моделей и методик автоматизированного анализа телеметрической информации в реальном масштабе времени для пилотируемых орбитальных станций с использованием специализированного языка программирования» на соискание учёной степени кандидата технических наук по специальности 2.3.1. «Системный анализ, управление и обработка информации, статистика (технические науки)».

В результате рассмотрения установлено, что результаты диссертационной работы Д. А. Махалова вошли в:

– предложения по путям повышения надёжности управления КА за счёт повышения оперативности и качества анализа ТМИ, выполняемого в реальном времени;

– предложения по перспективным технологиям автоматизированного анализа телеметрической информации

при подготовке НТО по СЧ НИР «Астролябия» (КА-2), этап 2 по госконтракту № 1922730202382217000241851/851-0226А/19/238 от 27.12.2019 между Госкорпорацией «Роскосмос» и АО «ЦНИИмаш».

Председатель комиссии:
зам. начальника ЦУП по НИР, к.т.н. А. Ю. Кутоманов

Члены комиссии:
и. о. начальника отделения А. В. Донсков

зам. начальника отдела В. С. Паненко

и.о. главного научного сотрудника, к.ф.-м.н. А. М. Титов