

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

Профиль подготовки: «Программное обеспечение средств вычислительной техники и автоматизированных систем»

БАКАЛАВРСКАЯ РАБОТА

на тему

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ НАСТРОЙКИ БЛОКА

ИЗМЕРЕНИЯ ПОДАЧИ РАКЕТНОГО ТОПЛИВА

Студент

 31.05.2022
(подпись, дата)

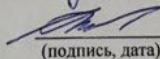
Емелин Сергей Владимирович
(ФИО полностью)

Руководитель

 31.05.22
(подпись, дата)

Н.С. Карамышева
(фамилия, инициалы)

Нормоконтролёр

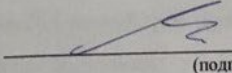
 31.05.2022
(подпись, дата)

А.С. Бычков
(фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 02.06.22 г. №

13)

Заведующий кафедрой


(подпись)

М.А. Митрохин
(фамилия, инициалы)

Работа защищена с отметкой хорошо (протокол заседания ГЭК от

20 июня 2022 № 12)

Секретарь ГЭК


(подпись)

Е.И. Гурин
(фамилия, инициалы)

Пенза 2022

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
« 26 » апреля 2022 г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Емелин Сергей Владимирович гр. 18ВВ2 факультета вычислительной техники
2. Тема работы «Программное обеспечение для настройки блока измерения подачи ракетного топлива»

Тема утверждена приказом ПГУ № 357/о от "26" апреля 2022 г.

3. Руководитель бакалаврской работы к.т.н., доцент каф. ВТ ПГУ Карамышева Н. С.

4. Задание на работу (назначение разработки, исходные данные и т.п)

Анализ использования программных приложений при создании ракетно-космической техники и определение способов разработки программного обеспечения для настройки блока измерения подачи ракетного топлива.

5. Перечень подлежащих разработке вопросов

Требуется произвести анализ предметной области, изучить протокол информационного взаимодействия с блоком датчиков измерения подачи ракетного топлива, выбрать среду разработки, определить интерфейс, спроектировать, реализовать, отладить и провести тестирование разработанного программного обеспечения.

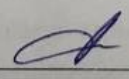
6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ предметной области	32	26.04.2022 – 28.05.2022	
2.	Анализ требований	32	29.04.2022 – 30.05.2022	
3.	Выбор технологий, применяемых при проектировании и разработки	30	1.04.2022 – 3.05.2022	
4.	Разработка программы	40	4.05.2022 – 8.05.2022	
5.	Разработка графического интерфейса	66	9.05.2022 – 15.05.2022	
6.	Отладка и тестирование	70	16.05.2022 – 23.05.2022	
7.	Оформление пояснительной записки	54	24.05.2022 – 31.05.2022	

Дата выдачи задания "26" апреля 2022 г.

Руководитель бакалаврской работы  26.04.2022.
(подпись, дата)

Задание к исполнению принял студент  26.04.2022
(подпись, дата)

Работу к защите допустить
Декан факультета  17.06.2022
(подпись, дата)

Реферат

Работа содержит 114 страниц формата А4, 48 рисунков, 2 таблицы, 4 источника, 4 приложения.

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, XML, ВТОРИЧНЫЙ БЛОК, ВНУТРЕННЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, КОСМИЧЕСКИЙ АППАРАТ, РАКЕТНО-КОСМИЧЕСКАЯ ТЕХНИКА

Объект разработки – программное обеспечение для блока измерения подачи ракетного топлива.

Цель работы – разработка программного обеспечения для настройки блока измерения подачи ракетного топлива. Это достигается путем сбора значений с вторичных преобразователей и их программированием ВПО. Необходимо решить задачу по настройке вторичных преобразователей блока измерения. В результате проведенной работы создано программное обеспечение, протестированы программные средства, удовлетворяющие поставленной задаче.

В пояснительной записке описана архитектура решения, обзор технологии разработки и отладки программы.

Графическая часть содержит UML-диаграммы приложения.

					ПГУ 1.090301.12.001 ПЗ				
Изм.	Лист	№ докум.	Подпись	Дата	Программное обеспечение для настройки блока измерения подачи ракетного топлива Пояснительная записка	Лит.	Лист	Листов	
Разраб.		Емелин С.В.					2	114	
Проверил		Карамышева Н.С.							
Н. Контр.		Бычков А.С.							
									
Утверд.		Митрохин М.А.							
						ФВТ, каф. ВТ Гр. 18ВВ2			

Содержание

Перечень принятых сокращений	4
Введение.....	5
1 Программное обеспечение для настройки блока измерения подачи ракетного топлива	6
1.1 Анализ технического задания	6
1.2 Актуальность решения	7
1.3 Выбор инструментов разработки	8
1.3.1 Язык разработки.....	8
1.3.2 Выбор среды разработки.....	9
1.3.3 Средство разработки пользовательского интерфейса.....	12
1.4 Описание архитектуры решения	13
1.5 Процесс проектирования и разработки	14
1.6 Тестирование и отладка.....	26
1.6.1 Выбор режима тестирования приложения	26
1.6.2 Тестирование приложения	27
2 Безопасность жизнедеятельности.....	33
2.1 Потенциально-опасные и вредные факторы	33
2.2 Пожарная безопасность	34
2.3 Электробезопасность	36
2.4 Определение класса условий труда.....	38
Заключение	41
Список использованных источников	42
Приложение А – Листинг программы.....	43
Приложение Б – <i>UML</i> диаграммы	75
Приложение В – Результаты тестирования	80
Приложение Г – Презентация	102

Перечень принятых сокращений

IDE (Integrated development environment) – интегрированная среда разработки

XML (Extensible Markup Language) – расширяемый язык разработки

QML (Qt Modeling Language) – декларативный язык программирования

UI (User Interface) – пользовательский интерфейс

GUI (Graphical User Interface) – графический интерфейс пользователя

API (Application Programming Interface) – программный интерфейс приложения

ПО – программное обеспечение

КА – космический аппарат

РКТ – ракетно-космическая техника

ПЗУ – постоянное запоминающее устройство

ВПО – внутреннее программное обеспечение

					ПГУ 1.090301.12.001 ПЗ	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

Введение

Запуск первой ракеты-носителя с жидкостным ракетным двигателем на орбиту является большим достижением человечества. Однако цифровизация, как катализатор изменили космическую отрасль промышленности.

При разработке и создании космических аппаратов (КА) принято обращать внимание на то, каким образом эти аппараты работают. В большинстве случаев объектом внимания становится конструкция аппарата, а также установленные в космический аппарат модули и устройства.

Учитывая сложность задач, выполняемых современными модулями, приборами, ни один КА не сможет выполнять свои функции без программного обеспечения (ПО), которое обеспечивает взаимодействие всех внутренних частей космического аппарата. Если аппаратное обеспечение является телом аппарата, то ПО является его интеллектом, которое позволяет людям управлять КА, даже находясь на другом конце Солнечной системы.

В ракетно-космической технике (РКТ) программное обеспечение находит применение и развитие в области научных исследований, проектирования и производства, испытаний и управления полетом, т.е. на всех этапах жизненного цикла ракетно-космической техники. Современное проектирование РКТ осуществляется с широким применением систем автоматизированного проектирования. В ходе производства программное обеспечение используется как интеллектуальное ядро систем контроля качества устройств и диагностического контроля. На контрольно-испытательных станциях и космодромах эффективное проведение испытаний возможно только при наличии высокотехнологичных аппаратно-программных комплексов.

В данной работе будет рассмотрена реализация программного обеспечения для настройки блока измерения подачи ракетного топлива. ПО позволяет ускорить и повысить качество настройки блоков измерения подачи ракетного топлива.

					ПГУ 1.090301.12.001 ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

1 Программное обеспечение для настройки блока измерения подачи ракетного топлива

1.1 Анализ технического задания

В данной работе реализуется программное обеспечение, которое должно обеспечить настройщиков блока измерения подачи ракетного топлива следующим функционалом:

- Изменение настроек программы для разных сценариев взаимодействия с блоком измерения подачи ракетного топлива;
- Запись внутреннего программного обеспечения во вторичный преобразователь каждого датчика блока;
- Прием данных в режиме первоначальной настройки;
- Запись констант в ПЗУ микроконтроллера вторичного преобразователя каждого датчика блока;
- Градуировка датчиков перемещения и температуры;
- Сохранение данных градуировки в файл с расширением *XML*;
- Просмотр данных ранее сделанных градуировок датчиков перемещения и температуры;
- Запись данных градуировки в ПЗУ микроконтроллера вторичного преобразователя каждого датчика блока;
- Чтение данных градуировки из ПЗУ микроконтроллера вторичного преобразователя каждого датчика блока;
- Контроль выходного кода датчиков;
- Возможность формирование отчетов в файл формата *XLSX* по данным градуировки каждого датчика блока;

Также стоит учитывать, что приложение должно быть совместимо с операционными системами семейства Windows, иметь удобный и понятный пользовательский интерфейс для работы с большим объемом данных.

					ПГУ 1.090301.12.001 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

1.2 Актуальность решения

В связи с широким использованием объектов ракетно-космической техники многоцелевого назначения и активным управлением ими возрастает актуальность применения программного обеспечения, позволяющего существенно облегчить и повысить оперативность и качество управления.

Переход на использование в своих изделиях цифровых технологий значительно повысить качество и сократить сроки создания РКТ в ракетно-космической промышленности. За прошедшие десятилетия развитие мирового космического рынка фактически охватило многие регионы мира, при этом разработка все новых космических программ, вступление в космическую деятельность новых государств и, как результат, естественное усиление конкуренции в значительной степени стимулируют развитие космонавтики, повышая ее коммерческий потенциал, развивая новые технологические разработки и открывая все новые сферы ее применения.

С каждым годом количество запусков космических аппаратов растет, количество успешных запусков увеличивается. Это можно увидеть на статистике приведенной на рисунке 1, приведенным ниже.

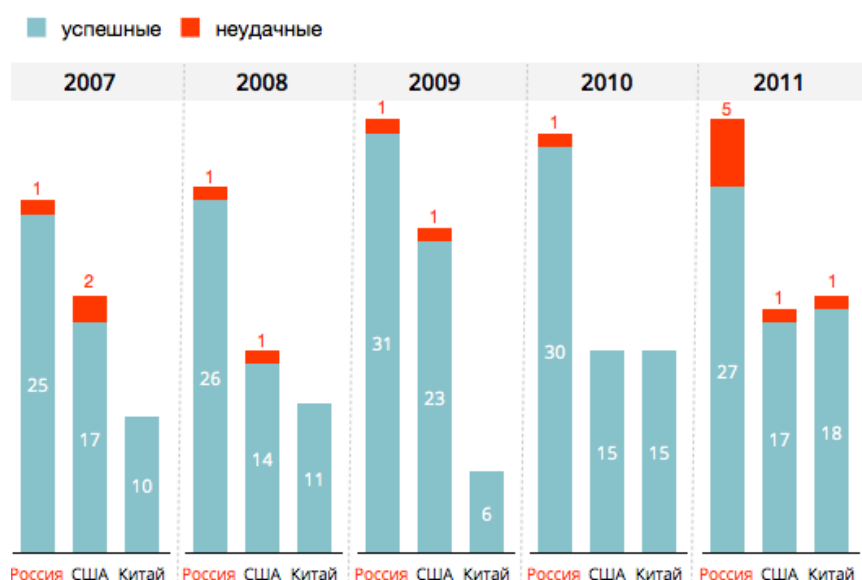


Рисунок 1 - Статистика роста запусков космических аппаратов

Исходя из этого, можно сказать, что потребность космической индустрии в программных продуктах, различном цифровом обеспечении только растет, вслед за технологическим прогрессом.

Одним из таких программных продуктов может послужить реализуемый проект. Разрабатываемое программное обеспечение включает в себя приём данных, настройка каждого датчика блока, программирование ПЗУ датчиков, мониторинг изменения показателей, автоматическое формирование отчетов, интуитивно понятный для пользователя интерфейс.

1.3 Выбор инструментов разработки

Для разработки данного программного обеспечения были использованы различные программные средства, позволяющие реализовать, расширить функциональность программы, ее внешний вид и стабильность работы, язык разработки и среда разработки.

1.3.1 Язык разработки

Исходя из требований, указанных выше, для разработки программного обеспечения был выбран язык программирования C++.

C++ - это мультипарадигмальный язык, включающий широкий спектр различных стилей и технологий программирования который подходит для выполнения различного рода задач, а так же является компилируемым языком, а это значит, что компилятор транслирует исходный код на C++ в исполняемый файл, который содержит набор машинных инструкций. является компилируемым языком, а это значит, что компилятор транслирует исходный код на C++ в исполняемый файл, который содержит набор машинных инструкций.

Данный язык использует объектно-ориентированный подход, который позволяет создавать конструкции (или объекты), способные принимать на вход данные и отдавать их другим объектам для построения между ними взаимодействия. Такой подход в программировании предоставляет

возможность не запоминать большие блоки данных в памяти, а работать с отдельными объектами, которые уже хранят в себе определенную структуру.

Одним из важных критериев становится производительность выбранного языка программирования, C++ стоит вторым по производительности среди других языков. Скорость написания кода на порядок выше у интерпретируемых языков, но если выбирать для конечного пользователя между временем, потраченным на разработку и скоростью и стабильностью работы, то выигрывает второе.

C++ универсален, компиляторы C++ есть на каждой операционной системе, большинство программ легко переносится с платформы на платформу.

Также плюсом является наличие бесплатных версий у нескольких платформ для разработки, таких как: *Microsoft Visual Studio*, *QT*, *Embarcadero RAD Studio* для *Windows* и *Mac Os*.

К описанным выше преимуществам можно добавить также большое количество библиотек и шаблонов разработки, позволяющие быстро найти решение возникающих проблем и потребностей в коде.

В процессе обучения более близко проходило ознакомление именно с этим семейством языка.

1.3.2 Выбор среды разработки

IDE (Integrated development environment – интегрированная среда разработки) – это комплекс программных средств для разработки программного обеспечения простым, быстрым и надежным способом. *IDE* включает в себя компилятор, отладчик, подсветку синтаксиса и настраиваемый интерфейс. *IDE* облегчает визуальное представление файлов и делает его более понятным для пользователя.

Рассмотрим наиболее популярные среды разработки: *Visual Studio*, *Eclipse* и *Qt Creator*.

Visual Studio – официальная, бесплатная, функциональная *IDE*, имеющая облачные хранилища. Однако данная среда довольно сложна в

					ПГУ 1.090301.12.001 ПЗ	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

освоении, при переходе на платную версию нередко теряются настройки и нарушается работа корпоративного сервера.

Eclipse ориентирована преимущественно на разработку *Java*-приложений, но полезна и для кодов на *C#*. Имеет хороший компилятор, высокую функциональность, а также данная среда разработки бесплатна. Из недостатков: ненадежность и сложность.

Учитывая достоинства и недостатки рассмотренных сред, выбор пал на *Qt Creator*.

Qt Creator – это *IDE* для разработки на языках программирования *C*, *C++*, *JavaScript* и *QML* которая предоставляет инструменты проектирования и разработки приложений для настольных и мобильных платформ, а так же включает в себя встроенный дизайнер интерфейсов. В работе [1] описано «данная библиотека является полностью объектно-ориентированной, что обеспечивает легкое расширение возможностей и создание новых компонентов».

Qt Creator предоставляет удобный интерфейс отладки программы, возможность остановить программу в любом месте написанного кода и проследить изменения значений в переменных отвечающих за различный функционал, указание строки в которой возникает ошибка и предлагает возможное решение.

Поддерживаются следующие режимы отладки:

- простой для отладки локально запущенных приложений, таких как *GUI* приложения на *Qt*;
- терминал для отладки локально запущенных процессов, которым требуется консоль, обычно это приложения без *GUI*;
- подключённый для отладки локальных процессов, запущенных вне *Qt Creator*;
- удалённый для отладки запущенных на другой машине процессов (используя *gdbserver*);

- ядро для отладки завершившихся аварийно процессов на *Unix*;
 - Post-mortem для отладки завершившихся аварийно процессов на Windows;
 - TRK для отладки процессов, запущенных на устройстве Symbian.
- Точки останова можно задать различными способами, а именно:
- останавливаться на заданной строчке заданного файла;
 - останавливаться при вызове функции с определенным именем;
 - останавливаться при обращении к данным по заданному адресу;
 - останавливаться при поимке исключения;
 - останавливаться при запуске или создании нового процесса;
 - останавливаться при выполнении системного вызова;
 - останавливаться при изменении в данных с адресами, заданными выражением.

Предоставляемый функционал отладки позволяет быстро исправить программу и написать чистый и правильный код.

Используемая среда разработки имеет уникальный механизм взаимодействия между объектами – сигналы и слоты. Он является центральной функцией в *Qt* которая отличает его от других сред разработки. Слотом является функция, которая вызывается в ответ на определенный сигнал. Сигнал выполняется только тогда, когда происходит определенное событие. Сигналы и слоты являются типобезопасным механизмом, в отличие от парадигмы событийного управления.

Достоинствами *Qt Creator* являются: удобный редактор исходного кода, уникальный механизм сигналов и слотов, кроссплатформенность и средство отладки программы, быстрая разработка и внедрение за счет комплексных *API*, переносимость на уровне исходного кода для любой операционной системы.

					ПГУ 1.090301.12.001 ПЗ	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

1.3.3 Средство разработки пользовательского интерфейса

После выбора языка программирования и среды разработки, появился вопрос о выборе средстве создания пользовательского интерфейса.

Для создания пользовательского интерфейса, был выбран предлагаемый средой разработки модуль, позволяющий решить данный вопрос.

Qt Widgets – встроенный модуль среды разработки *Qt Creator*, который представляет собой набор элементов (виджетов) для создания пользовательского интерфейса.

Виджеты – это блоки, используя которые реализуется создание пользовательского интерфейса. Виджеты это не просто визуальное отображение определенных блоков, а компонент, способный выполнять различные действия, например, реагировать на поступающие сигналы и события или отправлять сигналы другим виджетам.

Виджеты дают базовую возможность для отрисовки на экране и обработке событий пользовательского ввода, получать вводимые пользователем данные, предоставляет контейнер для группировки других виджетов, компоновку виджетов внутри контейнера, позволяют создать свои собственные элементы управления, наследуя классы уже существующих. Простым примером виджетов являются виджеты управления, например кнопки, выпадающие списки, полосы прокрутки и так далее. Каждому виджету возможно придать уникальный внешний вид - стиль, размер, шрифт, цвет.

Таким образом, создание различных меню, таблиц, списков, кнопок, полей ввода и вывода и создание уникального дизайна своей программы, становится легкой задачей для пользователя среды разработки.

1.4 Описание архитектуры решения

Необходимо разработать программное обеспечение, позволяющее настроить блок измерения подачи топлива.

Графическая часть программного обеспечения, разработанная с использованием технологии «*Qt Widgets*», должна предоставить пользователю интуитивно понятный интерфейс.

Пользовательский интерфейс должен обеспечивать удобство доступа к различным видам информации, представленным в программном обеспечении и позволять пользователю своевременно реагировать на изменения. Для решения поставленной задачи, требуется создать *ui*-блоки для работы пользователя с данными.

Для отображения данных используются графические элементы (поля вывода). Также должна присутствовать возможность сохранения и занесения в таблицы и файлы, полученные и измененные пользователем данные для мониторинга изменения и проверки правильности работы устройства. Такая возможность позволяет другому пользователю отследить произведенные изменения и внести свои корректировки в настройку. Пользователь должен видеть все значения и показатели, с которыми он работает для успешной настройки блока измерения. Блок должен быть настроен максимально точно с минимальными погрешностями.

Должна быть возможность создания отчетов с результатами настройки датчиков перемещения и температуры. Для этого присутствует заранее созданный шаблон *Excel* файла, в который заносятся все результаты по нажатию кнопки.

Для занесения данных в устройство, используются поля ввода, которые позволяют заносить настроенные значения в устройство и блоки работы с памятью. Благодаря этому, возможно получить и просмотреть данные которые установлены изначально, произвести настройку, занести измененные данные и отследить их правильность. Для проверки точности,

создаются блоки работы с памятью каждого датчика находящимся в устройстве. Только благодаря видимости параметров датчиков, возможно точно произвести настройку устройства.

1.5 Процесс проектирования и разработки

Основное окно, с которого настройщик начинает работу это *ui*-форма *Mainwindow*. *Ui*-форма это *UI*-файл который представляет собой страницу с виджетами позволяющими построить пользовательский интерфейс. Формы обрабатываются во время компиляции, что означает, что формы будут преобразованы в код *C++*, который может быть скомпилирован. В форме *Mainwindow* находятся все вкладки в которых реализуется требуемый функционал для настройки блока управления подачи ракетного топлива. В зависимости от того, какую вкладку выберет настройщик, тот класс с графическим интерфейсом и будет вызван.

Ниже представлен фрагмент реализации вызова классов в зависимости от нажатой кнопки:

```
void MainWindow::on_pushButton_clicked()
{
    settingsadapter = new settingsAdapter(this);
    settingsadapter -> setModal(true);
    settingsadapter-> show();
}

void MainWindow::on_pushButton_2_clicked()
{
    setting = new Setting(this);
    setting -> setModal(true);
    setting -> show();
}

void MainWindow::on_pushButton_4_clicked()
{
    functionControl = new functioncontrol(this);
    functionControl -> setModal(true);
    functionControl -> show();
}
```

					ПГУ 1.090301.12.001 ПЗ	Лист
						14
Изм.	Лист	№ докум.	Подпись	Дата		

}

Окно главного меню показано на рисунке 2.

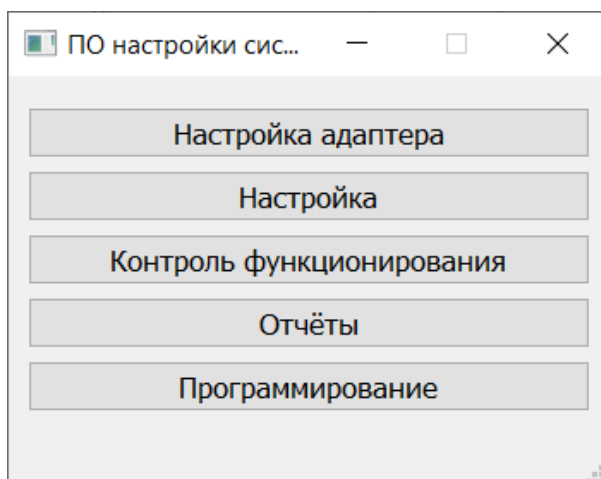


Рисунок 2 – Окно главного меню

Главное окно, имеет следующие кнопки: *pushButton*, *pushButton2*, *pushButton3*, *pushButton4*, *pushButton5*. Нажимая каждую кнопку, срабатывает сигнал *clicked()* и открывается нужная вкладка.

Нажав кнопку *pushButton2* из класса *Mainwindow*, открывается страница настройки. Тут используются виджеты для отображения выходных кодов датчиков перемещения и температуры *LineEdit*. Все данные принимаются в типе *string*. Каждое такое поле заблокировано для изменения настройщиком, во избежание случайного редактирования или удаления данных. А так же используются кнопки *btn_start_setting*, *btn_stop_setting*, *pushButton* и *pushButton_2*. По нажатию первой кнопки срабатывает сигнал *clicked()* и происходит вызов слота *disableStop* где описывается механика считывания выходных данных с датчиков находящимся в блоке, устанавливаются константы и блокируется кнопка которая была нажата и разблокируется кнопка *btn_stop_setting*.

Ниже представлен фрагмент срабатывания сигнала *clicked()* кнопки *btn_start_setting*:

```
void Setting::disableStop()
{
    ui->lndet_N0_DP1->setText(QString::number(N0));
```

```

        ui->lndt_N0_DP2->setText(QString::number(N0));
        ui->lndt_Nnom_DP1->setText(QString::number(Nnom));
        ui->lndt_Nnom_DP2->setText(QString::number(Nnom));
        ui->lndt_Range_DP1->
>setText(QString::number(RangeCode));
        ui->lndt_Range_DP2->
>setText(QString::number(RangeCode));
        ui->btn_start_setting->setEnabled(false);
        ui->btn_stop_setting->setEnabled(true);
    }

```

На рисунке 3 представлено окно настройки.

Рисунок 3 – Окно настройки

По кнопкам *pushButton* и *pushButton_2* происходит переход в окна настройки датчиков перемещения и температуры. Нажимая первую кнопку срабатывает сигнал *clicked()* и открывается *ui*-форма с настройкой датчиков перемещения. На данной вкладке используются виджеты *QComboBox* – для выбора датчика перемещения, *QLineEdit*, где выводятся считанные данные с датчика, и таблица *QTableView* куда настройщик записывает данные, *QGroupBox* – для группировки виджетов, тем самым создав для пользователя удобный интерфейс, а так же *QLabel* для отображения заводского номера блока. По нажатию кнопки *btn_start*, начинается считывание показателей с

датчика. Поля таблицы заблокированы для редактирования пользователем.

Таблица класса *QStandardItemModel*, состоит из четырех столбцов, и 13 строк которые имеют строковый тип данных.

Таблица создается следующим образом:

```
model = new QStandardItemModel(13, 4, this);
model->setHeaderData(0, Qt::Horizontal, "Прямой ход");
model->setHeaderData(1, Qt::Horizontal, "Обратный ход");
model->setHeaderData(2, Qt::Horizontal, "Прямой ход");
model->setHeaderData(3, Qt::Horizontal, "Обратный ход");
model->setHeaderData(0, Qt::Vertical, "-18мм");
model->setHeaderData(1, Qt::Vertical, "-15мм");
model->setHeaderData(2, Qt::Vertical, "-12мм");
model->setHeaderData(3, Qt::Vertical, "-9мм");
model->setHeaderData(4, Qt::Vertical, "-6мм");
model->setHeaderData(5, Qt::Vertical, "-3мм");
model->setHeaderData(6, Qt::Vertical, "0мм");
model->setHeaderData(7, Qt::Vertical, "3мм");
model->setHeaderData(8, Qt::Vertical, "6мм");
model->setHeaderData(9, Qt::Vertical, "9мм");
model->setHeaderData(10, Qt::Vertical, "12мм");
model->setHeaderData(11, Qt::Vertical, "15мм");
model->setHeaderData(12, Qt::Vertical, "18мм");
```

По кнопке записать, настройщик записывает данные в таблицу, каждое нажатие, добавляет одну запись в таблицу, где x является считанным показателем.

Код слота реагирующего на сигнал нажатия представлен ниже:

```
QString x = ui->lineEdit_liner4->text();
model->setData(model->index(curRow, curCol), x);

bool isColEven = (curCol % 2) == 0;
curRow = isColEven ? curRow + 1 : curRow - 1;
if (curRow == model->rowCount() || curRow == -1)
{
    curCol++;
    isColEven = (curCol % 2) == 0;
    curRow = isColEven ? 0 : model->rowCount() - 1;
}
```

```
countClickLin++;
if(countClickLin==52)
    ui->btn_write->setEnabled(false);
```

Так же реализованы кнопки сохранения и открытия файла. Загружаемый и сохраняемый файл должен быть в формате *XML*. «*XML* — расширяемый язык разметки для создания логической структуры данных, их хранения и передачи в виде, удобном и для компьютера, и для человека. Отличается простотой синтаксиса и универсальностью. *XML* позволяет описывать документы с помощью тегов, которые можно задавать самостоятельно» – рассмотрено в работе [2].

Для сохранения и загрузки файлов используются объекты класса *QXmlStreamReader* для чтения и *QXmlStreamWriter* для записи. Чтобы подключить данные классы требуется подключить библиотеки *QXmlStreamReader* и *QXmlStreamWriter*. Далее требуется создать объект класса который требуется. Механизм создания и считывания данных из файла, используется на протяжении всего проекта. Нажимая кнопку сохранить срабатывает сигнал *clicked()* и выполняется код слота представленный ниже:

```
QXmlStreamWriter xmlWriter(&file);
xmlWriter.setAutoFormatting(true);
xmlWriter.writeStartDocument();

QString dateTime =
QDateTime::currentDateTime().toString("dd.MM.yyyy
hh.mm.ss");
QString factoryNumber = ui->factory_numberLi->text();
const ushort rows = static_cast<ushort>(model-
>rowCount());
const ushort columns = static_cast<ushort>(model-
>columnCount());
QString prepareData[rows][columns];
for (ushort row = 0; row < rows; ++row)
    for (ushort column = 0; column < columns; ++column)
        prepareData[row][column] = model->data(model-
>index(row, column)).toString();

xmlWriter.writeStartElement("data");
xmlWriter.writeStartElement("date");
xmlWriter.writeCharacters(dateTime);
xmlWriter.writeEndElement();
```

```

xmlWriter.writeStartElement("number");
xmlWriter.writeCharacters(factoryNumber);
xmlWriter.writeEndElement()

xmlWriter.writeStartElement("table");

for (ushort row = 0; row < rows; ++row)
    for (ushort column = 0; column < columns; ++column)
    {
        xmlWriter.writeStartElement("row");

xmlWriter.writeCharacters(prepareData[row][column]);
        xmlWriter.writeEndElement();
    }

xmlWriter.writeEndElement();
xmlWriter.writeEndElement();
xmlWriter.writeEndDocument();

```

Создается *XML*-файл с тегами *data*, *date* *datetime* *table* и *row*. Тег *data* является корневым тегом, где находятся все другие. *Date* является родительским для тега *datetime* в котором хранится дата создания файла. В теге *table* находится дочерний тег *row* в котором хранятся все выходные данные с датчика записанные настройщиком.

Далее рассмотрим работу кнопки открыть. По нажатию данной кнопки срабатывает сигнал *clicked()* и вызывается слот *on_btn_open_clicked()*. После нажатия настройщик наблюдает перед собой окно выбора файла, директория заранее прописанная как для сохранения файла, так и открытия. Настройщику требуется только выбрать файл с которым он хочет работать. После выбора файла происходит считывание данных и занесение их в таблицу. Данные заносятся в том же порядке в котором и заносились, поэтому достоверность правильности считанных данных гарантируется. Данная реализация находится в вызываемом по нажатию слоте *on_btn_open_clicked()*.

Ниже представлена часть кода слота считывающая данные из таблицы:

```

while (!xmlReader.atEnd() && !xmlReader.hasError())
{
    xmlReader.readNext();

    if (xmlReader.isStartElement() && xmlReader.name() ==
"row")

```

```

        {
            if(row < model->rowCount() && col < model-
>columnCount())
                model->setData(model-
>index(row,col),xmlReader.readElementText());

            col++;
            if(col >= model->columnCount()){
                col = 0;
                row++;
            }
        }
    }
}

```

Цикл проходит по всему документу, проверяет все теги, из них выбирает тег *row*. И считывает все что находится в этих тегах. Таким образом и считываются данные из файла.

По кнопке отменить, происходит отмена предыдущей записи, данные которые были записаны в таблицу по одному нажатию стираются из ячейки. Цикл отслеживает текущий индекс массива и очищает нужную ячейку.

Ниже приведен код слота реализующий данный механизм.

```

bool isColEven = (curCol % 2) == 0;
curRow = isColEven ? curRow - 1 : curRow + 1;
if(curRow == model->rowCount() || curRow == -1)
{
    curCol--;
    isColEven = (curCol % 2) == 0;
    curRow = isColEven ? model->rowCount() - 1 : 0;
}

```

Нажимая кнопку «Отменить» срабатывает сигнал *clicked()* и реализуется программный код слота *on_btn_clear_clicked()*, обнуляются индексы массива, а все ячейки очищаются от данных.

Программный код слота *on_btn_clear_clicked()* приведен ниже:

```

for (int row = 0; row<model->rowCount(); ++row){

    for(int col = 0; col<model->columnCount(); ++col){
        QStandardItem *item = new QStandardItem();
        model->setItem(row,col, item);
        curCol=0;
        curRow=0;
    }
}

```

Сейчас рассмотрим окно настройки датчика температуры отображенном на рисунке 4.

Рисунок 4 – Окно настройки датчика температуры

В этом окне реализуется все то же, что и в настройке линейности, принимаются данные от датчика температуры и настройщик проводит настройку датчика температуры. Настройка температуры происходит в классе *Settingtemp*. В *Ui* форме этого класса используются виджеты *QLabel*, *QLineEdit*, *QTableView* и *QGroupBox*. Предназначение описанных виджетов были даны выше, здесь они выполняют ту же самую функцию что и в окне настройки линейности класса *linearitysetting*. Кнопки работают по такому же принципу. Настройщик получает выходные данные из датчика которые отображаются в виджетах типа *QLineEdit*, записывает выходные данные в таблицу реализованную с помощью класса *QStandardItemModel*. Работа с таблицей происходит таким же образом что и в описанном выше классе, запись и удаление данных, сохранение данных в файл и запись в таблицу данных из файла.

Для прохождения различных видов контроля используется окно контроля функционирования. Данное окно реализовано в *ui*-файле и использует класс *functioncontrol*. В данном окне используются такие элементы графического интерфейса как: *QLabel*, *QLineEdit* *QPushButton*.

QLabel предназначен для различных названий, *QLineEdit* для вывода получаемых значений с датчиков. *QPushButton* - это кнопки.

По нажатию кнопки «Старт» начинается считывание показателей двух датчиков перемещения и датчика температуры. Данный функционал описан в слоте *onTimeoutSetting()*. Вызывается графический интерфейс и указатель обращается к виджету *QLineEdit* *ui->lineEdit_voltageFun1->setText(QString::number(result));*. Данные записываются в типе string, поля виджета, заблокированы от изменений настройщика. В данные поля поступают исключительно данные с датчиков. А так же блокируется кнопка «Старт» и разблокируется кнопка «Стоп». Для этого разработан слот *disableStop()*.

```
ui->btn_start_func->setEnabled(false);  
ui->btn_stop_func->setEnabled(true);
```

По кнопке «Стоп», вызывается слот *disableStart()* и считывание данных прекращается, а так же блокируется от нажатия кнопка «Стоп» и разблокируется кнопка «Старт».

```
ui->btn_start_func->setEnabled(true);  
ui->btn_stop_func->setEnabled(false);
```

Создание отчетов с результатами измерения датчиков перемещения и температуры реализуется в классе reports и включает в себя 3 кнопки для отчета по каждому датчику (датчик перемещения 1, датчик перемещения 2, датчик температуры). Нажимая на кнопки срабатывает сигнал *clicked()* и в зависимости от того, какая кнопка была нажата срабатывает слот *on_report_dp1_clicked()*, *on_report_dp2_clicked()*, *on_report_dt_clicked()*. В каждом слоте реализуется похожий функционал, отличие только в количестве строк и столбцов для датчика температуры. Так же был создан специальный класс для работы с Excel файлами *workexcel*, который вызывается при срабатывании слотов. В данном классе разрабатываются методы для установки пути к файлу Excel (*setTemplatePath*), запись данных в ячейку (*writeToCell*), где получается указатель на ячейку документа и

					ПГУ 1.090301.12.001 ПЗ	Лист
						22
Изм.	Лист	№ докум.	Подпись	Дата		

происходит запись данных в выбранный диапазон ячеек, запуск приложения *Excel* с открытием выбранного шаблона (*openDocument*), сохранение документа *Excel* с использованием заводского номера, номера канала и диапазона (*saveDocument*), формирование отчета, содержащего результаты проверки выходного сигнала блока (*reportLinearityCheck*). При срабатывании слота мы создаем и обращаемся к объекту класса *workexcel* который позволяет использовать функции этого класса:

```
WorkExcel* workExcel = new WorkExcel();
QString filex1(QDir::currentPath() +
"\\Templates\\LinearityDP1.xlsx");
workExcel->setTemplatePath(filex1);

const short int result = workExcel->openDocument();
workExcel->reportLinearityCheck(data, &factoryNumber);
workExcel->saveDocument(&factoryNumber, exValue);
workExcel->closeDocument();
```

Вкладка программирование и использует ui-блок и класс *programming*. В данной вкладке используются виджеты *QLabel*, *QLineEdit*, *QTableView* и *QGroupBox*, *QToolBox*. Последний виджет представляет собой блок с вкладками, в которые можно помещать другие виджеты. Данный виджет используется для удобства представления настройщику таблиц данными требующимися для работы с памятью. Виджет содержит в себе 3 вкладки с таблицами для двух датчиков перемещения и датчика температуры. На каждой вкладке находятся кнопки для работы с таблицей «Открыть», «Сохранить», «Записать в ПЗУ», «Прочитать из ПЗУ». Вкладка программирование показана на рисунке 5.

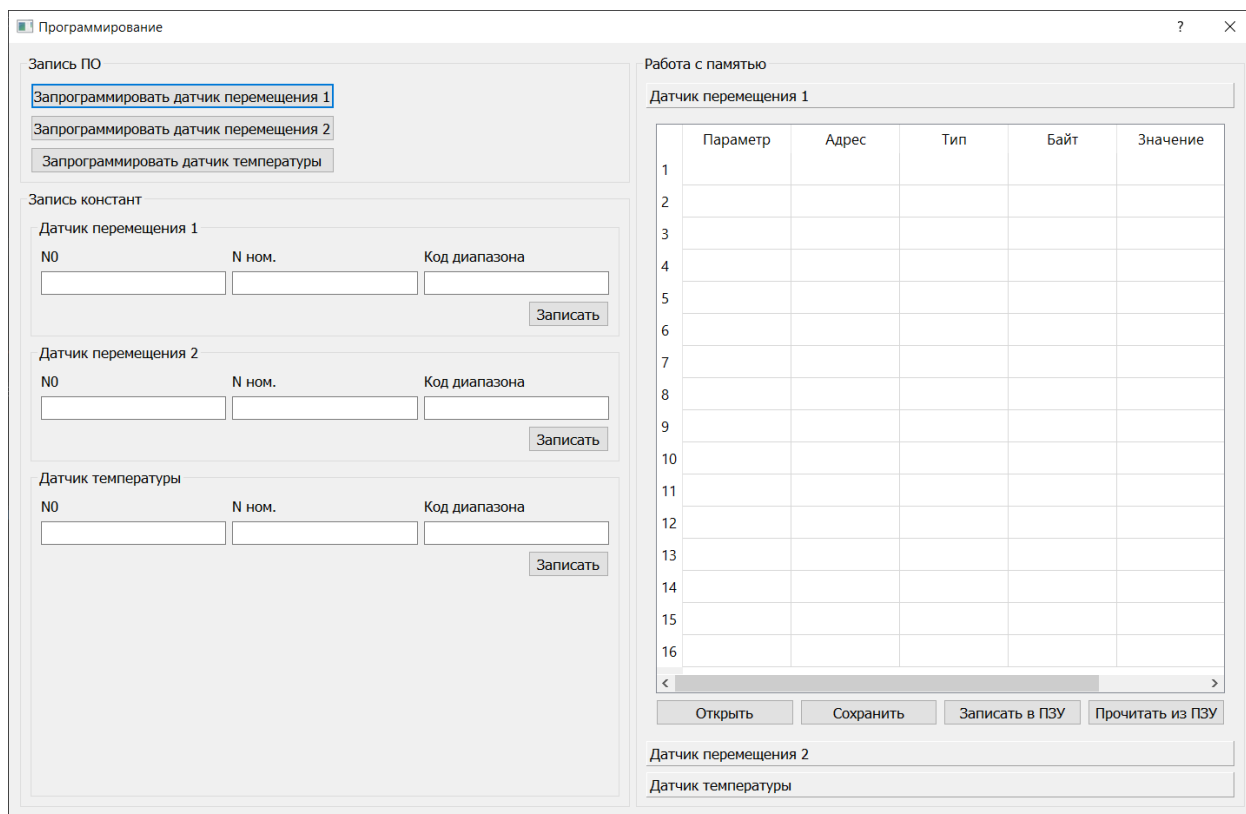


Рисунок 5 – Вкладка программирования

В таблице поля имеют тип данных *string* и как для прошлых окон заблокировано для редактирования настройщиком. Создание таблиц происходит следующим образом:

```

programming_model = new QStandardItemModel(16, 5, this);
programming_model->setHeaderData(0, Qt::Horizontal,
"Параметр");
programming_model->setHeaderData(1, Qt::Horizontal, "Адрес");
programming_model->setHeaderData(2, Qt::Horizontal, "Тип");
programming_model->setHeaderData(3, Qt::Horizontal, "Байт");
programming_model->setHeaderData(4, Qt::Horizontal,
"Значение");

programming_modelDP2 = new QStandardItemModel(16, 5, this);
programming_modelDP2->setHeaderData(0, Qt::Horizontal,
"Параметр");
programming_modelDP2->setHeaderData(1, Qt::Horizontal,
"Адрес");
programming_modelDP2->setHeaderData(2, Qt::Horizontal, "Тип");
programming_modelDP2->setHeaderData(3, Qt::Horizontal, "Байт");
programming_modelDP2->setHeaderData(4, Qt::Horizontal,
"Значение");

programming_modelDT = new QStandardItemModel(15, 5, this);
programming_modelDT->setHeaderData(0, Qt::Horizontal,
"Параметр");

```

```

    programing_modelDT->setHeaderData(1, Qt::Horizontal,
"Адрес");
    programing_modelDT->setHeaderData(2, Qt::Horizontal, "Тип");
    programing_modelDT->setHeaderData(3, Qt::Horizontal,
"Байт");
    programing_modelDT->setHeaderData(4, Qt::Horizontal,
"Значение");

```

programing_model – таблица для датчика перемещения 1,
programing_modelDP2 – таблица для датчика перемещения 2,
programing_modelDT – таблица для датчика температуры.

Так как кнопки для каждой вкладки разные, то при нажатии и срабатывании сигнала *clicked()*, вызывается слот соответствующий выбранному датчику. Настройщик выбирает файл из заранее прописанной директории, для занесения данных в таблицу. Программно установлен фильтр для файлов. Файлы должны быть в формате *XML*.

```

    QString filter = "Файлы .xml |(*xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this,
"Выбрать файл", path, filter);

```

Далее файл читается объектом класса *QXmlStreamReader xmlReader*, и данные из него записываются в таблицу работы с памятью. Запись происходит с помощью метода *setData()*.

```

    programing_model->setData(programing_model->index(row +
counter, column), xmlReader.readElementText());

```

По кнопке записать и прочитать в ПЗУ происходит запись данных в блок. По сигналу *clicked()*, вызывается слот соответствующий выбранному датчику и выполняется программный код слота. Чтобы настройщик видел прогресс записи и считки данных в ПЗУ, был добавлен виджет *QProgressBar* позволяющий отслеживать прогресс записи и считки данных.

Создание виджета можно наблюдать ниже:

```

main_layout.addWidget(&progress_bar);
setLayout(&main_layout);
progress_bar.setGeometry(25, 50, 450, 20);
progress_bar.setMinimum(0);
progress_bar.setMaximum(10);
progress_bar.setValue(0);

```

Для удобной разработки был создан класс *wrpsd* где прописана логика слотов вызывающихся по нажатию кнопок работы с памятью ПЗУ находящихся в виджете *QToolBox*. При вызове слота, он обращается к классу *wrpsd* и выполняет работу.

После завершения записи или считки из ПЗУ появляется окно реализованное с помощью члена класса *QMessageBox information*. Данный класс реализует окно вывода информации.

Запись ВПО в ПЗУ и констант используется тот же виджет *QToolBox* и классы *writeconst* для записи констант в датчики перемещения и температуры, и *programmingBlock*. В этих классах реализуется тот же функционал работы с виджетом и выводом сообщений.

1.6 Тестирование и отладка

1.6.1 Выбор режима тестирования приложения

Существует достаточно много видов тестирования программного продукта. Были рассмотрены следующие виды тестирования.

Функциональное тестирование.

Функциональное тестирование – вид тестирования, направленный на проверку того, какие функции приложения реализованы, и что они работают верным образом. Данный метод тестирования является одним из самых применяемых, так как позволяет наиболее полно проверить программу на соответствие требованиям. «Тестирование в перспективе «требования» использует спецификацию функциональных требований к системе как основу для дизайна тестовых случаев. В этом случае необходимо сделать список того, что будет тестироваться, а что нет, приоритезировать требования на основе рисков, а на основе этого приоритезировать тестовые сценарии. Это позволит сфокусироваться и не упустить при тестировании наиболее важный функционал» - рассмотрено в работе[3].

Нефункциональное тестирование.

					ПГУ 1.090301.12.001 ПЗ	Лист
						26
Изм.	Лист	№ докум.	Подпись	Дата		

Нефункциональное тестирование – вид тестирования, направленный на проверку нефункциональных особенностей приложения, таких как удобство использования, совместимость, производительность, безопасность и т.д.

Интеграционное тестирование.

Интеграционное тестирование – вид тестирования, направленный на проверку взаимодействия между несколькими частями приложения. Этот вид тестирования позволяет проверить корректность взаимодействия частей ПО между собой.

Было принято решение осуществить проверку программы с помощью функционального тестирования. Такой выбор обусловлен тем, что к программе были предъявлены в основном функциональные требования, выполнение которых нужно проверить.

1.6.2 Тестирование приложения

Проверка программы с помощью функционального тестирования отражена в таблице 1, а результаты показаны в приложении В (Результаты тестирования).

Таблица 1 - Описание функциональных тестов

№ Теста	Модуль	Описание	Входные данные	Результат
ПФ.1.1	Настройка адаптера	Указать путь к среде разработки Keil	—	Путь к Keil указан.
ПФ.1.2	Программирование	Запись ВПО в ПЗУ датчика перемещения 1	—	ВПО записано в ПЗУ датчика перемещения 1.
ПФ.1.3		Запись ВПО в ПЗУ датчика перемещения 2	—	ВПО записано в ПЗУ датчика перемещения 2.
ПФ.1.4		Запись ВПО в ПЗУ датчика	—	ВПО записано в ПЗУ датчика температуры.

№ Теста	Модуль	Описание	Входные данные	Результат
		температуры		
ПФ.1.5		Запись констант в ПЗУ датчика перемещения 1	—	Константы записаны в ПЗУ датчика перемещения 1.
ПФ.1.6		Запись констант в ПЗУ датчика перемещения 2	—	Константы записаны в ПЗУ датчика перемещения 2.
ПФ.1.7		Запись констант в ПЗУ датчика температуры	—	Константы записаны в ПЗУ датчика температуры.
ПФ.1.8	Настройка	Приём выходных кодов от датчиков перемещения 1, перемещения 2, температуры	—	Выходные коды приняты.
ПФ.1.9	Настройка линейности датчиков перемещения	Получить выходные коды для настройки линейности датчика перемещения 1	—	Выходные коды для настройки линейности датчика перемещения 1 получены.
ПФ.1.10		Запись выходного кода датчика перемещения 1 в таблицу градуировки		Выходной код датчика перемещения 1 для положения «-18 мм» записан в таблицу градуировки.
ПФ.1.11		Получить выходные коды для настройки линейности		Выходные коды для настройки линейности датчика перемещения 2

№ Теста	Модуль	Описание	Входные данные	Результат
		датчика перемещения 2		получены.
ПФ.1.12		Запись выходного кода датчика перемещения 2 в таблицу градуировки		Выходной код датчика перемещения 2 для положения «-18 мм» записан в таблицу градуировки.
ПФ.1.13		Отмена последней записи выходного кода датчика перемещения 1		Отмена последней записи выходного кода датчика перемещения 1 произведено.
ПФ.1.14		Очищение таблицы градуировки от данных		Очищение таблицы градуировки от данных произведено.
ПФ.1.15		Сохранение выходных данных в файл	Содержимое таблицы градуировки	Данные сохранены в файл Успешно (Рисунок – 26 -27).
ПФ.1.16		Загрузка выходных данных в таблицу градуировки из файла	Содержимое файла	Данные с файла считаны и занесены в таблицу Успешно (Рисунок – 28 -29).
ПФ.1.17	Настройка линейности датчика температуры	Получить выходные коды для настройки линейности датчика	—	Выходные коды для настройки датчика температуры получены.

№ Теста	Модуль	Описание	Входные данные	Результат
		температуры		
ПФ.1.18		Запись выходного кода датчика температуры в таблицу градуировки		Запись выходного кода датчика температуры в таблицу градуировки произведена.
ПФ.1.19		Отмена последней записи выходного кода датчика температуры		Отмена последней записи выходного кода датчика температуры произведена.
ПФ.1.20		Очищение таблицы градуировки от данных		Очищение таблицы градуировки от данных произведено.
ПФ.1.21		Сохранение выходных данных в файл		Сохранение выходных данных произведено.
ПФ.1.22		Загрузка выходных данных в таблицу градуировки из файла		Загрузка выходных данных в таблицу градуировки из файла произведена.
ПФ.1.23	Контроль функционирования	Проверить выходные коды датчика перемещения 1 после настройки	«-18 мм»	Выходные коды датчика соответствуют настроенным.
ПФ.1.24		Проверить выходные коды датчика	«18 мм»	Выходные коды датчика соответствуют

№ Теста	Модуль	Описание	Входные данные	Результат
		перемещения 1 после настройки		настроенным.
ПФ.1.25		Проверить выходные коды датчика перемещения 2 после настройки	«-18мм»	Выходные коды датчика соответствуют настроенным.
ПФ.1.26		Проверить выходные коды датчика перемещения 2 после настройки	«18мм»	Выходные коды датчика соответствуют настроенным.
ПФ.1.27		Проверить выходные коды датчика температуры после настройки	Температура °C 200	Выходные коды датчика соответствуют настроенным..
ПФ.1.28	Отчеты	Создание файла отчета с результатами измерения датчика перемещения 1	Содержимое файла градуировки датчика перемещения 1	Создан файл отчета с результатами измерения датчика перемещения 1.
ПФ.1.29		Создание отчета с результатами измерения датчика перемещения 2	Содержимое файла градуировки датчика перемещения 2	Создан файл отчета с результатами измерения датчика перемещения 2
ПФ.1.30		Создание отчета с результатами измерения датчика температуры	Содержимое файла градуировки датчика температуры	Создан файл отчета с результатами измерения датчика температуры.

Программное обеспечение отлажено с помощью среды разработки *Qt Creator*.

Отладка проводилась с использованием построения пошагового прохождения кода программы, точек останова и просмотра содержимого стеков, переменных и регистров. А так же в процессе разработки программного обеспечения был использован класс *qDebug()*, предназначенный для отладки и позволяющий зафиксировать выходные данные при работе программы.

Отладка и тестирование проводилось в процессе разработки, а так же по завершению написания программы.

В ходе тестирования были выявлены и устранены проблемы касающиеся, синтаксиса программного кода, вызывающие ошибки и способствовали не правильной работе чтения с блока измерения и записи данных в таблицы.

Полный набор вариантов использования программы пользователя продемонстрирован в приложении Б (*UML*-диаграмма вариантов использования)

					ПГУ 1.090301.12.001 ПЗ	Лист
						32
Изм.	Лист	№ докум.	Подпись	Дата		

2 Безопасность жизнедеятельности

Прогрессирующее научно-техническое развитие современного общества создает задачу поддержания безопасности исполнения людьми своих трудовых обязанностей. Осуществлением данной задачи занимается отдельная наука о безопасности труда и жизнедеятельности человека.

Безопасность жизнедеятельности (БЖД) - это комплекс мероприятий, направленных на обеспечение безопасности человека в среде обитания, сохранение его здоровья, разработку методов и средств защиты путем снижения влияния вредных и опасных факторов до допустимых значений, выработку мер по ограничению ущерба в ликвидации последствий чрезвычайных ситуаций мирного и военного времени.

«Охрана здоровья трудящихся, обеспечение безопасности условий труда, ликвидация профессиональных заболеваний и производственного травматизма составляет одну из главных забот человеческого общества. Обращается внимание на необходимость широкого применения прогрессивных форм научной организации труда, сведения к минимуму ручного, малоквалифицированного труда, создания обстановки, исключаяющей профессиональные заболевания и производственный травматизм» - рассмотрено в источнике [4].

2.1 Потенциально-опасные и вредные факторы

При эксплуатации ПО необходимо обращать внимание на воздействие вредных факторов на сотрудников. Вредный фактор рабочей среды - производственный фактор, воздействие которого на работника в определенных условиях может привести к заболеванию или отрицательному влиянию на здоровье потомства. Вредные и опасные факторы делятся на четыре категории: физические факторы, химические факторы, биологические факторы и факторы тяжести трудового процесса.

К физическим факторам относят микроклимат помещения, неионизирующие электромагнитные поля и излучения, электростатическое и

					ПГУ 1.090301.12.001 ПЗ	Лист
						33
Изм.	Лист	№ докум.	Подпись	Дата		

постоянное магнитное, геомагнитное поля, источники ионизирующего излучения, производственный шум, ультразвук, инфразвук, вибрацию, аэрозоли преимущественно фиброгенного действия – пыли, световую среду, аэроионный состав воздуха.

В работе [4] описано «химические факторы – это химические вещества, смеси, получаемые химическим синтезом и/или для контроля которых используют методы химического анализа».

К биологическим факторам относятся микроорганизмы-продуценты, живые клетки и споры, патогенные микроорганизмы.

«Факторы тяжести трудового процесса оценивают по тяжести труда. Тяжесть труда - характеристика трудового процесса, отражающая преимущественную нагрузку на опорно-двигательный аппарат и функциональные системы организма (сердечно-сосудистую, дыхательную и др.), обеспечивающие его деятельность» - описывает [4] источник.

2.2 Пожарная безопасность

Требования по присвоению категории пожарной безопасности касаются помещений, которые имеют производственное, техническое или складское значение.

Категорирование производится на основе следующих факторов:

- наличие горючих веществ и материалов и их количество;
- проектирование помещения и проводимые в них технологические процессы.

Категории, установленные для помещений рассмотрены в таблице 2.

Офис, который рассчитан на использование разработанного приложения, относится к низшей категории по взрывопожарной и пожарной безопасности (Д), поскольку в помещении находятся негорючие вещества и материалы.

Таблица 2 – Категорирование помещений по взрывопожарной и пожарной опасности

					ПГУ 1.090301.12.001 ПЗ	Лист
						34
Изм.	Лист	№ докум.	Подпись	Дата		

Категория помещения	Характеристики веществ и материалов, находящихся (обращающихся) в помещении
А (повышенная взрывопожароопасность)	Горючий газ, легковоспламеняющиеся жидкости с температурой вспышки не более 28 градусов по Цельсию
Б (взрывопожароопасность)	Горючие пыли или волокна, легковоспламеняющиеся жидкости с температурой вспышки более 28 градусов по Цельсию
В1-В4 (пожароопасность)	Горючие и трудногорючие жидкости, твердые горючие и трудногорючие вещества и материалы
Г (умеренная пожароопасность)	Негорючие вещества и материалы в горячем, раскаленном или расплавленном состоянии
Д (пониженная пожароопасность)	Негорючие вещества и материалы в холодном состоянии

Наиболее частые причины пожаров – нарушение правил пожарной безопасности или технологических процессов, неправильная эксплуатация оборудования. Как на предприятии в целом, так и в отдельных помещениях необходимо проводить противопожарную профилактику.

Мероприятия по пожарной безопасности делятся на:

- технические (соблюдение противопожарных правил, норм при проектировании зданий, при устройстве и размещении оборудования);
- мероприятия режимного характера (запрещение курения в неустановленных местах, производства огневых работ в пожароопасных помещениях);
- эксплуатационные (своевременные профилактические осмотры, ремонты, испытания технологического оборудования).

Также необходимо обеспечить предприятие средствами тушения пожаров и сигнализации.

«Основные огнегасительные вещества – вода, пена, песок, инертные газы, сухие (твердые) огнегасительные вещества и т.д. Так на основе данных веществ используются следующие первичные средства пожаротушения: переносимые и передвижные огнетушители, пожарные краны и средства обеспечения их использования, пожарный инвентарь (лопаты, молотки, конусы и т.д.), одеяла для изоляции очага возгорания» - изложено в работе [4].

Территория предприятия с офисом, рассчитанная под эксплуатацию разработанного ПО, окружена металлическим забором, въезд на территорию осуществляется через центральные ворота, что важно для эвакуации при пожаре. На каждом этаже здания имеются планы эвакуации. В помещении присутствуют средства пожаротушения, установлен извещатель пожарный ручной. Применяются углеводородные огнетушители.

Таким образом, на предприятии соблюдаются противопожарные правила и нормы, имеется свободный доступ к средствам пожаротушения и запасному выходу. Расположение рабочих мест оптимально.

2.3 Электробезопасность

Помещения, опасные с точки зрения поражения людей током подразделяются на:

- особо опасные (особо сырые помещения, с химически активной или органической средой);
- с повышенной опасностью (помещения с наличием токопроводящей пыли, с высокой температурой, помещения, имеющие токопроводящие полы);
- без повышенной опасности (в помещении отсутствуют условия, создающие повышенную или особую опасность поражения людей электрическим током).

					ПГУ 1.090301.12.001 ПЗ	Лист
						36
Изм.	Лист	№ докум.	Подпись	Дата		

По опасности поражения электротоком данное помещение относится к помещениям без повышенной опасности, поскольку текущее помещение характеризуется отсутствием условий, создающих повышенную или особую опасность. К данным условиям относятся: сырость и токопроводящая пыль, токопроводящие полы, высокие температуры, химически активная или органическая среды, а также возможность одновременного прикосновения человека к металлоконструкциям зданий, имеющим соединение с землей, технологическими аппаратами, механизмам и т.п. – с одной стороны, и к металлическим корпусам электрооборудования – с другой.

С точки зрения эксплуатации электрооборудования помещения, считающиеся безопасными, должны соответствовать следующим критериям:

- температура в помещении ниже 30 градусов Цельсия;
- отсутствие токопроводящей пыли;
- влажность помещения не более 60 процентов;
- помещение хорошо проветривается;
- полы выполнены из непроводящего ток материала;
- заполнение площади помещения предметами из металла менее 20 процентов.

В свою очередь, персонал, эксплуатирующие электропроводящее оборудование, должен знать правила и требования технической эксплуатации, а также правила оказания первой помощи пострадавшему от действия электрического тока.

В число системы электробезопасности текущего предприятия входят: защита от прямого и косвенного прикосновения к токоведущим и токопроводящим частям оборудования, применение защитного заземления, использование защитного автоматического отключения питания. Работники проходят обучение безопасным методам и приемам выполнения работ с токопроводящим оборудованием, а также проходят проверку знаний и требований, предъявляемых к организации и выполнению работ.

2.4 Определение класса условий труда

«На здоровье и работоспособность человека в процессе труда оказывает влияние совокупность факторов производственной (рабочей) среды и трудового процесса. Гигиенические критерии оценки условий труда позволяют оценить степень отклонений параметров производственной среды и трудового процесса от действующих гигиенических нормативов» - описано в [4] работе.

В частности, при организации рабочего места программиста должны быть соблюдены следующие основные условия:

- оптимальное размещение оборудования, входящего в состав рабочего места;
- достаточное рабочее пространство, позволяющее осуществлять все необходимые движения и перемещения;
- необходимо естественное и искусственное освещение для выполнения поставленных задач;
- уровень акустического шума не должен превышать допустимого значения.

Напряженность труда показывает степень воздействия нагрузок, связанных с рабочей деятельностью, на эмоциональные процессы, центральную нервную систему и органы чувств. Уровень напряженности оценивают по интеллектуальным, сенсорным и эмоциональным нагрузкам, а также их монотонности и режиму работы.

В ходе анализа трудовой деятельности учитываются следующие критерии: нагрузки интеллектуального характера, сенсорные нагрузки, эмоциональные нагрузки, режим работы.

По данным параметрам оценивается общая напряженность трудового процесса.

Для работника, эксплуатирующего текущее ПО, представленные нагрузки интеллектуального характера были отнесены к следующим классам:

- «Содержание работы» - 2 класс;

					ПГУ 1.090301.12.001 ПЗ	Лист
						38
Изм.	Лист	№ докум.	Подпись	Дата		

- «Восприятие сигналов (информации) и их оценка – 2 класс;
- «Распределение функций по степени сложности задания» – 2 класс;
- «Характер выполняемой работы» – 2 класс.

Сенсорные нагрузки:

- «Длительность сосредоточенного наблюдения (в процентах от времени смены)» - 1 класс (15%);
- «Плотность сигналов (световых, звуковых) и сообщений в среднем за 1 час работы» - 1 класс (1-3 сигнала);
- «Число производственных объектов одновременного наблюдения» - 1 класс (1-2 объекта);
- «Размер объекта различения при длительности сосредоточенного внимания (% от времени) – 1 класс (0 объектов);
- «Работа с оптическими приборами (микроскоп, лупа и т.п.) при длительности сосредоточенного наблюдения (% от времени)» - 1 класс;
- «Наблюдение за экраном видеотерминала (ч в смену)» - 1 класс;
- «Нагрузка на слуховой анализатор» - 1 класс;
- «Нагрузка на голосовой аппарат (суммарное количество часов, наговариваемых в неделю)» - 1 класс.

Эмоциональные нагрузки:

- «Степень ответственности за результат собственной деятельности. Значимость ошибки» - 2 класс;
- «Степень риска для собственной жизни» - 1 класс;
- «Ответственность за безопасность других лиц» - 1 класс;
- «Количество конфликтных производственных ситуаций за смену» - 1 класс.

Монотонность нагрузок:

- «Число элементов (приемов), необходимых для реализации простого задания или многократно повторяющихся операций» - 2 класс;
- «Продолжительность выполнения простых производственных заданий или повторяющихся операций» - 2 класс;

- «Время активных действий (в процентах к продолжительности смены)» - 2 класс (от 95%).;

- «Монотонность производственной обстановки (время пассивного наблюдения за ходом техпроцесса, в процентах от времени смены)» - 1 класс (до 5%).

Режим работы:

- «Фактическая продолжительность рабочего дня» - 1 класс;

- «Сменность работы» - 1 класс;

- «Наличие регламентированных перерывов и их продолжительность (без учета обеденного перерыва)» - 3.1 класс.

Итак, по результатам данных показателей можно дать общую оценку напряженности трудового процесса. 7 показателей из 23 отнесены ко 2 классу, 1 показатель – к классу 3.1, остальные 25 – к 1 классу. Значит, общий трудовой процесс является «Допустимым» (2 класс).

Полученные показатели отражают степень опасности условий труда на организм. На основании рассмотренных характеристик разрабатывается комплекс мероприятий, направленных на повышение эффективности деятельности в рамках предприятия, а также на минимизацию вредного воздействия на здоровье человека.

Таким образом, благоприятные условия труда имеют большое значение как для облегчения труда, так и для повышения положительного влияния на производительность труда.

Заключение

В рамках данной выпускной квалификационной работы было разработано программное обеспечение для настройки блока измерения подачи ракетного топлива. Разработанное ПО предоставляет возможность записывать внутреннее программное обеспечение в микроконтроллер каждого вторичного преобразователя (для 2 датчиков перемещения и 1 датчика температуры), получать данные с блока вторичных преобразователей, настроить вторичные преобразователи внутри блока, проводить калибровку, сохранять данные калибровки в файл с расширением *XML*, считывать параметры из ПЗУ, записывать параметры в ПЗУ, формировать отчёты по настройке датчиков перемещения и температуры. Так же приобретены навыки работы с *XML* и *EXCEL* файлами и создания графического интерфейса.

В процессе разработки был спроектирован и реализован графический интерфейс при помощи технологии *Qt Widgets*.

В ходе работы были использованы язык программирования *C++*, среда разработки *Qt Creator*.

После тестирования данного программного обеспечения было установлено, что ПО полностью готово к дальнейшей работе. Программа выполнена в соответствии с техническим заданием и в установленные сроки.

					ПГУ 1.090301.12.001 ПЗ	Лист
						41
Изм.	Лист	№ докум.	Подпись	Дата		

Список использованных источников

1. Шлее, М. Qt 5.10. Профессиональное программирование на C++ / М. Шлее. – М.: БХВ-Петербург, 2018. – 1072 с.
2. Эрик, Р. Изучаем XML / Р. Эрик. – М.: Символ-Плюс. 2001. – 402с.
3. Научная электронная библиотека [Электронный ресурс] / Режим доступа <https://www.elibrary.ru> свободный (дата обращения 27.04.2022).
4. Фролов, А. В. Безопасности жизнедеятельности. Охрана труда: Учебное пособие для вузов / А. В. Фролов. – Ростов н/Дю: Феникс, 2008. — 750 с.

					ПГУ 1.090301.12.001 ПЗ	Лист
						42
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А

(Обязательное)

Программное обеспечение для настройки блока измерения подачи ракетного

топлива

Листинг программы

					ПГУ 1.090301.12.001 ПЗ	Лист
						43
Изм.	Лист	№ докум.	Подпись	Дата		

Файл «mainwindow.cpp»

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "settingsadapter.h"

MainWindow::MainWindow(QWidget *parent) : QMainWindow(parent), ui(new
Ui::MainWindow)
{
    ui->setupUi(this);
    const QString title = "ПО настройки системы подачи топлива";
    setWindowTitle(title);
    QWidget::setFixedSize(350,240);

    model = new QStandardItemModel(13, 4);
    model->setHeaderData(0, Qt::Horizontal, QObject::tr("Параметр"));
    model->setHeaderData(1, Qt::Horizontal, QObject::tr("Адрес"));
    model->setHeaderData(2, Qt::Horizontal, QObject::tr("Тип"));
    model->setHeaderData(3, Qt::Horizontal, QObject::tr("Байт"));
    model->setHeaderData(4, Qt::Horizontal, QObject::tr("Значение"));
    model->setHeaderData(5, Qt::Horizontal, QObject::tr("Представление в
Hex"));
    model->setHeaderData(6, Qt::Horizontal, QObject::tr("Контрольная
сумма"));

    //readSetting("PZUDPl.ini");
}

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

void MainWindow::on_pushButton_clicked()
{
    settingsadapter = new settingsAdapter(this);
    settingsadapter -> setModal(true);
    settingsadapter-> show();
}

void MainWindow::on_pushButton_2_clicked()
{
    setting = new Setting(this);
    setting -> setModal(true);
    setting -> show();
}

void MainWindow::on_pushButton_4_clicked()
{
    functionControl = new functioncontrol(this);
    functionControl -> setModal(true);
    functionControl -> show();
}

void MainWindow::on_pushButton_3_clicked()
{
    psi = new PSI(this);
    psi->setModal(true);
    psi -> show();
}
```

					ПГУ 1.090301.12.001 ПЗ	Лист
						44
Изм.	Лист	№ докум.	Подпись	Дата		

```
void MainWindow::on_pushButton_5_clicked()
{
    programmings = new programming(this);
    programmings->setModal(true);
    programmings->show();
}
```

Файл «linearitysetting.cpp»

```
#include "linearitysetting.h"
#include "ui_linearitysetting.h"
```

```
linearitysetting::linearitysetting(QWidget *parent) : QDialog(parent), ui(new
Ui::linearitysetting)
{
    ui->setupUi(this);

    QWidget::setFixedSize(960, 530);
    model = new QStandardItemModel(13, 4, this);
    model->setHeaderData(0, Qt::Horizontal, "Прямой ход");
    model->setHeaderData(1, Qt::Horizontal, "Обратный ход");
    model->setHeaderData(2, Qt::Horizontal, "Прямой ход");
    model->setHeaderData(3, Qt::Horizontal, "Обратный ход");

    model->setHeaderData(0, Qt::Vertical, "-18мм");
    model->setHeaderData(1, Qt::Vertical, "-15мм");
    model->setHeaderData(2, Qt::Vertical, "-12мм");
    model->setHeaderData(3, Qt::Vertical, "-9мм");
    model->setHeaderData(4, Qt::Vertical, "-6мм");
    model->setHeaderData(5, Qt::Vertical, "-3мм");
    model->setHeaderData(6, Qt::Vertical, "0мм");
    model->setHeaderData(7, Qt::Vertical, "3мм");
    model->setHeaderData(8, Qt::Vertical, "6мм");
    model->setHeaderData(9, Qt::Vertical, "9мм");
    model->setHeaderData(10, Qt::Vertical, "12мм");
    model->setHeaderData(11, Qt::Vertical, "15мм");
    model->setHeaderData(12, Qt::Vertical, "18мм");

    ui->tableView->setModel(model);
    ui->btn_stop->setEnabled(false);
    ui->btn_write->setEnabled(false);
    ui->btn_cancel->setEnabled(false);
    ui->btn_clear->setEnabled(false);
    ui->btn_save->setEnabled(false);
    ui->btn_save->setEnabled(false);

    ui->lineEdit_liner->setEnabled(false);
    ui->lineEdit_liner2->setEnabled(false);
    ui->lineEdit_liner3->setEnabled(false);
    ui->lineEdit_liner4->setEnabled(false);

    connect(ui->btn_start, &QPushButton::clicked, this,
&linearitysetting::disableStop);
    connect(ui->btn_stop, &QPushButton::clicked, this,
&linearitysetting::disableStart);

    timerDP1 = new QTimer(this);
    timerDP2 = new QTimer(this);
    connect(timerDP1, &QTimer::timeout, this,
&linearitysetting::generateValueDP1);
    connect(timerDP2, &QTimer::timeout, this,
&linearitysetting::generateValueDP2);
```

					ПГУ 1.090301.12.001 ПЗ	Лист
						45
Изм.	Лист	№ докум.	Подпись	Дата		

```

        isDP = false;
        ui->tableView->setEditTriggers(QTableView::NoEditTriggers);
    }

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

    void linearitysetting::disableStop()
    {
        if (isDP)
            timerDP2->start(1000);
        else
            timerDP1->start(1000);

        ui->btn_start->setEnabled(false);
        ui->btn_stop->setEnabled(true);
        ui->btn_write->setEnabled(true);
    }

    void linearitysetting::disableStart()
    {
        if (isDP)
            timerDP2->stop();
        else
            timerDP1->stop();

        ui->btn_start->setEnabled(true);
        ui->btn_stop->setEnabled(false);
    }

    void linearitysetting::on_comboBox_currentIndexChanged(int index)
    {
        switch (index)
        {
            case 0: isDP = false; break;
            case 1: isDP = true; break;
        }
    }

    void linearitysetting::generateValueDP1()
    {
        unsigned int number = static_cast<unsigned int>(rand() %
differenceValueLi + 1);
        unsigned int result = minimalValueLi + number;
        ui->lineEdit_liner->setText(QString::number(result));

        unsigned int number2 = static_cast<unsigned int>(rand() %
differenceValueLi2 + 1);
        unsigned int result2 = minimalValueLi2 + number2;
        ui->lineEdit_liner2->setText(QString::number(result2));

        ui->lineEdit_liner3->setText(QString::number(result2-result));

        unsigned short int moveCode1 = rand() % 300;
        ui->lineEdit_liner4->setText(QString::number(moveCode1));
    }

```

```

void linearitysetting::generateValueDP2()
{
    unsigned int number3 = static_cast<unsigned int>(rand() %
differenceValueLi3 + 1);
    unsigned int result3 = minimalValueLi3 + number3;
    ui->lineEdit_liner->setText(QString::number(result3));

    unsigned int number4 = static_cast<unsigned int>(rand() %
differenceValueLi4 + 1);
    unsigned int result4 = minimalValueLi4 + number4;
    ui->lineEdit_liner2->setText(QString::number(result4));

    ui->lineEdit_liner3->setText(QString::number(result4-result3));

    unsigned int number6 = static_cast<unsigned int>(rand() %
differenceSpace2 + 1);
    unsigned int result6 = minSpace3 + number6;
    ui->lineEdit_liner4->setText(QString::number(result6));
}

void linearitysetting::on_btn_write_clicked()
{
    ui->btn_save->setEnabled(true);
    ui->btn_cancel->setEnabled(true);
    ui->btn_clear->setEnabled(true);

    QString x = ui->lineEdit_liner4->text();
    model->setData(model->index(curRow, curCol), x);

    bool isColEven = (curCol % 2) == 0;
    curRow = isColEven ? curRow + 1 : curRow - 1;
    if(curRow == model->rowCount() || curRow == -1)
    {
        curCol++;
        isColEven = (curCol % 2) == 0;
        curRow = isColEven ? 0 : model->rowCount() - 1;
    }
    countClickLin++;
    if(countClickLin==52)
        ui->btn_write->setEnabled(false);
}

void linearitysetting::on_btn_cancel_clicked()
{
    bool isColEven = (curCol % 2) == 0;
    curRow = isColEven ? curRow - 1 : curRow + 1;
    if(curRow == model->rowCount() || curRow == -1)
    {
        curCol--;
        isColEven = (curCol % 2) == 0;
        curRow = isColEven ? model->rowCount() - 1 : 0;
    }
    ui->btn_write->setEnabled(true);
    countClickLin--;
    model->clearItemData(model->index(curRow, curCol));

    if (countClickLin==0){
        ui->btn_cancel->setEnabled(false);
        ui->btn_clear->setEnabled(false);
    }
}

```

```

void linearitysetting::on_btn_exit_clicked()
{
    close();
}

void linearitysetting::on_btn_clear_clicked()
{
    for (int row = 0; row<model->rowCount(); ++row){

        for(int col = 0; col<model->columnCount(); ++col){
            QStandardItem *item = new QStandardItem();
            model->setItem(row,col, item);
            curCol=0;
            curRow=0;
        }

        model->index(curRow,curCol);
        countClickLin=0;
        if(countClickLin == 0)
            ui->btn_write->setEnabled(true);
        ui->btn_cancel->setEnabled(false);
        ui->btn_clear->setEnabled(false);
    }

void linearitysetting::on_btn_save_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
    "\\savedDataXML");

    bool isExists = dir.exists(path);
    if (!isExists)
        dir.mkdir(path);

    QString savePath = QDir::toNativeSeparators(path +
    "\\LinearitySetting.xml");
    QString selectedPath = QFileDialog::getSaveFileName(this, "Сохранить
файл", savePath);
    bool isSelected = selectedPath.isEmpty();
    if (isSelected)
        return;

    QFile file;
    file.setFileName(selectedPath);
    file.open(QIODevice::WriteOnly);

    bool isOpen = file.isOpen();
    if (!isOpen)
    {
        QMessageBox message;
        message.setText("Файл по адресу " + selectedPath + "не удалось
открыть!");
        message.show();
        return;
    }

    QXmlStreamWriter xmlWriter(&file);
    xmlWriter.setAutoFormatting(true);
    xmlWriter.writeStartDocument();

    QString dateTime = QDateTime::currentDateTime().toString("dd.MM.yyyy
hh.mm.ss");
    QString factoryNumber = ui->factory_numberLi->text();
    const ushort rows = static_cast<ushort>(model->rowCount());
    const ushort columns = static_cast<ushort>(model->columnCount());

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						48
Изм.	Лист	№ докум.	Подпись	Дата		

```

QString prepareData[rows][columns];
for (ushort row = 0; row < rows; ++row)
    for (ushort column = 0; column < columns; ++column)
        prepareData[row][column] = model->data(model->index(row,
column)).toString();

xmlWriter.writeStartElement("data");
xmlWriter.writeStartElement("date");
xmlWriter.writeCharacters(dateTime);
xmlWriter.writeEndElement();

xmlWriter.writeStartElement("number");
xmlWriter.writeCharacters(factoryNumber);
xmlWriter.writeEndElement();

xmlWriter.writeStartElement("table");

for (ushort row = 0; row < rows; ++row)
    for (ushort column = 0; column < columns; ++column)
    {
        xmlWriter.writeStartElement("row");
        xmlWriter.writeCharacters(prepareData[row][column]);
        xmlWriter.writeEndElement();
    }

xmlWriter.writeEndElement();
xmlWriter.writeEndElement();
xmlWriter.writeEndDocument();

file.close();
}

void linearitysetting::on_btn_open_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\savedDataXML");
    QString filter = "*.xml (*xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);

    bool isEmpty = selectedPath.isEmpty();
    if (isEmpty)
        return;

    QFile file;
    file.setFileName(selectedPath);

    file.open(QIODevice::ReadOnly | QIODevice::Text);
    bool isOpen = file.isOpen();
    if (!isOpen)
    {
        QMessageBox messageBox;
        messageBox.setWindowTitle("Предупреждение");
        messageBox.setText("При попытке открыть файл произошла ошибка!");
        messageBox.show();
    }

    QXmlStreamReader xmlReader(&file);
    int row = 0;
    int col = 0;

    while (!xmlReader.atEnd() && !xmlReader.hasError())
    {
        xmlReader.readNext();

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						49
Изм.	Лист	№ докум.	Подпись	Дата		

```

        if (xmlReader.isStartElement() && xmlReader.name() == "row")
        {
            if(row < model->rowCount() && col < model->columnCount())
                model->setData(model->
>index(row,col),xmlReader.readElementText());

            col++;
            if(col >= model->columnCount()){
                col = 0;
                row++;
            }
        }
    }

    file.close();
}

```

Файл «functioncontrol.cpp»

```

#include "functioncontrol.h"
#include "ui_functioncontrol.h"
#include "constansts.h"

functioncontrol::functioncontrol(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::functioncontrol)
{
    ui->setupUi(this);
    QWidget::setWindowTitle("Контроль функционирования");
    QWidget::setFixedSize(1020, 367);

    connect(ui->btn_start_func, &QPushButton::clicked, this,
&functioncontrol::disableStop);
    connect(ui->btn_stop_func, &QPushButton::clicked, this,
&functioncontrol::disableStart);

    ui->lineEdit_voltageFun1->setEnabled(false);
    ui->lineEdit_voltageFun2->setEnabled(false);
    ui->lineEdit_voltageFun3->setEnabled(false);
    ui->lineEdit_voltageFun4->setEnabled(false);
    ui->lineEdit_codeADCFun->setEnabled(false);
    ui->lineEdit_spaceledFun1->setEnabled(false);
    ui->lineEdit_spaceledFun2->setEnabled(false);
    ui->lineEdit_spacemmFun1->setEnabled(false);
    ui->lineEdit_spacemmFun2->setEnabled(false);
    ui->lineEdit_voltageVFun1->setEnabled(false);
    ui->lineEdit_voltageVFun2->setEnabled(false);
    ui->lineEdit_TCFun->setEnabled(false);
    ui->lineEdit_codTFun->setEnabled(false);
    ui->lineEdit_gostTFun->setEnabled(false);

    timerfunc = new QTimer(this);
    connect (timerfunc, &QTimer::timeout, this,
&functioncontrol::onTimeoutSetting);
}

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

void functioncontrol::disableStart()

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дата		

```

{
    timerfunc ->stop();
    ui->btn_start_func->setEnabled(true);
    ui->btn_stop_func->setEnabled(false);
}

void functioncontrol::disableStop()
{
    timerfunc ->start(500);
    ui->btn_start_func->setEnabled(false);
    ui->btn_stop_func->setEnabled(true);
}

void functioncontrol::onTimeoutSetting()
{
    unsigned int number = static_cast<unsigned int>(rand() %
differenceValueLi + 1);
    unsigned int result = minimalValueLi + number;
    ui->lineEdit_voltageFun1->setText(QString::number(result));

    unsigned int number2 = static_cast<unsigned int>(rand() %
differenceValueLi2 + 1);
    unsigned int result2 = minimalValueLi2 + number2;
    ui->lineEdit_voltageFun2->setText(QString::number(result2));

    unsigned int number3 = static_cast<unsigned int>(rand() %
differenceValueLi3 + 1);
    unsigned int result3 = minimalValueLi3 + number3;
    ui->lineEdit_voltageFun3->setText(QString::number(result3));

    unsigned int number4 = static_cast<unsigned int>(rand() %
differenceValueLi4 + 1);
    unsigned int result4 = minimalValueLi4 + number4;
    ui->lineEdit_voltageFun4->setText(QString::number(result4));

    //Код АЦП
    unsigned int number5 = static_cast<unsigned int>(rand() %
differenceValueCodeAdc + 1);
    unsigned int result5 = minCodeAdc + number5;
    ui->lineEdit_codeADCFun->setText(QString::number(result5));

    unsigned int number6 = static_cast<unsigned int>(rand() %
differenceSpace2 + 1);
    unsigned int result6 = minSpace3 + number6;
    ui->lineEdit_spaceledFun2->setText(QString::number(result6));

    double mmspace = static_cast<double>(result6 / 10.0);
    ui->lineEdit_spacemmFun2->setText(QString::number(mmspace, 'f', 1));

    unsigned int number7 = static_cast<unsigned int>(rand() % differenceCodeT
+ 1);
    unsigned int result7 = minCodeT + number7;
    ui->lineEdit_codTFun->setText(QString::number(result7));

    unsigned short int moveCode1 = rand() % 300;
    double moveMM1 = static_cast<double>(moveCode1 / 10.0);
    ui->lineEdit_spacemmFun1->setText(QString::number(moveMM1, 'f', 1));

    ui->lineEdit_spaceledFun1->setText(QString::number(moveCode1));
    ui->lineEdit_voltageVFun1->setText(QString::number(result2-result));

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						51
Изм.	Лист	№ докум.	Подпись	Дата		

```

ui->lineEdit_voltageVFun2->setText(QString::number(result4-result3));

unsigned short int tempGOST = rand() % 1800 + 1;
double tempGOSTD = (1500 + tempGOST) / 100.0;
ui->lineEdit_gostTFun->setText(QString::number(tempGOSTD, 'f', 2));

double tempC = tempGOSTD + 0.5;
ui->lineEdit_TCFun->setText(QString::number(tempC, 'f', 2));

}

```

Файл «programming.cpp»

```

#include "programming.h"
#include "ui_programming.h"
#include "programmingBlock.h"

programming::programming(QWidget *parent) : QDialog(parent), ui(new
Ui::programming)
{
    ui->setupUi(this);

    programing_model = new QStandardItemModel(16, 5, this);
    programing_model->setHeaderData(0, Qt::Horizontal, "Параметр");
    programing_model->setHeaderData(1, Qt::Horizontal, "Адрес");
    programing_model->setHeaderData(2, Qt::Horizontal, "Тип");
    programing_model->setHeaderData(3, Qt::Horizontal, "Байт");
    programing_model->setHeaderData(4, Qt::Horizontal, "Значение");

    programing_modelDP2 = new QStandardItemModel(16, 5, this);
    programing_modelDP2->setHeaderData(0, Qt::Horizontal, "Параметр");
    programing_modelDP2->setHeaderData(1, Qt::Horizontal, "Адрес");
    programing_modelDP2->setHeaderData(2, Qt::Horizontal, "Тип");
    programing_modelDP2->setHeaderData(3, Qt::Horizontal, "Байт");
    programing_modelDP2->setHeaderData(4, Qt::Horizontal, "Значение");

    programing_modelDT = new QStandardItemModel(15, 5, this);
    programing_modelDT->setHeaderData(0, Qt::Horizontal, "Параметр");
    programing_modelDT->setHeaderData(1, Qt::Horizontal, "Адрес");
    programing_modelDT->setHeaderData(2, Qt::Horizontal, "Тип");
    programing_modelDT->setHeaderData(3, Qt::Horizontal, "Байт");
    programing_modelDT->setHeaderData(4, Qt::Horizontal, "Значение");

    readSetting(programing_model, "PZUDP1.ini");
    readSetting(programing_modelDP2, "PZUDP2.ini");
    readSetting(programing_modelDT, "PZUDT.ini");

    ui->tableView1->setModel(programing_model);
    ui->tableView1_2->setModel(programing_modelDP2);
    ui->tableView1_3->setModel(programing_modelDT);
    ui->tableView1->setEditTriggers(QTableView::NoEditTriggers);
    ui->tableView1_2->setEditTriggers(QTableView::NoEditTriggers);
    ui->tableView1_3->setEditTriggers(QTableView::NoEditTriggers);

    ui->toolBox->setCurrentIndex(0);
}

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

void programming::readSetting(QStandardItemModel* model, QString PZUFileName)
{

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						52
Изм.	Лист	№ докум.	Подпись	Дата		

```

QFile *filePZU = new QFile();
QString fileName = QDir::toNativeSeparators(QDir::currentPath() + "/" +
PZUFileName);
filePZU->setFileName(fileName);

if (!QFile::exists(fileName))
    return;

for (ushort row = 0; row < 16; ++row)
    for (ushort column = 0; column < 5; ++column)
        cardROMDevice[row][column] = "";

// Заполнение таблицы констант ПЗУ.
if (filePZU->open(QIODevice::ReadOnly))
{
    QTextStream stream(filePZU);
    QString line;
    unsigned short int index = 0;

    while (!stream.atEnd())
    {
        line = stream.readLine();
        cardROMDevice[index][0] = line.mid(0, line.indexOf('\t'));
// Адрес.
        line.remove(1, line.indexOf('\t'));
        cardROMDevice[index][1] = line.mid(1, line.indexOf('\t') - 1);
// Количество байт полезной информации.
        line.remove(1, line.indexOf('\t'));
        cardROMDevice[index][2] = line.mid(1, line.indexOf('\t') - 1);
// Тип данных.
        line.remove(1, line.indexOf('\t'));
        cardROMDevice[index][3] = line.mid(1, line.indexOf('\t') - 1);
// Название параметра, а в data[4][i] хранится значение.
        line.remove(1, line.indexOf('\t'));
        ++index;
    }

    if (stream.status() != QTextStream::Ok)
        return;

    filePZU->close();
    delete filePZU;
    //return;
}
else
    return;

for (unsigned short int index = 0; index < 16; ++index)
{
    // Столбец "Параметр".
    QStandardItem *item = new QStandardItem(cardROMDevice[index][0]);
    model->setItem(index, 0, item);

    // Столбец "Адрес".
    item = new QStandardItem(cardROMDevice[index][1]);
    model->setItem(index, 1, item);

    // Столбец "Тип".
    item = new QStandardItem(cardROMDevice[index][2]);
    model->setItem(index, 2, item);

    // Столбец "Байт".
    item = new QStandardItem(cardROMDevice[index][3]);
    model->setItem(index, 3, item);
}

```

```

    }

}

//ДТ
void programming::on_pushButton_17_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\Templates");
    QString filter = "Файлы .xml |(*.xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);

    bool isEmpty = selectedPath.isEmpty();
    if (isEmpty)
        return;

    QFile file;
    file.setFileName(selectedPath);

    file.open(QFile::ReadOnly | QFile::Text);
    bool isOpen = file.isOpen();
    if (!isOpen)
    {
        QMessageBox messageBox;
        messageBox.setWindowTitle("Предупреждение");
        messageBox.setText("При попытке открыть файл произошла ошибка!");
        messageBox.show();
    }

    programing_modelDT->setData(programing_modelDT->index(6, 4), "100");
    programing_modelDT->setData(programing_modelDT->index(7, 4), "1100");
    programing_modelDT->setData(programing_modelDT->index(8, 4), "12");

    file.close();
}

//ДП1
void programming::on_pushButton_9_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\Templates");
    QString filter = "Файлы .xml |(*.xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);

    bool isEmpty = selectedPath.isEmpty();
    if (isEmpty)
        return;

    QFile file;
    file.setFileName(selectedPath);

    file.open(QFile::ReadOnly | QFile::Text);
    bool isOpen = file.isOpen();
    if (!isOpen)
    {
        QMessageBox messageBox;
        messageBox.setWindowTitle("Предупреждение");
        messageBox.setText("При попытке открыть файл произошла ошибка!");
        messageBox.show();
    }

    programing_model->setData(programing_model->index(0, 4), "100");
    programing_model->setData(programing_model->index(1, 4), "1100");

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						54
Изм.	Лист	№ докум.	Подпись	Дата		

```

        programing_model->setData(programing_model->index(2, 4), "10");

        QDomStreamReader xmlReader(&file);
        const ushort row = 3;
        const ushort column = 4;
        bool isLess;
        const ushort rowCount = 13;
        ushort counter = 0;

        while (!xmlReader.atEnd() && !xmlReader.hasError())
        {
            xmlReader.readNext();

            if (xmlReader.isStartElement() && xmlReader.name() == "row")
            {
                isLess = counter < rowCount;
                if (isLess)
                {
                    programing_model->setData(programing_model->index(row +
counter, column), xmlReader.readElementText());
                    ++counter;
                }
                else
                    break;
            }
        }

        file.close();
    }

    //ДП2
    void programming::on_pushButton_14_clicked()
    {
        QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\Templates");
        QString filter = "Файлы .xml |(*.xml)";
        QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);

        bool isEmpty = selectedPath.isEmpty();
        if (isEmpty)
            return;

        QFile file;
        file.setFileName(selectedPath);

        file.open(QFile::ReadOnly | QFile::Text);
        bool isOpen = file.isOpen();
        if (!isOpen)
        {
            QMessageBox messageBox;
            messageBox.setWindowTitle("Предупреждение");
            messageBox.setText("При попытке открыть файл произошла ошибка!");
            messageBox.show();
        }

        programing_modelDP2->setData(programing_modelDP2->index(0, 4), "100");
        programing_modelDP2->setData(programing_modelDP2->index(1, 4), "1100");
        programing_modelDP2->setData(programing_modelDP2->index(2, 4), "11");

        file.close();
    }
}

```

```

void programming::on_pushButton_2_clicked()
{
    programmingdp = new programmingBlock(DP1);
    programmingdp->setModal(true);
    programmingdp->show();
}

void programming::on_pushButton_4_clicked()
{
    programmingdp = new programmingBlock(DP2);
    programmingdp->setModal(true);
    programmingdp->show();
}

void programming::on_pushButton_3_clicked()
{
    programmingdp = new programmingBlock(DT);
    programmingdp->setModal(true);
    programmingdp->show();
}

void programming::on_pushButton_8_clicked()
{
    writeconst = new writeconsts(DP1);
    writeconst->setModal(true);
    writeconst->show();
}

void programming::on_pushButton_10_clicked()
{
    writeconst = new writeconsts(DP2);
    writeconst->setModal(true);
    writeconst->show();
}

void programming::on_pushButton_11_clicked()
{
    writeconst = new writeconsts(DT);
    writeconst->setModal(true);
    writeconst->show();
}

void programming::on_pushButton_18_clicked()
{
    wr = new wrpsd(BT5);
    wr->setModal(true);
    wr->show();
}

void programming::on_pushButton_19_clicked()
{
    wr = new wrpsd(BT6);
    wr->setModal(true);
    wr->show();
}

void programming::on_pushButton_12_clicked()
{
    //WRITE
    wr = new wrpsd(BT1);
    wr->setModal(true);
    wr->show();
}

void programming::on_pushButton_13_clicked()

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						56
Изм.	Лист	№ докум.	Подпись	Дата		

```

{ //READ
    wr = new wrpsd(BT2);
    wr->setModal(true);
    wr->show();
}

void programming::on_pushButton_15_clicked()
{
    wr = new wrpsd(BT3);
    wr->setModal(true);
    wr->show();
}

void programming::on_pushButton_16_clicked()
{
    wr = new wrpsd(BT4);
    wr->setModal(true);
    wr->show();
}

void programming::on_pushButton_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
    "\\savedDataXML");

    bool isExists = dir.exists(path);
    if (!isExists)
        dir.mkdir(path);

    QString savePath = QDir::toNativeSeparators(path +
    "\\LinearitySetting.xml");
    QString selectedPath = QFileDialog::getSaveFileName(this, "Сохранить
    файл", savePath);
    bool isSelected = selectedPath.isEmpty();
    if (isSelected)
        return;

    QMessageBox::information(this, "Сохранение данных", "Данные успешно
    сохранены по адресу: " + savePath);
}

void programming::on_btn_saveprogrdp2_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
    "\\savedDataXML");

    bool isExists = dir.exists(path);
    if (!isExists)
        dir.mkdir(path);

    QString savePath = QDir::toNativeSeparators(path +
    "\\LinearitySetting.xml");
    QString selectedPath = QFileDialog::getSaveFileName(this, "Сохранить
    файл", savePath);
    bool isSelected = selectedPath.isEmpty();
    if (isSelected)
        return;

    QMessageBox::information(this, "Сохранение данных", "Данные успешно
    сохранены по адресу: " + savePath);
}

void programming::on_btn_saveprogrdt_clicked()
{

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						57
Изм.	Лист	№ докум.	Подпись	Дата		

```

    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\savedDataXML");

    bool isExists = dir.exists(path);
    if (!isExists)
        dir.mkdir(path);

    QString savePath = QDir::toNativeSeparators(path +
"\\LinearitySetting.xml");
    QString selectedPath = QFileDialog::getSaveFileName(this, "Сохранить
файл", savePath);
    bool isSelected = selectedPath.isEmpty();
    if (isSelected)
        return;

    QMessageBox::information(this, "Сохранение данных", "Данные успешно
сохранены по адресу: " + savePath);
}

```

Файл «programmingBlock.cpp»

```

#include "programmingBlock.h"
#include "ui_programmingBlock.h"

programmingBlock::programmingBlock(SensorType sensorType, QWidget *parent) :
QDialog(parent), ui(new Ui::programmingDpl)
{
    ui->setupUi(this);

    switch (sensorType)
    {
    case DP1:
        setWindowTitle("Программирование датчика перемещения 1");
        break;

    case DP2:
        setWindowTitle("Программирование датчика перемещения 2");
        break;

    case DT:
        setWindowTitle("Программирование датчика температуры");
        break;
    }

    this->sensorType = sensorType;

    main_layout.addWidget(&progress_bar);
    setLayout(&main_layout);
    progress_bar.setGeometry(25, 50, 450, 20);
    progress_bar.setMinimum(0);
    progress_bar.setMaximum(10);
    progress_bar.setValue(0);

    timer = new QTimer();
    connect (timer, &QTimer::timeout, this, &programmingBlock::TimerTick);
    timer->setInterval(500);
    timer->start();
}

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

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						58
Изм.	Лист	№ докум.	Подпись	Дата		

```

void programmingBlock::TimerTick()
{
    int value = progress_bar.value();
    bool isLess;

    isLess = progress_bar.value() < progress_bar.maximum();

    if (isLess)
        progress_bar.setValue(++value);
    else
    {
        switch (sensorType)
        {
            case DP1:
                QMessageBox::information(this, "Сообщение", "ПО записано в датчик
перемещения 1");
                break;

            case DP2:
                QMessageBox::information(this, "Сообщение", "ПО записано в датчик
перемещения 2");
                break;

            case DT:
                QMessageBox::information(this, "Сообщение", "ПО записано в датчик
температуры");
                break;
        }

        timer->stop();
        close();
    }
}

```

Файл «reports.cpp»

```

#include "reports.h"
#include "ui_reports.h"

QString filex2(QDir::currentPath() + "\\Templates\\LinearityDP2.xlsx");
QString filex3(QDir::currentPath() + "\\Templates\\TemperatureDT.xlsx");
PSI::PSI(QWidget *parent) : QDialog(parent), ui(new Ui::PSI)
{
    ui->setupUi(this);
    setWindowTitle("Отчёты");
    setFixedSize(406, 150);
}

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

void PSI::on_report_dpl_clicked()
{
    ushort exValue = 1;
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\savedDataXML");
    QString filter = ".xml (*xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);
}

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						59
Изм.	Лист	№ докум.	Подпись	Дата		

```

bool isEmpty = selectedPath.isEmpty();
if (isEmpty)
    return;

QFile file;
file.setFileName(selectedPath);

file.open(QFile::ReadOnly);
bool isOpen = file.isOpen();
if (!isOpen)
{
    QMessageBox messageBox;
    messageBox.setWindowTitle("Предупреждение");
    messageBox.setText("При попытке открыть файл произошла ошибка!");
    messageBox.show();
}

QXmlStreamReader xmlReader(&file);
QString factoryNumber;
const ushort rowCount = 13;
const ushort rows = 13;
const ushort columns = 4;
QString data[rows][columns];
ushort row, column;
row = column = 0;
bool isCycle;

while (!xmlReader.atEnd() && !xmlReader.hasError())
{
    xmlReader.readNext();

    if (xmlReader.isStartElement() && xmlReader.name() == "number")
        factoryNumber = xmlReader.readElementText();

    if (xmlReader.isStartElement() && xmlReader.name() == "row")
    {
        isCycle = row < rowCount;
        if (isCycle)
        {
            data[row][column] = xmlReader.readElementText();
            ++row;
        }
        else
        {
            ++column;
            row = 0;
            data[row][column] = xmlReader.readElementText();
            ++row;
        }
    }
}

file.close();

WorkExcel* workExcel = new WorkExcel();
QString filex1(QDir::currentPath() + "\\Templates\\LinearityDP1.xlsx");
workExcel->setTemplatePath(filex1);

const short int result = workExcel->openDocument();
workExcel->reportLinearityCheck(data, &factoryNumber);
workExcel->saveDocument(&factoryNumber, exValue);
workExcel->closeDocument();
}

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						60
Изм.	Лист	№ докум.	Подпись	Дата		

```

void PSI::on_report_dp2_clicked()
{
    int exValue = 2;
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\savedDataXML");
    QString filter = ".xml (*xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);

    bool isEmpty = selectedPath.isEmpty();
    if (isEmpty)
        return;

    QFile file;
    file.setFileName(selectedPath);

    file.open(QFile::ReadOnly);
    bool isOpen = file.isOpen();
    if (!isOpen)
    {
        QMessageBox messageBox;
        messageBox.setWindowTitle("Предупреждение");
        messageBox.setText("При попытке открыть файл произошла ошибка!");
        messageBox.show();
    }

    file.close();
}

void PSI::on_report_dt_clicked()
{
    int exValue = 3;
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\savedDataXML");
    QString filter = ".xml (*xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);

    bool isEmpty = selectedPath.isEmpty();
    if (isEmpty)
        return;

    QFile file;
    file.setFileName(selectedPath);

    file.open(QFile::ReadOnly);
    bool isOpen = file.isOpen();
    if (!isOpen)
    {
        QMessageBox messageBox;
        messageBox.setWindowTitle("Предупреждение");
        messageBox.setText("При попытке открыть файл произошла ошибка!");
        messageBox.show();
    }

    file.close();
}

```

Файл «setting.cpp»

```

#include "setting.h"
#include "ui_setting.h"

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						61
Изм.	Лист	№ докум.	Подпись	Дата		

```

#include "linearitysetting.h"
#include "settingtemp.h"

Setting::Setting(QWidget *parent) : QDialog(parent), ui(new Ui::Setting)
{
    ui->setupUi(this);
    QWidget::setFixedSize(1000, 470);
    QWidget::setWindowTitle("Настройка");

    ui->btn_stop_setting->setEnabled(false);

    connect(ui->btn_start_setting, &QPushButton::clicked, this,
    &Setting::disableStop);
    connect(ui->btn_stop_setting, &QPushButton::clicked, this,
    &Setting::disableStart);

    ui->lndt_N0_DP1->setEnabled(false);
    ui->lndt_N0_DP2->setEnabled(false);
    ui->lndt_Nnom_DP1->setEnabled(false);
    ui->lndt_Nnom_DP2->setEnabled(false);
    ui->lndt_Range_DP1->setEnabled(false);
    ui->lndt_Range_DP2->setEnabled(false);
    ui->lineEdit_voltage->setEnabled(false);
    ui->lineEdit_voltage_2->setEnabled(false);
    ui->lineEdit_voltage3->setEnabled(false);
    ui->lineEdit_voltage4->setEnabled(false);
    ui->lineEdit_codeADC->setEnabled(false);
    ui->lineEdit_spaceled->setEnabled(false);
    ui->lineEdit_space2ed->setEnabled(false);
    ui->lineEdit_spacemm->setEnabled(false);
    ui->lineEdit_spacemm2->setEnabled(false);
    ui->lineEdit_voltageV->setEnabled(false);
    ui->lineEdit_voltageV_2->setEnabled(false);
    ui->lineEdit_TC->setEnabled(false);
    ui->lineEdit_codT->setEnabled(false);
    ui->lineEdit_gostT->setEnabled(false);

    timersetting = new QTimer(this);
    connect (timersetting, &QTimer::timeout, this,
    &Setting::onTimeoutSetting);
}

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

void Setting::on_pushButton_clicked()
{
    linearitySetting = new linearitysetting(this);
    linearitySetting->setModal(true);
    linearitySetting -> show();
}

void Setting::disableStart()
{
    timersetting ->stop();
    ui->btn_start_setting->setEnabled(true);
    ui->btn_stop_setting->setEnabled(false);
}

void Setting::disableStop()

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дата		

```

{
    timersetting ->start(500);
    ui->lnedt_N0_DP1->setText(QString::number(N0));
    ui->lnedt_N0_DP2->setText(QString::number(N0));
    ui->lnedt_Nnom_DP1->setText(QString::number(Nnom));
    ui->lnedt_Nnom_DP2->setText(QString::number(Nnom));
    ui->lnedt_Range_DP1->setText(QString::number(RangeCode));
    ui->lnedt_Range_DP2->setText(QString::number(RangeCode));
    ui->btn_start_setting->setEnabled(false);
    ui->btn_stop_setting->setEnabled(true);
}

void Setting::on_pushButton_2_clicked()
{
    settingtmp = new settingtemp(this);
    settingtmp ->setModal(true);
    settingtmp -> show();
}

void Setting::onTimeoutSetting()
{
    unsigned int number = static_cast<unsigned int>(rand() %
differenceValueLi + 1);
    unsigned int result = minimalValueLi + number;
    ui->lineEdit_voltage->setText(QString::number(result));

    unsigned int number2 = static_cast<unsigned int>(rand() %
differenceValueLi2 + 1);
    unsigned int result2 = minimalValueLi2 + number2;
    ui->lineEdit_voltage_2->setText(QString::number(result2));

    unsigned int number3 = static_cast<unsigned int>(rand() %
differenceValueLi3 + 1);
    unsigned int result3 = minimalValueLi3 + number3;
    ui->lineEdit_voltage3->setText(QString::number(result3));

    unsigned int number4 = static_cast<unsigned int>(rand() %
differenceValueLi4 + 1);
    unsigned int result4 = minimalValueLi4 + number4;
    ui->lineEdit_voltage4->setText(QString::number(result4));

    //Код АЦП
    unsigned int number5 = static_cast<unsigned int>(rand() %
differenceValueCodeAdc + 1);
    unsigned int result5 = minCodeAdc + number5;
    ui->lineEdit_codeADC->setText(QString::number(result5));

    //Перемещение (ед.к) 2
    unsigned int number6 = static_cast<unsigned int>(rand() %
differenceSpace2 + 1);
    unsigned int result6 = minSpace3 + number6;
    ui->lineEdit_space2ed->setText(QString::number(result6));

    // Перемещение (мм) 2
    double mmSPACE = static_cast<double>(result6 / 10.0);
    ui->lineEdit_spacemm2->setText(QString::number(mmSPACE, 'f', 1));

    unsigned int number7 = static_cast<unsigned int>(rand() % differenceCodeT
+ 1);
    unsigned int result7 = minCodeT + number7;
    ui->lineEdit_codT->setText(QString::number(result7));

    // Перемещение (мм) 1
    unsigned short int moveCode1 = rand() % 300;

```

```

double moveMM1 = static_cast<double>(moveCode1 / 10.0);
ui->lineEdit_spacemm->setText(QString::number(moveMM1, 'f', 1));

//Перемещение (ед.к) 1
ui->lineEdit_spaceled->setText(QString::number(moveCode1));

// К напряжения 1
ui->lineEdit_voltageV->setText(QString::number(result2-result));

// К напряжения 2
ui->lineEdit_voltageV_2->setText(QString::number(result4-result3));

// Темп по ГОСТ.
unsigned short int tempGOST = rand() % 1800 + 1;
double tempGOSTD = (1500 + tempGOST) / 100.0;
ui->lineEdit_gostT->setText(QString::number(tempGOSTD, 'f', 2));

//Темп в С
double tempC = tempGOSTD + 0.5;
ui->lineEdit_TC->setText(QString::number(tempC, 'f', 2));
}

```

Файл «settingsadapter.cpp»

```

#include "settingsadapter.h"
#include "ui_settingsadapter.h"
#include "QFileDialog"

settingsAdapter::settingsAdapter(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::settingsAdapter)
{
    ui->setupUi(this);
    QWidget::setWindowTitle("Настройка адаптера");

    ui->lineEdit_file->setEnabled(false);
}

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

void settingsAdapter::on_pushButton_clicked()
{
    QString filter = ".exe (*exe)";
    QString file_name = QFileDialog::getOpenFileName(this, "Выбрать файл",
"C://", filter);
    ui->lineEdit_file->setText(file_name);
}

```

Файл «settingstemp.cpp»

```

#include "settingstemp.h"
#include "ui_settingstemp.h"
#include "constansts.h"

settingtemp::settingtemp(QWidget *parent) : QDialog(parent), ui(new
Ui::settingtemp)
{
    ui->setupUi(this);
    QWidget::setFixedSize(730, 360);

    modeltmp = new QStandardItemModel(7, 3, this);
}

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						64
Изм.	Лист	№ докум.	Подпись	Дата		

```

modeltmp->setHeaderData(0, Qt::Horizontal, "Код температуры");
modeltmp->setHeaderData(1, Qt::Horizontal, "Код темп. по ГОСТ");
modeltmp->setHeaderData(2, Qt::Horizontal, "МИТ");
modeltmp->setHeaderData(0, Qt::Vertical, "0");
modeltmp->setHeaderData(1, Qt::Vertical, "25");
modeltmp->setHeaderData(2, Qt::Vertical, "50");
modeltmp->setHeaderData(3, Qt::Vertical, "100");
modeltmp->setHeaderData(4, Qt::Vertical, "150");
modeltmp->setHeaderData(5, Qt::Vertical, "200");
modeltmp->setHeaderData(6, Qt::Vertical, "280");

ui->tableView->setModel(modeltmp);
ui->tableView->resizeColumnsToContents();
ui->tableView->setEditTriggers(QTableView::NoEditTriggers);

ui->btn_stop_tmp->setEnabled(false);
ui->btn_write_tmp->setEnabled(false);
ui->btn_cancel_tmp->setEnabled(false);
ui->btn_clear_tmp->setEnabled(false);
ui->btn_savetmp->setEnabled(false);
ui->lineEdit_codTmp->setEnabled(false);
ui->lineEdit_codTmpGost->setEnabled(false);
ui->lineEdit_codTmpIT->setEnabled(false);

connect(ui->btn_start_tmp, &QPushButton::clicked, this,
&settingtemp::disableStop);
connect(ui->btn_stop_tmp, &QPushButton::clicked, this,
&settingtemp::disableStart);

timertmp = new QTimer(this);
connect (timertmp, &QTimer::timeout, this, &settingtemp::onTimeoutTmp);
}

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

void settingtemp::disableStop()
{
    timertmp->start(500);

    ui->btn_savetmp->setEnabled(true);
    ui->btn_start_tmp->setEnabled(false);
    ui->btn_stop_tmp->setEnabled(true);
    ui->btn_write_tmp->setEnabled(true);
}

void settingtemp::disableStart()
{
    timertmp->stop();

    ui->btn_start_tmp->setEnabled(true);
    ui->btn_stop_tmp->setEnabled(false);
}

void settingtemp::onTimeoutTmp()
{
    unsigned int number = static_cast<unsigned int>(rand() %
differenceValueCodTmp + 1);
    unsigned int result = minimalValueCodTmp + number;
    ui->lineEdit_codTmp->setText(QString::number(result));

    unsigned short int tempGOST = rand() % 1800 + 1;

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						65
Изм.	Лист	№ докум.	Подпись	Дата		

```

double tempGOSTD = (1500 + tempGOST) / 100.0;
ui->lineEdit_codTmpGost->setText(QString::number(tempGOSTD, 'f', 2));

double tempC = tempGOSTD + 0.12575;
ui->lineEdit_codTmpIT->setText(QString::number(tempC, 'f', 5));

}

void settingtemp::on_btn_write_tmp_clicked()
{
    ui->btn_cancel_tmp->setEnabled(true);
    ui->btn_clear_tmp->setEnabled(true);

    QString codT = ui->lineEdit_codTmp->text();
    QString tmpGost = ui->lineEdit_codTmpGost->text();
    QString tmpIT = ui->lineEdit_codTmpIT->text();
    modeltmp->setData(modeltmp->index(curRow, 0), codT);
    modeltmp->setData(modeltmp->index(curRow, 1), tmpGost);
    modeltmp->setData(modeltmp->index(curRow, 2), tmpIT);

    curRow++;
    countClickTmp++;

    if(countClickTmp==7)
        ui->btn_write_tmp->setEnabled(false);
}

void settingtemp::on_btn_cancel_tmp_clicked()
{
    curRow--;

    modeltmp->clearItemData(modeltmp->index(curRow, 2));
    modeltmp->clearItemData(modeltmp->index(curRow, 1));
    modeltmp->clearItemData(modeltmp->index(curRow, 0));

    ui->btn_write_tmp->setEnabled(true);
    countClickTmp--;

    if (countClickTmp==0){
        ui->btn_cancel_tmp->setEnabled(false);
        ui->btn_clear_tmp->setEnabled(false);
    }
}

void settingtemp::on_btn_exit_tmp_clicked()
{
    close();
}

void settingtemp::on_btn_clear_tmp_clicked()
{
    for (int row = 0; row<modeltmp->rowCount(); ++row){

        for(int col = 0; col<modeltmp->columnCount(); ++col){
            QStandardItem *item = new QStandardItem();
            modeltmp->setItem(row,col, item);
            curCol=0;
            curRow=0;

        }

        modeltmp->index(curRow, curCol);
        countClickTmp=0;
        if(countClickTmp == 0)

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						66
Изм.	Лист	№ докум.	Подпись	Дата		

```

        ui->btn_write_tmp->setEnabled(true);
        ui->btn_cancel_tmp->setEnabled(false);
        ui->btn_clear_tmp->setEnabled(false);
    }

void settingtemp::on_btn_open_tmp_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\savedDataXML");
    QString filter = ".xml (*xml)";
    QString selectedPath = QFileDialog::getOpenFileName(this, "Выбрать файл",
path, filter);

    bool isEmpty = selectedPath.isEmpty();
    if (isEmpty)
        return;

    QFile file;
    file.setFileName(selectedPath);

    file.open(QFile::ReadOnly | QFile::Text);
    bool isOpen = file.isOpen();
    if (!isOpen)
    {
        QMessageBox messageBox;
        messageBox.setWindowTitle("Предупреждение");
        messageBox.setText("При попытке открыть файл произошла ошибка!");
        messageBox.show();
    }

    QDomStreamReader xmlReader(&file);
    int row = 0;
    int col = 0;

    while (!xmlReader.atEnd() && !xmlReader.hasError())
    {
        xmlReader.readNext();

        if (xmlReader.isStartElement() && xmlReader.name() == "row")
        {
            if(row < modeltmp->rowCount() && col < modeltmp->columnCount())
                modeltmp->setData(modeltmp-
>index(row,col),xmlReader.readElementText());

            col++;
            if(col >= modeltmp->columnCount()){
                col = 0;
                row++;
            }
        }
    }

    file.close();
}

void settingtemp::on_btn_savetmp_clicked()
{
    QString path = QDir::toNativeSeparators(QDir::currentPath() +
"\\savedDataXML");

    bool isExists = dir.exists(path);
    if (!isExists)

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						67
Изм.	Лист	№ докум.	Подпись	Дата		

```

        dir.mkdir(path);

        QString savePath = QDir::toNativeSeparators(path +
"\\TemperatureSetting.xml");
        QString selectedPath = QFileDialog::getSaveFileName(this, "Сохранить
файл", savePath);
        bool isSelected = selectedPath.isEmpty();
        if (isSelected)
            return;

        QFile file;
        file.setFileName(selectedPath);
        file.open(QIODevice::WriteOnly);

        bool isOpen = file.isOpen();
        if (!isOpen)
        {
            QMessageBox message;
            message.setText("Файл по адресу " + selectedPath + "не удалось
открыть!");
            message.show();
            return;
        }

        QDomStreamWriter xmlWriter(&file);
        xmlWriter.setAutoFormatting(true);
        xmlWriter.writeStartDocument();

        const ushort rows = static_cast<ushort>(modeltmp->rowCount());
        const ushort columns = static_cast<ushort>(modeltmp->columnCount());
        QString prepareData[rows][columns];
        for (ushort row = 0; row < rows; ++row)
            for (ushort column = 0; column < columns; ++column)
                prepareData[row][column] = modeltmp->data(modeltmp->index(row,
column)).toString();

        QString dateTime = QDateTime::currentDateTime().toString("dd.MM.yyyy
hh.mm.ss");
        QString factoryNumber = ui->factory_number->text();

        xmlWriter.writeStartElement("data");
        xmlWriter.writeStartElement("date");
        xmlWriter.writeCharacters(dateTime);
        xmlWriter.writeEndElement();

        xmlWriter.writeStartElement("number");
        xmlWriter.writeCharacters(factoryNumber);
        xmlWriter.writeEndElement();

        xmlWriter.writeStartElement("table");

        for (ushort row = 0; row < rows; ++row)
            for (ushort column = 0; column < columns; ++column)
            {
                xmlWriter.writeStartElement("row");
                xmlWriter.writeCharacters(prepareData[row][column]);
                xmlWriter.writeEndElement();
            }

        xmlWriter.writeEndElement();
        xmlWriter.writeEndElement();
        xmlWriter.writeEndDocument();

        file.close();

```

```
}
```

Файл «workexcel.cpp»

```
#include "workexcel.h"
```

```
#define LC_row 13
```

```
#define LC_column 4
```

```
#define TC_row 7
```

```
#define TC_column 3
```

```
// Конструктор класса.
```

```
WorkExcel::WorkExcel(QObject *parent) : QObject(parent)
```

```
{
```

```
}
```

```
// Деструктор класса.
```

```
WorkExcel::~WorkExcel()
```

```
{
```

```
}
```

```
// Функция для установки пути к шаблонам в зависимости от типа отчета.
```

```
void WorkExcel::setTemplatePath(QString path)
```

```
{
```

```
    filePathExcel = path;
```

```
}
```

```
// Запись данных в одну ячейку.
```

```
void WorkExcel::writeToCell(const int row, const int column, const QVariant data)
```

```
{
```

```
    // Получение указателя на указанную ячейку [row][col] (нумерация начинается с единицы!).
```

```
    QAxObject *cell = sheet->querySubObject("Cells(QVariant&, QVariant&)", row, column);
```

```
    cell->setProperty("Value", data);
```

```
    delete cell;
```

```
}
```

```
// Запись данных в выбранный диапазон ячеек.
```

```
void WorkExcel::writeToCell(const QString row, const QString column, const QVariant data)
```

```
{
```

```
    QAxObject *range = sheet->querySubObject("Range(const QVariant&", QVariant(row + ":" + column));
```

```
    // Запись в выбранный диапазон ячеек вполне конкретного параметра data.
```

```
    range->dynamicCall("SetValue(const QVariant&", data);
```

```
    delete range;
```

```
}
```

```
// Запуск приложения Excel с открытием выбранного шаблона.
```

```
short WorkExcel::openDocument()
```

```
{
```

```
    // Получение указателя на Excel.
```

```
    excel = new QAxObject("Excel.Application", this);
```

```
    if (!excel)
```

```
        return -3;
```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.12.001 ПЗ

Лист

69

```

        excel->dynamicCall("SetVisible(bool)", true);

        // Получение указателя на книги.
        workbooks = excel->querySubObject("Workbooks");
        if (!workbooks)
            return -2;

        // Получение указателя на директорию, из которой необходимо загрузить
        книгу.
        workbook = workbooks->querySubObject("Open(const QString&)",
        filePathExcel);

        // Получение указателя на листы (вкладки снизу).
        sheets = workbook->querySubObject("Sheets");
        if (!sheets)
            return -1;

        // Получение указателя на конкретный лист Excel.
        sheet = sheets->querySubObject("Item(QString&)", QVariant("Лист1"));
        if (!sheet)
            return 0;

        return 1;
    }

    // Сохранение документа Excel с использованием заводского номера, номера
    канала и диапазона.
    bool WorkExcel::saveDocument(const QString *factoryNumber, const ushort
    value)
    {
        // Указание приложению Excel на запрет выводить служебные сообщения.
        excel->setProperty("DisplayAlerts", 0);

        savePath = QDir::toNativeSeparators(QDir::currentPath() + "\\Reports");
        QDir directory(savePath);

        if (!directory.exists())
            directory.mkpath(savePath);

        QString title = "Сохранение файла Excel";
        QString fileName;

        switch (value)
        {
        case 1:
            fileName = "Заводской-№" + *factoryNumber + "_Датчик-перемещения-1_";
            break;

        case 2:
            fileName = "Заводской-№" + *factoryNumber + "_Датчик-перемещения-2_";
            break;

        case 3:
            fileName = "Заводской-№" + *factoryNumber + "_Датчик-температуры-2_";
            break;
        }

        QString datetime =
        QDateTime::currentDateTime().toString("dd.MM.yyyy_HH.mm.ss");
        QString fullFileName = QDir::toNativeSeparators(savePath + "\\" +
        fileName + datetime + tr(".xlsx"));
        QString filter = "Файл Excel (*.xlsx);;Все файлы (*.*)";
    }

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						70
Изм.	Лист	№ докум.	Подпись	Дата		

```

        QString pathToFile =
QDir::toNativeSeparators(QFileDialog::getSaveFileName(nullptr, title,
fullFileName, filter));

        if (!pathToFile.isEmpty())
        {
            // Сохранение файла Excel в указанном каталоге.
            if (workbook->dynamicCall("SaveAs (const QString&)",
pathToFile).toBool())
                return true;
            else
                return false;
        }
        else
            return false;
    }

// Закрытие документа и приложения Excel.
void WorkExcel::closeDocument()
{
    excel->setProperty("DisplayAlerts", 1);

    delete sheet;
    delete sheets;
    workbook->dynamicCall("Close()");
    delete workbook;
    delete workbooks;
    excel->dynamicCall("Quit()");
    delete excel;
}

// Формирование отчета, содержащего результаты проверки выходного сигнала
блока.
void WorkExcel::reportLinearityCheck(QString data[13][4], const QString
*factoryNumber)
{
    writeToCell(10, 3, *factoryNumber);

    const ushort rows = 13;
    const ushort columns = 4;
    // Запись данных в файл Excel.
    for (ushort row = 0; row < rows; ++row)
        for (ushort column = 0; column < columns; ++column)
            writeToCell(10 + row, 6 + column, data[row][column]);
}

```

Файл «writeconsts.cpp»

```

#include "writeconsts.h"
#include "ui_writeconsts.h"

writeconsts::writeconsts(SensorType sensorType, QWidget *parent) :
    QDialog(parent),
    ui(new Ui::writeconsts)
{
    ui->setupUi(this);

    switch (sensorType)
    {
        case DP1:
            setWindowTitle("Запись констант в датчик перемещения 1");
            break;

        case DP2:

```

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.12.001 ПЗ

Лист

71

```

        setWindowTitle("Запись констант в датчик перемещения 2");
        break;

    case DT:
        setWindowTitle("Запись констант в датчик температуры");
        break;
    }

    this->sensorType = sensorType;

    main_layout.addWidget(&progress_bar);
    setLayout(&main_layout);
    progress_bar.setGeometry(25, 50, 450, 20);
    progress_bar.setMinimum(0);
    progress_bar.setMaximum(10);
    progress_bar.setValue(0);

    timer = new QTimer();
    connect (timer, &QTimer::timeout, this, &writeconsts::TimerTick);
    timer->setInterval(500);
    timer->start();
}

void writeconsts::TimerTick()
{
    int value = progress_bar.value();
    bool isLess;

    isLess = progress_bar.value() < progress_bar.maximum();

    if (isLess)
        progress_bar.setValue(++value);
    else
    {
        switch (sensorType)
        {
            case DP1:
                QMessageBox::information(this, "Сообщение", "Константы записаны в датчик перемещения 1");
                break;

            case DP2:
                QMessageBox::information(this, "Сообщение", "Константы записаны в датчик перемещения 2");
                break;

            case DT:
                QMessageBox::information(this, "Сообщение", "Константы записаны в датчик температуры");
                break;
        }

        timer->stop();
        close();
    }
}

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

```

Файл «wrpsd.cpp»

```
#include "wrpsd.h"
#include "ui_wrpsd.h"

wrpsd::wrpsd(SensorType sensorType, QWidget *parent) :
    QDialog(parent),
    ui(new Ui::wrpsd)
{
    ui->setupUi(this);
    switch (sensorType)
    {
        case BT1:
            setWindowTitle("Запись в ПЗУ датчика перемещения 1");
            break;

        case BT2:
            setWindowTitle("Чтение из ПЗУ датчика перемещения 1");
            break;

        case BT3:
            setWindowTitle("Запись в ПЗУ датчика перемещения 2");
            break;

        case BT4:
            setWindowTitle("Чтение из ПЗУ датчика перемещения 2");
            break;

        case BT5:
            setWindowTitle("Запись в ПЗУ датчика температуры");
            break;

        case BT6:
            setWindowTitle("Чтение из ПЗУ датчика температуры");
            break;
    }

    this->sensorType = sensorType;

    main_layout.addWidget(&progress_bar);
    setLayout(&main_layout);
    progress_bar.setGeometry(25, 50, 450, 20);
    progress_bar.setMinimum(0);
    progress_bar.setMaximum(10);
    progress_bar.setValue(0);

    timer = new QTimer();
    connect (timer, &QTimer::timeout, this, &wrpsd::TimerTick);
    timer->setInterval(500);
    timer->start();
}

void wrpsd::TimerTick()
{
    int value = progress_bar.value();
    bool isLess;

    isLess = progress_bar.value() < progress_bar.maximum();

    if (isLess)
        progress_bar.setValue(++value);
    else
    {

```

					ПГУ 1.090301.12.001 ПЗ	Лист
						73
Изм.	Лист	№ докум.	Подпись	Дата		

```

switch (sensorType)
{
case BT1:
    QMessageBox::information(this, "Сообщение", "Данные записаны в
ПЗУ датчика перемещения 1");
    break;

case BT2:
    QMessageBox::information(this, "Сообщение", "Данные считаны из
ПЗУ датчика перемещения 1");
    break;

case BT3:
    QMessageBox::information(this, "Сообщение", "Данные записаны в
ПЗУ датчика перемещения 2");
    break;

case BT4:
    QMessageBox::information(this, "Сообщение", "Данные считаны из
ПЗУ датчика перемещения 2");
    break;

case BT5:
    QMessageBox::information(this, "Сообщение", "Данные записаны в
ПЗУ датчика температуры");
    break;

case BT6:
    QMessageBox::information(this, "Сообщение", "Данные считаны из
ПЗУ датчика температуры");
    break;
}

timer->stop();
close();
}

}

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

```

Приложение Б

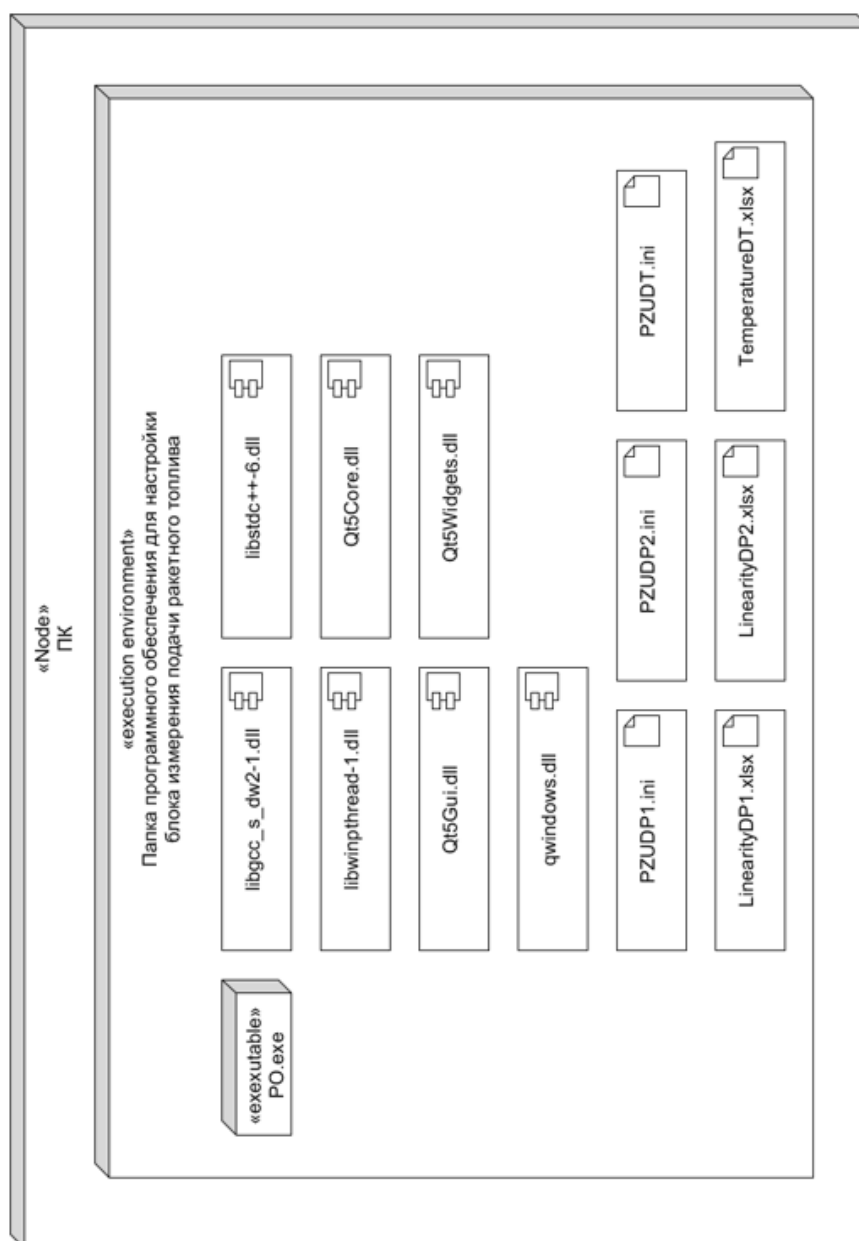
(Обязательное)

Программное обеспечение для настройки блока измерения подачи ракетного
топлива

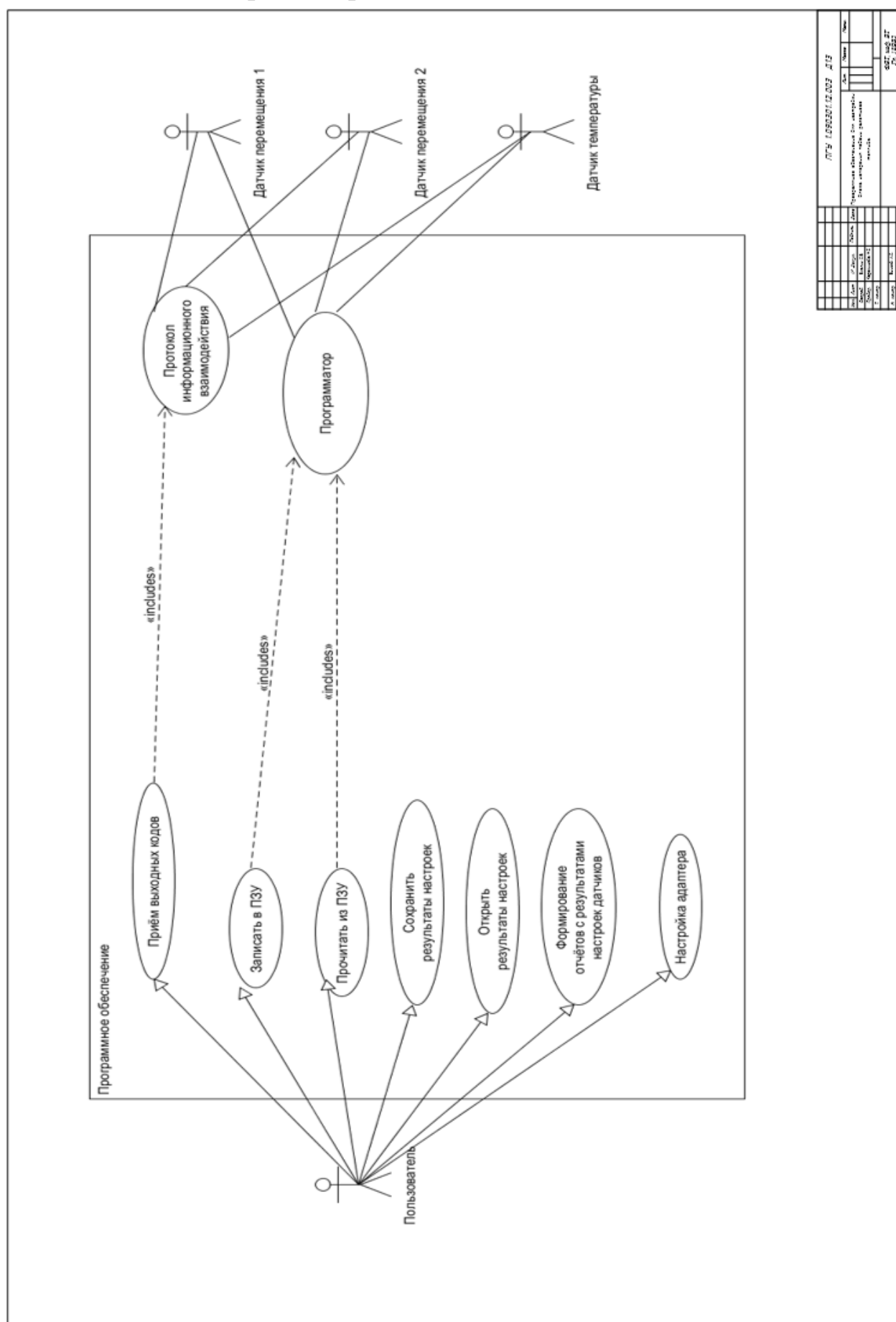
UML-диаграммы

					ПГУ 1.090301.12.001 ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дата		

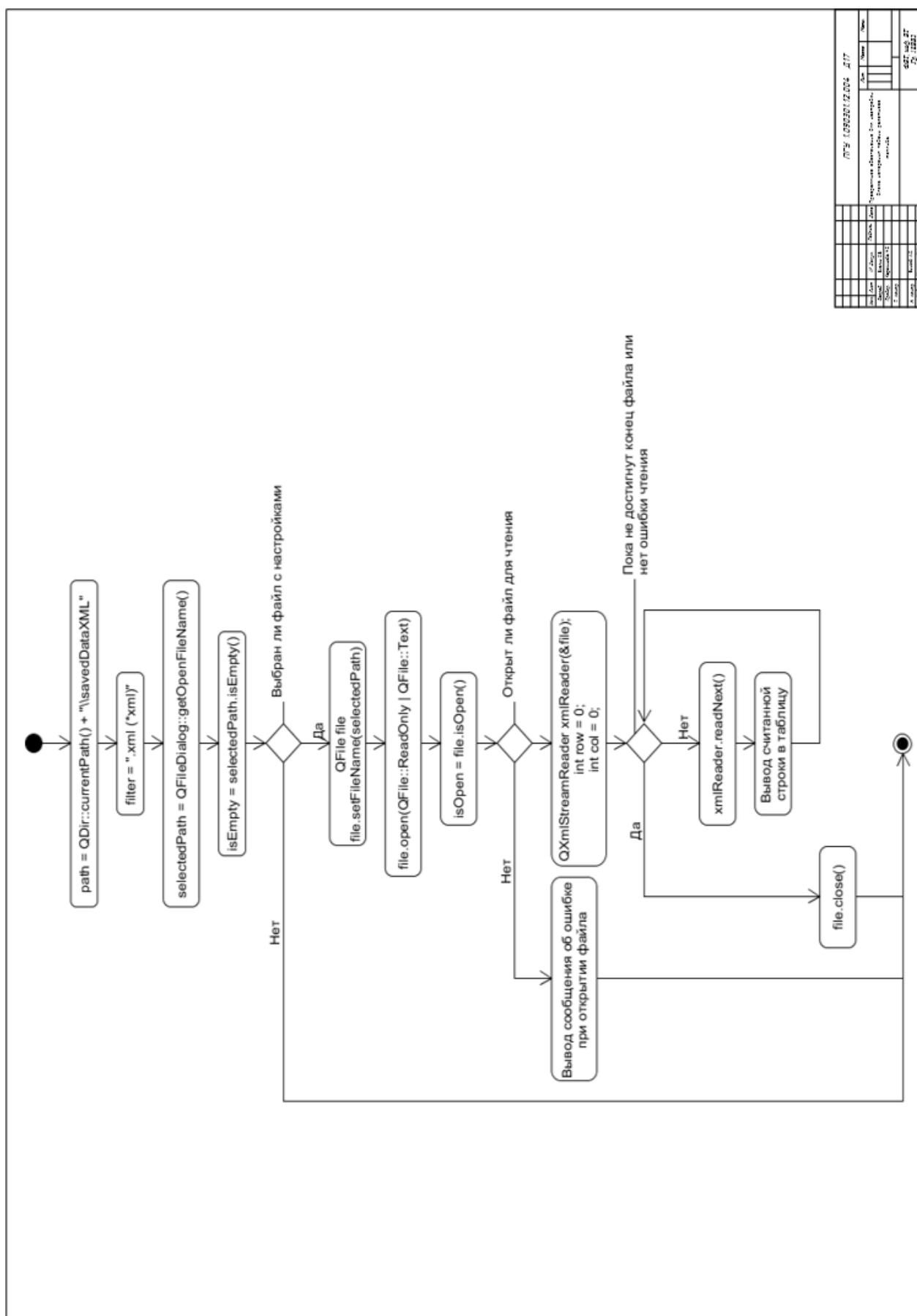
Б.1 - *UML* - Диаграмма развертывания

[illegible]

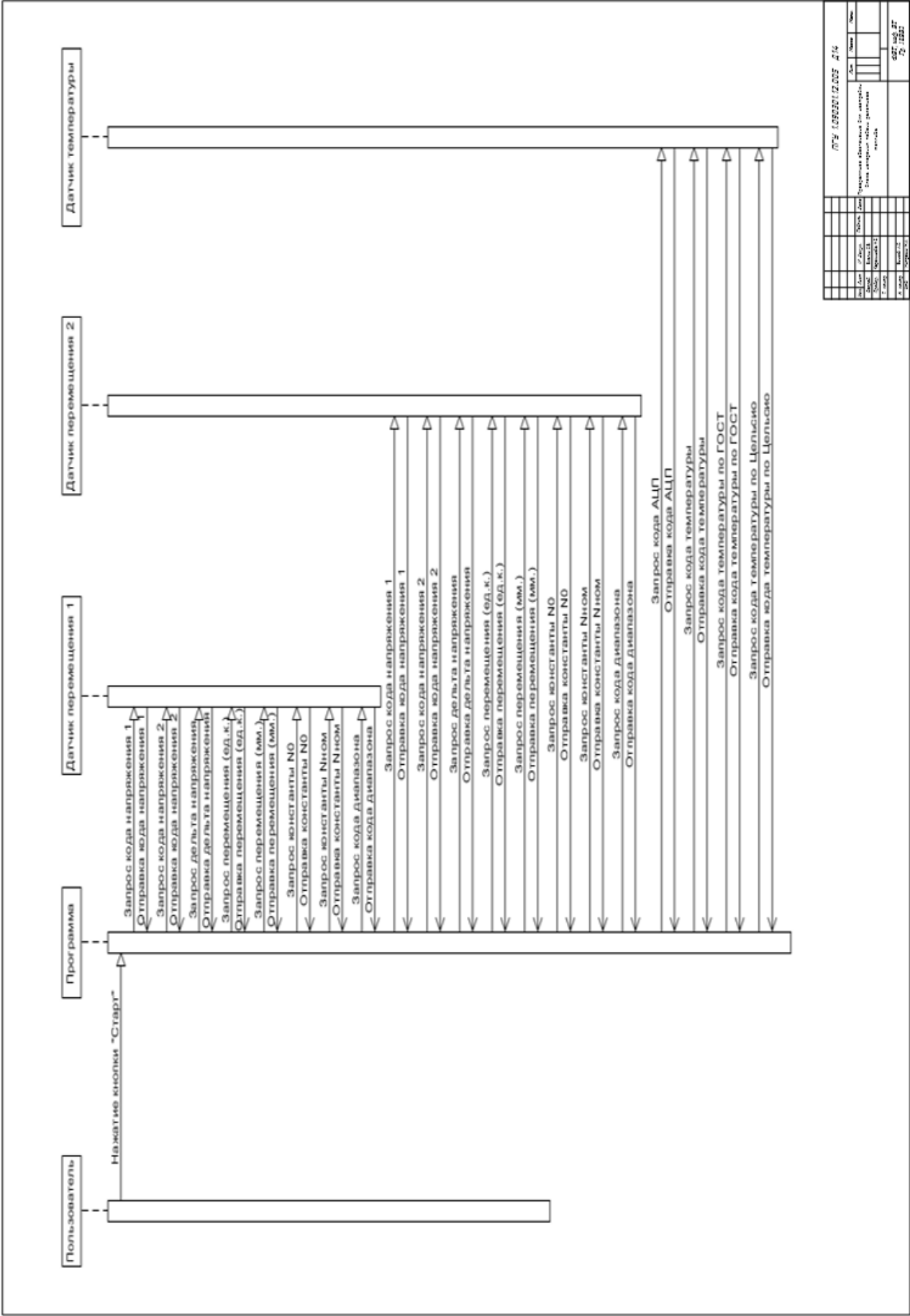
Б.2 – *UML* – Диаграмма прецедентов



Б.3 – *UML* – Диаграмма деятельности



Б.4 – UML – Диаграмма последовательностей



ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата
ПГУ 1.090301.12.001 ПЗ		Изм.	Лист	№ докум.	Подпись	Дата

Приложение В

(Обязательное)

Программное обеспечение для настройки блока измерения подачи ракетного
топлива

Результаты тестирования

					ПГУ 1.090301.12.001 ПЗ	Лист
						80
Изм.	Лист	№ докум.	Подпись	Дата		

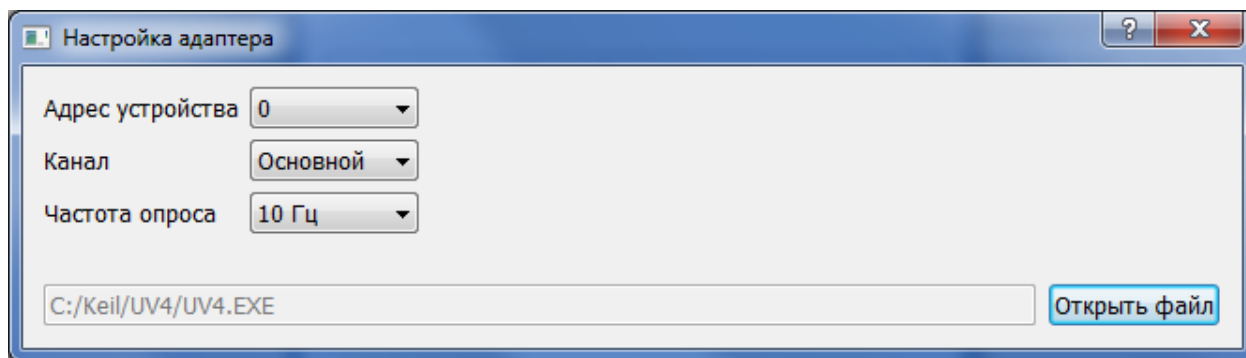


Рисунок В.1 — ПФ.1.1 Указание пути к среде разработки Keil

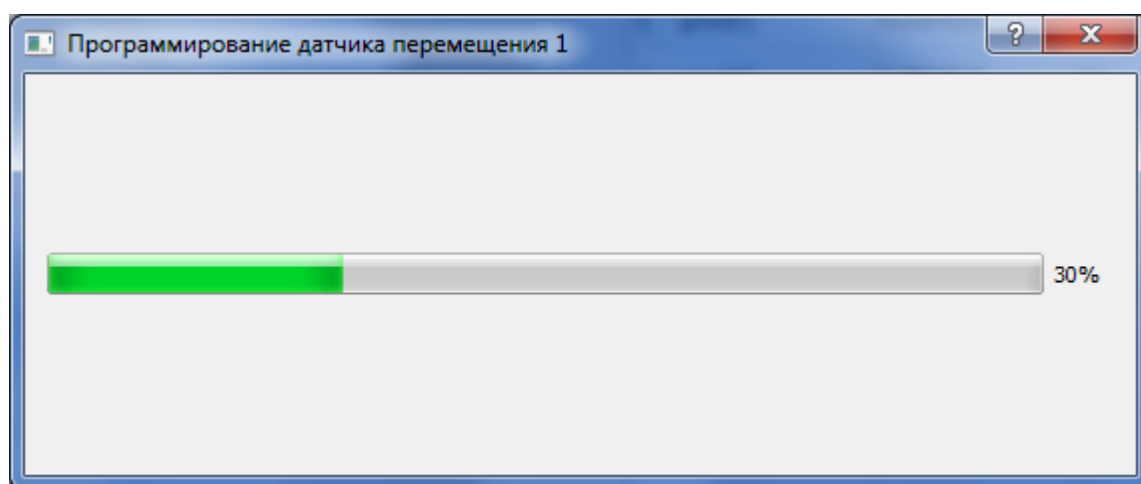


Рисунок В.2 — ПФ.1.2 Процесс записи ВПО в ПЗУ датчика перемещения 1

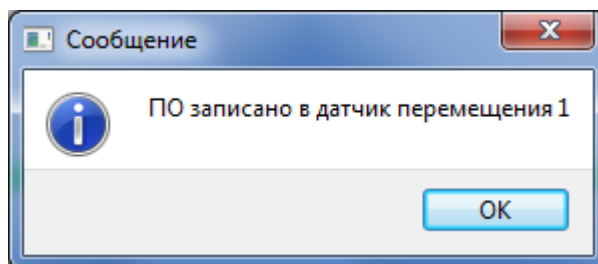


Рисунок В.3 — ПФ.1.2 Сообщение об успешной записи ВПО в датчик перемещения 1

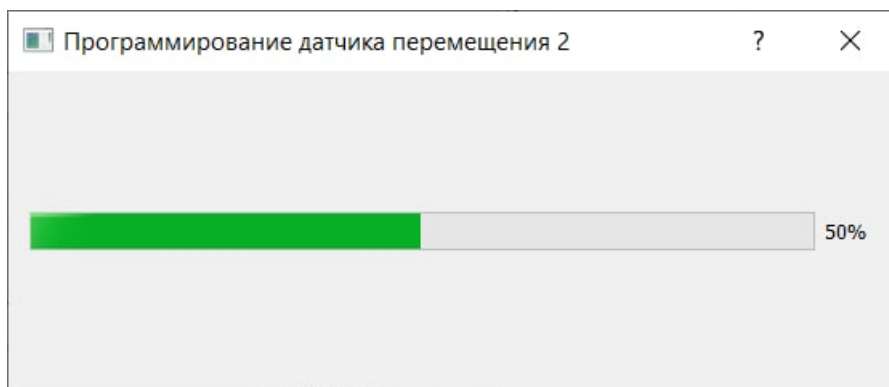


Рисунок В.4 — ПФ.1.3 Процесс записи ВПО в ПЗУ датчика перемещения 2

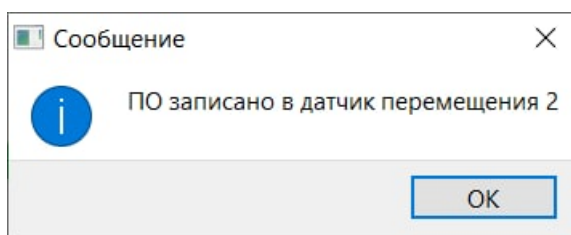


Рисунок В.5 — ПФ.1.3 Сообщение об успешной записи ВПО в датчик перемещения 2

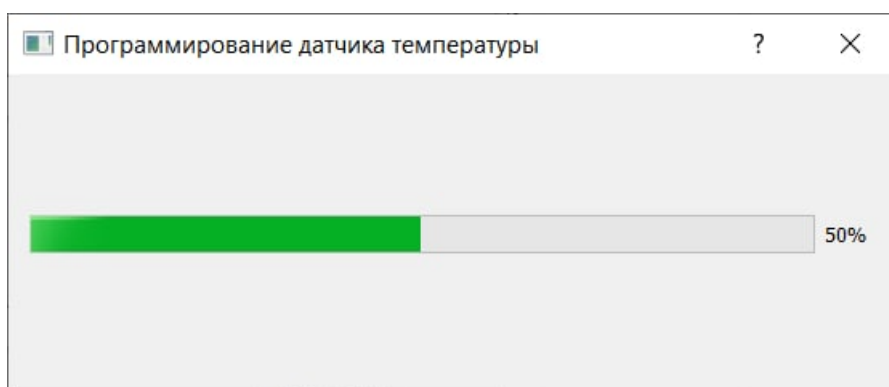


Рисунок В.6 — ПФ.1.4 Процесс записи ВПО в ПЗУ датчика температуры

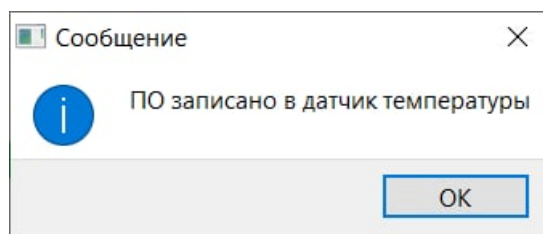


Рисунок В.7 — ПФ.1.4 Сообщение об успешной записи ВПО в датчик температуры

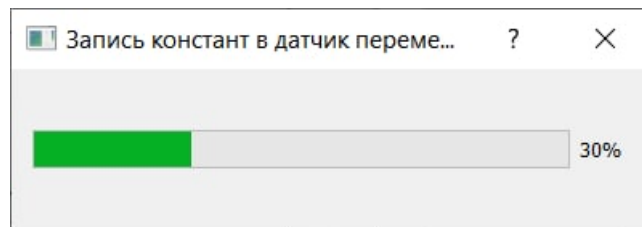


Рисунок В.8 — ПФ.1.5 Процесс записи констант в датчик перемещения 1

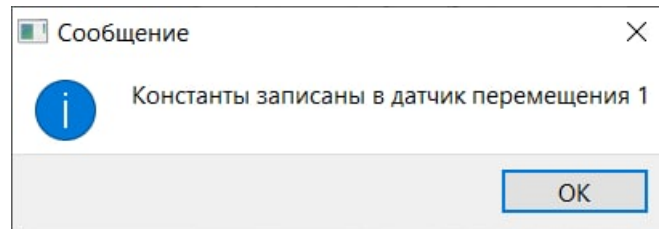


Рисунок В.9 — ПФ.1.5 Сообщение об успешной записи констант в датчик перемещения 1

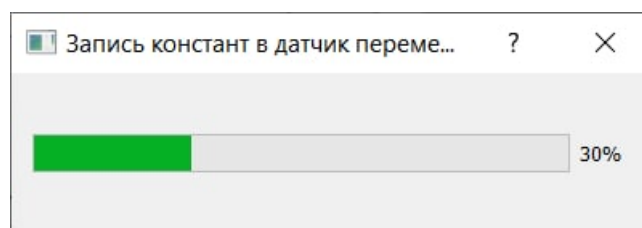


Рисунок В.10 — ПФ.1.6 Процесс записи констант в датчик перемещения 2

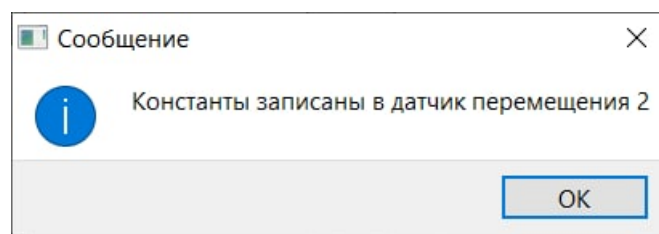


Рисунок В.11 — ПФ.1.6 Сообщение об успешной записи констант в датчик перемещения 2

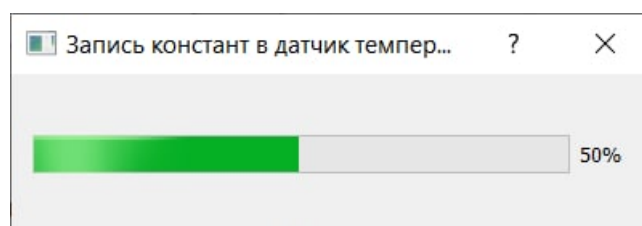


Рисунок В.12 — ПФ.1.7 Процесс записи констант в датчик температуры

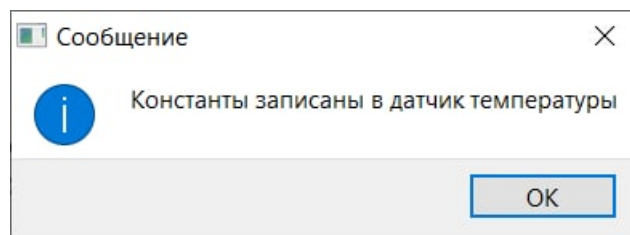


Рисунок В.13 — ПФ.1.7 Сообщение об успешной записи констант в датчик температуры

Заводской номер блока №123456	
Датчик перемещения 1	
Код напряжения 1	2349
Код напряжения 2	308
К Δ напряжения	152
Перемещение (ед.к.)	2657
Перемещение (мм)	15.2
N0	100
N ном.	1100
Код диапазона	10
Датчик перемещения 2	
Код напряжения 1	3413
Код напряжения 2	3666
К Δ напряжения	253
Перемещение (ед.к.)	501
Перемещение (мм)	50.1
N0	100
N ном.	1100
Код диапазона	10
Датчик температуры	
Код АЦП	2140
Код температуры	2490
Температура по ГОСТ °C	21.64
Температура °C	22.14

Старт Стоп

Настройка линейности Настройка температуры

Рисунок В.14 — ПФ.1.8 Приём выходных кодов от датчиков перемещения 1, перемещения 2, температуры

Настройка линейности

?

×

Заводской номер блока

№123456

Данные блока

Датчик перемещения 1

Код напряжения 1

2433

Код напряжения 2

2628

К Δ напряжения

195

Перемещение(ед.к)

2

Таблица градуировки

	Прямой ход	Обратный ход	Прямой ход	Обратный ход
-18мм				
-15мм				
-12мм				
-9мм				
-6мм				
-3мм				
0мм				
3мм				
6мм				
9мм				
12мм				
15мм				
18мм				

Управление

Старт

Стоп

Записать

Отменить

Очистить

Сохранить

Открыть

Выход

Рисунок В.15 — ПФ.1.9 Получение выходных кодов для настройки линейности датчика перемещения 1

Настройка линейности

?

×

Заводской номер блока

№123456

Данные блока

Датчик перемещения 1

Код напряжения 1

2433

Код напряжения 2

2628

К Δ напряжения

195

Перемещение(ед.к)

2

Таблица градуировки

	Прямой ход	Обратный ход	Прямой ход	Обратный ход
-18мм	2			
-15мм				
-12мм				
-9мм				
-6мм				
-3мм				
0мм				
3мм				
6мм				
9мм				
12мм				
15мм				
18мм				

Управление

Старт

Стоп

Записать

Отменить

Очистить

Сохранить

Открыть

Выход

Рисунок В.16 — ПФ.1.10 Запись выходного кода датчика перемещения 1 в таблицу градуировки

Настройка линейности

Заводской номер блока
№123456

Данные блока

Датчик перемещения 2

Код напряжения 1: 3412

Код напряжения 2: 3684

К Δ напряжения: 272

Перемещение(ед.к): 1

Таблица градуировки

	Прямой ход	Обратный ход	Прямой ход	Обратный ход
-18мм				
-15мм				
-12мм				
-9мм				
-6мм				
-3мм				
0мм				
3мм				
6мм				
9мм				
12мм				
15мм				
18мм				

Управление

Старт Стоп Записать Отменить Очистить Сохранить Открыть Выход

Рисунок В.17 — ПФ.1.11 Получение выходных кодов для настройки линейности датчика перемещения 2

Настройка линейности

Заводской номер блока
№123456

Данные блока

Датчик перемещения 2

Код напряжения 1: 3412

Код напряжения 2: 3684

К Δ напряжения: 272

Перемещение(ед.к): 1

Таблица градуировки

	Прямой ход	Обратный ход	Прямой ход	Обратный ход
-18мм	1			
-15мм				
-12мм				
-9мм				
-6мм				
-3мм				
0мм				
3мм				
6мм				
9мм				
12мм				
15мм				
18мм				

Управление

Старт Стоп Записать Отменить Очистить Сохранить Открыть Выход

Рисунок В.18 — ПФ.1.12 Запись выходного кода датчика перемещения 2 в таблицу градуировки

Настройка линейности

?

×

Заводской номер блока
№123456

Данные блока

Датчик перемещения 1

Код напряжения 12404

Код напряжения 22559

К Δ напряжения155

Перемещение(ед.к)2

Таблица градуировки

	Прямой ход	Обратный ход	Прямой ход	Обратный ход
-18мм	2	2	2	
-15мм	101	101	101	101
-12мм	203	203	203	203
-9мм	301	301	301	301
-6мм	404	404	404	404
-3мм	500	500	500	500
0мм	600	600	600	600
3мм	703	703	703	703
6мм	804	804	804	804
9мм	900	900	900	900
12мм	1001	1001	1001	1001
15мм	1102	1102	1102	1102
18мм	1200	1200	1200	1200

Управление

Старт

Стоп

Записать

Отменить

Очистить

Сохранить

Открыть

Выход

Рисунок В.19 — ПФ.1.13 Отмена последней записи выходного кода датчика перемещения 1

Настройка линейности

?

×

Заводской номер блока
№123456

Данные блока

Датчик перемещения 1

Код напряжения 12404

Код напряжения 22559

К Δ напряжения155

Перемещение(ед.к)2

Таблица градуировки

	Прямой ход	Обратный ход	Прямой ход	Обратный ход
-18мм				
-15мм				
-12мм				
-9мм				
-6мм				
-3мм				
0мм				
3мм				
6мм				
9мм				
12мм				
15мм				
18мм				

Управление

Старт

Стоп

Записать

Отменить

Очистить

Сохранить

Открыть

Выход

Рисунок В.20 — ПФ.1.14 Очищение таблицы градуировки от данных

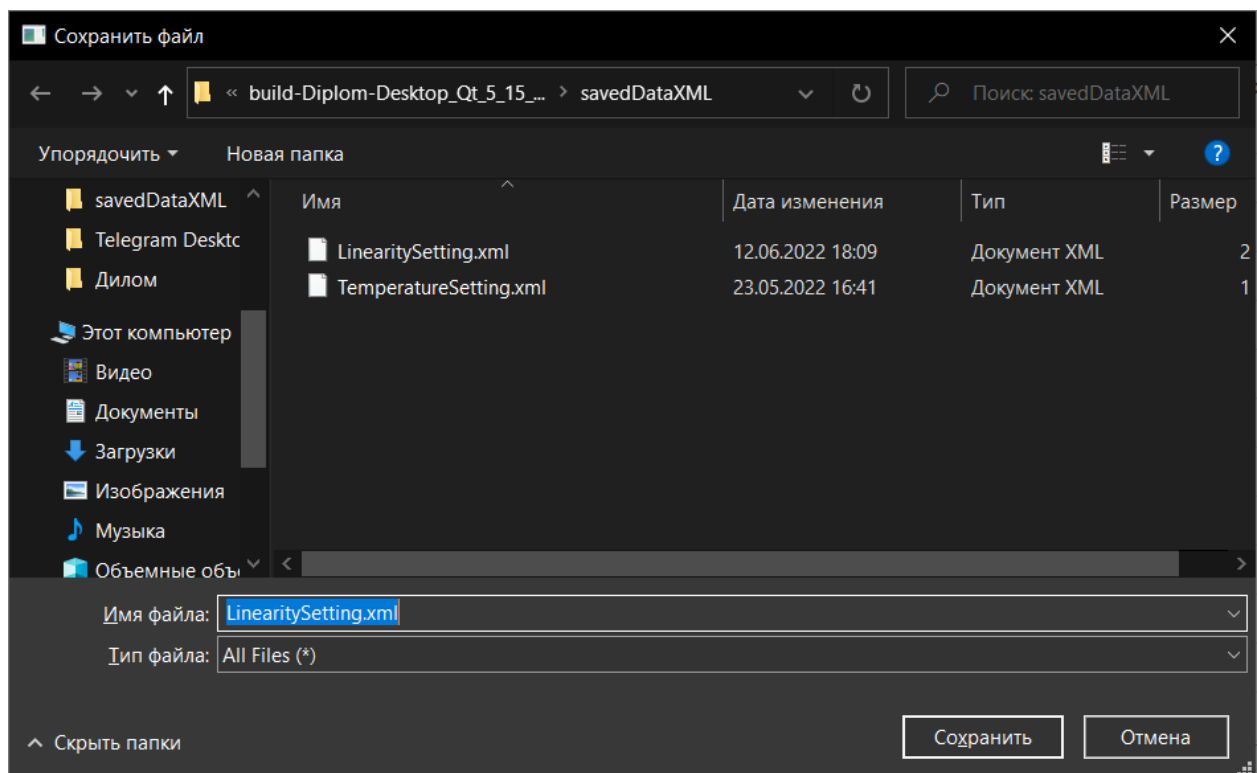


Рисунок В.21 — ПФ.1.15 Сохранение выходных данных в файл

LinearitySetting.xml – Блокнот

Файл Правка Формат Вид Справка

```
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <date>12.06.2022 18.13.40</date>
  <number>№123456</number>
  <table>
    <row>2</row>
    <row>2</row>
    <row>2</row>
    <row>2</row>
    <row>101</row>
    <row>101</row>
    <row>101</row>
    <row>101</row>
    <row>203</row>
    <row>203</row>
    <row>203</row>
    <row>203</row>
    <row>301</row>
    <row>301</row>
    <row>301</row>
    <row>301</row>
    <row>404</row>
    <row>404</row>
    <row>404</row>
    <row>404</row>
    <row>500</row>
    <row>500</row>
    <row>500</row>
    <row>500</row>
    <row>600</row>
    ...
  </table>
</data>
```

Стр 1, столб 1 100% UNIX (LF) UTF-8

Рисунок В.22 — ПФ.1.15 Сохранение выходных данных в файл

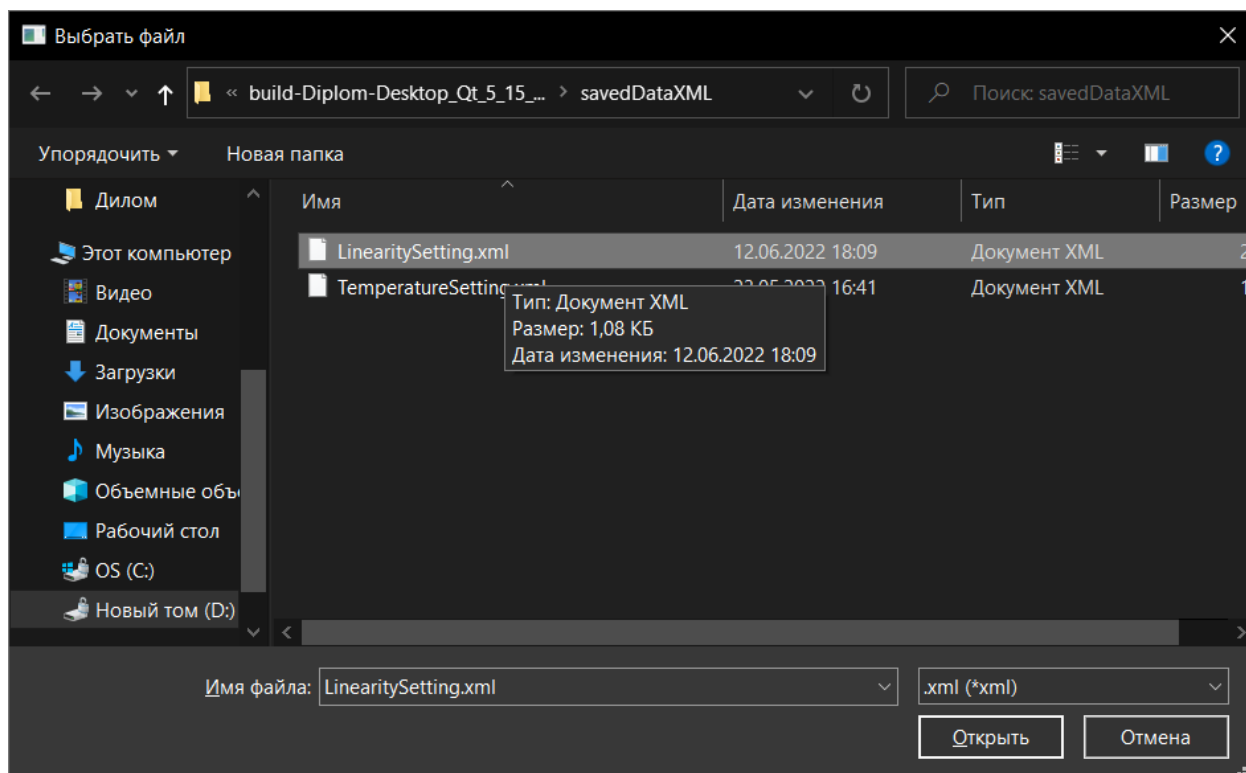


Рисунок В.23 — ПФ.1.16 Загрузка выходных данных в таблицу градуировки из файла

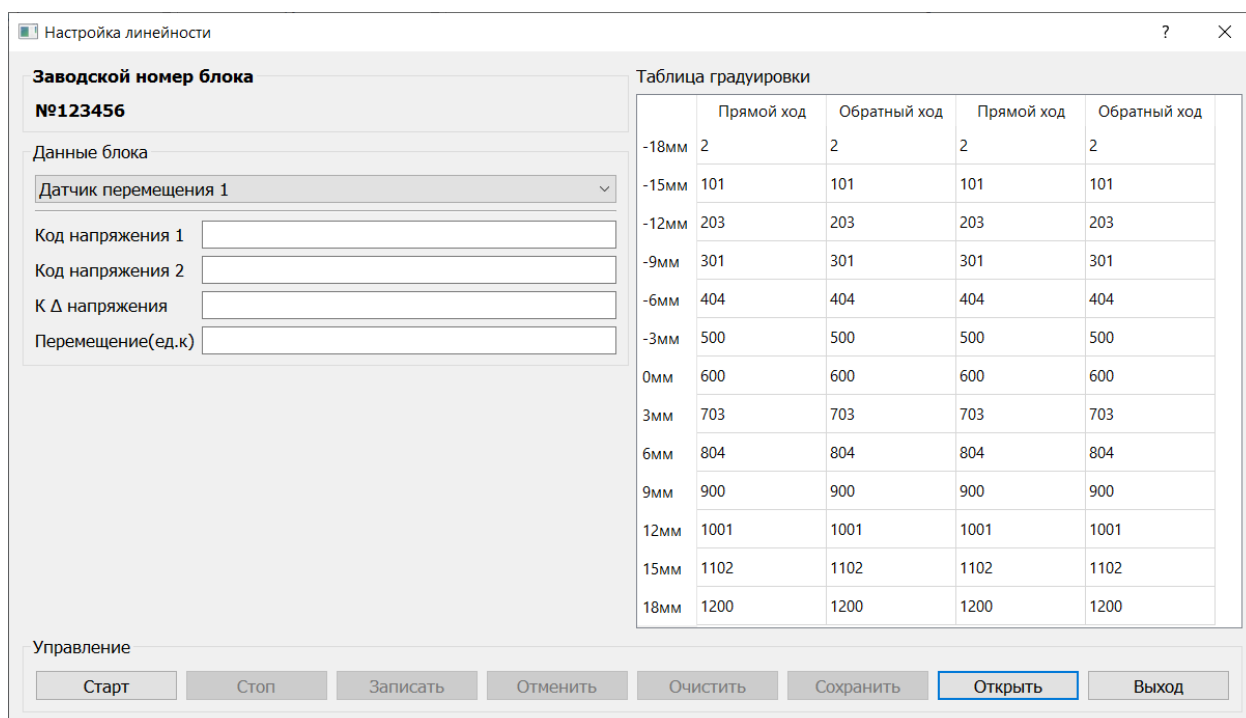


Рисунок В.24 — ПФ.1.16 Загрузка выходных данных в таблицу градуировки из файла

Настройка температуры

Заводской номер блока
№123456

Данные блока

Код температуры: 2418

Код темп. по ГОСТ: 0.10

МИТ: 0.22575

Таблица градуировки

	Код температуры	Код темп. по ГОСТ	МИТ
0			
25			
50			
100			
150			
200			
280			

Управление

Старт Стоп Записать Отменить Очистить Сохранить Открыть Выход

Рисунок В.25 — ПФ.1.17 Получить выходные коды для настройки датчика температуры

Настройка температуры

Заводской номер блока
№123456

Данные блока

Код температуры: 2418

Код темп. по ГОСТ: 0.10

МИТ: 0.22575

Таблица градуировки

	Код температуры	Код темп. по ГОСТ	МИТ
0	2418	0.10	0.22575
25			
50			
100			
150			
200			
280			

Управление

Старт Стоп Записать Отменить Очистить Сохранить Открыть Выход

Рисунок В.26 — ПФ.1.18 Запись выходного кода датчика температуры в таблицу градуировки

Настройка температуры

Заводской номер блока
№123456

Данные блока

Код температуры: 2493

Код темп. по ГОСТ: 280.10

МИТ: 280.22575

Управление

Старт Стоп Записать **Отменить** Очистить Сохранить Открыть Выход

Таблица градуировки

	Код температуры	Код темп. по ГОСТ	МИТ
0	2418	0.10	0.22575
25	2301	25.46	25.58575
50	2322	50.32	50.44575
100	2459	100.23	100.35575
150	2327	150.18	150.30575
200	2346	200.43	200.55575
280			

Рисунок В.27 — ПФ.1.19 Отмена последней записи выходного кода датчика температуры

Настройка температуры

Заводской номер блока
№123456

Данные блока

Код температуры: 2493

Код темп. по ГОСТ: 280.10

МИТ: 280.22575

Управление

Старт Стоп Записать Отменить Очистить **Сохранить** Открыть Выход

Таблица градуировки

	Код температуры	Код темп. по ГОСТ	МИТ
0			
25			
50			
100			
150			
200			
280			

Рисунок В.28 — ПФ.1.20 Очищение таблицы градуировки от данных

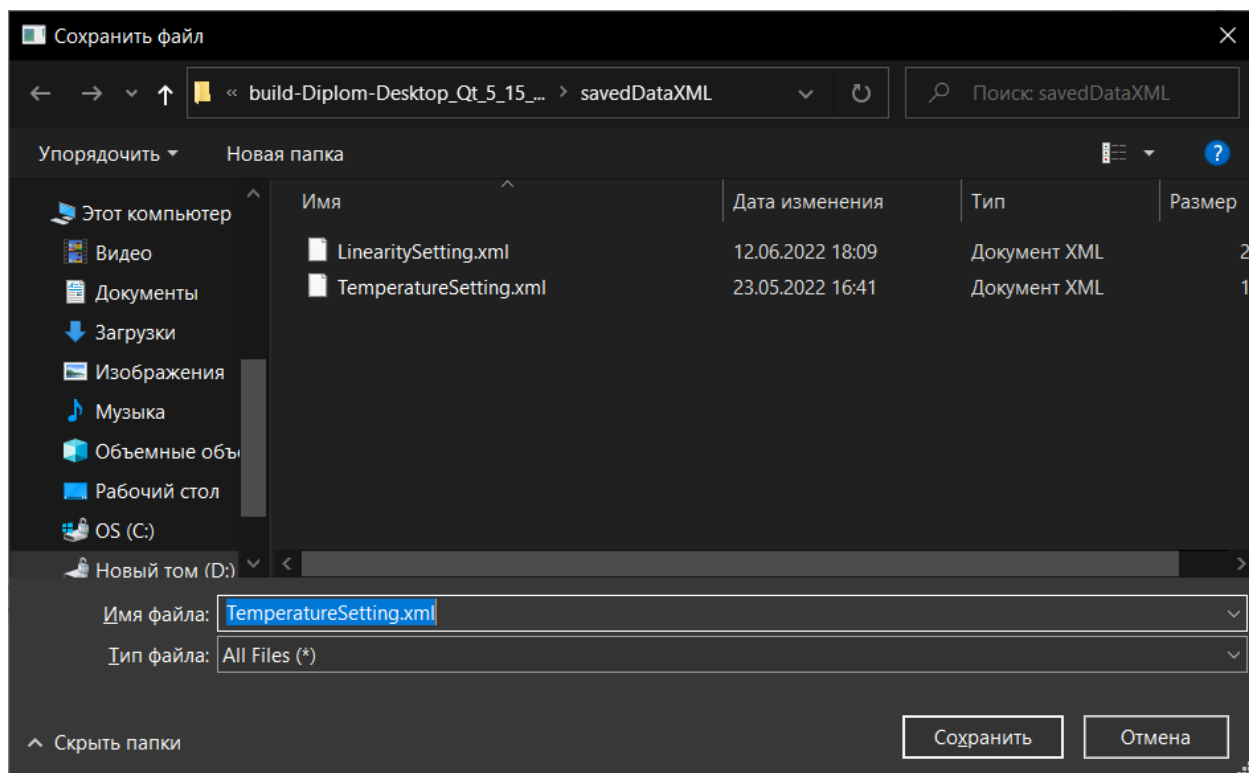


Рисунок В.29 — ПФ.1.21 Сохранение выходных данных в файл

TemperatureSetting.xml – Блокнот

ФайлПравкаФорматВидСправка

<?xml version="1.0" encoding="UTF-8"?>
<data>
<date>14.06.2022 20.00.24</date>
<number>№123456</number>
<table>
<row>2418</row>
<row>0.10</row>
<row>0.22575</row>
<row>2301</row>
<row>25.46</row>
<row>25.58575</row>
<row>2322</row>
<row>50.32</row>
<row>50.44575</row>
<row>2459</row>
<row>100.23</row>
<row>100.35575</row>
<row>2327</row>
<row>150.18</row>
<row>150.30575</row>
<row>2346</row>
<row>200.43</row>
<row>200.55575</row>
<row>2493</row>
<row>280.10</row>
<row>280.22575</row>
</table>
</data>

Стр 1, столб 1

100%

UNIX (LF)

UTF-8

Рисунок В.30 — ПФ.1.21 Сохранение выходных данных в файл

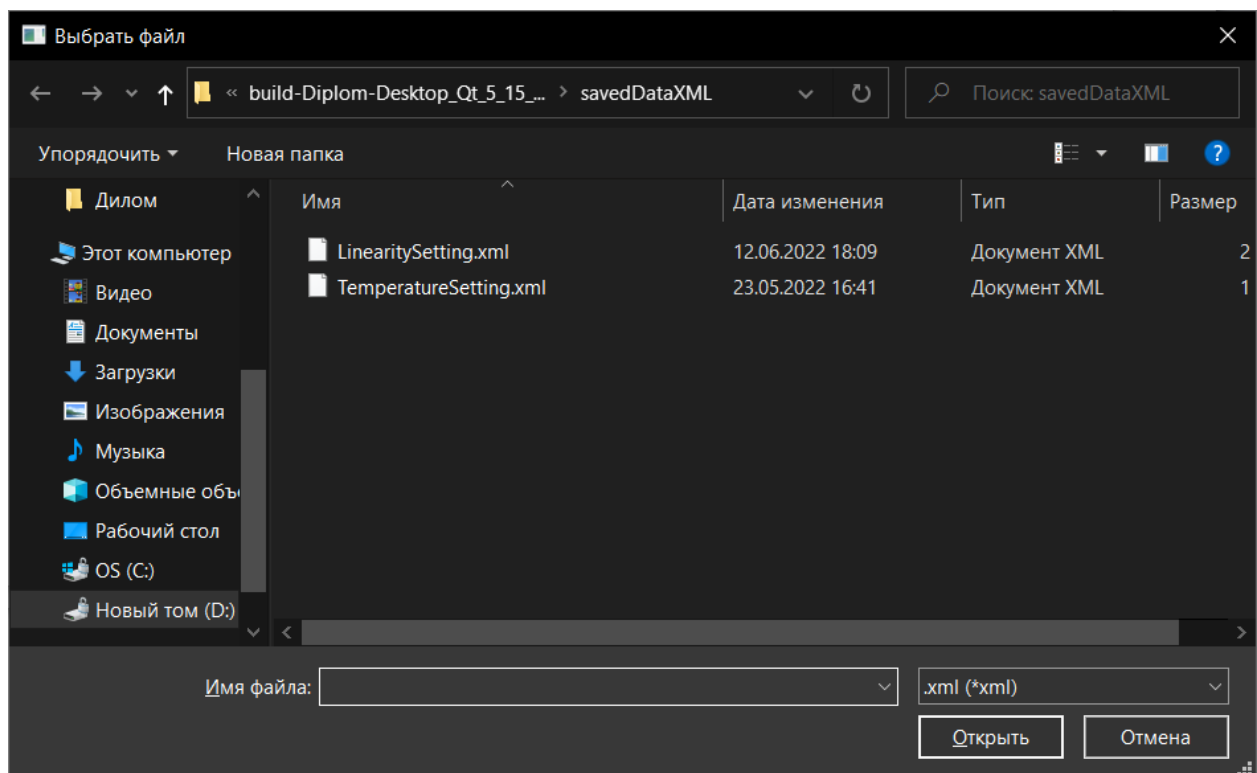


Рисунок В.31 — ПФ.1.22 Загрузка выходных данных в таблицу градуировки из файла

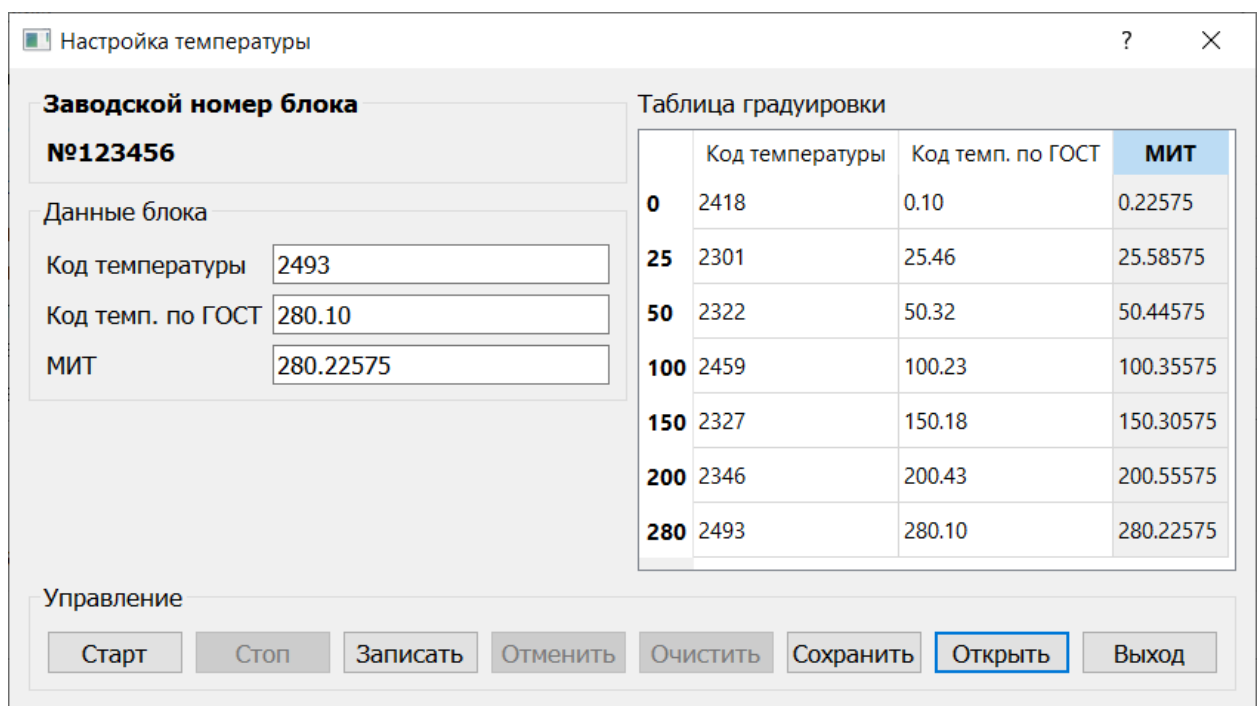


Рисунок В.32 — ПФ.1.22 Загрузка выходных данных в таблицу градуировки из файла

Контроль функционирования

Заводской номер блока
№123456

Датчик перемещения 1		Датчик перемещения 2		Датчик температуры	
Код напряжения 1	2433	Код напряжения 1	3412	Код АЦП	2205
Код напряжения 2	2628	Код напряжения 2	3684	Код температуры	2418
К Δ напряжения	195	К Δ напряжения	272	Температура по ГОСТ °C	0.10
Перемещение (ед.к)	2	Перемещение (ед.к)	1	Температура °C	0.22575
Перемещение (мм)	0.2	Перемещение (мм)	0.1		

Старт Стоп

Рисунок В.33 — ПФ.1.23 Проверка выходных кодов датчика перемещения 1 после настройки

Контроль функционирования

Заводской номер блока
№123456

Датчик перемещения 1		Датчик перемещения 2		Датчик температуры	
Код напряжения 1	2386	Код напряжения 1	3309	Код АЦП	2205
Код напряжения 2	2686	Код напряжения 2	3595	Код температуры	2418
К Δ напряжения	300	К Δ напряжения	286	Температура по ГОСТ °C	0.10
Перемещение (ед.к)	1200	Перемещение (ед.к)	1201	Температура °C	0.22575
Перемещение (мм)	120.0	Перемещение (мм)	120.1		

Старт Стоп

Рисунок В.34 — ПФ.1.24 Проверка выходных кодов датчика перемещения 1 после настройки

Контроль функционирования

Заводской номер блока
№123456

Датчик перемещения 1		Датчик перемещения 2		Датчик температуры	
Код напряжения 1	2433	Код напряжения 1	3412	Код АЦП	2205
Код напряжения 2	2628	Код напряжения 2	3684	Код температуры	2418
К Δ напряжения	195	К Δ напряжения	272	Температура по ГОСТ °C	0.10
Перемещение (ед.к)	2	Перемещение (ед.к)	1	Температура °C	0.22575
Перемещение (мм)	0.2	Перемещение (мм)	0.1		

Старт Стоп

Рисунок В.35 — ПФ.1.25 Проверка выходных кодов датчика перемещения 2 и датчика температуры после настройки

Контроль функционирования

Заводской номер блока
№123456

Датчик перемещения 1		Датчик перемещения 2		Датчик температуры	
Код напряжения 1	2386	Код напряжения 1	3309	Код АЦП	2205
Код напряжения 2	2686	Код напряжения 2	3595	Код температуры	2418
К Δ напряжения	300	К Δ напряжения	286	Температура по ГОСТ °C	0.10
Перемещение (ед.к)	1200	Перемещение (ед.к)	1201	Температура °C	0.22575
Перемещение (мм)	120.0	Перемещение (мм)	120.1		

Старт Стоп

Рисунок В.36 — ПФ.1.26 Проверка выходных кодов датчика перемещения 2 после настройки

Контроль функционирования

Заводской номер блока
№123456

Датчик перемещения 1		Датчик перемещения 2		Датчик температуры	
Код напряжения 1	2386	Код напряжения 1	3309	Код АЦП	2205
Код напряжения 2	2686	Код напряжения 2	3595	Код температуры	2418
К Δ напряжения	300	К Δ напряжения	286	Температура по ГОСТ °C	0.10
Перемещение (ед.к)	1200	Перемещение (ед.к)	1201	Температура °C	0.22575
Перемещение (мм)	120.0	Перемещение (мм)	120.1		

Старт Стоп

Рисунок В.37 — ПФ.1.27 Проверка выходных кодов датчика температуры после настройки

Сохранение файла Excel

« build-Diplom-Desktop_Qt_5_15_2_MinG... » Reports

Поиск: Reports

Упорядочить Новая папка

Имя Дата изменения Тип Размер

Нет элементов, удовлетворяющих условиям поиска.

Имя файла: Заводской-№№123456_Датчик-перемещения-1_12.06.2022_18.23.27.xlsx

Тип файла: Файл Excel (*.xlsx)

Скрыть папки Сохранить Отмена

Рисунок В.38 — ПФ.1.28 Создание файла отчета с результатами измерения датчика перемещения 1

Заводской №№123456. Данные перемещения 2.12.06.2022.18.15.31.xlsx - Microsoft Excel (Сбой активации продукта)

Главная Вставка Разметка страницы Формулы Данные Рецензирование Вид

Times New Roma 12

Буфер обмена Вставить Вырезать Копировать Формат по образцу

Шрифт

Выравнивание

Общий

Условное форматирование

Стили

Вставка Удалить Формат

Автоподсказка

Заполнить

Сортировка

Найти и выделить

Рецензирование

И22 1201

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27

А В С D E F G H I J K L M N O P Q R

Результаты измерений выходного сигнала датчика перемещения 2

Заводской номер	Номер градуировочной точки	Величина перемещения, мм	Выходной сигнал, единиц			
			1 цикл		2 цикл	
			прямой ход	обратный ход	прямой ход	обратный ход
№123456	1	0	1	300	603	902
	2	3	1	300	603	1000
	3	6	1	300	701	1000
	4	9	1	404	701	1000
	5	12	103	404	701	1000
	6	15	103	404	701	1104
	7	18	103	404	800	1104
	8	21	103	502	800	1104
	9	24	202	502	800	1104
	10	27	202	502	800	1201
	11	30	202	502	902	1201
	12	33	202	603	902	1201
	13	36	300	603	902	1201

Лист1

Готово

Рисунок В.41 — ПФ.1.29 Создание файла отчета с результатами измерения датчика перемещения 2

Сохранение файла Excel

« build-Diplom-Desktop_Qt_5_15_2_MinG... » Reports

Поиск: Reports

Упорядочить Новая папка

Этот компьютер

Видео

Документы

Загрузки

Изображения

Музыка

Объемные объ...

Рабочий стол

OS (C:)

Новый том (D:)

Имя

Дата изменения

Тип

Размер

Нет элементов, удовлетворяющих условиям поиска.

Имя файла: Заводской-№№123456_Датчик-температуры-_12.06.2022_18.15.31

Тип файла: Файл Excel(*.xlsx)

Скрыть папки

Сохранить Отмена

Рисунок В.42 — ПФ.1.30 Создание файла отчета с результатами измерения датчика температуры

Заводской №№123456 Датчик температуры_12.06.2022_18.15.31.xlsx - Microsoft Excel (Свой активации продукта)

Главная Вставка Разметка страницы Формулы Данные Рецензирование Вид

Буфер обмена Вставить Вырезать Копировать Формат по образцу Шрифт Выравнивание

Общий Условное форматирование Форматировать как таблицу Стили Вставить Удалить Формат

Автосумма Заполнить Сортировка Найти и выделить

Результаты проверки измерений выходного сигнала датчика температуры

Заводской номер	Номер градуировочной точки, j	Значение температуры T_i , °C	Выходной код, единиц	Выходной сигнал, °C	
				Датчик температуры	МНП
	1	(0±1)	2418	0.10	0.22575
	2	(25±2)	2301	25.46	25.58575
	3	(50±2)	2322	50.32	50.44575
	4	(100±2)	2459	100.23	100.35575
	5	(150±2)	2327	150.18	150.30575
	6	(200±3)	2346	200.43	200.55575
	7	(280±3)	2493	280.10	280.22575

Лист1

Рисунок В.43 — ПФ.1.30 Создание файла отчета с результатами измерения датчика температуры

Приложение Г

(Обязательное)

Программное обеспечение для настройки блока измерения подачи ракетного
топлива

Презентация

					ПГУ 1.090301.12.001 ПЗ	Лист
						102
Изм.	Лист	№ докум.	Подпись	Дата		

Программное обеспечение настройки блока измерения подачи ракетного топлива

Выполнил:
студент группы 18BB2

Емелин С.В.

Руководитель:
к.т.н., доцент, кафедры ВТ Карамышева Н.С.

Цель проекта

- Разработать программное обеспечение для
настройки блока измерения подачи ракетного
топлива

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.12.001 ПЗ

Лист

103

Актуальность

Актуальность разработки ПО обусловлена:

- ▶ широким использованием объектов ракетно-космической техники
- ▶ стремительным развитием ракетно-космической техники
- ▶ потребностью космической индустрии в программных продуктах



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.12.001 ПЗ

Лист

104

Постановка задачи

- ▶ прием данных в режиме первоначальной настройки
- ▶ градуировка датчиков перемещения и температуры
- ▶ сохранение данных градуировки в файл с расширением *XML*
- ▶ просмотр данных ранее сделанных градуировок датчиков перемещения и температуры
- ▶ контроль выходного кода датчиков
- ▶ возможность формирования отчётов в файл формата *XLSX* по данным градуировки каждого датчика блока

Используемые программные средства

- ▶ Qt Framework
- ▶ Qt Widgets
- ▶ Qt Creator IDE
- ▶ Язык программирования C++

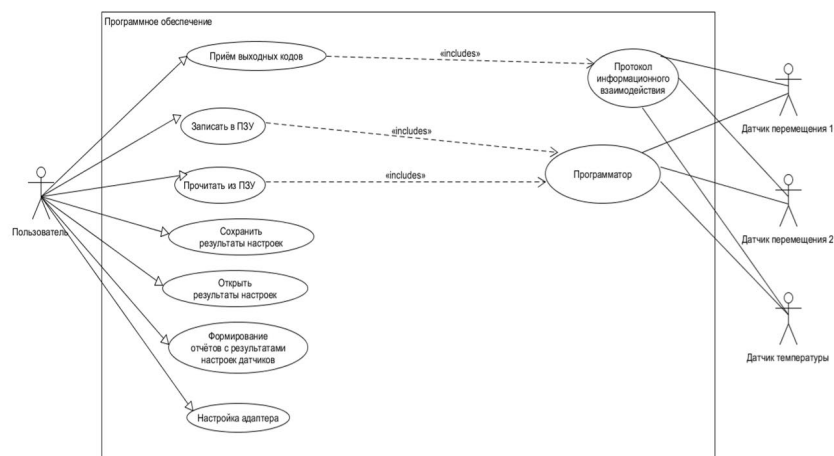
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.12.001 ПЗ

Лист

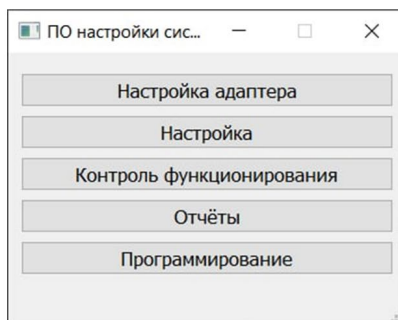
105

UML – диаграмма вариантов использования



Графический интерфейс ПО

► Окно главного меню



Графический интерфейс ПО

▸ Окно настройки адаптера

Настройка адаптера

Адрес устройства 0

Канал Основной

Частота опроса 10 Гц

Открыть файл

Графический интерфейс ПО

▸ Окно программирования

Программирование

Запись ПО

Запрограммировать датчик перемещения 1

Запрограммировать датчик перемещения 2

Запрограммировать датчик температуры

Запись констант

Датчик перемещения 1

№ N нон. Код диапазона

Записать

Датчик перемещения 2

№ N нон. Код диапазона

Записать

Датчик температуры

№ N нон. Код диапазона

Записать

Работа с памятью

Датчик перемещения 1

Параметр	Адрес	Тип	Байт	Значение
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

Открыть Сохранить Записать в ПЗУ Прочитать из ПЗУ

Датчик перемещения 2

Датчик температуры

Графический интерфейс ПО

▸ Окно настройки

The 'Настройка' window contains the following fields and controls:

- Заводской номер блока**: №123456
- Датчик перемещения 1**:
 - Код напряжения 1
 - Код напряжения 2
 - К Δ напряжения
 - Перемещение (ед.к.)
 - Перемещение (мм)
 - №0
 - № ном.
 - Код диапазона
- Датчик перемещения 2**:
 - Код напряжения 1
 - Код напряжения 2
 - К Δ напряжения
 - Перемещение (ед.к.)
 - Перемещение (мм)
 - №0
 - № ном.
 - Код диапазона
- Датчик температуры**:
 - Код АЦП
 - Код температуры
 - Температура по ГОСТ °C
 - Температура °C
- Buttons**: Старт, Стоп
- Navigation**: Настройка линейности, Настройка температуры

Графический интерфейс ПО

▸ Окно настройки линейности

The 'Настройка линейности' window contains the following fields and controls:

- Заводской номер блока**: №123456
- Данные блока**: Датчик перемещения 1 (dropdown menu)
- Код напряжения 1**
- Код напряжения 2**
- К Δ напряжения**
- Перемещение(ед.к)**
- Таблица градуировки**:

	Прямой ход	Обратный ход	Прямой ход	Обратный ход
-18мм				
-15мм				
-12мм				
-9мм				
-6мм				
-3мм				
0мм				
3мм				
6мм				
9мм				
12мм				
15мм				
18мм				
- Управление**: Старт, Стоп, Записать, Отменить, Очистить, Сохранить, Открыть, Выход

Графический интерфейс ПО

▸ Окно настройки температуры

Настройка температуры

Заводской номер блока
№123456

Данные блока

Код температуры

Код темп. по ГОСТ

МИТ

Управление

Старт Стоп Записать Отменить Очистить Сохранить Открыть Выход

Таблица градуировки

	Код температуры	Код темп. по ГОСТ	МИТ
0			
25			
50			
100			
150			
200			
280			

Графический интерфейс ПО

▸ Окно контроля функционирования

Контроль функционирования

Заводской номер блока
№123456

Датчик перемещения 1

Код напряжения 1

Код напряжения 2

К Δ напряжения

Перемещение (ед.к)

Перемещение (мм)

Датчик перемещения 2

Код напряжения 1

Код напряжения 2

К Δ напряжения

Перемещение (ед.к)

Перемещение (мм)

Датчик температуры

Код АЦП

Код температуры

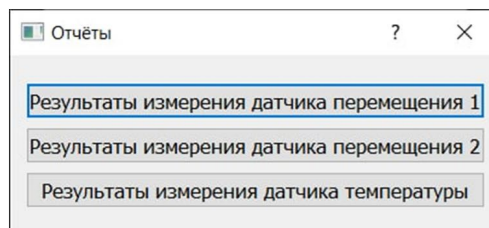
Температура по ГОСТ °C

Температура °C

Старт Стоп

Графический интерфейс ПО

▸ Окно отчетов



Результаты тестирования

▸ Тестирование сохранения файла настройки линейности

```
LinearitySetting.xml – Блокнот
Файл  Правка  Формат  Вид  Справка
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <date>12.06.2022 18.13.48</date>
  <number>№123456</number>
  <table>
    <row>2</row>
    <row>2</row>
    <row>2</row>
    <row>2</row>
    <row>101</row>
    <row>101</row>
    <row>101</row>
    <row>101</row>
    <row>203</row>
    <row>203</row>
    <row>203</row>
    <row>203</row>
    <row>301</row>
    <row>301</row>
    <row>301</row>
    <row>301</row>
    <row>404</row>
    <row>404</row>
    <row>404</row>
    <row>404</row>
    <row>500</row>
    <row>500</row>
    <row>500</row>
    <row>500</row>
  </table>
</data>
```

Результаты тестирования

- Тестирование сохранения файла настройки температуры

```
TemperatureSetting.xml – Блокнот
Файл Правка Формат Вид Справка
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <date>14.06.2022 20.00.24</date>
  <number>№123456</number>
  <table>
    <row>2418</row>
    <row>0.10</row>
    <row>0.22575</row>
    <row>2301</row>
    <row>25.46</row>
    <row>25.58575</row>
    <row>2322</row>
    <row>50.32</row>
    <row>50.44575</row>
    <row>2459</row>
    <row>100.23</row>
    <row>100.35575</row>
    <row>2327</row>
    <row>150.18</row>
    <row>150.30575</row>
    <row>2346</row>
    <row>200.43</row>
    <row>200.55575</row>
    <row>2493</row>
    <row>280.10</row>
    <row>280.22575</row>
  </table>
</data>
```

Результаты тестирования

- Проверка выходных кодов датчика температуры после настройки

Контроль функционирования

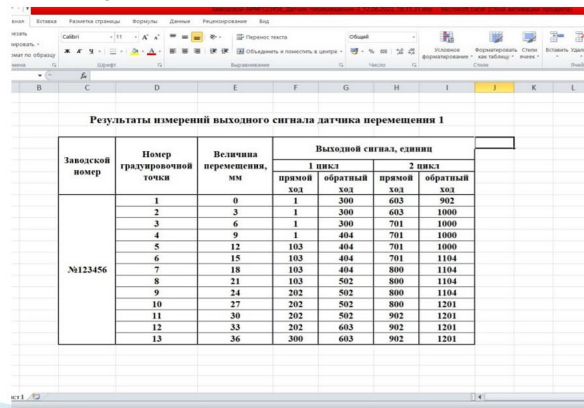
Заводской номер блока
№123456

Датчик перемещения 1		Датчик перемещения 2		Датчик температуры	
Код напряжения 1	2386	Код напряжения 1	3309	Код АЦП	2205
Код напряжения 2	2686	Код напряжения 2	3595	Код температуры	2418
К Д напряжения	300	К Д напряжения	286	Температура по ГОСТ °C	0.10
Перемещение (ед.к)	1200	Перемещение (ед.к)	1201	Температура °C	0.22575
Перемещение (мм)	120.0	Перемещение (мм)	120.1		

Старт Стоп

Результаты тестирования

- Проверка создания отчета с данными настройки датчика перемещения



Результаты измерений выходного сигнала датчика перемещения 1

Заводской номер	Номер градуировочной точки	Величина перемещения, мм	Выходной сигнал, единиц			
			1 канал		2 канал	
			прямой хол	обратный хол	прямой хол	обратный хол
N123456	1	0	1	300	603	902
	2	3	1	300	603	1000
	3	6	1	300	701	1000
	4	9	1	404	701	1000
	5	12	103	404	701	1000
	6	15	103	404	701	1104
	7	18	103	404	800	1104
	8	21	103	502	800	1104
	9	24	202	502	800	1104
	10	27	202	502	800	1201
	11	30	202	502	902	1201
	12	33	202	603	902	1201
	13	36	300	603	902	1201

Заключение

- В рамках данной выпускной квалификационной работы было разработано программное обеспечение для настройки блока измерения подачи ракетного топлива. Разработанное ПО предоставляет возможность записывать внутреннее программное обеспечение в микроконтроллер каждого вторичного преобразователя, получать данные с блока вторичных преобразователей, проводить калибровку, сохранять данные калибровки в файл с расширением *XML*, считывать параметры из ПЗУ, записывать параметры в ПЗУ, формировать отчёты по настройке датчиков перемещения и температуры.

Спасибо за внимание!



					ПГУ 1.090301.12.001 ПЗ	Лист
						113
Изм.	Лист	№ докум.	Подпись	Дата		