МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество «Казахский национальный исследовательский технический университет имени К.И.Сатпаева»

SATBAYEV UNIVERSITY

Институт Автоматики и информационных технологий

Кафедра Робототехники и технических средств автоматики

6В07111 – Робототехника и мехатроника

Акимов Максим Александрович

Разработка системы управления автоматизированной системой охлаждения с использованием ПЛК на базе семейства Arduino

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

6В07111 – Робототехника и мехатроника

Алматы 2023

Некоммерческое Акционерное Общество «Казахский Национальный Исследовательский Технический Университет имени К.И.Сатпаева»

SATBAYEV UNIVERSITY

Институт Автоматики и информационных технологий

Кафедра «Робототехники и технических средств автоматики»

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой РТиТСА
кандидат технических наук
_____Ожикенов К. А.
«26» Мая 2023 г.

ДИПЛОМНЫЙ ПРОЕКТ

Тема: «Разработка системы управления автоматизированной системой охлаждения с использованием ПЛК на базе семейства Arduino»

по специальности 6B07111 – Робототехника и мехатроника

Выполнил Акимов Максим

Рецензент
phd, ассоциированный профессор
_____ Балбаев Г.К.
подпись ФИО
«25» май 2023 ж.

Научный руководитель
Магистр технических наук,
старший преподаватель
_____ Кальменов Е.Т.
«25» май 2023 ж.

Алматы 2023

Некоммерческое Акционерное Общество «Казахский Национальный Исследовательский Технический Университет имени К.И.Сатпаева»

SATBAYEV UNIVERSITY

Институт Автоматики и информационных технологий

Кафедра «Робототехники и технических средств автоматики»

6В07111 – Робототехника и мехатроника

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой РТиТСА
кандидат технических наук
_____ Ожикенов К. А.
«26» мая 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся Акимову Максиму Александровичу
Тема: Разработка системы управления автоматизированной системой охлаждения с использованием ПЛК на базе семейства Arduino
Утверждена приказом Ректора Университета №408пп от «23» май 2023 г.
Срок сдачи законченной работы «21» декабря 2022 г.
Исходные данные к дипломному проекту: Arduino UNO, Ethernet Shield
Перечень подлежащих разработке вопросов в дипломном проекте:
а) Исследовать проблему использования ПЛК в производстве
б) Изучить ПЛК, представляющие конкурентное преимущество и протоколы связи
в) Приступить к разработке PLCDuino, подготовить описание к работе
г) Смоделировать элементы для дальнейшего использования
д) Рассчитать необходимые параметры
Перечень графического материала (с точным указанием обязательных чертежей):
Представлены 18 слайдов презентации работы
Рекомендуемая основная литература: *из* 24 *наименований* 24

ГРАФИК
подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Исследовательская часть	24.01.2023	Выполнено
Теоретическая часть	29.02.2023	Выполнено
Практическая часть	18.03.2023	Выполнено
Специальная часть	25.03.2023	Выполнено

Подписи
консультантов и нормоконтролера на законченный проект с указанием относящихся к ним разделов проекта

Наименования разделов	Консультанты, И.О.Ф. (уч. степень, звание)	Дата подписания	Подпись
Нормоконтролер	Ермухамед Т.К., магистр технических наук, старший преподаватель	25.05.2023	
Основная часть	Ермухамед Т.К., магистр технических наук, старший преподаватель	25.05.2023	
Расчетная часть	Ермухамед Т.К., магистр технических наук, старший преподаватель	25.05.2023	

Научный руководитель _____ Кальменов Е.Т.

Задание принял к исполнению обучающийся _____ Акимов М.А.

Дата «25» мая 2023 г.

СОДЕРЖАНИЕ

АҢДАТПА

Бұл дипломдық жұмыс Arduino отбасы (PLCDuino) негізіндегі PLC көмегімен автоматтандырылған салқындату жүйесін басқару жүйесін әзірлеуге арналған. Бұл басқару жүйесі Arduino және Ethernet Shield кеңеюіне негізделген PLC-ден тұрады және жұмыс қабілеттілігін көрсету үшін 6 басқарылатын салқындатқыштар, температура сенсорлары, релелер және жарық индикаторларының негізін қамтитын сервер салқындатқыш бөлмесінің макеті салынды. .

Бұл жобаның негізгі мақсаты - қымбат өнеркәсіптік автоматтандыру жабдықтарын сатып алмай-ақ көп нәрсені модельдеу және біріктіру. PLCDuino негізгі мақсаты орта және тұрмыстық өнеркәсіпте автоматтандыру процесін орталықтандырылған басқаруды енгізу болып табылады. Бұдан басқа, жаһандық мақсаттар туралы айтатын болсақ, бұл жоба шағын және орта бизнеске үлкен шығынға ұшырамай, қолжетімді бағамен жабдықты алуға мүмкіндік береді, сонымен қатар адам әсерін жеделдету және азайту арқылы Қазақстандағы өндіріс сапасын ілгерілетеді.

Жұмыстың теориялық бөлімінде мынадай іргелі сұрақтар қарастырылады: PLC неліктен өндіріс үшін соншалықты маңызды? Қандай байланыс протоколы таңдалды және неге? Бұл жүйені бұзудан қалай қорғауға болады? Бұл жобаның мақсаты қандай? Сонымен қатар, параметрлік зерттеулер мен математикалық есептеулер де ұсынылған, оларға қатысты таңдау ағымдағы жұмыста қолданылатын салқындату түрінің пайдасына жасалды.

Тәжірибелік бөлімде негізгі екпін құрылғыны ең негізгіден прототипке дейін әзірлеуге аударылады, оған мыналар кіреді: математикалық есептеулер, электрлік схемалар, 3D модельдеу, өндіріс технологиясы, техникалық іске асыру, бағдарламалау, топология және т.б.

Бүкіл жүйе толығымен жұмыс істейді, жөндеуден өткен және жаппай өндіруге және өнеркәсіпте енгізуге жатады.

АННОТАЦИЯ

Данная дипломная работа посвящена разработке системы управления автоматизированной системой охлаждения с использованием ПЛК на базе семейства Arduino (PLCDuino). Состоит данная система управления из самого ПЛК на базе Arduino и Ethernet Shield расширения, а для демонстрации работоспособности был построен макет серверной чиллерной комнаты, который включает в себя основу из 6-ти управляемых куллеров, температурных датчиков, реле и световых индикаторов.

Основная задача данного проекта состоит в возможности моделировать и интегрировать большинство вещей, не покупая при этом дорогое оборудование для промышленной автоматизации. В то время как основная цель PLCDuino - это внедрение централизованного управления процессом автоматизации на средних и домашних производствах. Помимо этого, говоря о более глобальных целях, данных проект даст возможность малому и среднему бизнесу обзавестись оборудованием по доступной цене, не принося огромные убытки, а также продвинет качество производств в Казахстане, через ускорение и уменьшения человеческого влияния.

В исследовательской и теоретической частях работы рассмотрены такие фундаментальные вопросы, как: Почему ПЛК так важен для производств? Какой протокол связи был выбран и почему? Как защитить данную систему от взлома? Какую цель преследует данный проект? Помимо этого, также представлены параметрические исследования и математические расчеты, относительно которых был сделан выбор в пользу типа охлаждения, что используется в актуальной работе.

В практической же части основной упор идет в разработку устройства с самых основ до прототипа, что включает в себя: математические расчеты, электрические принципиальные схемы, 3D моделирование, технология изготовления, техническая реализация, программирование, топология и прочее.

Вся система полностью функциональная, отлажена и подлежит массовому производству и внедрению в промышленность.

ABSTRACT

This thesis is devoted to the development of a control system for an automated cooling system using a PLC based on the Arduino family (PLCDuino). This control system consists of the PLC itself based on Arduino and Ethernet Shield expansion, and to demonstrate the operability, a mock-up of a server chiller room was built, which includes a base of 6 controlled coolers, temperature sensors, relays and light indicators.

The main goal of this project is to be able to model and integrate most things without buying expensive industrial automation equipment. While the main goal of PLCDuino is the introduction of centralized control of the automation process in medium and home industries. In addition, speaking of more global goals, this project will enable small and medium-sized businesses to acquire equipment at an affordable price without causing huge losses, and will also promote the quality of production in Kazakhstan, through acceleration and reduction of human impact.

In the theoretical part of the work, such fundamental questions are considered as: Why is the PLC so important for production? What communication protocol was chosen and why? How to protect this system from hacking? What is the purpose of this project? In addition, parametric studies and mathematical calculations are also presented, in relation to which the choice was made in favor of the type of cooling that is used in the current work.

In the practical part, the main emphasis is on the development of the device from the very basics to the prototype, which includes: mathematical calculations, electrical circuit diagrams, 3D modeling, manufacturing technology, technical implementation, programming, topology, and more.

The whole system is fully functional, debugged and subject to mass production and implementation in industry.

ВВЕДЕНИЕ

На сегодняшний день невозможно представить ни одно крупное производство или предприятие без использования систем автоматизации. Причина этому - сложность производственных процессов, которое делает ручное управление невозможным, в то время как системы автоматизации более рентабельны, чем наем обслуживающего персонала, поскольку они быстрее и надежнее. В настоящее время автоматизация необходима для любой сферы промышленности - от строительства до производства любых товаров в огромных количествах. Несмотря на разнообразие сфер и областей применения, все системы автоматизации имеют один и тот же фундаментальный принцип и схожую структуру с программируемым логическим контроллером в основе (ПЛК), выступающим в качестве основной логической единицы системы.

В данной работе рассмотрены основы работы контроллера ПЛК, его структурные и функциональные компоненты, а также применение в промышленном производстве и обозначены задачи, которые может решать ПЛК для различных отраслей.

Цель проекта: как уже было сказано выше [АННОТАЦИЯ], это внедрение централизованного управления процессом автоматизации на средних и домашних производствах. Если углубиться в суть проблемы, то как известно, в Казахстане слабо развито промышленное производство, причиной этому является дороговизна покупаемого зарубежного оборудования для автоматизации и ускорения производства. Примером такого оборудования является – ПЛК ОВЕН [2.1], что на рынке оценивается в 575.000 тенге, который не могуь позволить себе многие малые и средние прозводства. Именно это и сподвигло создать дешевый и доступный аналог программируемого логического контроллера (ПЛК) на основе всеми известной плате Arduino.

Актуальность: было проведено аналитическое исследование, какие промышленные структуры в Казахстане могут быть потенциально заинтересованы в данном продукте.[2.9] Затем, был фактический опрос этих самых производств на предмет сотрудничества и установки у себя данной системы PLCDuino, что показало плодотворные результаты. В конечном итоге, система управления автоматизированной системой охлаждения с использованием ПЛК на базе семейства Arduino была представлена на стартап конкурсе «Open Space» [2.9], где этот проект прошел в финал, получил фидбэк от спонсоров и менторов, что способствовало заинтересованности в дальнейшей поддержке данной системы.

Новизна: данная идея не является инноваторской, попытки создать полноценный ПЛК из платы Arduino уже были, в таких странах как: Индия, Испания и Америка. Но не один из вышеперечисленных инноваторов не

решился продвигать эту идею вплоть до полноценного прототипа, тем самым, идею внедрить PLCDuino в полноценное производство можно считать инноваторской. Что касательно отечественного рынка, данный новаторский проект не может иметь аналогов по цене/качеству, так как эта технология появилась относительно недавно и реальных исследований не проводилось не в одной из стран мира.

1 ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

1.1 Общие понятия об программируемом логическом контроллере

Современные промышленные фабрики и прочие отрасли производства все больше внимания уделяют автоматизации производственных процессов, снижению потребности в человеческом участии в принятии решений, облегчению работы сотрудников и, самое главное, повышению производительности. Автоматизация является основным фактором развития промышленности и требует использования современных технологий для управления различными этапами производства и упрощения жизни людей. Программируемые логические контроллеры (ПЛК) были разработаны, чтобы помочь людям контролировать и мониторить объекты. Эти контроллеры имеют множество входов и выходов для подключения периферийных устройств, накопителей, систем сбора данных, человеко-машинных интерфейсов и многого другого. В настоящее время программируемые логические контроллеры используются в различных отраслях, таких как энергетика, связь, химическая промышленность, горнодобывающая промышленность, транспортировка нефти и газа, системы безопасности, коммунальные услуги, автоматизация складов, производство продуктов питания, транспорт и строительство. [2]

Программируемые контроллеры широко используются в различных отраслях промышленности, так как их можно комбинировать с множеством дополнительных компонентов для создания многофункциональной системы управления. [2] Эти контроллеры выгодны для систем управления предприятием, систем развития производства и других механизмов или настроек, так как обеспечивают быстрый экономический эффект, улучшают качество производства, снижают потребность в ручном труде.

Однако, перед их использованием необходимо проверить точность программ ПЛК. Этот процесс можно выполнить с помощью средств проверки моделей. Обычно это длительная и дорогостоящая процедура, для которой требуется эксперт в данной области, знакомый с системой и различными инструментами проверки модели, которые могут оценить код в контроллере. [1] Кроме того, эта проверка часто требует преобразования кода ПЛК в язык, распознаваемый средством проверки модели, что может изменить поведение ПЛК.

Рисунок 1.1.1 – Промышленный ПЛК 6ED1052-1MD00-0BA6 Siemens для простых задач автоматизации

По своей структуре, программируемый логический контроллер — это микропроцессорное устройство, которое может собирать, преобразовывать, обрабатывать, хранить данные и генерировать управляющие команды. Он имеет ограниченное количество входов и выходов, подключенных к датчикам, ключам и исполнительным устройствам, связанным с управляемым объектом. Как правило, предназначен для работы исключительно в режиме реального времени.

Рисунок 1.1.2 – Принцип работы ПЛК в виде визуальной схемы

ПЛК оснащен периферийными входами для взаимодействия с окружающей средой, которая может принимать дискретные сигналы, такие как состояние включения/выключения (например, определение конечного положения энкодера) или непрерывные аналоговые сигналы от датчиков (например, температура, давление, уровень и т. д.). Со стороны выхода, ПЛК имеет выходные периферийные устройства, подключенные к элементам, которые управляют автоматизированным процессом, либо изображающий дискретный сигнал управления включением/выключением. Пример таких устройств - контактор двигателя, катушка клапана, электромагнитное реле и сигнальная лампа.

Центральный процессор (Рисунок 1.1.2), является ядром системы ПЛК, он действует как логика управления между входами и выходами, определяет состояние выходов на основе входов, чтобы поддерживать минимальное отличие от желаемого состояния устройства. Программист создает программный алгоритм, чтобы определить, как ПЛК будет реагировать на изменения входных сигналов. Эта программа хранится в памяти ПЛК, а операционная система ПЛК обеспечивает ее циклическое повторное выполнение.

Что касательно архитектуры ПЛК, ЦП (центральный процессор) состоит из блока управления и процессора. Блок управления ЦП управляет связью между различными аппаратными компонентами ПЛК, в то время как процессор обрабатывает все вычисления и выполнение программы (например, релейную логику).

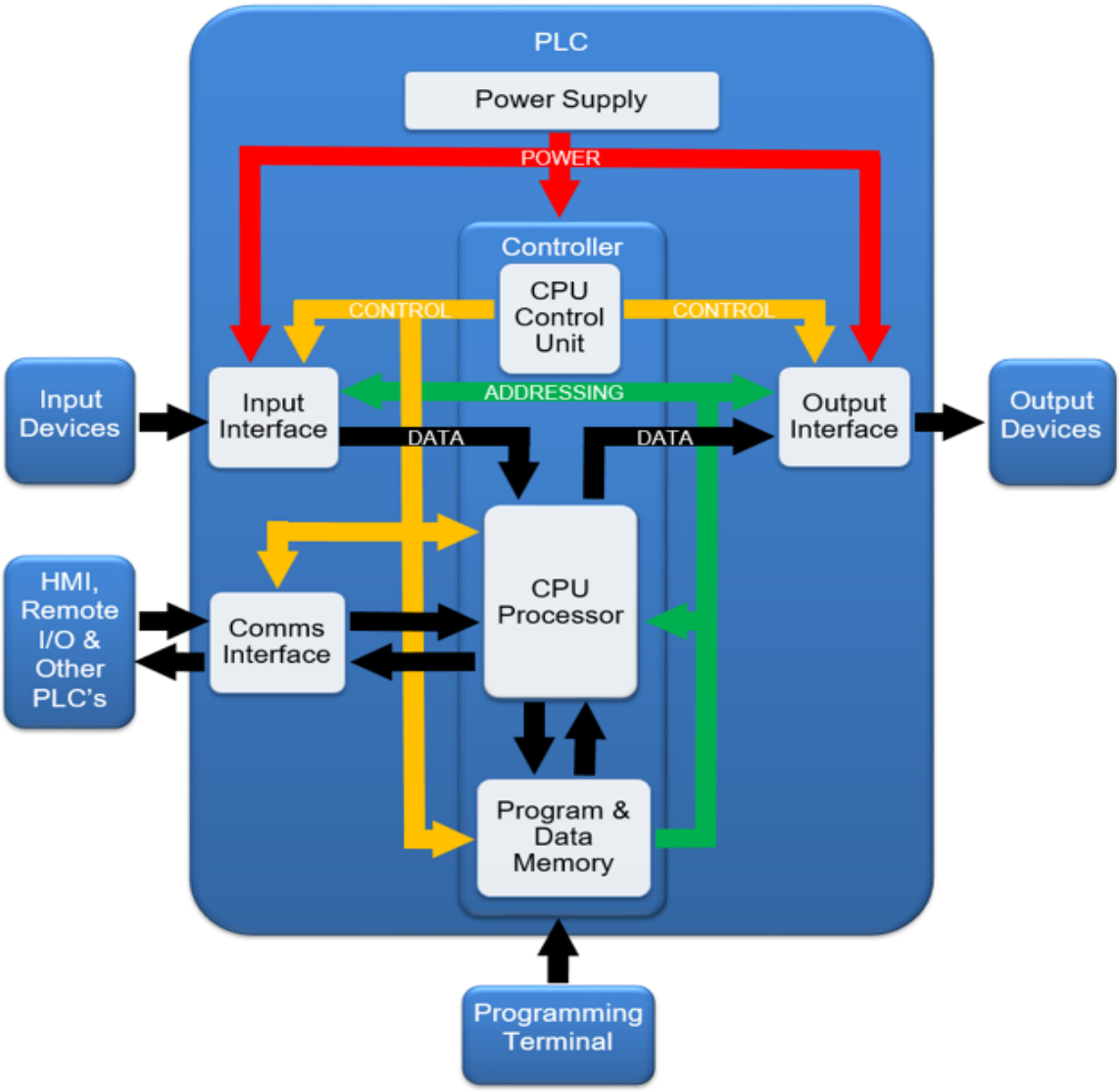


Рисунок 1.1.3 – Блок-схема архитектуры ПЛК [8]

Данные вводятся с устройств ввода, отправляются в процессор ЦП, а затем выводятся на устройства вывода. Процессор ЦП также обменивается данными с памятью программ и данных. После того как все необходимые данные собраны, программа обрабатывается циклически. Затем обрабатанные данные отправляются на выходной интерфейс для обработки и дальнейшего использования в управлении устройствами вывода. [8]

ЦП также управляет и обменивается данными с коммуникационным интерфейсом и периферийными устройствами.

В тоже время система адресации используется исключительно для организации данных, что совместно используются различными аппаратными компонентами.

Терминал программирования используется для формулирования программы ПЛК (например, релейной логики [9]), загрузки программы в контроллер и контроля/управления ПЛК и его программой. [8]

Блок питания отвечает за обеспечение и управление питанием различных аппаратных компонентов ПЛК. [8]

1.2 Возможные уязвимости программируемых логических контроллеров с точки зрения атак и обнаружения.

Программируемые логические контроллеры (ПЛК) как специализированные встроенные полевые устройства, ориентированные на определенные задачи, играют жизненно важную роль в современных промышленных системах управления (ICS), которые состоят из критической инфраструктуры. Чтобы удовлетворить растущие требования к рентабельности при одновременном повышении эффективности производства, в системы управления на основе ПЛК интегрируются коммерческие готовые программные и аппаратные средства, а также внешние сети, такие как Интернет. [4] Однако он также дает злоумышленникам возможность запускать злонамеренные, целенаправленные и изощренные кибератаки.

Далее будет отображена исследование уязвимости ПЛК и связанных с ними систем управления. Список уязвимостей был собран из исследовательских работ. [1] [4] [5] [7] [9] На рисунке 4 представлена разбивка уязвимостей на уровне компонентов и на уровне системы. Для ПЛК основными источниками уязвимости являются программы, прошивки и память. [4] Для систем управления, которые взаимодействуют с ПЛК, уязвимости обнаруживаются в промышленном прикладном программном обеспечении, промышленных протоколах связи и подключенных устройствах.

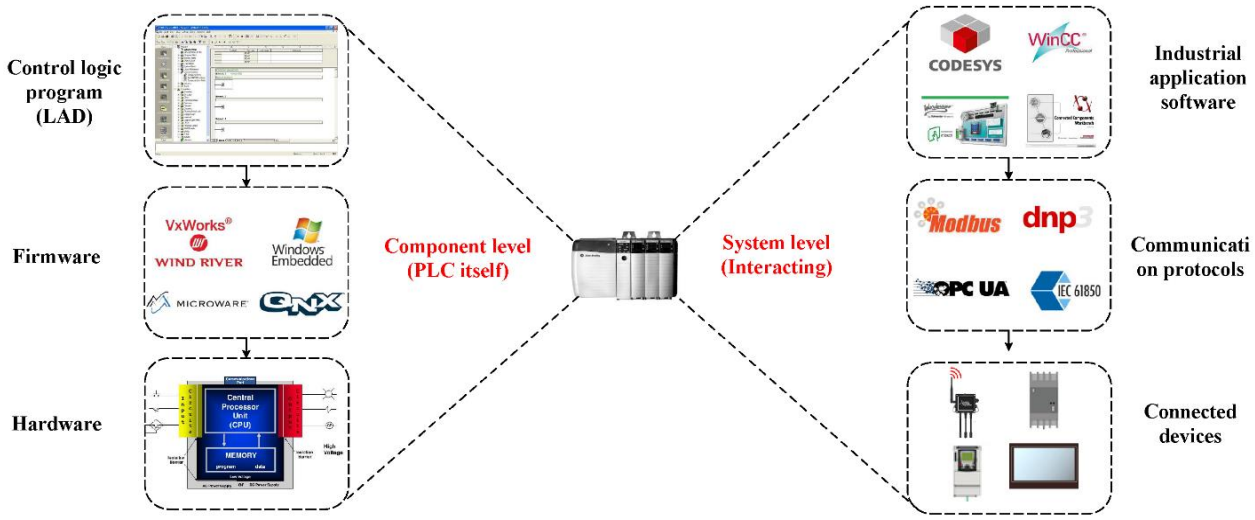


Рисунок 1.2.1 – Классификация на уровне компонентов и на уровне системы

Анализ на уровне основных компонентов - Хорошо известно, что при проектировании ПЛК не полностью учитывалась безопасность. Учитывая, что ПЛК применяются во всех аспектах инфраструктуры, замена устаревших ПЛК была бы крайне непрактичной и практически невозможной. Обе причины, как правило, приводят к различным атакам на ПЛК, использующим различные уязвимости в их конструкции. [4]

Уязвимости программы ПЛК - Как правило, управляющие программы пишутся с использованием LD. Учитывая то, как программы написаны или спроектированы, в программах существуют некоторые фатальные недостатки. Целостность и доступность ПЛК могут быть нарушены намеренно или случайно. С точки зрения программистов, могут быть созданы лазейки для хакеров, а программы ПЛК со скрытыми или незамеченными угрозами могут быть унаследованы; оба являются результатом отсутствия профессиональных знаний и навыков, таких как использование дублированных инструкций, отслеживание, отсутствие определенных катушек или выходов, обход или отказ в обслуживании. Что касается лестничной логики, то она уязвима для внедрения вредоносных программ из-за отсутствия аутентификации перед загрузкой программ в ПЛК. Жизненно важно подчеркнуть, что вредоносное ПО, вставленное в код лестничной логики, может находиться в состоянии покоя, которое может быть нарушено в любое время, что представляет большую потенциальную угрозу. Например, вредоносное ПО в LD под названием Ladder Logic Bombs LLB было описано Govil et al. [9]

Анализ на уровне системы - Помимо программирования, памяти и встроенного программного обеспечения, для получения функций ПЛК необходимо объединить их с промышленным прикладным программным обеспечением, протоколами связи и подключенными устройствами. Промышленное прикладное программное обеспечение может программировать, настраивать параметры, собирать данные и контролировать состояние ПЛК. Протоколы связи играют жизненно важную роль в обмене данными между различными устройствами. [7] Кроме того, порты подключенных устройств напрямую связаны с вводом/выводом ПЛК. Тем не менее, все они по-прежнему имеют негативное потенциальное влияние из-за своих собственных недостатков. [5]

Уязвимости программного обеспечения промышленных приложений - Прикладное программное обеспечение состоит в основном из программного обеспечения для программирования ПЛК, программного обеспечения для настройки и программного обеспечения SCADA. Как только программное обеспечение скомпрометировано, прямое управление ПЛК теряется, что приводит к загрузке вредоносного кода, изменению параметров ПЛК, раскрытию промышленных данных и так далее. Возьмем Stuxnet в качестве

несомненно показательного примера. Будучи первым, кто содержал руткиты, специально предназначенные для конкретных систем Siemens SCADA, Stuxnet использовал четыре нулевых дня, что было его уникальностью. Кроме того, Леверетт и Вайтман продемонстрировали, что CoDeSys, стороннее программное обеспечение для программирования, имело возможность изменять код внутри ПЛК, поддерживать целостность управления процессом и проверять каталоги на предмет обхода, что, возможно, стало основой для злонамеренных действий. эксплойты и вызвали атаки с внедрением кода. [5]

1.3 Исследование и внедрение Modbus TCP протокол для связи ПК - ПЛК

Modbus TCP/IP, также известный как протокол Modbus TCP или просто Modbus TCP, представляет собой тип протокола связи, используемый для управления оборудованием автоматизации. Он используется для подключения ПЛК, модулей ввода-вывода и шлюзов к другим полевым шинам или сетям ввода-вывода через Ethernet.

Этот протокол становится все более популярным в качестве стандарта автоматизации. В этом документе содержится информация о различных функциях Modbus, полезных для взаимодействия между различными типами оборудования автоматизации, а также о функциях, характерных для ПЛК. Он также группирует поддерживаемые типы сообщений в классы соответствия, которые различают те сообщения, которые реализуются повсеместно, и те, которые являются необязательными.

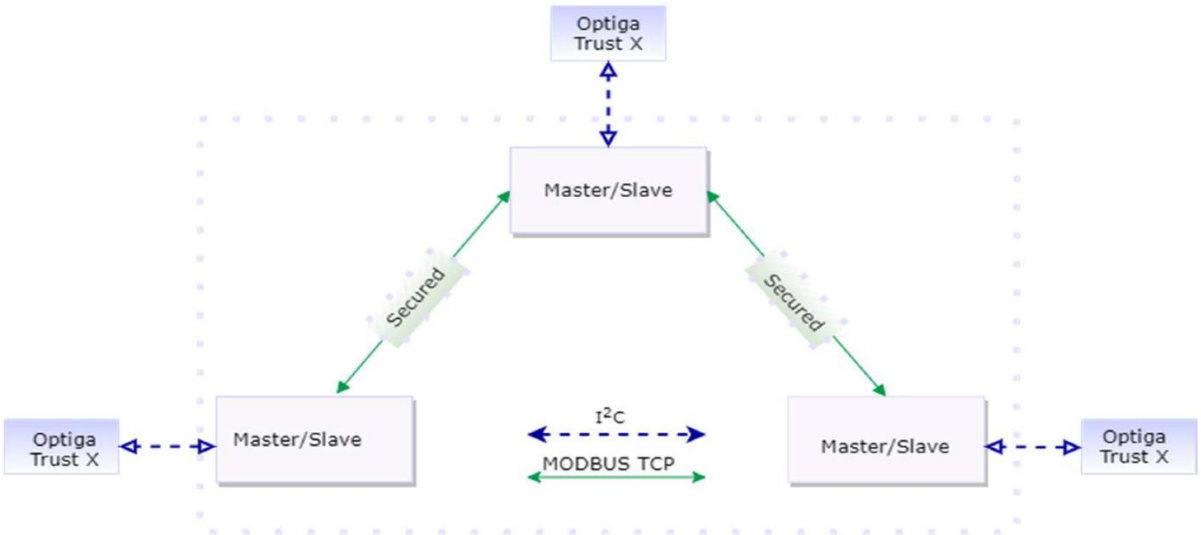


Рисунок 1.3.1 – Архитектура Modbus TCP [10]

Modbus TCP/IP сочетает в себе протокол управления передачей и интернет-протокол для отправки сообщений Modbus между совместимыми устройствами в разных системах. Для этого используется физическая сеть (Ethernet) с сетевым стандартом (TCP/IP) и прикладным протоколом (Modbus). Сообщение Modbus TCP/IP, по сути, представляет собой сообщение данных связи Modbus, упакованное в оболочку Ethernet TCP/IP. TCP/IP является протоколом транспортного уровня и не изменяет способ хранения или интерпретации данных в сообщении.

При передаче данных с компьютера на компьютер через Интернет TCP работает на верхнем уровне между двумя конечными системами, такими как браузер и веб-сервер. TCP выполняет надежную передачу потока байтов от одного процесса к другому.

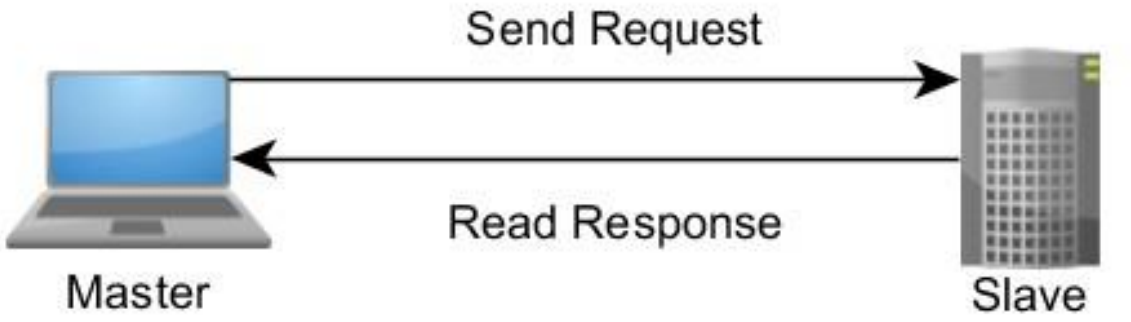


Рисунок 1.3.2 – Сетевые отношения Master-Slave [7]

Спецификация прикладного протокола Modbus основана на PDU и коде, который его обрабатывает. В этой спецификации описывается структура PDU, концепции данных, используемые протоколом, то, как функциональные коды используются для доступа к этим данным, а также реализация и ограничения каждого функционального кода. [7]

Формат Modbus PDU структурирован как функциональный код, за которым следует соответствующий набор данных. Размер и содержание этих данных определяется кодом функции, а общий размер PDU не должен превышать 253 байта. Каждый код функции имеет определенное поведение, которое подчиненные устройства могут настроить в соответствии с потребностями приложения. Спецификация PDU описывает фундаментальные принципы доступа к данным и манипулирования ими, однако ведомые устройства могут обрабатывать данные способом, который явно не указан в спецификации. [10]

Данные, доступные через Modbus, обычно хранятся в одном из четырех диапазонов адресов или банков данных: катушки, цифровые входы, регистры хранения и регистры ввода. В зависимости от отрасли или приложения эти

имена могут быть разными; например, регистры хранения могут называться выходными регистрами, а катушки могут называться цифровыми или дискретными выходами. Эти банки данных определяют тип и права доступа к хранимым в них данным. Ведомые устройства могут напрямую обращаться к этим данным, которые размещены на самом устройстве. Данные, доступные для Modbus, обычно являются частью основной памяти устройства. С другой стороны, ведущие устройства Modbus должны запрашивать доступ к этим данным, используя различные функциональные коды.

Поведение каждого блока описано в таблице 1.3.1.

Таблица 1.3.1 – Блоки модели данных Modbus [7]

Memory Block	Data Type	Master Access	Slave Access	Memory Block
Coils	Boolean	Read/Write	Read/Write	Coils
Discrete Inputs	Boolean	Read-only	Read/Write	Discrete Inputs
Holding Registers	Unsigned Word	Read/Write	Read/Write	Holding Registers

Эти блоки могут быть использованы для управления доступом к различным элементам данных и упростить доступ к различным типам данных на уровне приложения. Блоки чисто концептуальные. Это могут быть разные адреса памяти в данной системе, но они также могут перекрываться. Например, первый Coil может занимать в памяти то же место, что и первый бит слова, хранящегося в первом регистре. Система адресации определяется ведомым устройством, и его понимание каждого блока памяти составляет значительную часть структуры данных устройства.

[7] Концепция разницы между адресами памяти и номерами ссылок дополнительно усложняется индексацией, выбранной конкретным приложением. Как упоминалось ранее, регистр хранения номер один расположен по нулевому адресу. Как правило, ссылочные номера имеют один индекс, что означает, что начальное значение данного диапазона равно единице. Таким образом, 400 001 будет соответствовать регистру временного хранения 00001, который находится по адресу 0. Однако некоторые реализации предпочитают начинать свои диапазоны с нуля, поэтому 400 000 будет ссылаться на регистр временного хранения по нулевому адресу. Это показано в Таблице 1.3.2.

Таблица 1.3.2 – Регистры схем индексации [7]

Address	Register Number	Number (1-indexing, standard)	Number (0-indexing, alternative)
0	1	400001	400000
1	2	400002	400001
2	3	400003	400002

1.4 Параметрическое исследование автоматического управления системой охлаждением серверов в зданиях центров обработки данных

Экспериментально [6] проведено параметрическое исследование воздушного потока и управления охлаждением серверов центров обработки данных для различных проектных условий. Для целей тестирования была спроектирована и построена физическая масштабная модель центра обработки данных, вмещающая одну стойку из четырех серверов. Распределение температуры передней и задней стоек и серверов, а также индексы тепла подачи/возврата (SHI/RHI) используются для оценки тепловых характеристик центра обработки данных. Были проведены эксперименты [6] для параметрического изучения влияния коэффициента раскрытия перфорированной плитки, изменения мощности нагрузки серверов и плотности мощности стойки. Результаты показали, что перфорированная плитка с коэффициентом раскрытия 25% обеспечивает наилучшие результаты среди других коэффициентов раскрытия, оптимальная польза от холодного воздуха при охлаждении серверов получается при равномерной загрузке серверов по мощности увеличение удельной мощности снижение рециркуляции воздуха, но увеличить обвод воздуха и температуру серверов.

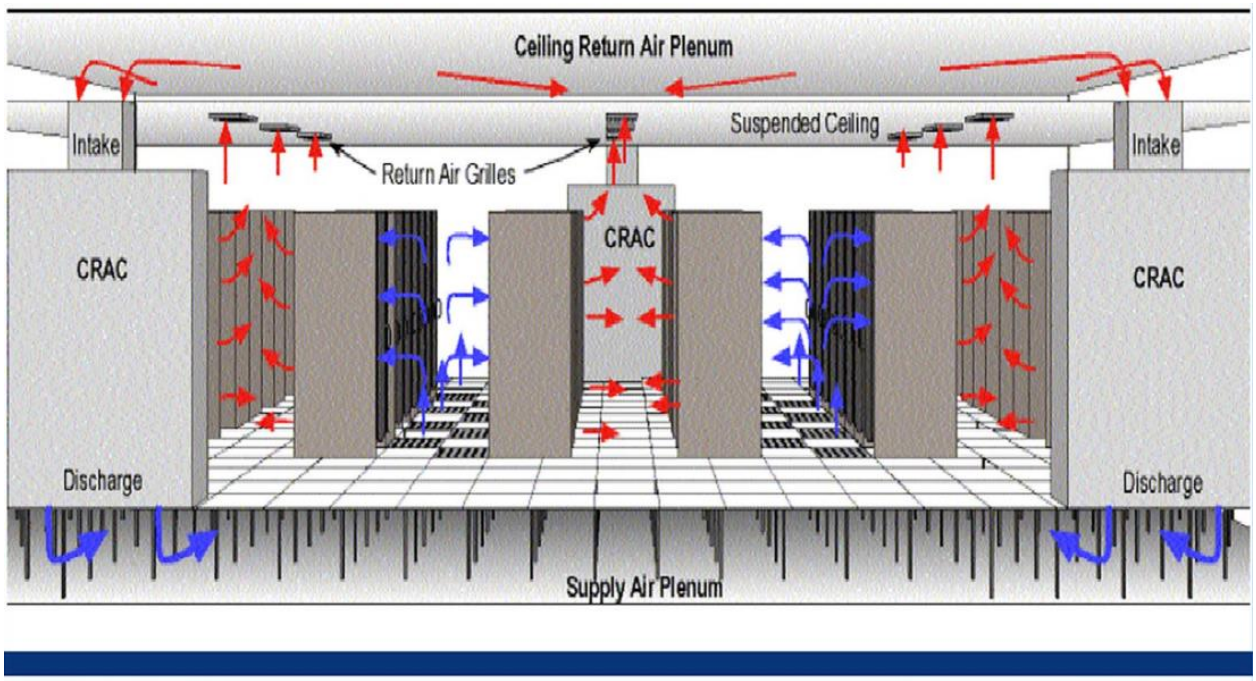


Рисунок 1.4.1 – Типовой центр обработки данных с открытым проходом [6]

Эффективное управление температурой в помещениях центров обработки данных может поддерживаться за счет правильного распределения воздуха в помещениях. Таким образом, для повышения эффективности охлаждения дата-центров необходимо изучить схемы воздушных потоков и распределения температуры внутри дата-центров. Производительность и эффективность управления температурным режимом центров обработки данных обычно оценивают безразмерными параметрами индекса подаваемого тепла (SHI) и индекса возвратного тепла (RHI) [13] [14]. Используя эти индексы, можно понять и оценить теплопередачу и управление температурой внутри центров обработки данных.

	Tile size	Aisle pitch	Cold-aisle size	Hot-aisle size
U.S.	2 ft (610 mm)	14 ft (4267 mm)	4 ft (1220 mm)	3 ft (914 mm)
Global	600 mm (23.6 in)	4200 mm (13.78 ft)	1200 mm (3.94 ft)	914 mm (3 ft)

Seven-title aisle pitch, equipment aligned on cold aisle

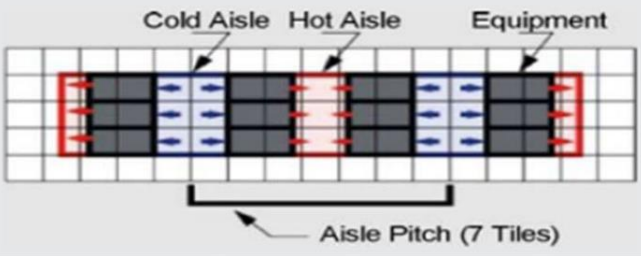


Рисунок 1.4.2 – Распределение шага между проходами и расположение стоек с разделением [6]

Разработчики центра обработки данных определили, что скорость воздуха не является важным фактором, поскольку тепловой комфорт человека не является основным фактором в центре обработки данных.

Шривастава и др. [15] провели исследование различных конфигураций центров обработки данных. Они пришли к выводу, что наиболее эффективной системой воздухораспределения является та, в которой холодный воздух подается с фальшпола, а возвратный воздух вытягивается с потолка.

Этот вывод подтверждает утверждение Накао и др. [16] о том, что наихудшее распределение воздуха и регулирование температуры в центрах обработки данных достигается, когда холодный воздух подается с потолка, а рециркуляционный воздух забирается с пола.

Исследования [17] [18] сравнивали и оценивали конфигурации с напольной и воздушной подачей. Было обнаружено, что, хотя подача под полом рекомендуется для правильного распределения воздуха и управления температурным режимом, это может привести к возникновению горячих точек на серверах, расположенных в верхней части стойки, из-за рециркуляции горячего воздуха.

Хэм и др. [19] разработали упрощенную модель сервера для имитации энергопотребления центров обработки данных при охлаждении. Они заявили, что предлагаемая модель сервера должна быть в состоянии решить проблемы с текущим методом моделирования на этапе проектирования, особенно когда в центре обработки данных высокая температура окружающей среды и используется экономайзер.

Фулпагар и др. [20] исследовали влияние препятствий напорной камеры на производительность центра обработки данных. Сообщалось, что засоры в нагнетательных камерах могут привести к снижению скорости потока холодного воздуха до 80%, что приведет к повышению температуры воздуха на выходе из стеллажей на 2,5 °C и увеличению вероятности образования горячих точек.

Алмоли и др. [21] провели расчетное гидродинамическое исследование жидкостного охлаждения стоек в центрах обработки данных и обнаружили, что это может привести к снижению нагрузки на КВКЗ по сравнению с традиционным воздушным охлаждением.

Шарма и др. [14] предложили два безразмерных параметра тепловых индексов, индекс приточного тепла (SHI) и индекс возвратного тепла (RHI), для оценки явлений рециркуляции горячего воздуха и обхода холодного воздуха для управления температурным режимом центра обработки данных. SHI рассчитывается как отношение тепла, полученного воздухом в холодном коридоре (за счет рециркуляции горячего воздуха) до его поступления в стеллажи, к общему теплу, полученному воздухом после выхода из стеллажей.

$$\mathrm{SHI} = \left(\frac{\delta \mathrm{Q}}{Q+\delta \mathrm{Q}}\right) = \frac{\text{Повышение энтальпии в холодном коридоре перед входом в сервер}}{\text{Суммарный подъем энтальпии холодного воздуха}} \quad (1.4.1)$$

Индекс возвратного тепла — это доля тепла, поглощаемого холодным воздухом при его прохождении через серверы, по отношению к общему количеству поглощаемого тепла.

$$\mathrm{RHI} = \left(\frac{\mathrm{Q}}{Q+\delta \mathrm{Q}}\right) = \frac{\text{Суммарный отбор тепла блоками КВКЗ}}{\text{Полная энтальпия подъема холодного воздуха}} \quad (1.4.2)$$

$$\mathrm{SHI} = \mathrm{RHI} \quad (1.4.3)$$

где Q — теплота, полученная холодным воздухом при прохождении его по серверам стойки за счет тепловыделения серверов, а δQ — теплота, полученная холодным воздухом перед входом в стойки. SHI можно переписать через температуру воздуха на входе и выходе из стойки и температуру приточного холодного воздуха следующим образом:

$$\mathrm{SHI} = \left(\frac{\sum(T_{in}^{r} - T_{ref})}{\sum(T_{out}^{r} - T_{ref})}\right) \quad (1.4.4)$$

где T_{in}^{r}, T_{out}^{r}, и T_{ref} – температура воздуха на входе в стойку, температура воздуха на выходе из стойки и температура воздуха на входе в приточную камеру соответственно.

Уравнения (1.4.1) и (1.4.4) показывают, что увеличение δQ вызывает увеличение (T_{in}^{r}) и SHI. Повышение (T_{in}^{r}) может привести к отказу системы и проблемам с надежностью. Увеличение (T_{in}^{r}) также вызывает генерацию энтропии, снижающую эффективность использования энергии. Следовательно, SHI можно использовать как показатель управления температурным режимом и энергоэффективности в центре обработки данных. Низкий RHI указывает на обход холодного воздуха и его смешивание с отработанным воздухом стойки без прохождения на серверы. Идеальные значения (SHI и RHI) составляют (0 и 1), в то время как типичные допустимые диапазоны SHI и RHI составляют SHI < 0,2 и RHI > 0,8. [6]

2 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

2.1 Промышленные ПЛК, конкурентные преимущества

По своей структуре, все нынешние промышленные ПЛК от таких крупных производителей, как OWEN, Festo, Siemens, Direct Logic, Advantech, Delta и прочие - не отличаются принципиально от своих аналогов. Задача любого ПЛК - значительно упрощать создание и управление сложными автоматизированными системами и отдельными устройствами, в том числе бытовыми. Сокращается этап разработки, упрощается установка и отладка за счет стандартизации аппаратных и программных компонентов, повышается надежность при эксплуатации. Кроме того, при необходимости упрощается ремонт и модернизация.

Для сравнения, был выбран самый распространенный и ходовой ОВЕН ПЛК160-24.А-М [М02] [22], на его примере будет проведено дальнейшее сравнение с PLCDuino.

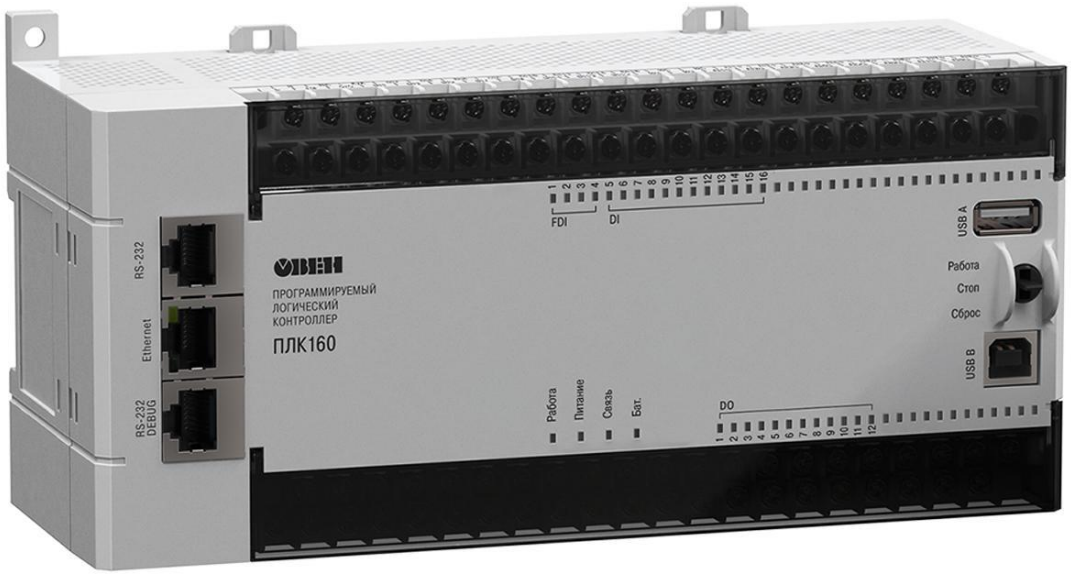


Рисунок 2.1.1 – Внешний вид ОВЕН ПЛК160-24.А-М [М02]

ОВЕН ПЛК160 [М02] из всех существующих аналогов является оптимальным выбором для построения систем автоматизации среднего уровня и распределенных систем управления, поскольку он оснащен программируемыми моноблочными контроллерами как с дискретными, так и с аналоговыми входами/выходами. Более подробное описание об ПЛК в разделе [1.1]

Преимущества ОВЕН ПЛК160 [М02] [22]:

– Наличие встроенных дискретных и аналоговых входов/выходов на борту.

– Скоростные входы для обработки энкодеров.

– Ведение архива работы оборудования или работа по заранее оговоренным сценариям при подключении к контроллеру USB-накопителей.

– Простое и удобное программирование в системе CODESYS V.2 через порты USB Device, Ethernet, RS-232 Debug.

– Передача данных на верхний уровень через Ethernet или GSM-сети (GPRS).

– 3 последовательных порта (RS-232, RS-485):

– увеличение количества входов-выходов;

– управление частотными преобразователями;

– подключение панелей операторов, GSM-модемов, считывателей штрих-кодов и т.д.

– Наличие двух исполнений по питанию (220 В и 24 В).

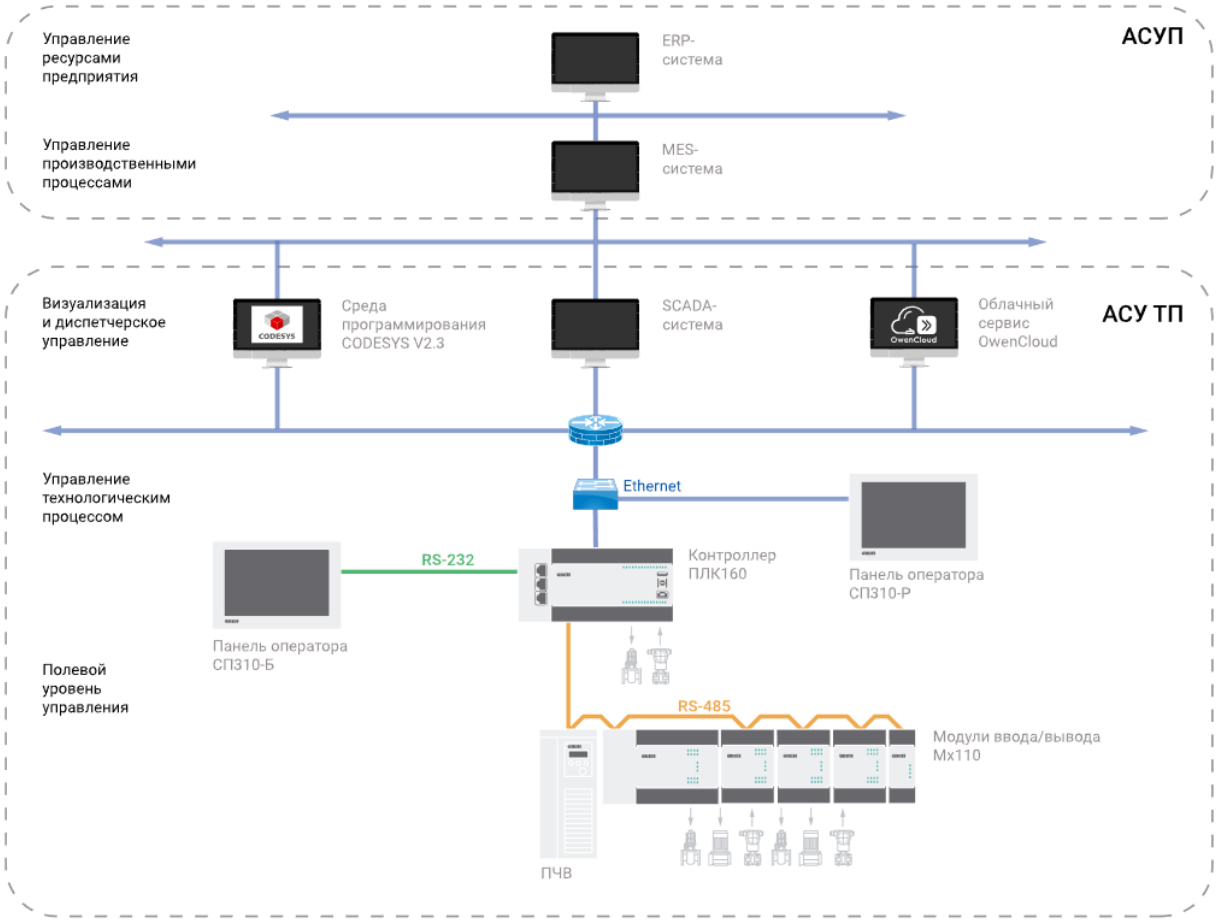


Рисунок 2.1.2 – Схема применения ПЛК160 [M02] [22]

Далее будут приведены технические характеристики устройства ПЛК160, но интересовать будут исключительно те, с которыми будет дальнейшее сравнение с PLCDuino

Таблица 2.1.1 – Технические характеристики ПЛК160 [M02] [22]

Напряжение питания	от 9 до 30 В постоянного тока при T > -20 °C от 9 до 26 В постоянного тока при -40 °C > T > -20 °C (номинальное 12 или 24 В)
Количество входов: - из них быстродействующих	16 4 (DI1-DI4)
Количество релейных выходных каналов	12
Количество аналоговых входов	8
Центральный процессор	RISC-процессор Texas Instruments Sitara AM1808
Объем оперативной памяти (SDRAM)	Пользовательская программа: 1 Мб Данные пользовательской программы: 128 Кб

	Heap: до 4 Мб1 RAM-диск: 8 Мб
Объем энергонезависимой памяти (FLASH)	6 Мбайт доступно для хранения файлов и архивов
Размер Retain-памяти (MRAM)	16 Кбайт
Масса, не более	1,2 кг
Средняя наработка на отказ2	60 000 ч
Средний срок службы	8 лет
Интерфейсы связи:	RS-485 RS-232 RS-232 Debug Ethernet 100 Base-T USB-Device USB-Host
Максимальный ток питания подключаемых устройств	До 250 мА3 (RS-232-Debug / RS-232) До 150 мА (USB-Host)

Принципиальное сравнение между ОВЕН ПЛК160 [М02] и PLCDuino будет приведено в [2.5]

2.2 Протокол коммуникации Modbus TCP, достоинства и недостатки, полный разбор с примерами

Общие понятия и представления об данном протоколе находятся в разделе [1.3]. В данном разделе отображаются преимущественные особенности в сопряжении – ПК–ПЛК–исполняющие устройства. Для начала, как будет выглядеть данная схема подключения без данного протокола –

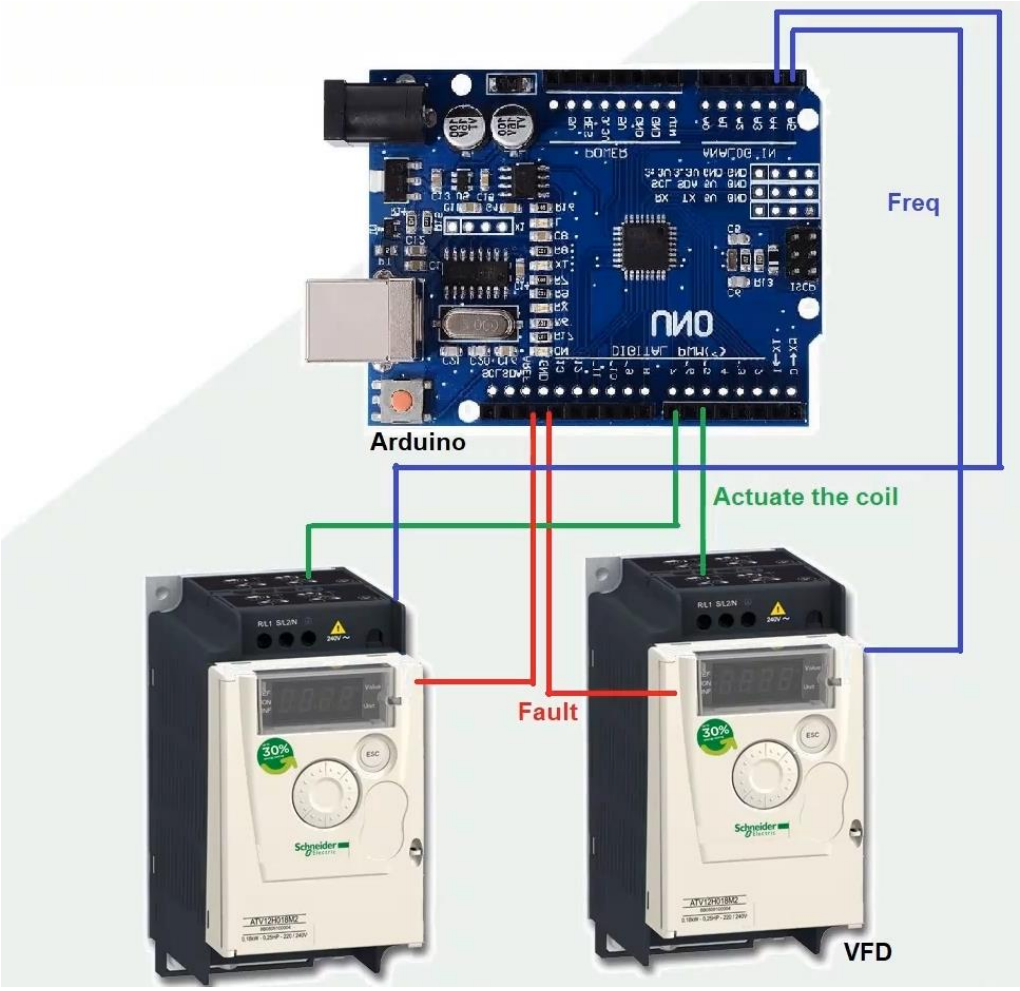


Рисунок 2.2.1 – Коммуникация VFD устройств к Arduino UNO без протокола Modbus TCP

VFD означает частотно-регулируемый привод (Variable Frequency Drive), который также называют приводом переменного тока или инвертором. Это устройство используется для управления скоростью двигателя переменного тока, а также другими параметрами.

Проблема заключается в том, что каждый из дополнительных VFD нужно подключать с помощью проводов к соответствующему пину, в данном случае к одному VFD подключается сразу 3 провода, что будет невозможно, если их количество увеличится в разы, пинов просто не хватит.

Далее, на примере тех же самых VFD, будет отображена схема подключения с использованием протокола Modbus TCP к Arduino UNO и Ethernet Shield.

Рисунок 2.2.2 – Коммуникация VFD устройств к PLCDuino по протоколу Modbus TCP

Количество проводов уменьшилось, что в данном случае играет очень большую роль. Если подключить вместо VFD одно-канальное реле, провод будет необходим лишь один. В данном случае, PLCDuino является буфером хранения «скриптов» с CoDeSys [2.3] и может регулировать и активировать все девайсы, что подключены к ней.

2.2.1 Modbus TCP Master-Slave Mode

Всего существует 3 метода подключения по протоколу Modbus TCP между CoDeSys SoftPLC и исполняющей PLCDuino:

– Оба CoDeSys SoftPLC и PLCDuino могут считывать и записывать данные от друг друга (Two Masters)

– Только CoDeSys SoftPLC может считывать/записывать данные с/в PLCDuino (CoDeSys SoftPLC Master, PLCDuino Slave)

– Только PLCDuino может считывать/записывать данные с/в CoDeSys SoftPLC (PLCDuino Master, CoDeSys SoftPLC Slave)

В моем сценарии, будет рассматриваться именно второя связь, где – CoDeSys SoftPLC является «Мастером» а Arduino – «Slave», потому что плата имеет ограниченное количество ресурсов.

Первый вариант тоже имеет свои минусы – предположим, имеется 1 активная переменная и 2 источника, что считывает ее.

По данному сценарию Master может отправить 4 типа запросов – Write INT data, Read INT data, Write BIT и Read BIT. PLCDuino, в свою очередь, будет реагировать соответственным образом.

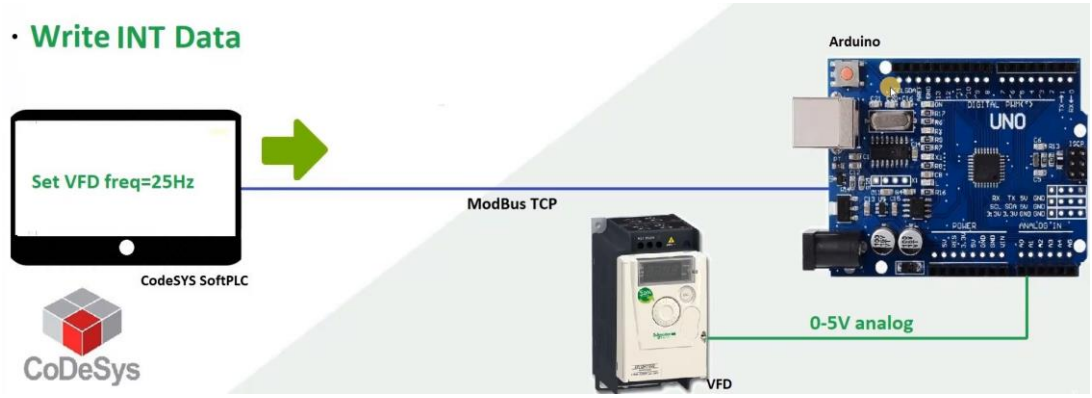


Рисунок 2.2.1.1 – Write INT data на примере с VFD

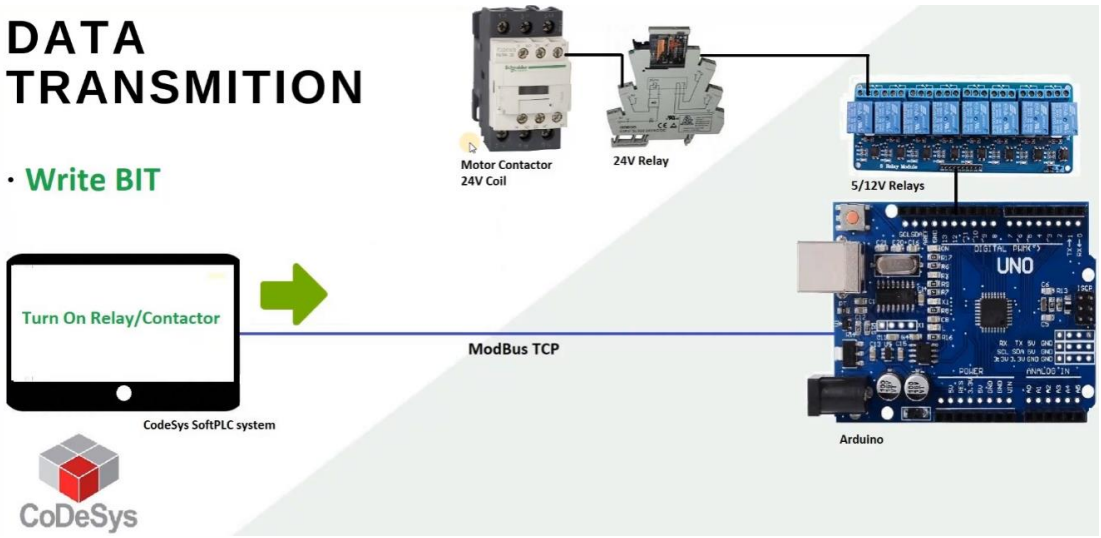


Рисунок 2.2.1.2 – Write BIT на примере с реле и Motor Contact Coil

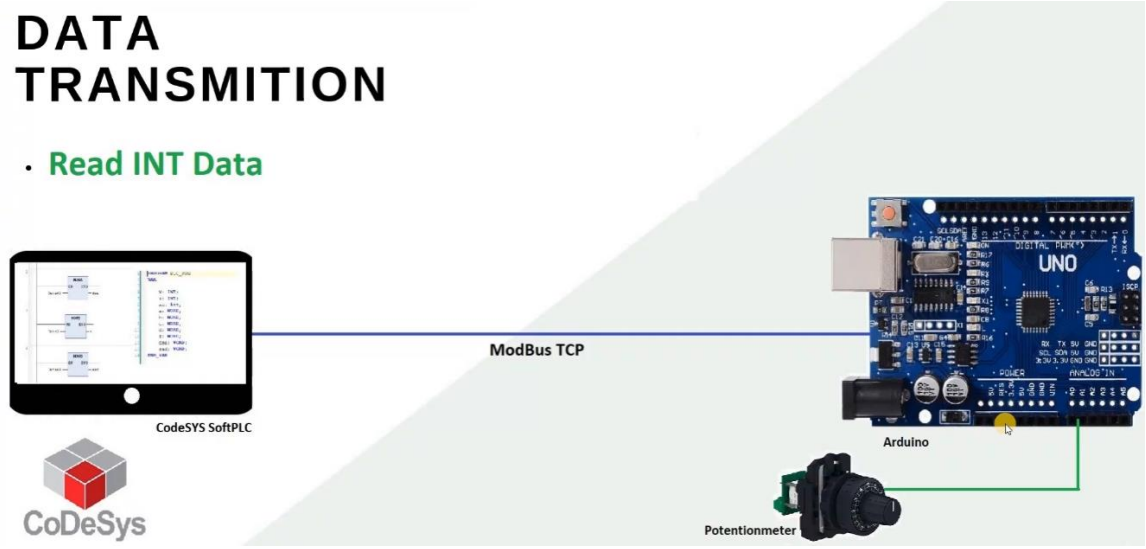


Рисунок 2.2.1.3 –Read INT data на примере с потенциометром

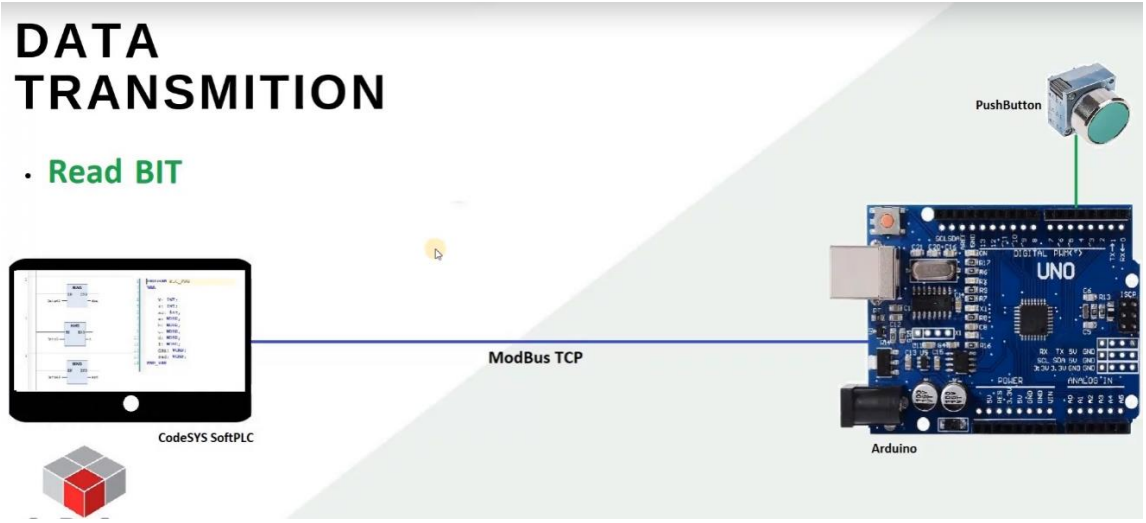


Рисунок 2.2.1.4 – Read BIT на примере с кнопкой

2.3 Софт для PLCDuino, разбор программы CoDeSys

Предисловие - До широкого использования ПЛК все автоматизированное управление управлялось схемами, состоящими из реле, переключателей, часов и счетчиков. Этот тип управления требовал большого количества проводов и часто заполнял большие шкафы электромеханическими реле. Электрикам приходилось либо сконструировать элементы управления, либо использовать готовую схему подключения реле. Эти схемы иллюстрировали, как были подключены все переключатели, датчики, двигатели, клапаны, реле и т. д. Схемы подключения реле являются предшественниками языка программирования лестничных диаграмм (LD),

который до сих пор является популярным языком программирования, используемым для программирования ПЛК.

Механические органы управления имели множество недостатков, например, занимали много места и требовали много времени и труда для установки и внесения каких-либо модификаций. Релейное управление обычно состоит из сотен реле, соединенных вместе проводами, идущими во всех направлениях. Если логическую функцию необходимо изменить или расширить, необходимо перемонтировать весь физический блок, что очень дорого с точки зрения труда. Кроме того, поскольку реле являются электромеханическими устройствами, они имеют ограниченный срок службы, что приводит к частым перебоям в работе и нарушению технологического процесса. [23]

Изначально CoDeSys разрабатывался для задач, требующих автономности, надежности и максимальной производительности при использовании минимального оборудования. Это позволило выйти за рамки традиционных систем ПЛК IEC 61131-3. В настоящее время автомобили, краны, экскаваторы, самосвалы, яхты, печатные станки, деревообрабатывающие станки, литейные и прокатные станки, а также сборочные машины крупнейших мировых брендов имеют один или группу встроенных контроллеров с CoDeSys. В 2011 году компания ITQ GmbH провела исследование по анализу характеристик и распространенности программных средств в области машиностроения и мобильных приложений в Европе [24]. По результатам 36% компаний используют CoDeSys и инструменты на его основе (Bosh Rexroth IndraWorks, Beckhoff TwinCAT и др.), а на универсальные инструменты, конкурирующие с CoDeSys вместе взятые, приходится 7%.

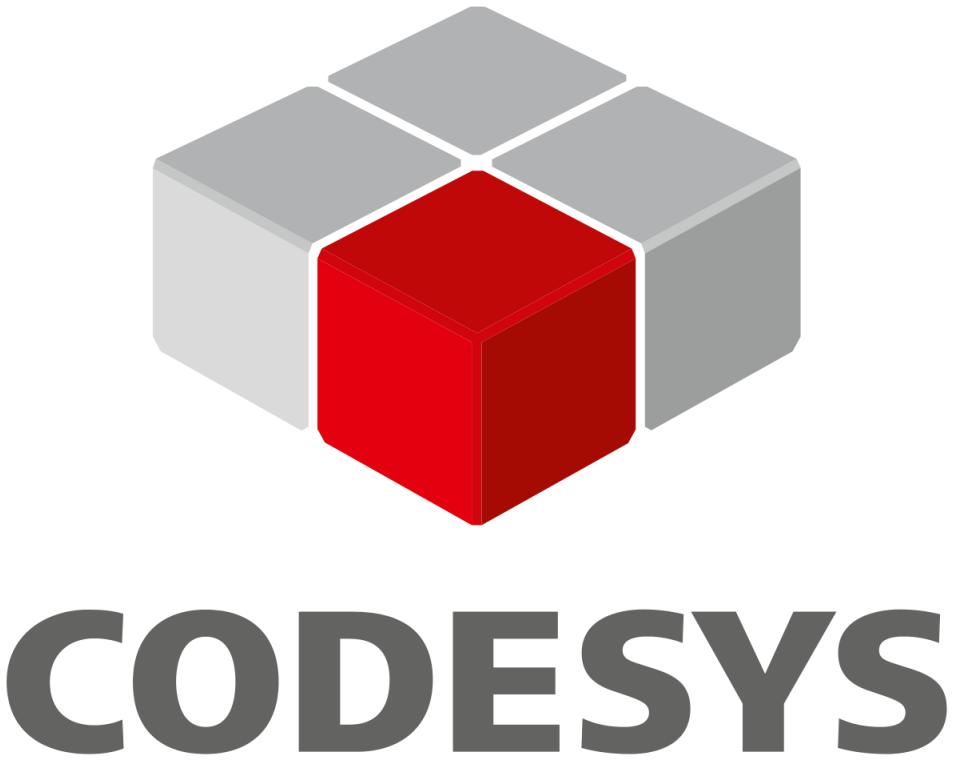


Рисунок 2.3.1 – Официальное лого всей линейки CoDeSys (начиная с версии 3.5)

Также хотелось бы отметить, что CoDeSys является бесплатным, что примечательно, учитывая, что это программное обеспечение высшего уровня, используемое некоторыми из крупнейших компаний в мире и он точно не уступает своим конкурентам.

Такие крупные компании, как ОВЕН программируют свои ПЛК на том же самом программном обеспечении.

Подводя итог, прототип PLCDuino по программному обеспечению не уступает «гигантам» рынка по производству промышленных ПЛК, что делает его заведомо потенциально хорошим продуктом.

2.4 Arduino IDE

Arduino IDE — это программа, которая позволяет пользователям писать свои собственные программы (скетчи) для платформы Arduino. Эта платформа предназначена для дизайнеров-любителей, которые хотят создавать простые системы автоматизации и робототехники. Язык программирования Arduino основан на C++ (с использованием компилятора AVR-GCC) и включает функции, облегчающие кодирование для тех, кто только начинает. IDE означает «Интегрированная среда разработки»

“Integrated Development Environment”, и представляет собой программу, созданную Arduino.cc. Он используется для редактирования, компиляции и загрузки кода на устройство Arduino. Большинство плат Arduino совместимы с этим программным обеспечением с открытым исходным кодом, которое можно легко установить и использовать для компиляции кода.

Arduino хорошо известен своим экономичным программным обеспечением и тысячами проектов, которые были созданы с его помощью. Он написан на модифицированной и упрощенной версии C++, что делает его доступным для пользователей всех уровней, от школьников до опытных программистов. Программное обеспечение совместимо с MAC, Windows и Linux, поэтому для него не требуется мощная система.

В случае с данным проектом, написанный мною скетч будет являться «буффером» для хранения скриптов, что будут вызываться с помощью кода [3.3], написанного в CoDeSys, поэтому Arduino не будет даже проводить логические операции.

2.5 Обзор технических деталей устройства (Arduino UNO, Ethernet Shield, Relay, датчик температуры DS18B20, драйвер двигателя L298N)

Далее, вкратце будут рассмотрены технические характеристики каждой детали по отдельности и почему были выбраны именно эти детали.

1) Arduino UNO.

Для физических вычислений доступно множество других микроконтроллеров, но ни один из них не может сравниться с преимуществами микроконтроллера Arduino. Он недорогой, самый дешевый модуль стоит не более 9600 тенге. Кроме того, он кроссплатформенный, работает на операционных системах MAC OS, Windows и Linux. Arduino IDE также удобна и проста для понимания даже для новичков, а библиотека C++ может быть расширена. Кроме того, можно создать собственную версию платы для конкретных целей и модифицировать ее на платах Arduino. Все эти функции делают плату Arduino идеальной для создания роботов, контроллеров и бытовой техники.

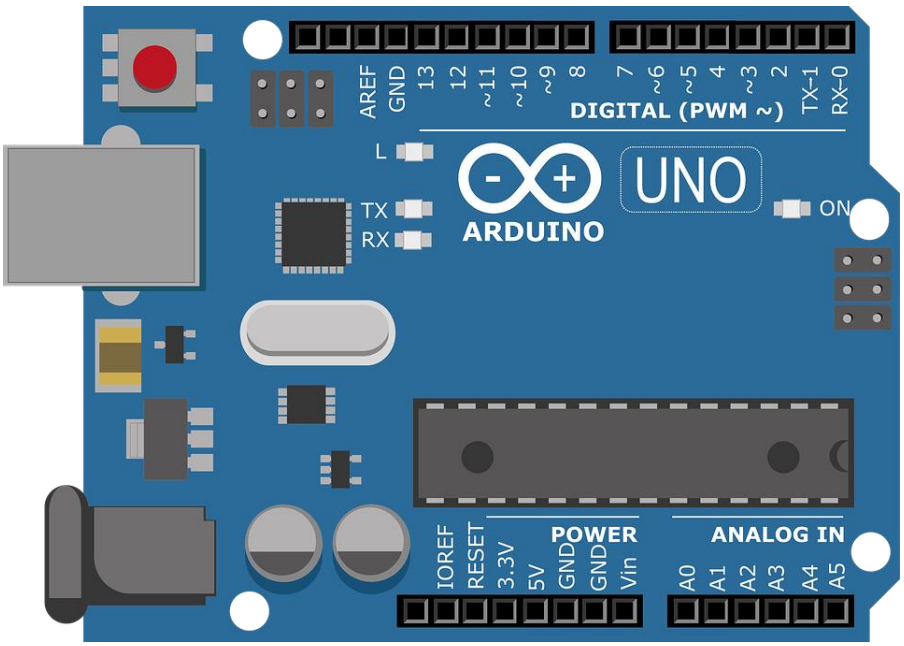


Рисунок 2.5.1 – Внешний вид платы Arduino UNO

Как было предложено в [2.1] сравнение по характеристикам PLCDuino с ПЛК160 [M02] ОВЕН

Таблица 2.5.1 – Сравнение по техническим характеристикам ПЛК160 [M02] и PLCDuino

ПЛК	PLCDuino	ПЛК160 [M02] ОВЕН
Тактовая частота контроллера	16 МГц	
Флешь-память программ	32 Кб из которых 512 байт используются для загрузчика (bootloader)	6 Мбайт доступно для хранения файлов и архивов
Оперативная память ОЗУ(RAM)	2 Кб	Пользовательская программа: 1 Мб Данные пользовательской программы: 128 Кб Heap: до 4 Мб1 RAM-диск: 8 Мб
Энергонезависимая память	EEPROM 1 Кб	16 Кб
Цифровые Входы/Выходы:	20 шт., 6 из которых могут использоваться как выходы ШИМ (PWM) и 6 как Аналоговые входы(АЦП)	16 4 (DI1-DI4)
Рабочее напряжение	4,5 В - 5,5 В	от 9 до 30 В постоянного тока при T > -20 °C от 9 до 26 В постоянного тока при -40 °C > T > -20 °C (номинальное 12 или 24 В)

Микроконтроллер	ATmega328	RISC-процессор Texas Instruments Sitara AM1808
Интерфейсы связи	Поддерживает интерфейсы I2C (TWI) и SPI интерфейс UART TTL (5 В) драйвера USB COM Ethernet 100 Base-T	RS-485 RS-232 RS-232 Debug Ethernet 100 Base-T USB-Device USB-Host
Постоянный ток на один выход не более	40 мА	До 250 мА3 (RS-232-Debug / RS-232) До 150 мА (USB-Host)
Средняя наработка на отказ2	[4.1]	60 000 ч
Средний срок службы	[4.1]	8 лет
Цена на данный момент	432957 тенге	12300 тенге

Как видно, характеристики ПЛК ОВЕН выше, но и цена сильно высокая, таким образом, рассматривая дешевый ПЛК, то PLCDuino ему соответствует.

2) Ethernet Shield.

Плата расширения Arduino Ethernet оснащена контроллером Wiznet W5100 Ethernet, который обеспечивает доступ Arduino к Интернету. Эта микросхема поддерживает стек сетевых протоколов IP и позволяет использовать протоколы TCP и UDP. Он может обрабатывать до четырех соединений с открытыми сокетами одновременно. Для программирования платы для работы в глобальной сети рекомендуется использовать библиотеку Ethernet. Плата подключается к Arduino через специальный разъем, который состоит из металлических штырей («папа») с одной стороны и гнезда («мама») с другой стороны. Эта конструкция позволяет подключать к Arduino несколько плат расширения, одну над другой. подключены к Arduino, один над другим.

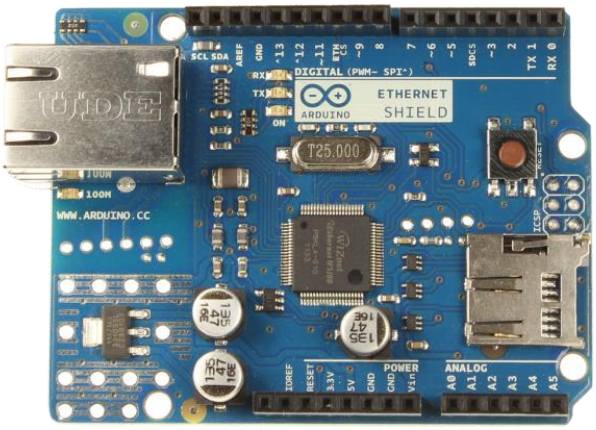


Рисунок 2.5.2 – Внешний вид платы расширения Arduino Ethernet

3) Relay.

Этот 8-канальный релейный модуль (5В) предназначен для Arduino и других микроконтроллеров для управления устройствами с высоким входным током. При подаче на вход сигнала низкого напряжения реле сработает. Более доступного модуля по доступной цене просто нет.

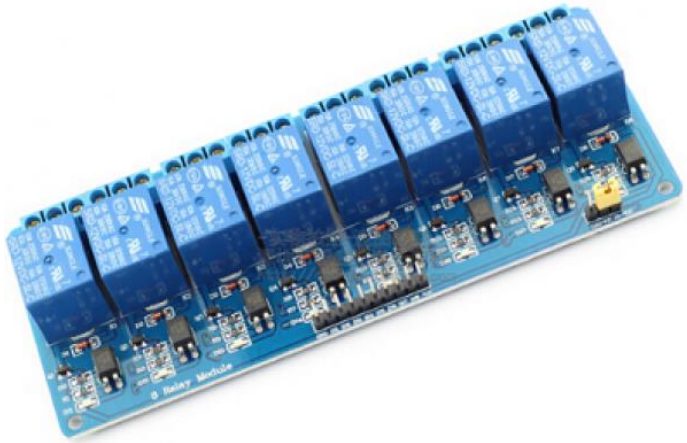


Рисунок 2.5.3 – Внешний вид платы 8-ми канального реле

4) Датчик температуры DS18B20.

Водонепроницаемый датчик температуры DS18B20 (длиной 1 метр) можно использовать для измерения температуры воды в аквариуме или чайнике. Он также подходит для использования на открытом воздухе, так как устойчив к дождю. Садоводы найдут его полезным для измерения температуры почвы в теплице или в саду. При отсутствии медицинского термометра этот датчик можно использовать и для измерения температуры тела.

Рисунок 2.5.4 – Внешний вид датчика температуры DS18B20

5) Драйвер двигателя L298N.

Модуль драйвера L298N представляет собой двухканальный драйвер двигателя постоянного тока на основе H-моста, который можно использовать для управления двумя двигателями постоянного тока или одним четырехпроводным двухфазным шаговым двигателем. Подавая сигнал ШИМ на соответствующие выходы, можно одновременно управлять направлением и скоростью двух двигателей постоянного тока.

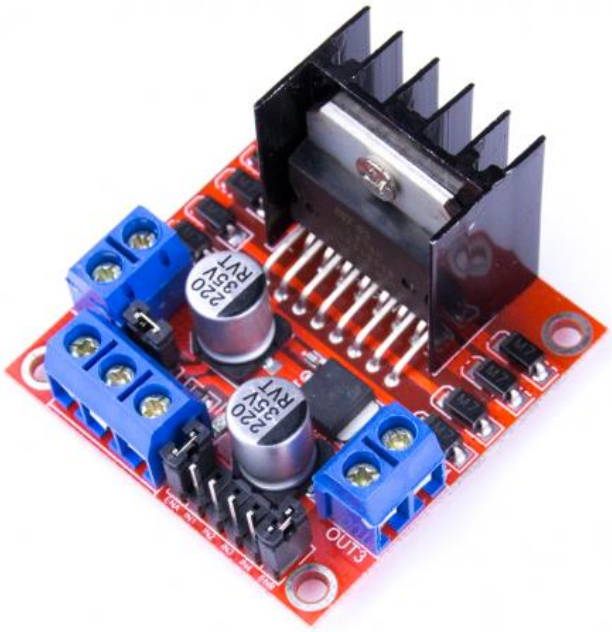


Рисунок 2.5.5 – Внешний вид драйвера двигателя L298N

2.6 Выбор программы 3D моделирования

Небольное предисловие. 3D-моделирование — это вид компьютерной графики, предполагающий создание трехмерных визуальных объектов с помощью специализированного программного обеспечения. Другими словами, это создание объемных изображений с помощью специальных программ.

Во первых, 3D-моделирование является неотъемлемой частью индустрии развлечений, поскольку оно используется в фильмах, анимации и видеоиграх. Без 3D-моделирования расстановка метавселенных была бы невозможна.

Во вторых, онн также используется для создания прототипов и визуализации зданий и интерьеров для презентаций проектов.

В третьих, 3D-моделирование можно использовать для изготовления деталей, украшений и медицинских протезов, которые затем можно распечатать на 3D-принтере или изготовить на другом устройстве.

В данном проекте необходим именно третий пункт, для создания корпуса ПЛК и моделирования примерной чиллерной комнаты. Как известно, не все 3D программы приспособленны для этого, они также делятся по степени надобности на полигональное моделирование и на моделирование в САПР программах.

Суть полигонального моделирования заключается в создании моделей с использованием полигонов — поверхностей, заданных точками. Точками можно манипулировать, формируя модель, подчеркивая ее эстетические и интуитивные качества. Это скорее художественная работа, так как обычно нет ссылки на фактические единицы измерения.

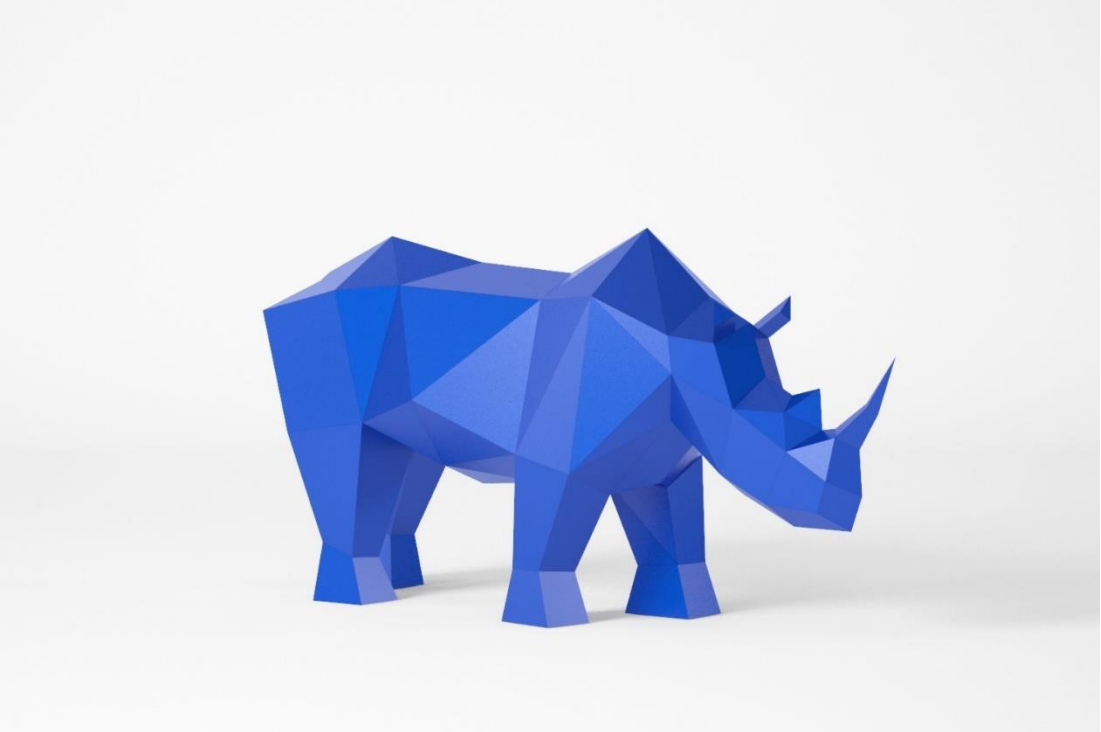


Рисунок 2.6.1 – Пример работы в полигональном моделировании

По причине того, что все мерки в таких программах мнимые, то этот вариант не подходит для создания точных моделей.

Что касательно моделирования в САПР программах (CAD — Computer-Aided Design) — это проектирование, которое использует формулы, а не полигоны, для определения моделей. Это позволяет создавать чрезвычайно точные конструкции, вплоть до долей миллиметра, и поэтому широко используется для создания моделей, которые будут производиться в массовом производстве. Примеры таких моделей включают детали для заводов, автомобилей, двигателей, зданий, мебели и самолетов.

Именно по этой причине среда моделирования для дальнейшего проектирования была выбрана именно САПР, а если более точно – Autodesk Fusion 360. Данная программа вобрала в себя: удобность, наличие студенческой лицензии, практичность, точные мерки и приятный интерфейс.

Рисунок 2.6.2 – Пример моей собственной работы в САПР программе Fusion 360

2.7 Прототип внедрения на производство, 3D модель серверной комнаты с типом охлаждения на основе исследований

На основе исследований, приведенных в пункте [1.4], был выбран следующий вид чиллерных комнат, для прототипа внедрения в производство (Рисунок 2.7.1 и Рисунок 2.7.2).

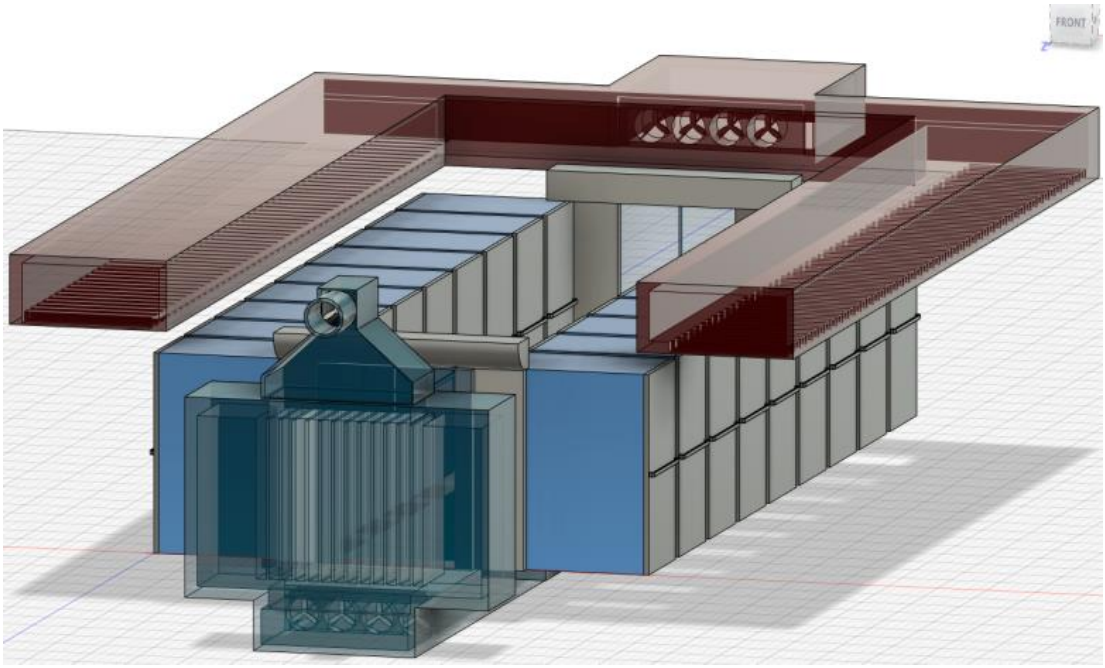


Рисунок 2.7.1 – 3D модель чиллерной комнаты в программе Fusion 360, вид сбоку

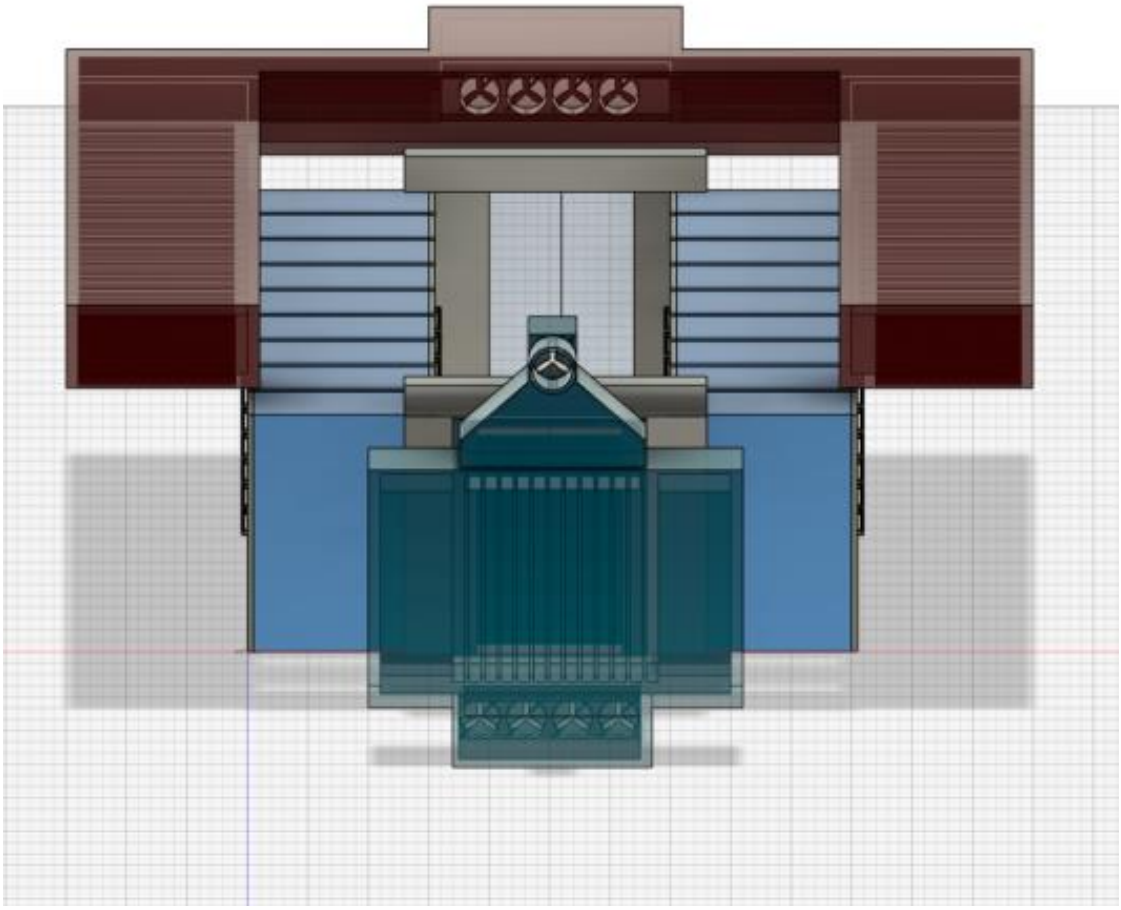


Рисунок 2.7.2 – 3D модель чиллерной комнаты в программе Fusion 360, вид спереди

Создание горячих и холодных коридоров помогает предотвратить смешивание горячего и холодного воздуха, позволяя более эффективно охлаждать более мощное оборудование.

Холодный контур (охлаждающий коридор) обозначен синим цветом и располагается снизу вдоль всей чиллерной комнаты – предназначается для охлаждения воздуха и подачи на вдув серверов. В нижней части холодного контура установлены 2 пары вентиляторов, которые управляют потоком воздуха. В верхней части холодного контура расположен основной вентилятор, нагнетающий холодный воздух в холодильную камеру.

Горячий контур (горячий коридор) обозначен красным цветом и находится сверху – через который выводится горячий воздух. В горячем контуре имеются также 2 пары вентиляторов, которые отводят горячий воздух из серверного помещения наружу.

2.8 Operational Tasks

Operational tasks — это критически важные действия и процессы отдела/команды, которые по определению имеют приоритет над всеми другими видами деятельности отдела. В группах информационных технологий (ИТ) это ежедневное и ночное производство.

В данном случае, под Operational tasks подразумевается план осуществления проекта, который в моем случае является таковым:

1) Техническая реализация продукта и создание первого прототипа
2) Экономический расчет с будущем перспективой
3) Создание бизнес-плана
4) Нахождение потенциальных производственных компаний
5) Участие в стартап конкурсах [2.9]
6) Демонстрация прототипа заинтересованным компаниям-производстам
7) Подписание договора на показание услуг по внедрению PLCDuino
8) Установка PLCDuino системы на производстве
9) Участие в госзакупках
10) Распространение продукта с помощью медиа и СМИ

2.9 Стартап конкурс «Open Space», потенциально заинтересованные производства и команда IQZ

«Open Space» является одной из стратегий открытых инноваций, используемых компанией AGT Global. Эта инициатива направлена на оценку и рассмотрение поступающих высокотехнологичных проектов и предложений о возможном сотрудничестве с инвесторами, партнерствами, холдинговыми компаниями или другими организациями внутри компании.

Данный проект, был представлен в данном стартап конкурсе с прохождением в финал, что является одним из главных факторов того, что данный проект имеет серьезные дальнейшие перспективы, так как победа в данном конкурсе будет означать полную экономическую поддержку со стороны спонсоров.

Название проекта: Разработка системы управления автоматизированной чиллерной с использованием ПЛК на базе семейства Arduino (PLCDuino)

Информация о конкурсанте	
Стадии развития Проекта	• идея (замысел) • разработка (есть макет, пробник, пример, тестовый вариант, тизер/ требует доработки и финансирования +) • завершен (требует финансирования) (Выбор одного из списка/галочка)
ФИО участника (полностью) и фото	Акимов Максим Александрович 
Год рождения	29.10.2000
E-mail: нужно адрес gmail.com	nukem.mainer@gmail.com
Удост, данные ИИН	Удост №040921939, ИИН: 001029500073
WhatsApp, Телеграмм	+, +
Город	Алматы
Номер телефона	87770244875
Информация о Проекте	(Выбор одного из списка/галочка)

Рисунок 2.9.1 – Фрагмент одобренной заявки на участие в конкурсе «Open Space»

Касательно заинтересованных компаний производств, были собраны необходимые сведения, был проведен опрос данных компаний, что могли бы позволить установить данную технологию на некоторые цеха. К данным производствам можно отнести: Приборостроительный завод «Saiman», ТОО «Хлебобараночный комбинат «Аксай», ТОО "Маслодел", "ТОО «Vita Bottlers Казахстан»".

При получении экономической поддержки со стартап-конкурса «Open Space», по operational tasks [2.8], начнется реализация данного проекта в массы.

Однако, данный проект является крупным, требует участия не только автора данного проекта, по этой причине автор проекта Акимов Максим Александрович создал команду из студентов-робототехников, под названием

– «Invictus Qazaqstan» (IQZ). По этой причине, полная установка данного продукта на производство не будет занимать много времени.

Рисунок 2.9.2 – Эксклюзивный label команды IQZ

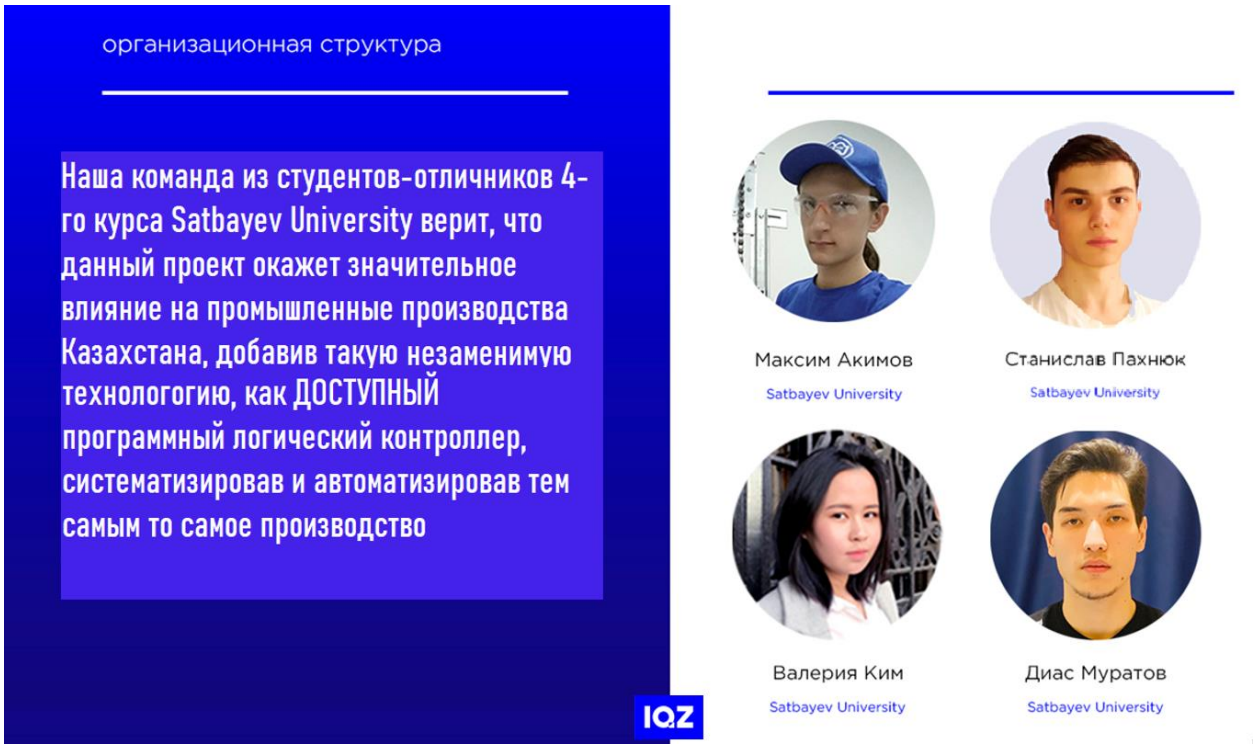


Рисунок 2.9.3 – Организационная структура команды IQZ

3 ПРАКТИЧЕСКАЯ ЧАСТЬ

3.1 Топология проекта

Первое, с чего надо начать, это с создания топологии, на которую будет дальнейшее ориентирование при дальнейшем проектировании. Топология данного проекта для промышленности выглядит следующим образом.

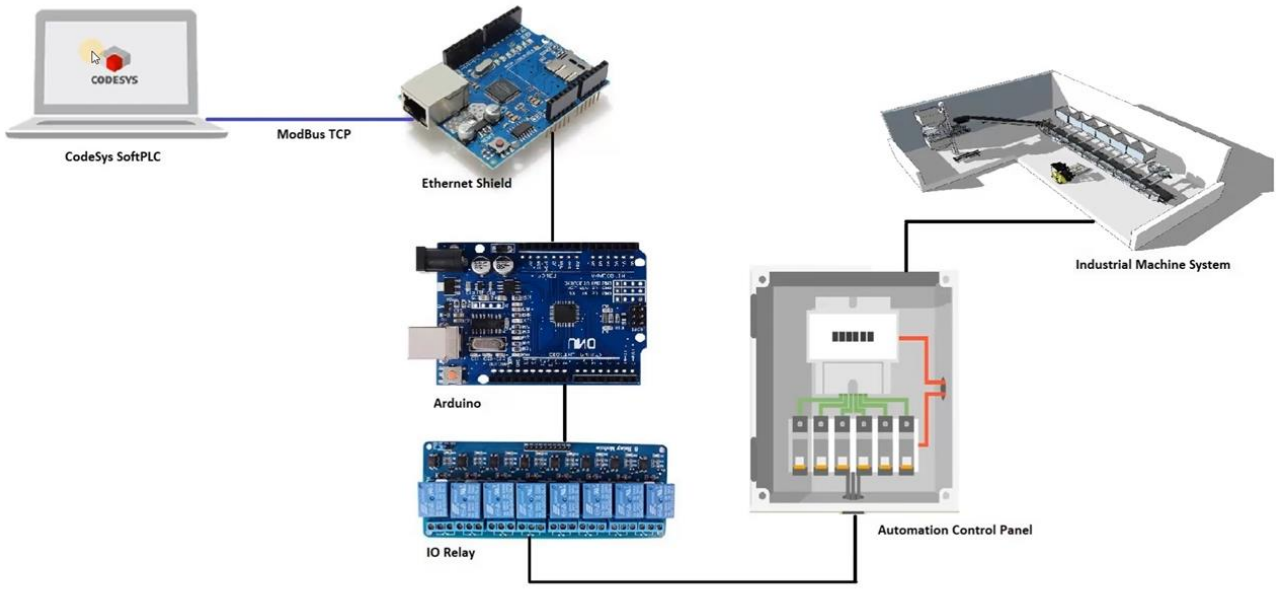


Рисунок 3.1.1 – Визуальная топология проекта

Что необходимо для реализации автоматизированного производства на PLCduino:

– Arduino UNO

– Arduino Ethernet shield

– PC/Laptop с Windows и CoDeSys

– 8-CH Arduino Relay external card with 5v 700mA minimum power supply

– Производство (в данном случае это будет модель чиллерной в сетевой комнате)

Вдобавок, все программное обеспечение, которое будет использоваться, является бесплатны, отметив, что это очень высокопрофессиональное программное обеспечение, [2.3] используемое крупнейшими компаниями [2.1] по всему миру, и оно ни в коем случае не может быть плохое.

Почему именно такая топология? По данной топологии я подключаю ПК (С установленной CoDeSys) по Modbus TCP к Ethernet Shield, который присоединён к Arduino, потому что прошивать контроллеры PLC через CoDeSys возможно только через данный протокол [2.2]. В тоже время Arduino

нельзя вывести напрямую к панели управления автоматикой какого-либо производства, потому что питание для контрольной панели необходимо 24V, по этой причине между Arduino и панелью управления автоматикой выступает 8-ми канальное реле в качестве посредника.

В данной схеме, Arduino является I/O Buffer-ом и коммуникатором между программируемый средой и панелью управления автоматикой.

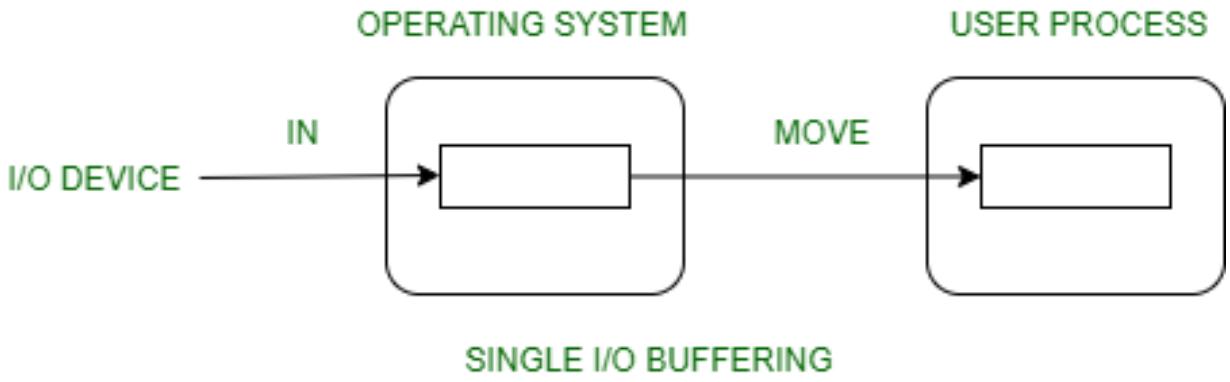


Рисунок 3.1.2 – Input/Output Buffer с единичной буферизацией

3.2 Электрическая принципиальная схема проекта

Далее будет отображена электрическая принципиальная схема чиллерного макета, спроектированная в программе Sprint LayOut.

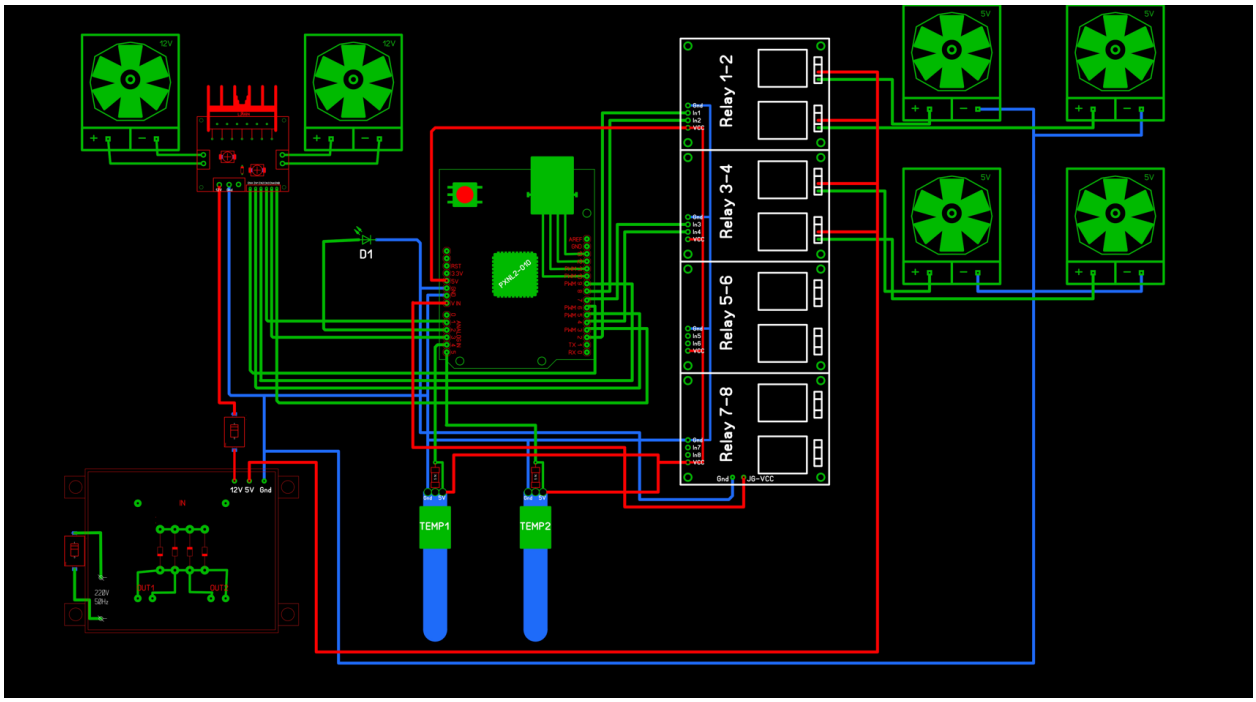


Рисунок 3.2.1 – Электрическая принципиальная схема чиллерной комнаты

По данной схеме – В середине находится плата Arduino UNO, к которой присоединен Ethernet Shield, пины 13,12,11,10 отведены на связь через Modbus TCP протокол с платой, 4 из 8-ми канального реле подключены к соответствующим пинам, а те в свою очередь к 4-ем куллерам на разрыв, питание все исходит из блока питания для ПК. Помимо этого, снизу схемы располагаются 2 датчика температуры, соединенные через резистор на 10 КОм. Слева сверху схемы находится драйвер двигателя, к которому присоединены еще 2 куллера, оборотами которых будет вестись прямое управление. Световой индикатор подключен ко 2-ому аналоговому пину.

Первая пара куллеров, оборотами которых будет вестись управление – 12V, остальные на 5V.

3.3 Кодовая реализация и соединение по протоколу Modbus TCP, «псевдокод»

В данном разделе будет отображена сокращенная step-by-step программная реализация проекта.

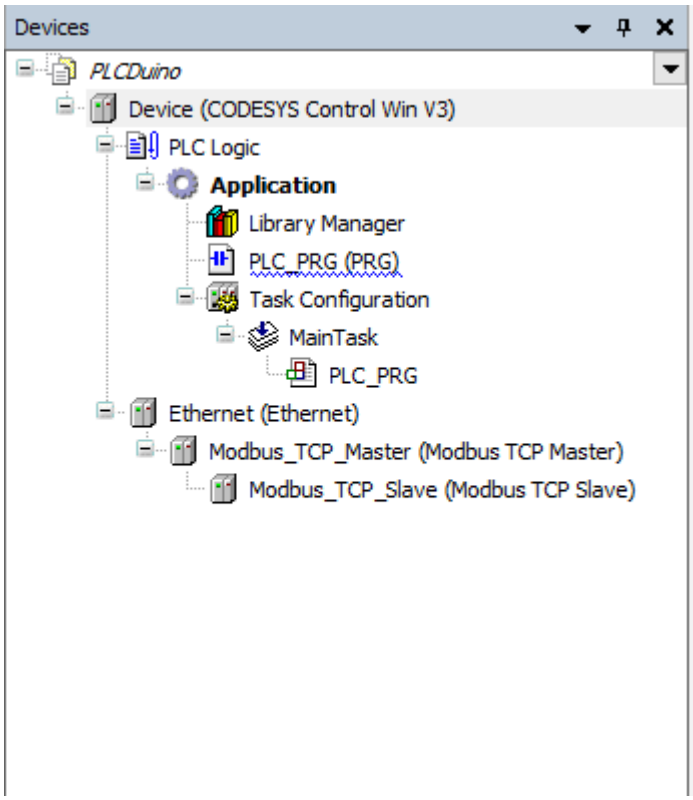


Рисунок 3.3.1 – Окно Devices в программе CoDeSys

Первым шагом нужно было создать PLC_PRG, основное окно программирования любого ПЛК, затем были созданы 3 ступени связи – Ethernet –> Modbus TCP Master –> Modbus TCP Slave.

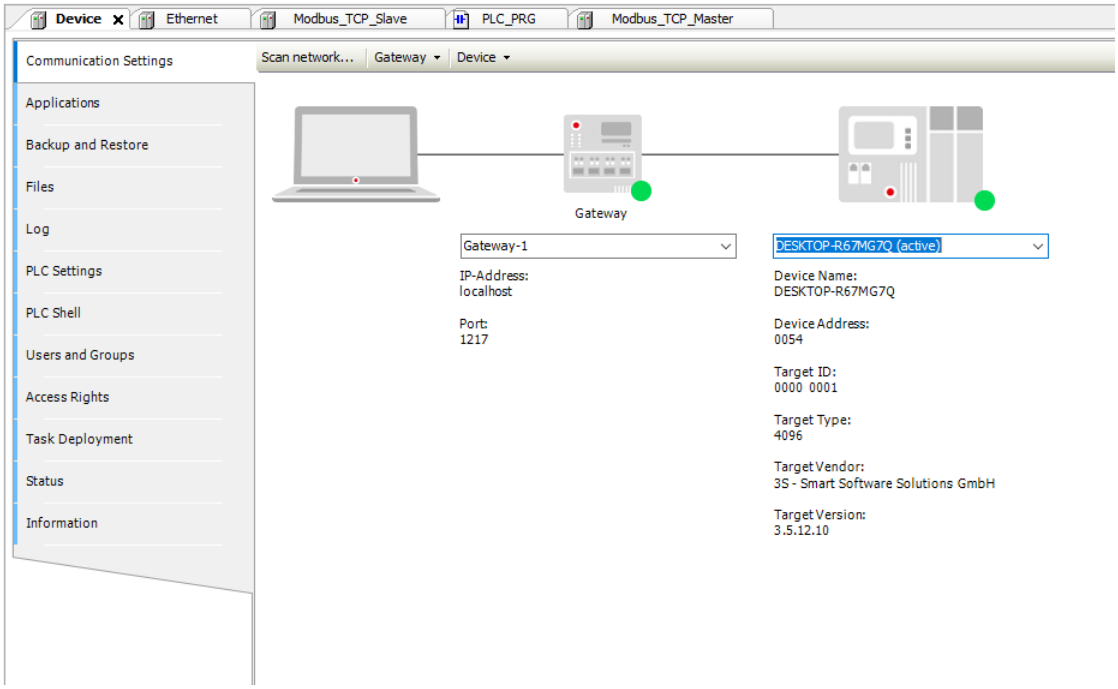


Рисунок 3.3.2 – Окно Communication Settings в программе CoDeSys

Следующий шаг состоит в том, чтобы по Ethernet сети установить сопряжение между ПК и ПЛК (Master-Slave Mode [2.2.1]). Как видно из Рисунка 3.3.2, связь по сетевому шлюзу (Gateway) по IP адресу прошла успешно. После этого был включен CoDeSys Control Win SysTray для сопряжения с ПЛК, был введен IP адрес роутера (IP Address), маска сети (Subnet Mask), стандартный сетевой шлюз (Default Gateway), – (Рисунок 3.3.3) что также было успешно. Для ПЛК был создан отдельный IP адрес, что отличается от IP адреса роутера.

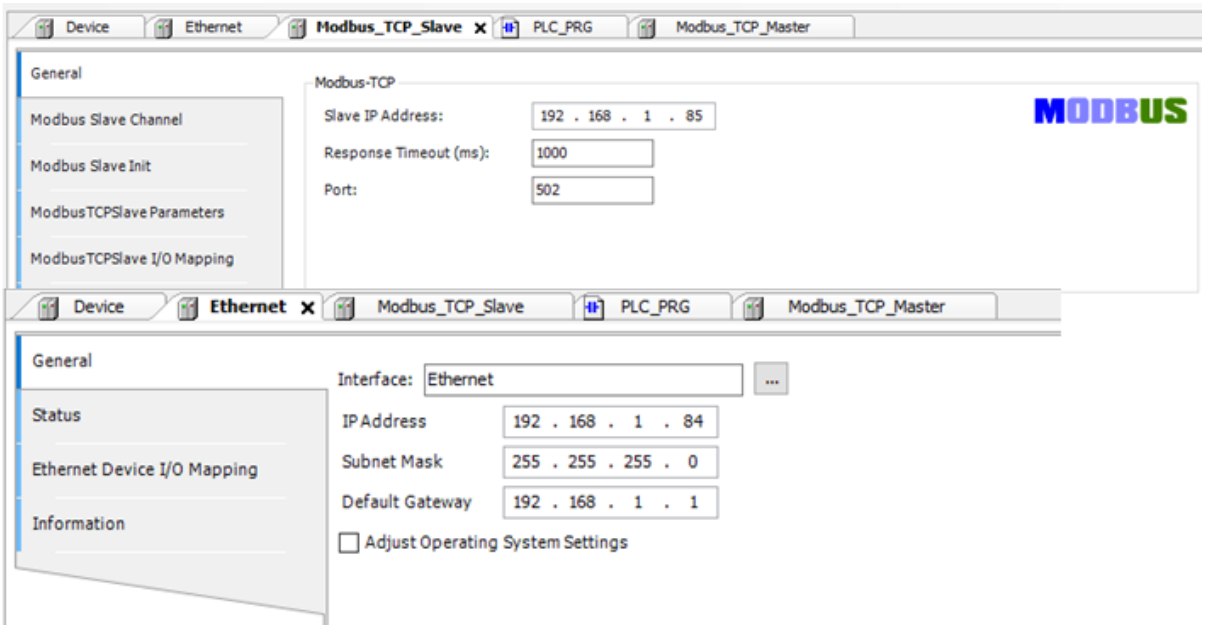
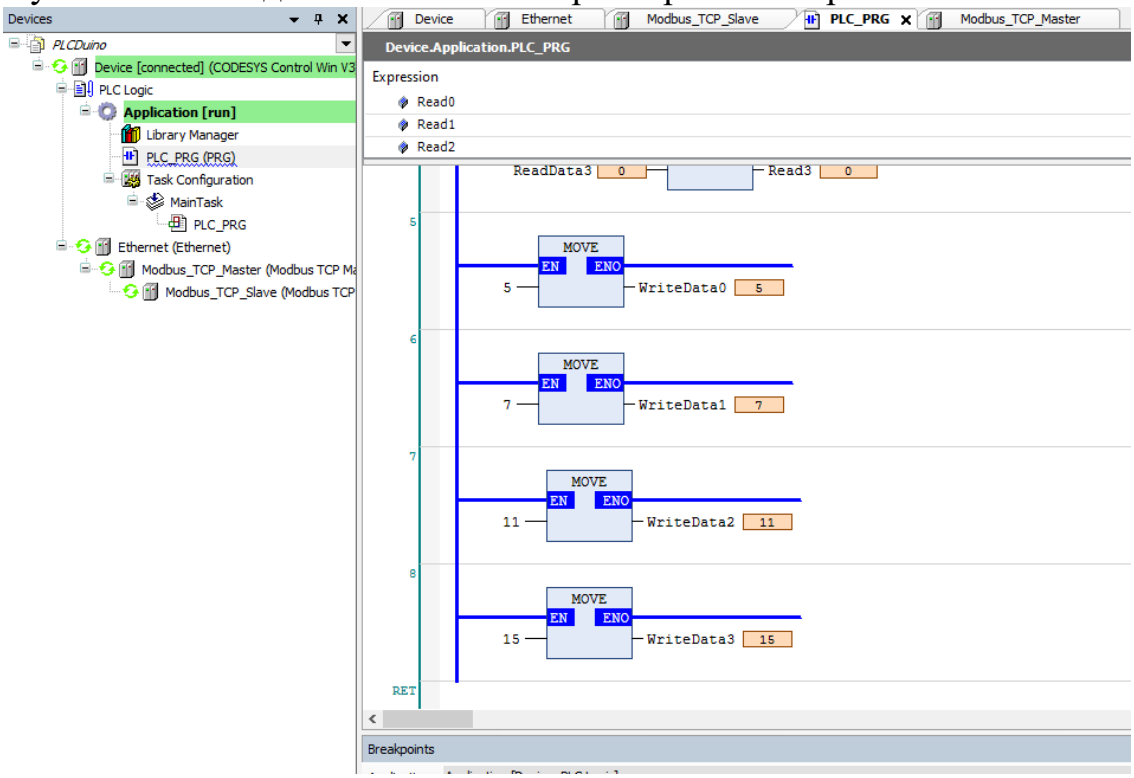


Рисунок 3.3.3 – Задавание сетевых параметров и настроек ПК и PLCDuino



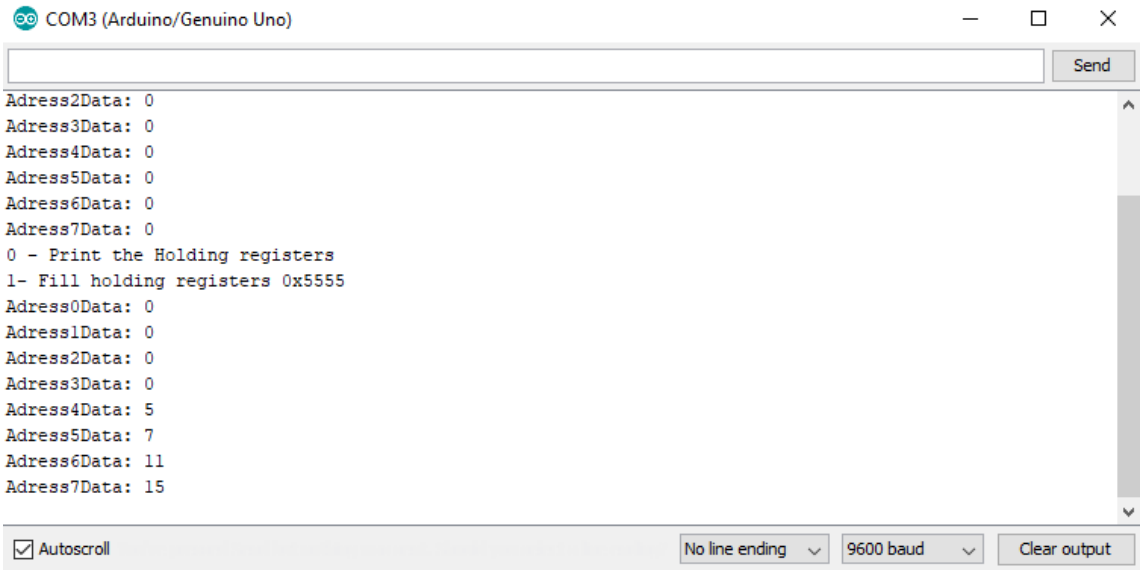


Рисунок 3.3.4 – Демонстрация сопряжения между ПК и PLCDuino

Чтобы проверить работоспособность данного сопряжения, был написал код [Приложение С], отправляющий 4 бита информации и принимающий 4 бита. Далее, в Serial Port отправляется сигнал, являющийся триггером срабатывания отправки и принятия информации. Как видно из Рисунка 3.3.4, в программе CoDeSys в блоке MOVE было отправлено 4 бита информации, содержащие в себе разные цифры (5,7,11,15). Serial Port в Arduino IDE принял их и вывел их на монитор, данные записались в WriteData. Тоже самое и с обратной стороны, на Рисунке 3.3.5 обратная схема передачи информации, где из Arduino IDE посылаются биты информации, а CoDeSys их принимает и записывает их в ReadData.

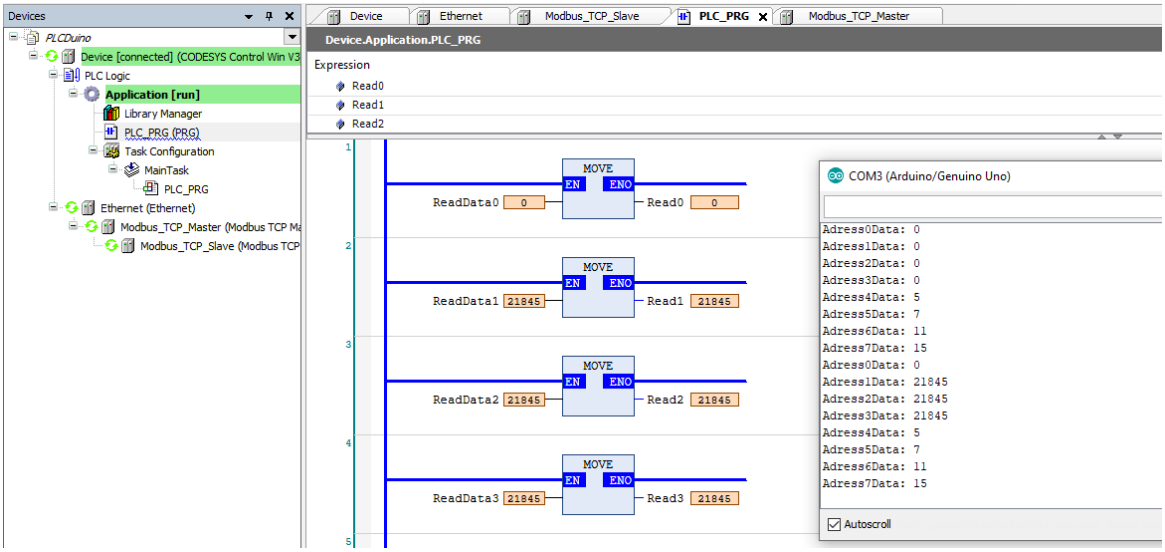


Рисунок 3.3.5 – Отправка информации из Arduino IDE в CoDeSys

После проверки сопряжения между CoDeSys и PLCDuino, была создана схема на языке LD [2.3].

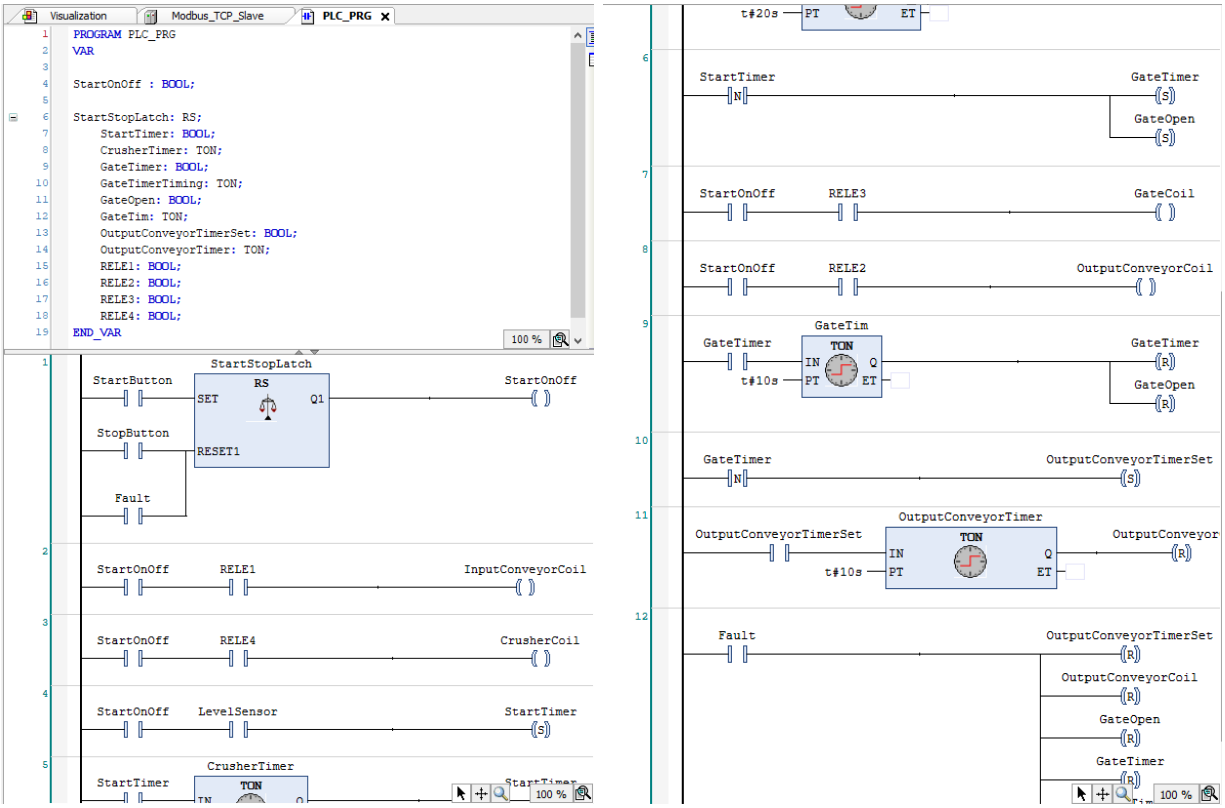


Рисунок 3.3.6 – Полная схема работы чиллерной на языке LD в среде CoDeSys

Полное обьяснение работы схемы описано в разделе [3.3.1].

Variable	Mapping	Channel	Address	Type	Unit	Description
		Channel 0	%QW0	ARRAY [0..3] OF WORD		Write Multiple Registers
		Channel 0[0]	%QW0	WORD		0x0000
StartButton		Bit0	%QX0.0	BOOL		
StopButton		Bit1	%QX0.1	BOOL		
InputConveyor...		Bit2	%QX0.2	BOOL		
OutputConvey...		Bit3	%QX0.3	BOOL		
CrusherCoil		Bit4	%QX0.4	BOOL		
GateCoil		Bit5	%QX0.5	BOOL		
		Bit6	%QX0.6	BOOL		
		Bit7	%QX0.7	BOOL		
		Bit8	%QX1.0	BOOL		
		Bit9	%QX1.1	BOOL		
		Bit10	%QX1.2	BOOL		
		Bit11	%QX1.3	BOOL		
		Bit12	%QX1.4	BOOL		
		Bit13	%QX1.5	BOOL		
		Bit14	%QX1.6	BOOL		
		Bit15	%QX1.7	BOOL		
InputConveyorFreq		Channel 0[1]	%QW1	WORD		0x0001
OutputConveyorFreq		Channel 0[2]	%QW2	WORD		0x0002
CrusherFreq		Channel 0[3]	%QW3	WORD		0x0003
		Channel 1	%IW0	ARRAY [0..0] OF WORD		Read Holding Registers
		Channel 2	%IW1	ARRAY [0..3] OF WORD		Read Holding Registers
Temp1		Channel 2[0]	%IW1	WORD		0x0006
Temp2		Channel 2[1]	%IW2	WORD		0x0007
		Channel 2[2]	%IW3	WORD		0x0008
		Channel 2[3]	%IW4	WORD		0x0009

Рисунок 3.3.7 – Полный список «слов» чиллерной

На Рисунке 3.3.7 отображаются созданные «слова» (WORD), что хранят в себе биты информации той или иной переменной, в каждый из низ записываются либо значение BOOL, как к примеру в канале Channel 0[0], либо записываются целочисленные значения, как в каналах Channel 0 [1-3] и Channel 2[0-1]. Но также здесь имеются и каналы, что наоборот считывают информацию – Channel 1-2.

Рисунок 3.3.8 – Код чиллерной в Arduino IDE

Полная версия этого кода находится в [Приложение В], а объяснение в разделе [3.3.1]

3.3.1 «Псевдокод»

Псевдокод — это способ выражения логики программы без необходимости написания реального кода. Это позволяет спланировать структуру и поток вашей программы, прежде чем писать ее на языке программирования. Либо для отображения уже готового кода, но объяснив его максимально доступным для понимания языком.

Таблица 3.3.1.1 – Псевдокод Arduino IDE

Инициализация
#Включение библиотек <Ethernet, SPI, MgsModbus, microDS18B20>
Создание МАК адреса с сеткой {0x90 , 0xA2 , 0xDA , 0x0E ,0x94 , 0xB5};
Создание IP адреса у PLCDuino (192,168,1,85);
Задаем адрес локального роутера(192,168,1,1); Задаем адрес маски подсети (255,255,255,0);
Создаем переменные bool, int;
Задаем имена переменных к конкретным пинам;
Область единовременного включения
Задавание пинам их состояния (OUTPUT, INPUT);
Включения серийного порта;
Активация Ethernet подключения (МАК адрес, ip, адрес локального роутера, маска сети);
Считывание Кодесисом четырех битов [0-3]=0;
Запись в Кодесис четырех битов [4-7]=0;
Область основной части кода (цикл)
Присваивание каждой переменной состояние каждого бита [0]>>0-5)&1
ЕСЛИ (кнопка СТАРТ включена И кнопка СТОП выключена) {
ТО состояние битов отправляются на пины;
ЕСЛИ (происходит считывание с датчика температуры 1);
ТО переменная датчика ds2 запрашивает состояное температуры;
И бит номер 6 записывает его состояние;
ЕСЛИ (происходит считывание с датчика температуры 2);
ТО переменная датчика ds запрашивает состояное температуры;
И бит номер 7 записывает его состояние; }
ИНАЧЕ {
Все переменные получают негативный сигнал; }
ЕСЛИ (нажата кнопка СТОП) {
ТО (аварийный светодиод включается); }
ЕСЛИ (Серийный порт включен) {
Переменная inByte считывает значения переменных;
ДО ТЕХ ПОР ПОКА (I не станет 4) { Серийный монитор выводит состояние битов;}
И Серийный монитор выводит значения с битов считывания и записи;

Серийный монитор обновляется если ввести цифру 0; } Сервер включен и работает }

Код в Arduino IDE выполняет исключительно роль исполняющего устройства (Slave), логические задачи он не выполняет.

Таблица 3.3.1.2 – Псевдокод CoDeSys

ЕСЛИ включена кнопка СТАРТ ТО RS триггер выводится в состояние SET и активируется переменная включения ЕСЛИ нажата кнопка СТОП ИЛИ активировалась переменная аварии ТО RS триггер выводится в состояние RESET и переменная выключается ЕСЛИ переменная старта и первое/второе/третье/четвертое реле включены ТО первый/второй/третий/четвертый вентилятор включается ЕСЛИ активна переменная включения И датчик температуры 1/2 считывает температуру ТО включается переменная таймера 1/2 Таймер 1/2 начинает отсчет и при окончании происходит RESET переменной таймера 1/2 ЕСЛИ на переменную СТОП подается сигнал ТО переменная включения деактивируется ЕСЛИ переменная аварии включается ТО все переменные сбрасываются на ноль И включается таймер аварии

Код в CoDeSys выполняет исключительно роль командующего устройства (Master), логические задачи он выполняет.

3.4 Визуализация в CoDeSys, отображение панели управления ПЛК

Визуализация в CoDeSys сделана на подобии панелей управления с реальных производств, где установлены реальные промышленные ПЛК типа [2.1].

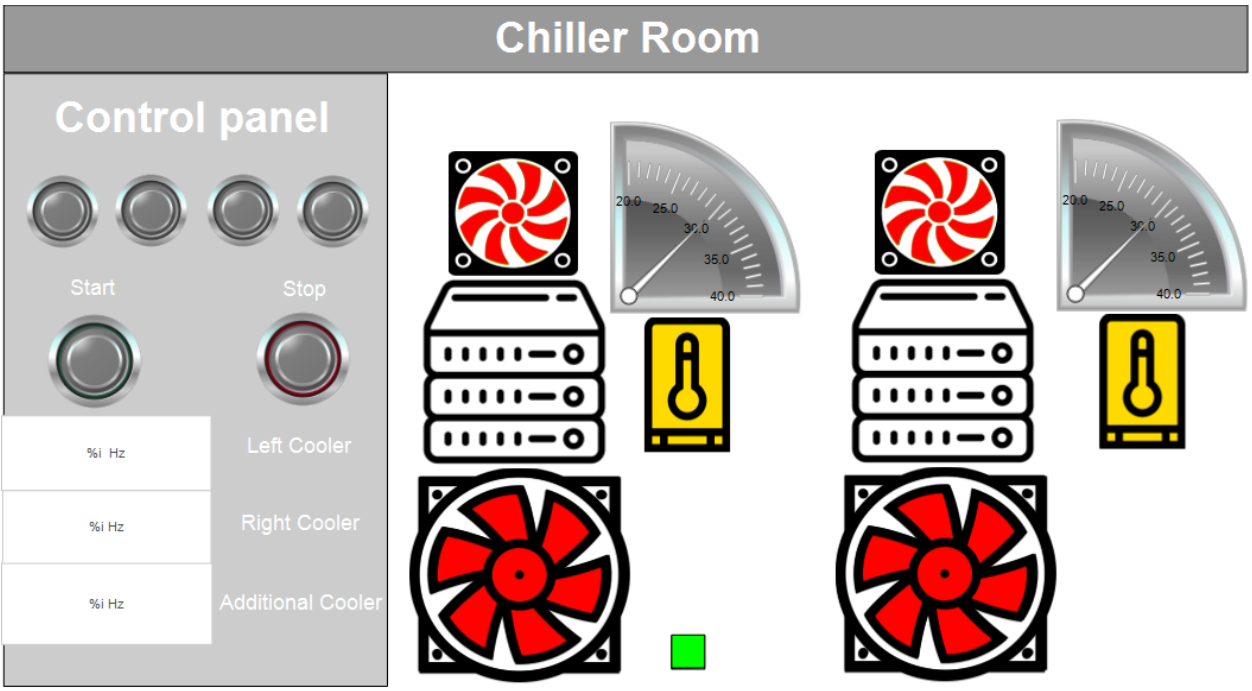


Рисунок 3.4.1 – Визуализация, контрольная панель чиллерной комнаты

Control panel – Контрольная панель управления, кнопки включения системы (Start), отключения (Stop) и каждого куллера по отдельности.

Left cooler, Right cooler, Additional cooler – Задавание скорости оборотов конкретных куллеров (RPM).

Верхние и нижние куллера – Каждый куллер имеет 3 индикатора цвета – белый (система отключена), зеленый (система работает и куллер в рабочем состоянии) и красный (система отключена).

Стрелочные индикаторы – Вывод актуальной на момент времени информации с датчиков температуры DS18B20.

Индикаторы температуры – Состояние датчиков температуры DS18B20 с двумя индикаторами цвета – белый (система отключена) и желтый (датчики в рабочем состоянии и считывают температуру).

Зеленый индикатор – Индикатор аварии – зеленый (нет аварии) и красный (авария).

3.5 3D моделирование корпуса ПЛК

Как было описано в [2.6], 3D модель корпуса для PLCDuino сделана также на Fusion 360.

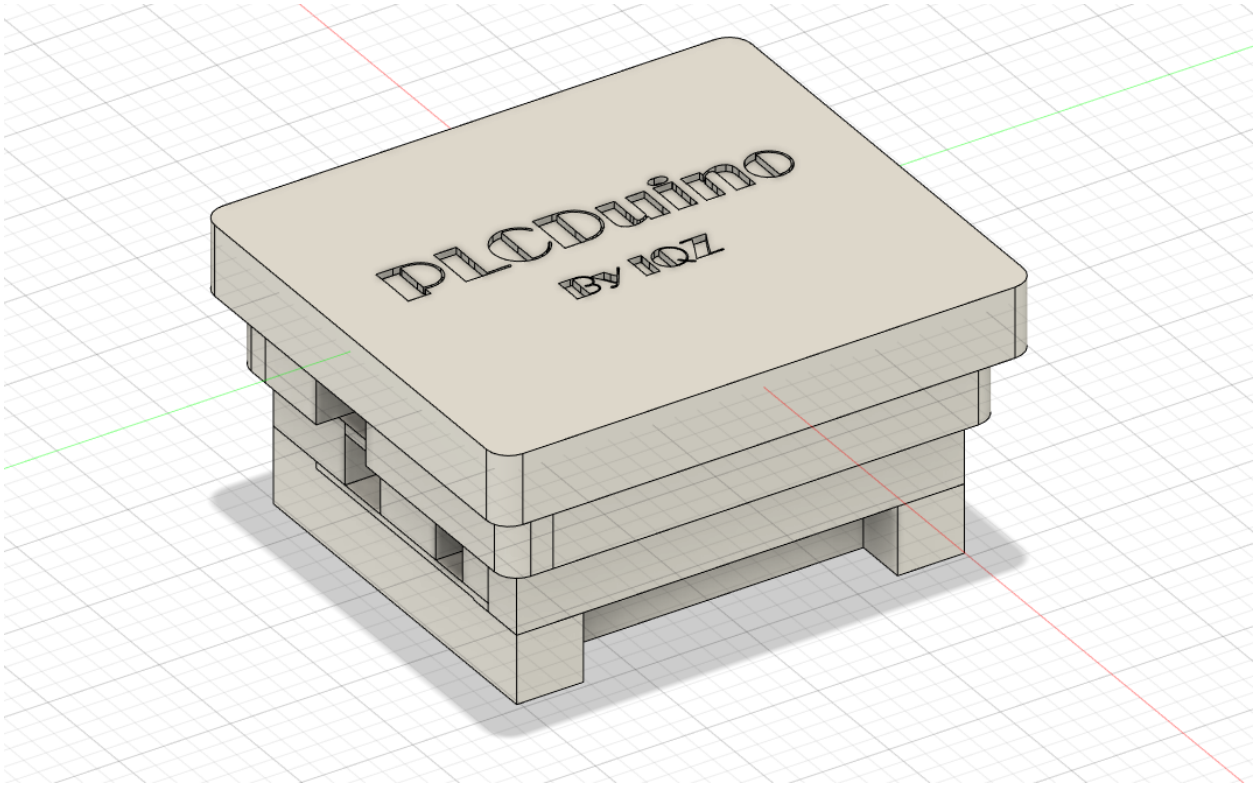


Рисунок 3.5.1 – 3D модель корпуса в программе Fusion 360

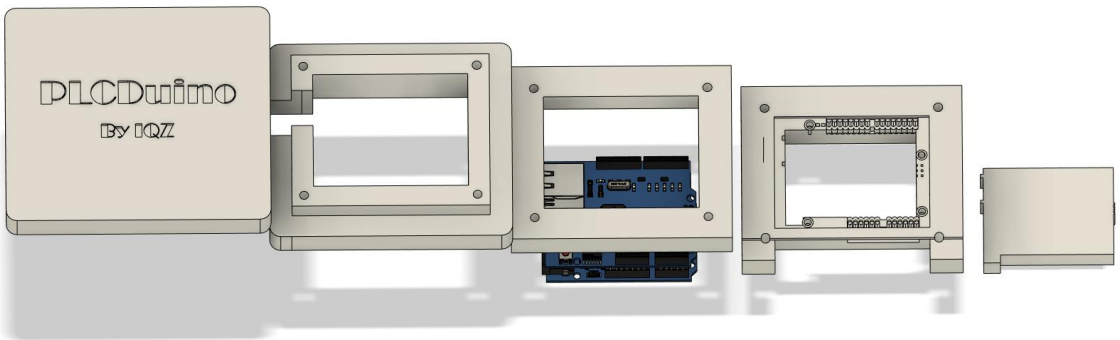


Рисунок 3.5.2 – 3D модель корпуса в программе Fusion 360 в разрезе

Для того, чтобы спроектировать точный корпус для PLCDuino, необходимо было спроектировать сначала компоненты, что будут помещены туда, а именно – Arduino UNO и Ethernet Shield.

Данных компонентов в открытом доступе не было найдено, поэтому пришлось моделировать каждый из элементов по отдельности.

Точность моделирования данных плат предсталены на Рисунке 3.5.3 и Рисунке 3.5.4.

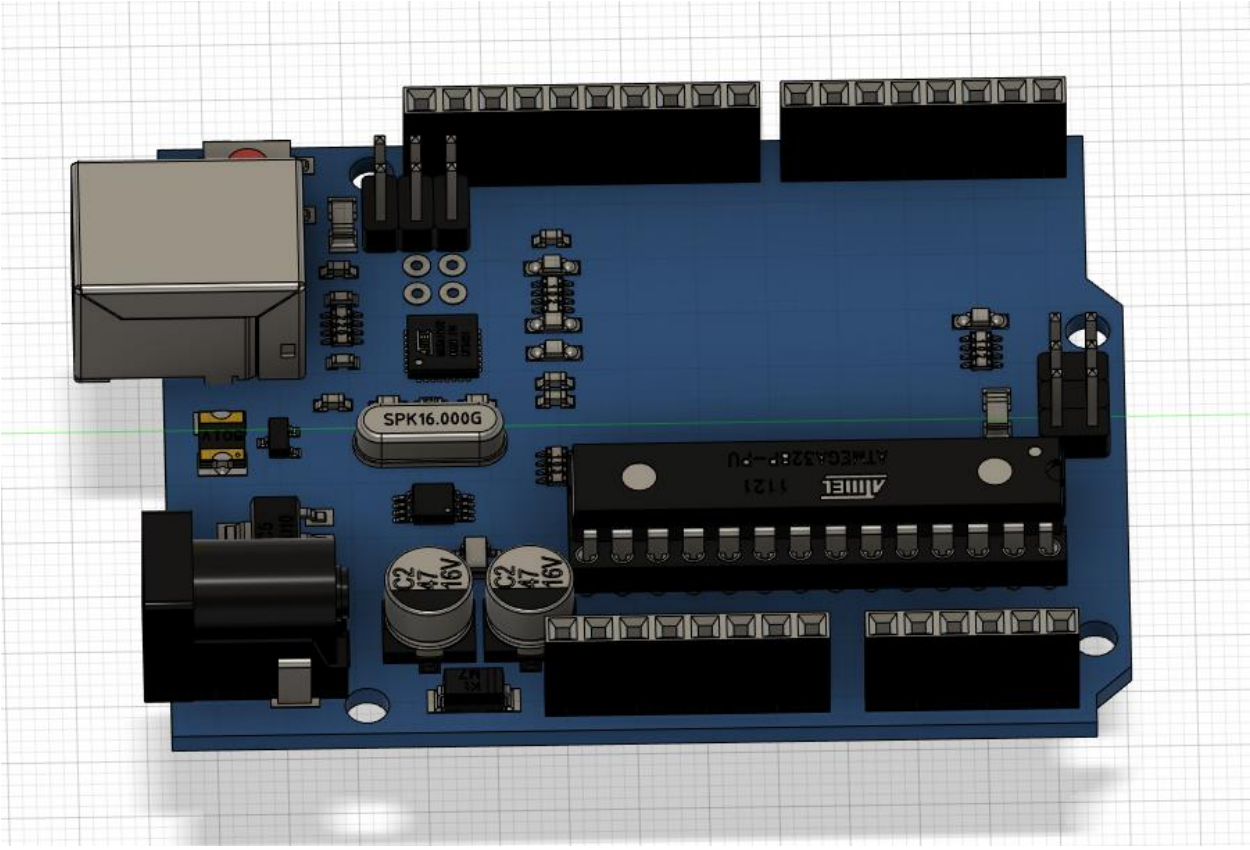


Рисунок 3.5.3 – 3D модель Arduino UNO в программе Fusion 360

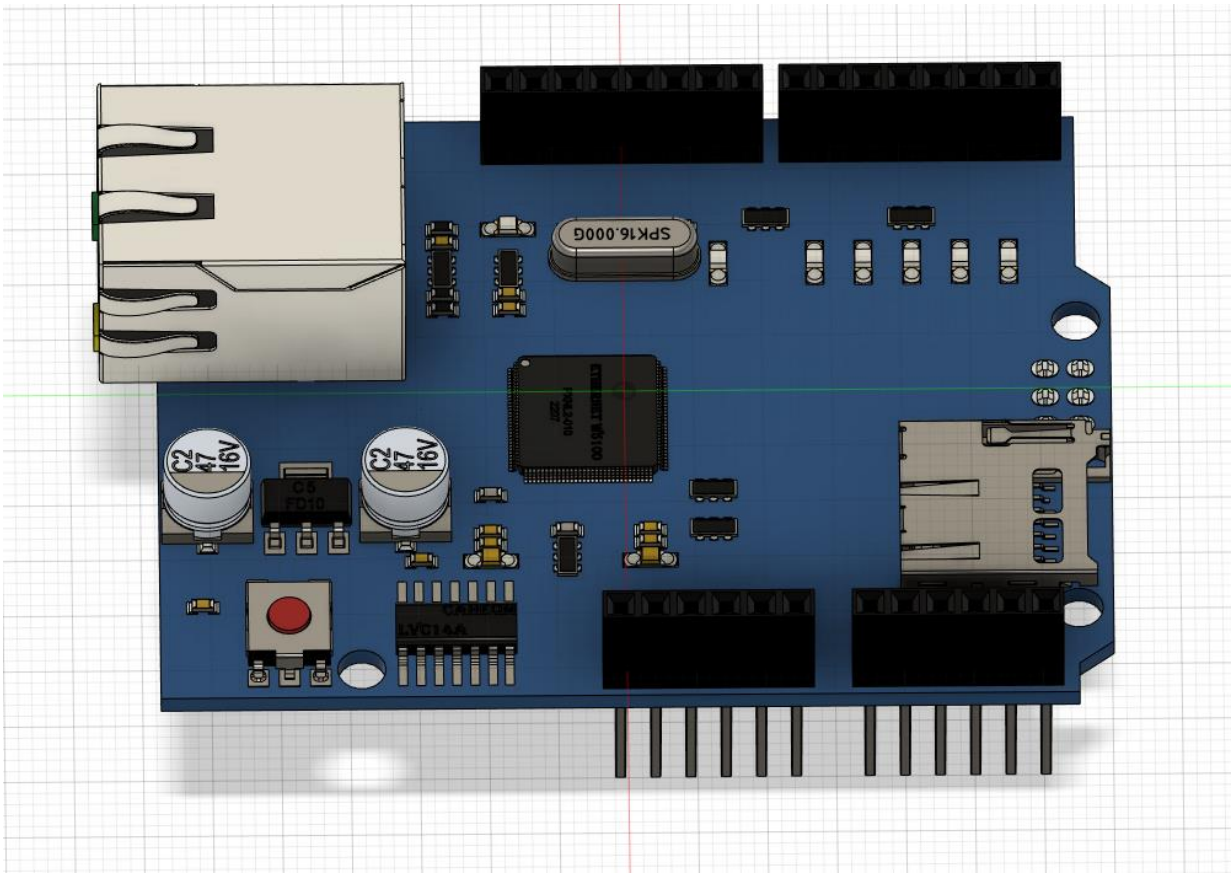


Рисунок 3.5.4 – 3D модель Ethernet Shield в программе Fusion 360

Данные модели являются точными копиями по размерам и меркам, что и их оригиналы. Делались данные модели по данным меркам, взятые из datasheet данных самих устройств.

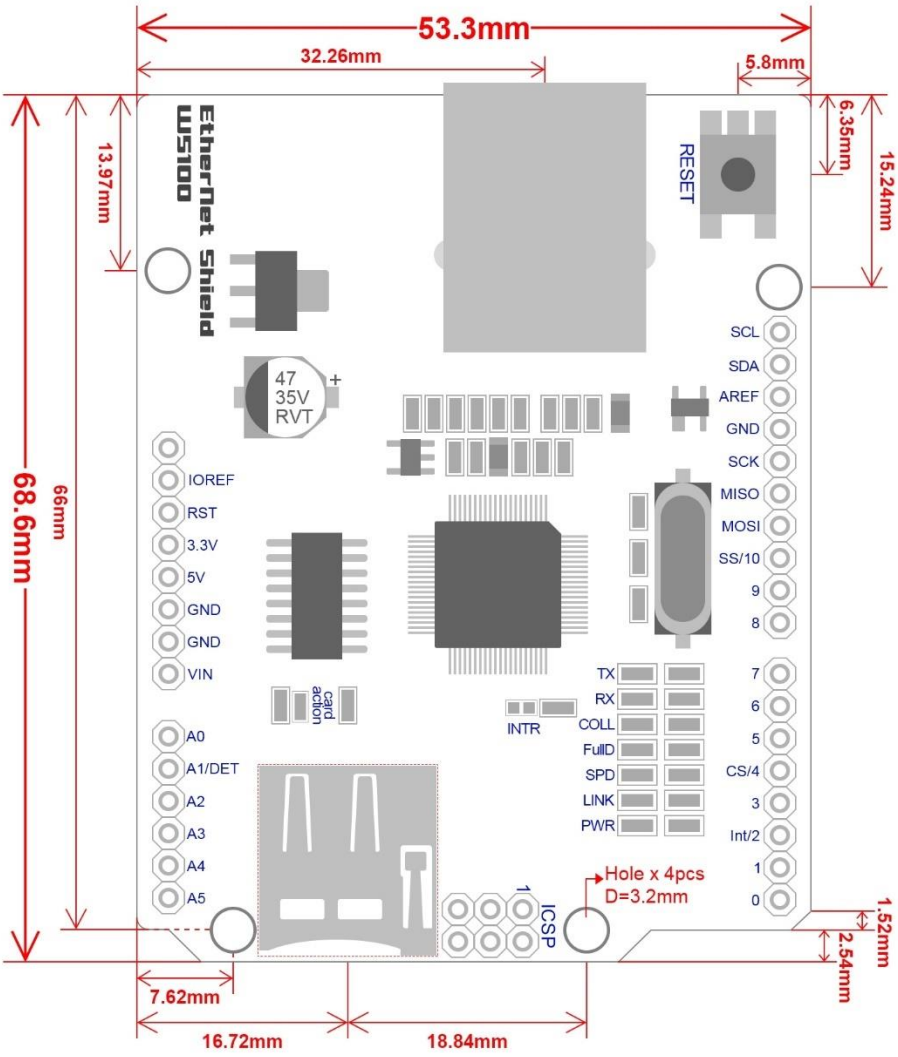


Рисунок 3.5.5 – Мерки старой версии Ethernet Shield

Рисунок 3.5.6 – Распечатанный на 3D принтере корпус PLCDuino

3.6 Создание макета и демонстрация работоспособности

В данной главе находятся рисунки предварительного внешнего вида устройства, дальнейшие разработки добавят лишь внешний вид устройству, но функционал полностью останется неизменным.

Также показано основные рабочие моменты, более подробная работа показана на видеозаписи работы устройства.

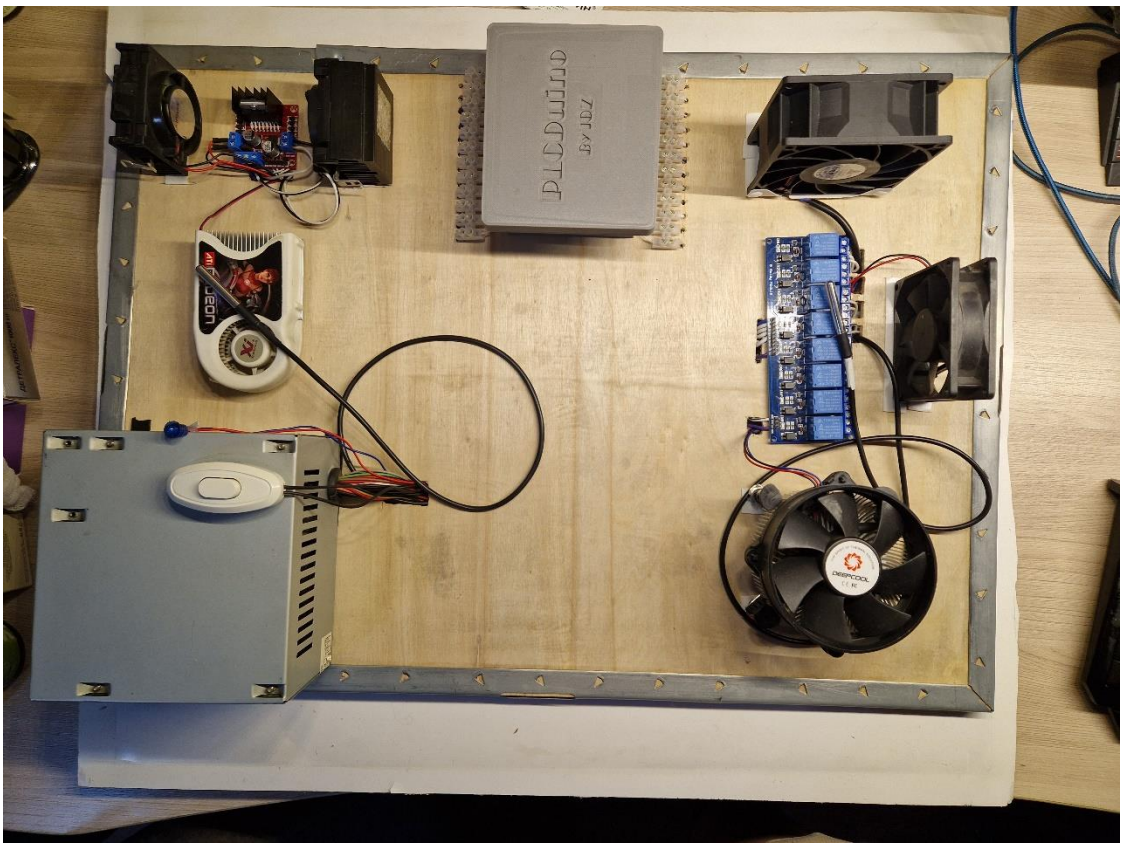


Рисунок 3.6.1 – Внешний вид макета чиллерной комнаты на PLCDuino

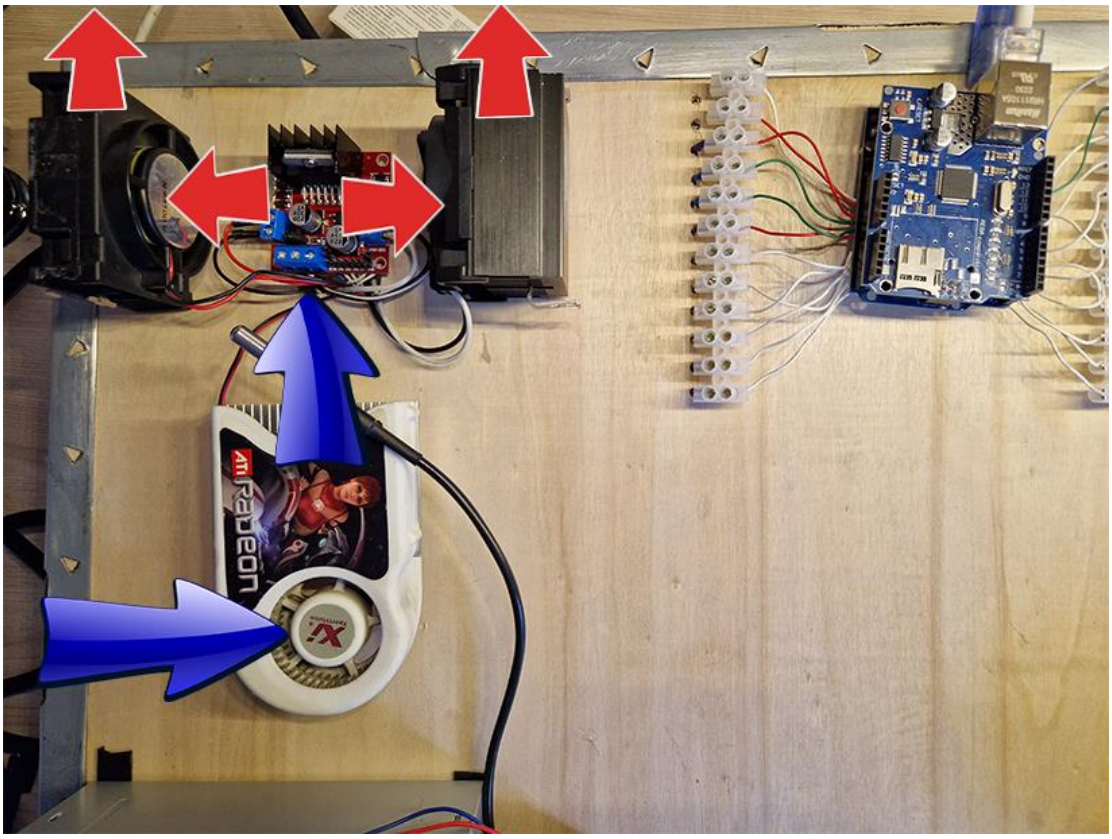


Рисунок 3.6.2 – Сегмент с куллерами и управляемыми оборотами

В данном сегменте имеются 3 куллера с управляемыми оборотами, количество оборотов на куллерах (Rotors Per Moment) устанавливается через контрольную панель PLCDuino.

Как видно на Рисунке 3.6.2, в нижний куллер через верх подается холодный воздух, в данном случае он является «холодным коридором». Дальше, холодный воздух охлаждает условный сервер, коим является драйвер двигателя L298N. В то время, 2 соседних куллера забирают горячий воздух и прогоняют его наружу, создавая тем самым «горячий коридор». Датчик температуры фиксирует и отправляет данные об состоянии данной комнаты на контрольную панель PLCDuino.

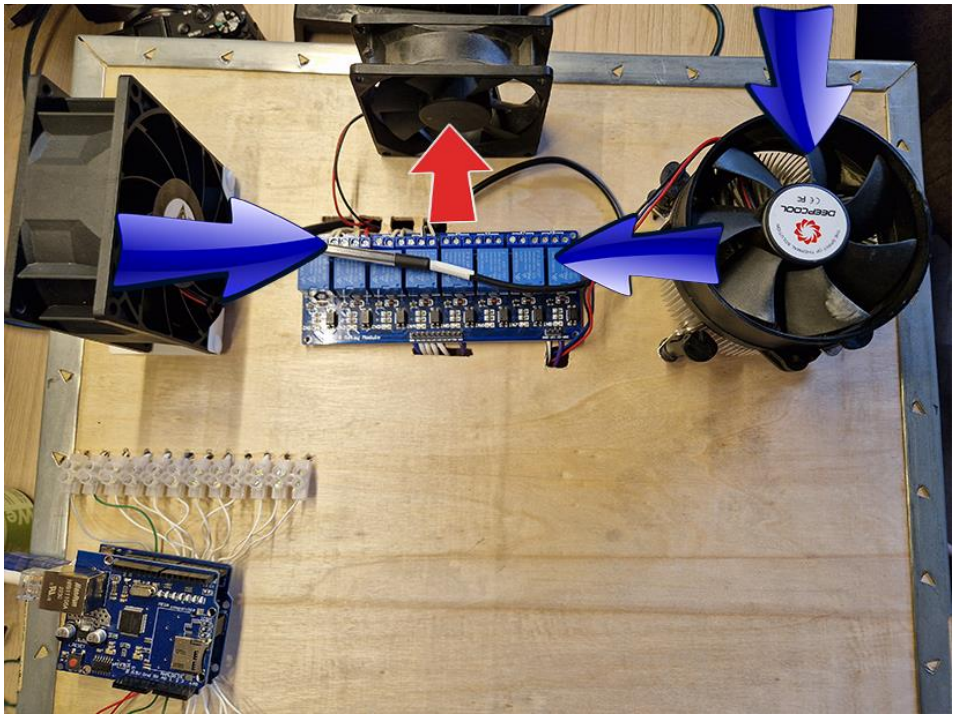


Рисунок 3.6.3 – Сегмент с мощными куллерами

Следующий сегмент состоит из трех мощных куллеров, еще одним условным сервером, являющимся 8-ми канальным реле и датчиком измерения температуры. В данном случае, забор холодного воздуха происходит из двух источников, сбоку и сверху – 2 «холодных коридора». Тем временем третий куллер забирает горячий воздух с условного сервера и выпускает прочь через «горячий коридор».

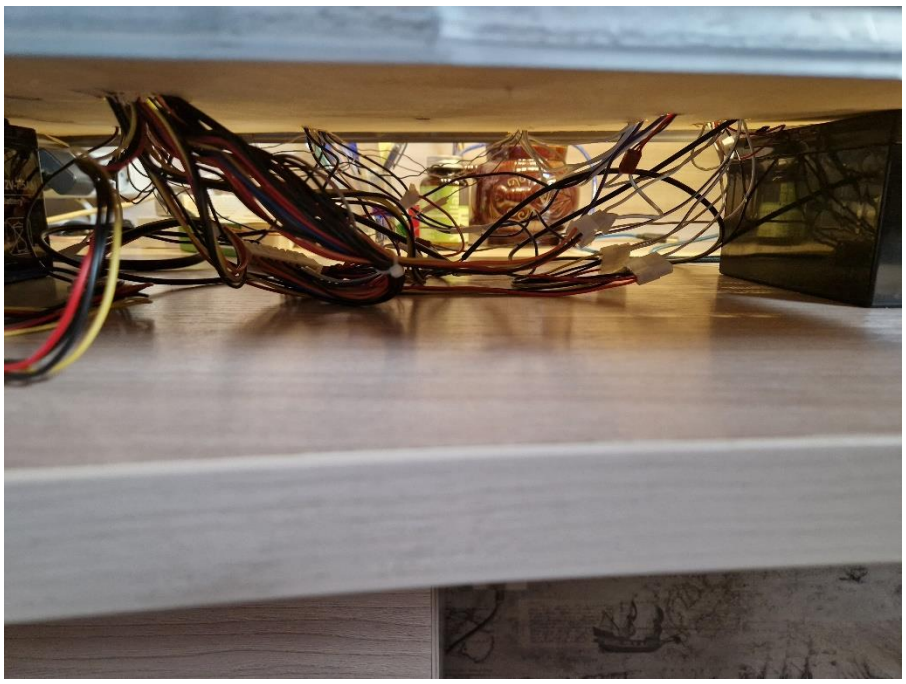


Рисунок 3.6.4 – Проводка по схеме [3.2]

Вся проводка проведена под макетом, выполнена качественно, запаяна и изолирована.

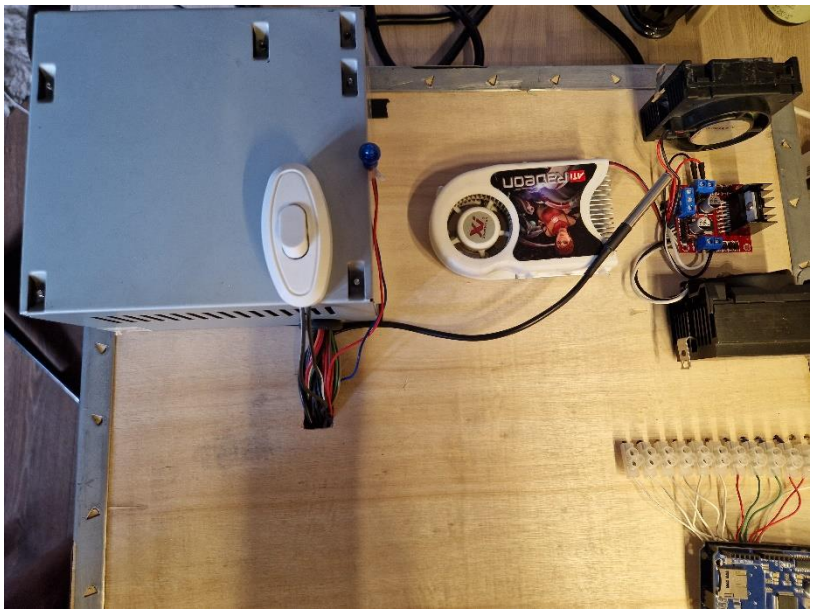


Рисунок 3.6.5 – Дополнительные элементы макета (блок питания, тумблер на питание и светодиод)

4 СПЕЦИАЛЬНАЯ ЧАСТЬ

4.1 Расчет надежности

Для максимальной эффективности проекта, система должна работать как можно дольше без отказа, по этой причине ниже будут расписаны основные показатели надежности:

1) Показатели безотказности – это способность объекта оставаться работоспособным в течение определенного периода времени. Для начала выведу информацию по каждому устройству, сколько времени каждый компонент может оставаться в рабочем состоянии.
   А) Arduino UNO. T_{UNO} = 88000 часов.
   Б) Arduino Ethernet Shield. T_{ES} = 70000 часов.
   В) Светодиоды. T_{L} = 50000 часов.
   Г) 8-ми канальное реле. T_{RL} = 100000 часов.
   Д) Датчик температуры SNS-TMP10K-1M. T_{TEMP} = 80000 часов.
   Е) Драйвер двигателя L298N. T_{LN} = 130000 часов.
   Ж) Блок питания компьютера. T_{PW} = 100000 часов.
   З) Вентилятор для корпуса ID-Cooling NO-8025-SD. T_{C} = 300000 часов.
   Каждый из этих показателей были рассчитаны заранее и взяты с официальной технической документации (datasheet).
2) Значение интенсивности отказов элемента - это вероятность отказа, которая определяется на основе предположения, что объект не вышел из строя до указанного момента времени.

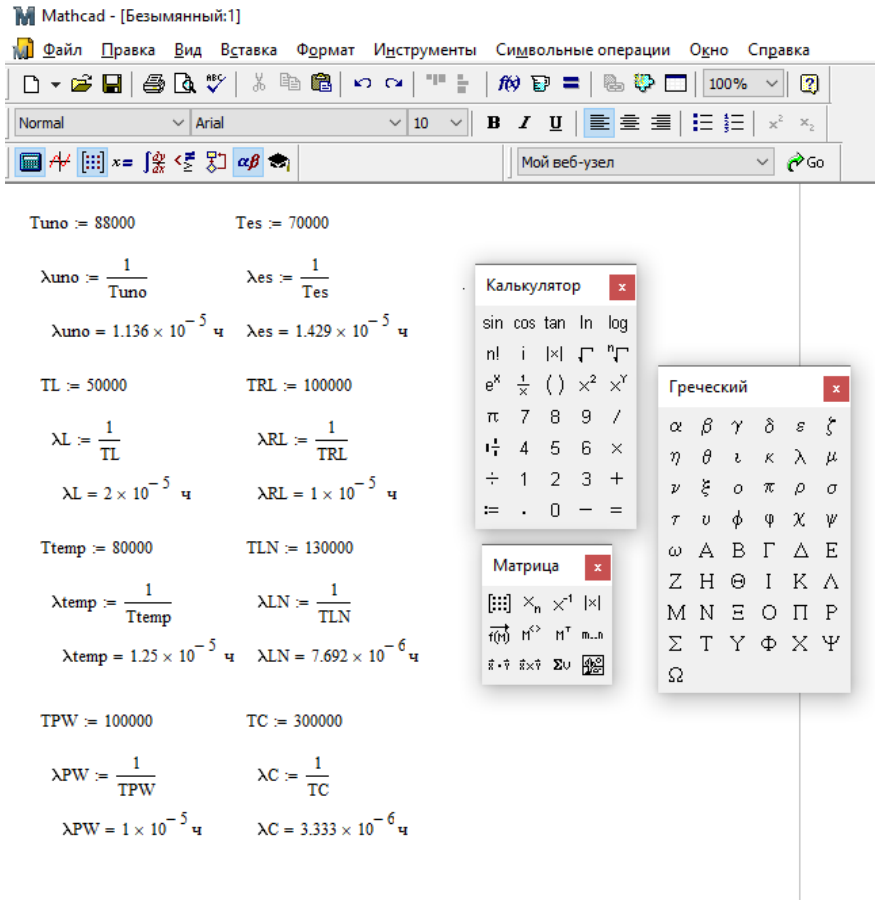


Рисунок 4.1.1 – Расчет в Matcad значения интенсивности отказов элемента

3) Вероятность безотказной работы элементов системы надежности при экспоненциальном законе распределения времени до отказа.

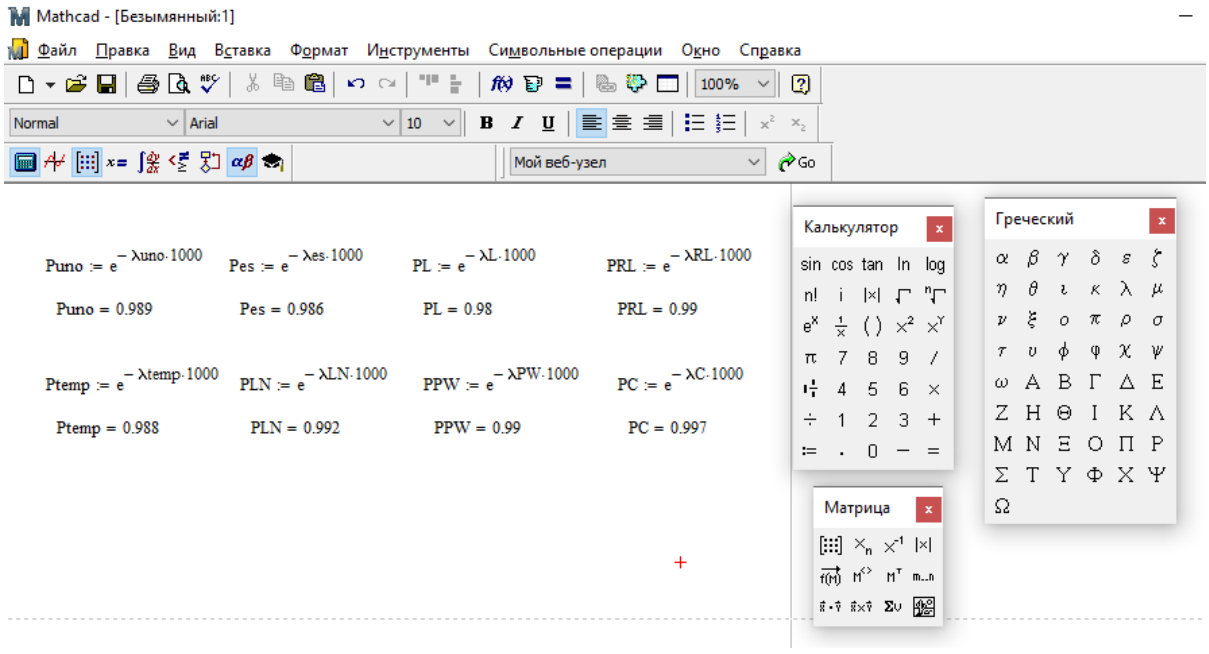


Рисунок 4.1.2 – Расчет в Matcad значения вероятности безотказной работы

Расчет был проведен относительно времени t, где t=1000 часов

4) Общие показатели надежности всей системы

Вероятность безотказной работы всей системы в течении времени t=1000 часов с последовательной структурой.

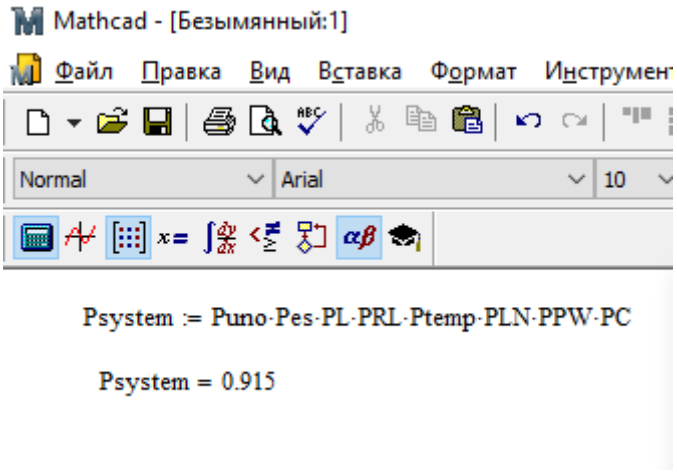


Рисунок 4.1.3 – Расчет в Matcad значения вероятности безотказной работы

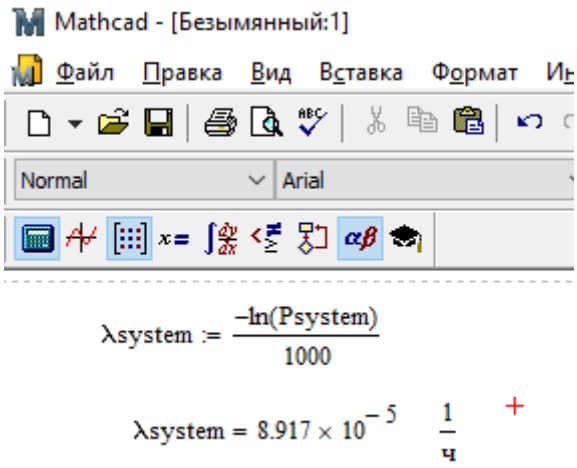


Рисунок 4.1.4 – Расчет в Matcad значения интенсивности отказов всей системы

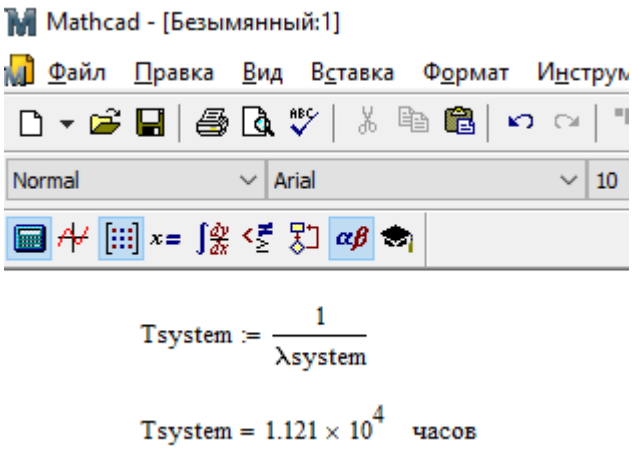


Рисунок 4.1.5 – Расчет в Matcad значения средней наработки до отказа системы

Если система будет работать 8 часов в неделю и 250 дней в году (нормированный рабочих график), средняя наработка в годах будет 5.6 лет.

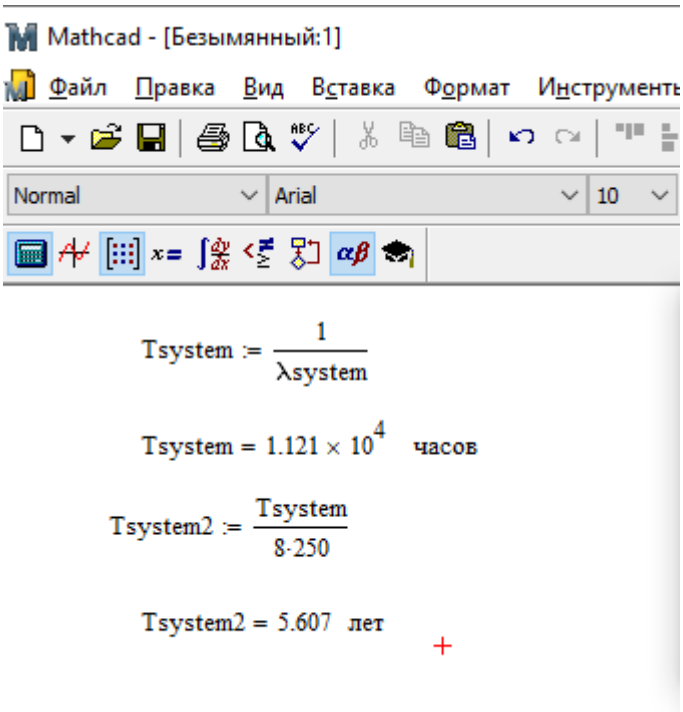


Рисунок 4.1.6 – Расчет в Matcad значения средней наработки в годах

Анализ надежности автоматизированной системы управления вентиляцией показал, что она имеет высокую вероятность безотказной работы в течение 1000 часов, что составляет 0,915. Это объясняет, почему использовались микроконтроллер Arduino Uno. Дополнительно могут быть приняты организационные меры для дальнейшего повышения надежности,

такие как повышение бдительности оператора и обслуживающего персонала при контроле слабых мест в системе, обеспечение более качественного обслуживания системы управления, обеспечение своевременного и качественного профилактического обслуживания.

4.2 Математическая статистика

Математическая статистика — это область прикладной математики, которая включает сбор, описание, проверку и интерпретацию числовых данных. Математические теории, используемые в статистике, основаны на дифференциальном и интегральном исчислении, линейной алгебре и теории вероятностей.

Изучение случайного процесса PLCDuino начинается с рассмотрения трех определяемых в терминах процесса объектов:

1) Математическое ожидание случайного среднего процесса X(t):

$$m_x(t) = \ \mathrm{M}\left(U * e^{-t^2}\right) = e^{-t^2} M(U) = e^{-t^2} m_U = 5e^{-t^2} \qquad (4.2.1)$$

2) Дисперсия случайного процесса X(t):

$$D_x(t) = \ \mathrm{D}\left(U * e^{-t^2}\right) = e^{-t^2} D(U) = e^{-t^2} D_U = 0{,}001e^{-t^2} \qquad (4.2.2)$$

3) Корреляционная функция случайного процесса X(t):

$$K_x(t_1 t_2) = \ 0{,}001e^{-t_1^2}\, e^{-t_2^2} = 0{,}001e^{-(t_1^2+t_2^2)} \qquad (4.2.3)$$

В основе расчетов случайных процессов PLCDuino лежит взятое за основу X(t)=U*e^{-t^2}, где U – случайная величина с характеристиками $m_U = 5$, а $D_U = 0{,}001$.

4.3 Расчет энергоемкости макета и промышленного проекта

При учете, что стоимость за киловат в г.Алматы на 2023 год -7,05тг, то расчет энергоемкости для макетного образца и для его промышленного варианта будет.

Таблица 4.3.1 – Расчет энергоемкости макета проекта

Макет проекта. Arduino Uno - 2.5Вт Arduino Ethernet плата расширения - 2.5Вт Датчик температуры SNS-TMP10K-1M (X2) - 0.5Вт Модуль реле на 8 каналов с опторазвязкой - 0.25 Вт Блок питания ATX QD-400PNR OEM - 6Вт Вентилятор для корпуса ID-Cooling NO-8025-SD (X6) - 2.4Вт Общее потребление 26.4 Вт/ч - 0,18тг/ч

Таблица 4.3.2 – Расчет энергоемкости промышленного проекта

Промышленный проект. Arduino Uno - 2.5Вт Arduino Ethernet плата расширения - 2.5Вт Датчик температуры SNS-TMP10K-1M (X2) - 0.5Вт Модуль реле на 8 каналов с опторазвязкой - 0.25 Вт Блок питания ATX QD-400PNR OEM - 6Вт Реле 24V – 1.2 Вт Канальный вентилятор смешанного типа ВЕНТС ТТ 100 (X4) - 60Вт Общее потребление 379.2вт/ч - 2,67тг

4.4 Экономический расчет

Экономический расчет проекта также разделен на 2 части – макетная часть и промышленная часть, цена актуальные на 01.03.2023.

Таблица 4.4.1 – Экономический расчет макетного проекта в тенге

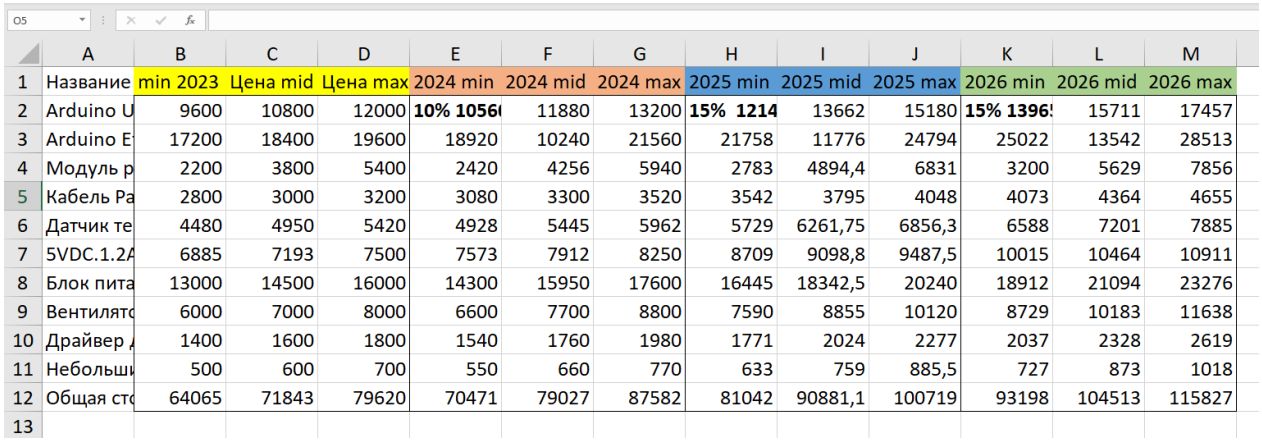
Название детали	Цена min	Цена mid	Цена max
Arduino Uno	9600	10800	12000
Arduino Ethernet плата расширения	17200	18400	19600
Модуль реле на 8 каналов с опторазвязкой	2200	3800	5400
Кабель Patch-Cord	2800	3000	3200
Датчик температуры SNS-TMP10K-1M (X2)	4480	4950	5420
5VDC.1.2A Yealink Блок питания 1.2A	6885	7193	7500
Блок питания ATX QD-400PNR OEM	13000	14500	16000
Вентилятор для корпуса ID-Cooling NO-8025-SD (X6)	6000	7000	8000
Драйвер двигателя L298N	1400	1600	1800
Небольшие детали (светодиоды, переключатели)	500	600	700
Общая стоимость	64065	71843	79620

Таблица 4.4.2 – Экономический расчет промышленного проекта в тенге

Название детали	Цена min	Цена mid	Цена max
Arduino Uno	9600	10800	12000
Arduino Ethernet плата расширения	17200	18400	19600
Модуль реле на 8 каналов с опторазвязкой	2200	3800	5400
Кабель Patch-Cord	2800	3000	3200
Датчик температуры SNS-TMP10K-1M (X2)	4480	4950	5420
5VDC.1.2A Yealink Блок питания 1.2A	6885	7193	7500
Реле промежуточные серии MY4 24B (X4)	3400	4100	4800
Автоматический выключатель 2P BA47-29 IEK (X2)	3000	3500	4000
Канальный вентилятор смешанного типа ВЕНТС ТТ 100 (X4)	86400	105200	124000
Общая стоимость	135965	160943	185920

Таблица 4.4.3 – Дополнительный экономический расчет на прочие расходы

Расходы на установку и содержание	Цена mid
Инструменты (одноразово)	50000
Транспортные расходы	20000
Фонд оплаты труда (сборка, программирование, настройка)	200000
Расходники (провода, болты, гайки, смазывающее вещество)	6000
Пластик для 3D принтера (для макетной части)	12000
Монтаж вентиляции (железная конструкция)	5100/метр



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Название	min 2023	Цена mid	Цена max	2024 min	2024 mid	2024 max	2025 min	2025 mid	2025 max	2026 min	2026 mid	2026 max
2	Arduino U	9600	10800	12000	**10% 1056**	11880	13200	**15% 1214**	13662	15180	**15% 1396**	15711	17457
3	Arduino E	17200	18400	19600	18920	10240	21560	21758	11776	24794	25022	13542	28513
4	Модуль р	2200	3800	5400	2420	4256	5940	2783	4894,4	6831	3200	5629	7856
5	Кабель Ра	2800	3000	3200	3080	3300	3520	3542	3795	4048	4073	4364	4655
6	Датчик те	4480	4950	5420	4928	5445	5962	5729	6261,75	6856,3	6588	7201	7885
7	5VDC.1.2A	6885	7193	7500	7573	7912	8250	8709	9098,8	9487,5	10015	10464	10911
8	Блок пита	13000	14500	16000	14300	15950	17600	16445	18342,5	20240	18912	21094	23276
9	Вентилято	6000	7000	8000	6600	7700	8800	7590	8855	10120	8729	10183	11638
10	Драйвер д	1400	1600	1800	1540	1760	1980	1771	2024	2277	2037	2328	2619
11	Небольши	500	600	700	550	660	770	633	759	885,5	727	873	1018
12	Общая сто	64065	71843	79620	70471	79027	87582	81042	90881,1	100719	93198	104513	115827
13													

Рисунок 4.4.1 – Экономический расчет индекса роста цен на 3 года с учетом роста инфляции

ЗАКЛЮЧЕНИЕ

Данный дипломный проект создавался не столько для защиты, сколько для дальнейшей реализации в качестве готового решения недорогой автоматизации производств, с помощью внедрения PLCDuino. Также, есть огромная уверенность в том, что данный продукт будет реализован в дальнейшем на целевых компаниях-производствах по одной простой причине, что каждая ступень, каждый расчет, каждая техническая реализация от идеи и до прототипа представлена в данной дипломной работе.

Исходя из научных журналов, что были прочитаны и приведены в [СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ], были выбраны исключительно лучшие решения для реализации системы охлаждения и реализации самой системы PLCDuino, что не имеет аналогов на отечественном рынке.

Чертежи, электрические схемы и 3D модели защищены авторским правом по причине того, что создавались полностью с нуля и не имеют в интернете аналогов. Помимо этого, вышеперечисленные технические инструменты полностью соответствуют стандартам качества.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

[1] Siboulet, É.; Pottier, L.; Ranger, T.; Riera, B. Fresh Approaches for Structured Text Programmable Logic Controllers Programs Verification. *Processes* 2023, *11*, 687. https://doi.org/10.3390/pr11030687

[2] Банников Евгений Викторович Использование ПЛК в промышленности // International scientific review. 2019. №LV. URL: https://cyberleninka.ru/article/n/ispolzovanie-plk-v-promyshlennosti

[3] Wonnacott, R.; Ching, D.S.; Chilleri, J.; Safta, C.; Rashkin, L.; Reichardt, T.A. Industrial PLC Network Modeling and Parameter Identification Using Sensitivity Analysis and Mean Field Variational Inference. Sensors 2023, 23, 2416. https://doi.org/10.3390/s23052416

[4] Wang, Z.; Zhang, Y.; Chen, Y.; Liu, H.; Wang, B.; Wang, C. A Survey on Programmable Logic Controller Vulnerabilities, Attacks, Detections, and Forensics. Processes 2023, 11, 918. https://doi.org/10.3390/pr11030918

[5] Saban, M.; Bekkour, M.; Amdaouch, I.; El Gueri, J.; Ait Ahmed, B.; Chaari, M.Z.; Ruiz-Alzola, J.; Rosado-Muñoz, A.; Aghzout, O. A Smart Agricultural System Based on PLC and a Cloud Computing Web Application Using LoRa and LoRaWan. Sensors 2023, 23, 2725. https://doi.org/10.3390/s23052725

[6] Nada, S.A. & Elfeky, Karem & Attia, Ali & Alshaer, W.. (2017). Experimental parametric study of servers cooling management in data centers buildings. Heat and Mass Transfer. 53. doi:10.1007/s00231-017-1966-y

[7] What is the Modbus Protocol & How Does It Work? https://www.ni.com/ru-ru/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/the-modbus-protocol-in-depth.html

[8] PLC Architecture and Types: With Comparison Table https://ladderlogicworld.com/plc-architecture/

[9] Govil, N.; Agrawal, A.; Tippenhauer, N.O. On ladder logic bombs in industrial control systems. In Proceedings of the Computer Security, Oslo, Norway, 14–15 September 2017; pp. 110–126.

[10] Tidrea, A.; Korodi, A.; Silea, I. Elliptic Curve Cryptography Considerations for Securing Automation and SCADA Systems. Sensors 2023, 23, 2686. https://doi.org/10.3390/s23052686

[11] Handel, T.G., Sandford, M.T. (1996). Hiding data in the OSI network model. In: Anderson, R. (eds) Information Hiding. IH 1996. Lecture Notes in Computer Science, vol 1174. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-61996-8_29

[12] Almoli A, Thompson A, Kapur N, Summers J, Thompson H, Hannah G (2011) Computational fluid dynamic investigation of liquid rack cooling in data centers. Appl Energy 89:150–155

[13] Herrlin MK (2006) A new tool for evaluating and designing the thermal environment in telecom central offices. In: Proceedings of telecommunications energy conference, pp 1–5

[14] Sharma RK, Bash CE, Patel CD (2002) Dimensionless parameters for evaluation of thermal design and performance of large scale data centers. In: Proceedings of AIAA2002-3091, American Institute of Aeronautics and Astronautics Conference

[15] Shrivastava S, Sammakia B, Schmidt R, Iyengar M (2005) Comparative analysis of different data center airflow management configurations. In: Proceedings of the ASME/Pacific Rim technical conference and exhibition on integration and packaging of MEMS, NEMS, and electronic systems: advances in electronic packaging, PART A, pp 329–336

[16] Nakao M, Hayama H, Nishioka M (1991) Which cooling air supply system is better for a high heat density room: underfloor or overhead. Proc Int Telecommun Energy Conf (INTELEC) 12(4):393–400

[17] Sorell V, Escalante S, Yang J (2005) Comparison of overhead and underfloor air delivery systems in a data center environment using CFD modeling. ASHRAE Trans 111(2):756–764

[18] Herrlin M, Belady C (2006) Gravity assisted air mixing in data centers and how it affects the rack cooling effectiveness. In: Proceedings of the inter society conference on thermal phenomena (ITherm), San Diego, CA, May 30-June 2

[19] Ham S-W, Kim M-H, Choi B-N, Jeong J-W (2014) Simplified server model to simulate data center cooling energy consumption. Energy Build 86:328–339

[20] Fulpagare Y, Mahamuni G, Bhargav A (2015) Effect of plenum chamber obstructions on data center performance. Appl Therm Eng 80(31):187–195

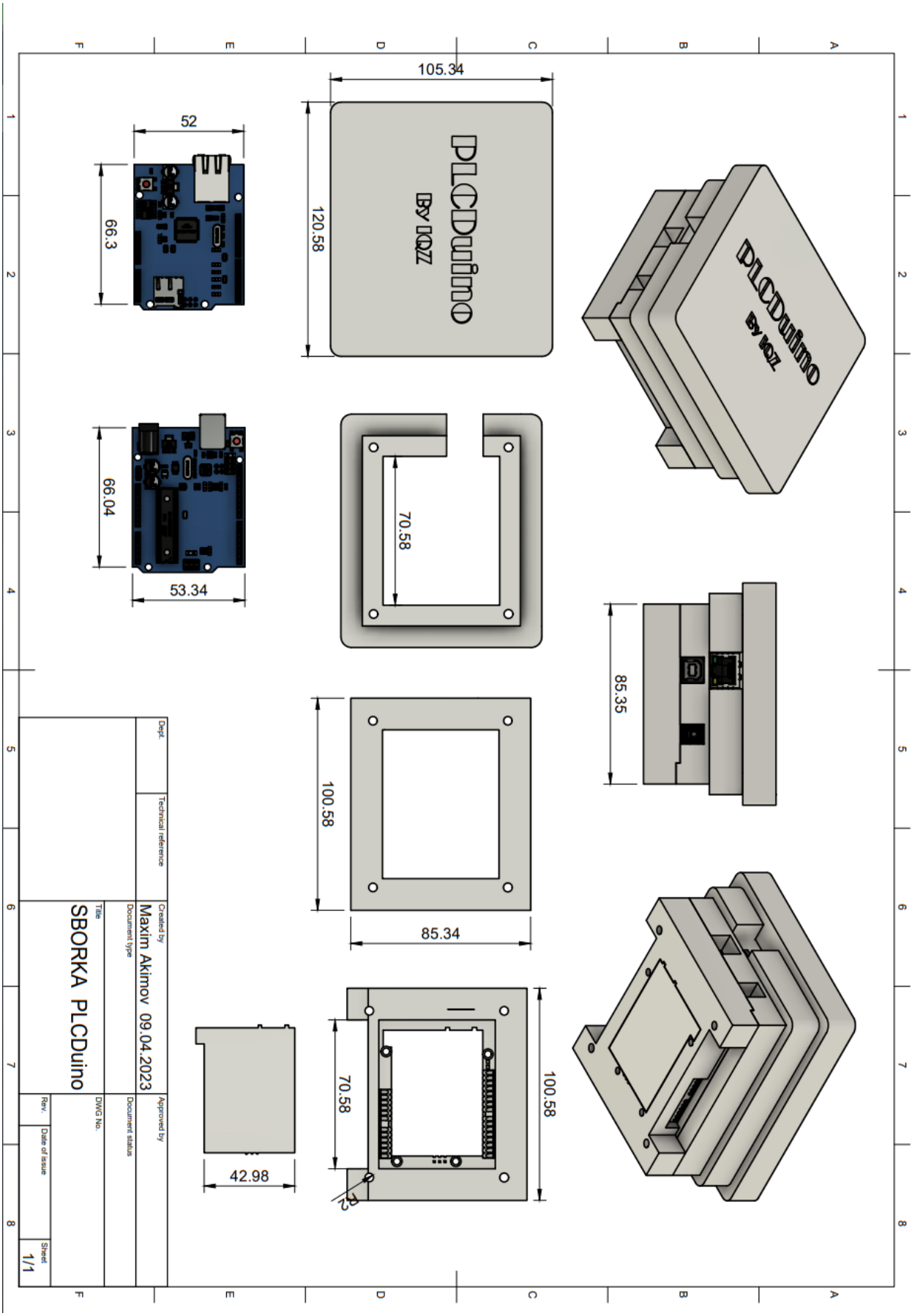
[21] Almoli A, Thompson A, Kapur N, Summers J, Thompson H, Hannah G (2011) Computational fluid dynamic investigation of liquid rack cooling in data centers. Appl Energy 89:150–155

[22] Официальный сайт ОВЕН https://owen.ru/product/plk160

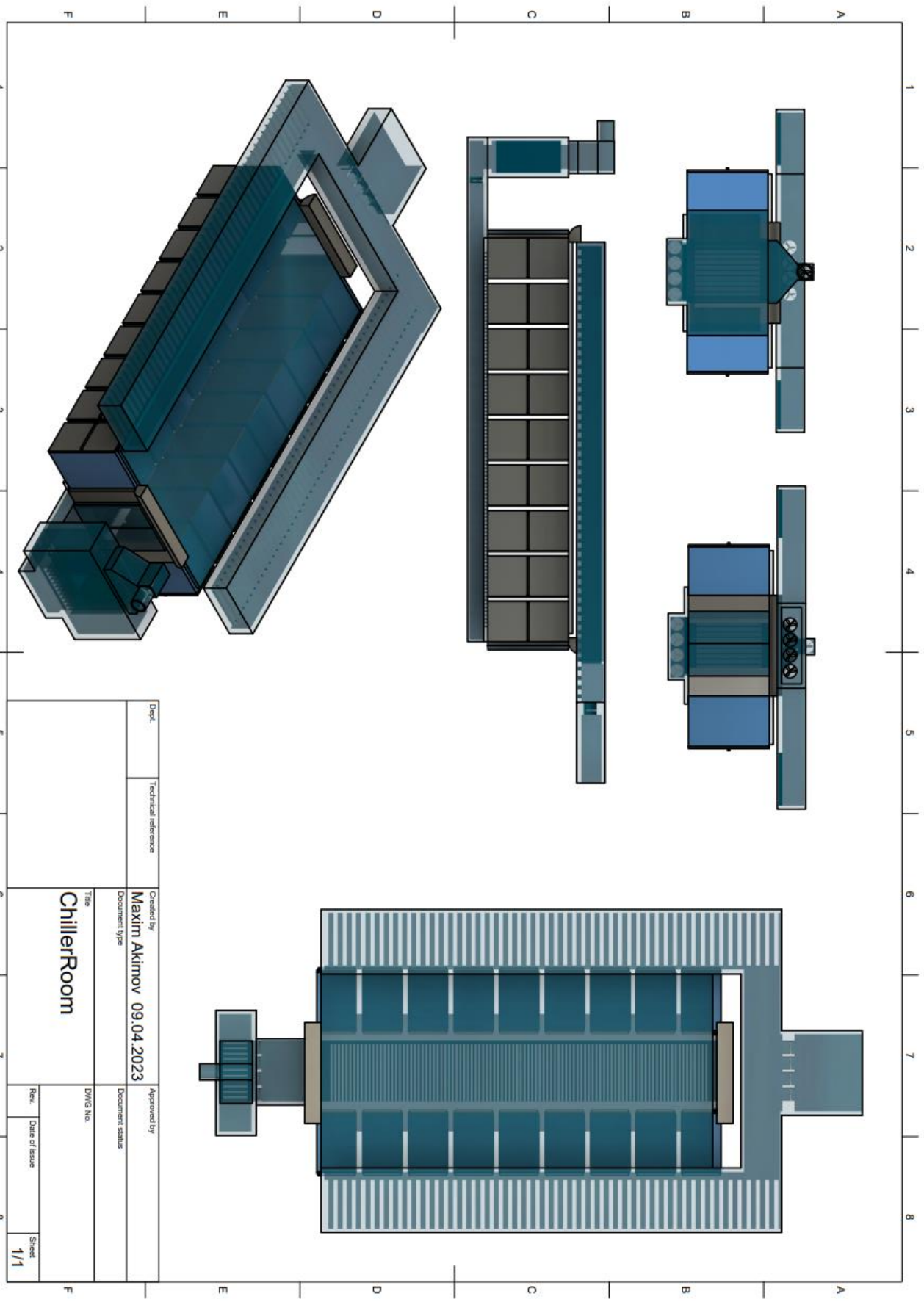
[23] Hanssen, Dag H. Programmable logic controllers: a practical approach to IEC 61131-3 using CODESYS. John Wiley & Sons, 2015.

[24] Jorn Linke. Der SPS-Benchmark: Das Ergebnis. Computer Automation. 2011. 9.

Приложение А



Приложение Б



Приложение В

```
#include <SPI.h>
#include <Ethernet.h>
#include "MgsModbus.h"
#include <DS_raw.h>
#include <microDS18B20.h>
#include <microOneWire.h>
MgsModbus Mb;
int inByte=0;
byte mac[] = {0x90 , 0xA2 , 0xDA , 0x0E ,0x94 , 0xB5};
IPAddress ip(192,168,1,85);
IPAddress gateway(192,168,1,1);
IPAddress subnet(255,255,255,0);
bool StartStop ,Conveyor1Coil , Conveyor2Coil ,CrusherCoil , GateCoil , StopButton;
bool Fault , LevelSensor ;
int Conveyor1Freq , Conveyor2Freq , CrusherFreq , Temp1 , Temp2;
int enA = 6;
int in1 = 5;
int in2 = 9;
int LED = A2;
int enB = 3;
int in3 = A1;
int in4 = A3;
MicroDS18B20 <A5> ds;
MicroDS18B20 <A4> ds2;
void setup()
{
StartStop=0;
Conveyor1Coil=0;
Conveyor2Coil=0;
CrusherCoil=0;
GateCoil=0;
Conveyor1Freq=0;
Conveyor2Freq =0;
CrusherFreq=0;
StopButton=0;
Temp1=0;
Temp2=0;
pinMode(2 , OUTPUT);
pinMode(4 , OUTPUT);
pinMode(7 , OUTPUT);
```

Продолжение приложения В.1

```
pinMode(8 , OUTPUT);
pinMode(enA, OUTPUT);
pinMode(in1, OUTPUT);
pinMode(in2, OUTPUT);
pinMode(enB, OUTPUT);
pinMode(in3 , OUTPUT);
pinMode(in4 , OUTPUT);
pinMode(LED, OUTPUT);
pinMode(A5, INPUT);
pinMode(A4, INPUT);
Serial.begin(9600);
Serial.println("Serial interface started");
Ethernet.begin(mac , ip , gateway ,subnet);
Serial.println("Ethernet interface started");
//-----------CodeSys Read
Mb.MbData[0]=0;
Mb.MbData[1]=0;
Mb.MbData[2]=0;
Mb.MbData[3]=0;
//-----------CodeSys write
Mb.MbData[4]=0;
Mb.MbData[5]=0;
Mb.MbData[6]=0;
Mb.MbData[7]=0;
Serial.println("0 - Print the Holding registers");
}
void loop(){
Conveyor1Freq=        Mb.MbData[1];
Conveyor2Freq=        Mb.MbData[2];
CrusherFreq=          Mb.MbData[3];
//Temp1=              Mb.MbData[6];
//Temp2=              Mb.MbData[7];
StartStop=           (Mb.MbData[0]>>0)&1;
Conveyor1Coil=        (Mb.MbData[0]>>5)&1;
Conveyor2Coil=        (Mb.MbData[0]>>2)&1;
CrusherCoil=         (Mb.MbData[0]>>3)&1;
GateCoil=            (Mb.MbData[0]>>4)&1;
StopButton=          (Mb.MbData[0]>>1)&1;
LevelSensor= digitalRead(A0);
if (LevelSensor==0) LevelSensor=1 ; else LevelSensor=0;
Fault= digitalRead(A1);
```

Продолжение приложения В.2

```
if (Fault==0) Fault=1 ; else Fault=0;
if (Fault==1)
  Mb.MbData[4]|=(1<<0);
else
  Mb.MbData[4]&=~(1<<0);
if (LevelSensor==1)
  Mb.MbData[4]|=(1<<1);
else
  Mb.MbData[4]&=~(1<<1);
if(StartStop==1 && StopButton==0)
{
digitalWrite(2 ,!Conveyor1Coil);
digitalWrite(4 ,!Conveyor2Coil);
digitalWrite(7,!CrusherCoil);
digitalWrite(8 ,!GateCoil);
digitalWrite(in1, HIGH);
digitalWrite(in2, LOW);
analogWrite(enA, CrusherFreq);
digitalWrite(in3, HIGH);
digitalWrite(in4, LOW);
analogWrite(enB, Conveyor1Freq);
digitalWrite(LED, LOW);
  if (ds2.readTemp()){
  ds2.requestTemp();
  Mb.MbData[6]=(ds2.getTemp());}
  if (ds.readTemp()){
  ds.requestTemp();
  Mb.MbData[7]=(ds.getTemp());}
}
else
{
digitalWrite(2 ,HIGH);
digitalWrite(4 ,HIGH);
digitalWrite(7 ,HIGH);
digitalWrite(8 ,HIGH);
digitalWrite(in1, LOW);
digitalWrite(in2, LOW);
digitalWrite(in3, LOW);
digitalWrite(in4, LOW);
}
```

Продолжение приложения В.3

```
if (StopButton==1)
{
digitalWrite(LED, HIGH);
}
if (Serial.available() >0){
inByte= Serial.read();
   for (int i =1 ; i<4 ; i++){Serial.print("Motor number");Serial.print(i);Serial.print("Frequency Value: ");Serial.println(Mb.MbData[i]); }
   Serial.print("Actuators word ");Serial.println(Mb.MbData[0] , BIN);
   Serial.print("Sensors word ");Serial.println(Mb.MbData[4] , BIN);
   Serial.print("Sensor1:"); Serial.println(ds2.getTemp());
   Serial.print("Sensor2:"); Serial.println(ds.getTemp());
   Serial.println("0 - Print the Holding registers");
 }
  Mb.MbsRun();
 }
```