Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Г.М. Цибульский
подпись
«____»_____2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 — Информационные системы и технологии

Разработка информационной системы мониторинга параметров серверного помещения

Руководитель _____ доцент, канд. техн. наук Р.В. Брежнев
подпись, дата
Выпускник _____ И.И. Павленко
подпись, дата

Красноярск 2019

Продолжение титульного листа бакалаврской работы по теме: Разработка информационной системы мониторинга параметров серверного помещения.

Нормоконтролер _____ Р.В. Брежнев

подпись, дата

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Г.М. Цибульский
подпись
«____»_____2019 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту Павленко Игорю Игоревичу

Группа КИ15-12Б, направление 09.03.02 «Информационные системы и технологии», профиль 09.03.02.05 «Информационные системы и технологии в административном управлении».

Тема выпускной квалификационной работы «Разработка информационной системы мониторинга параметров серверного помещения».

Утверждена приказом по университету № 6220/с от 08.05.2019

Руководитель Брежнев Р.В. кандидат технических наук, доцент кафедры искусственного интеллекта ИКИТ СФУ.

Исходные данные для ВКР: задание на практическую работу, полученное в рамках прохождения производственной практики на предприятии.

Перечень разделов ВКР:

– введение;

– обзор существующих систем мониторинга;

– вывод к главе 1;

– проектирование и разработка программно-аппаратной системы;

– вывод к главе 2;

– заключение;

– список использованных источников;

– приложение А – Г (Техническое задание, пояснительная записка к схеме подключения, скетчи работы датчиков, акт о внедрении, отчет «Антиплагиат», плакаты презентации).

Перечень графического материала: презентация «Разработка информационной системы мониторинга параметров серверного помещения».

Руководитель ВКР _____ Р.В. Брежнев
подпись

Задание принял к исполнению _____ И.И. Павленко
подпись

«___»_____2019 г.

График

Выполнения выпускной квалификационной работы студентом направления 09.03.02 «Информационные системы и технологии», профиля 09.03.02.05 «Информационные системы и технологии в административном управлении».

График выполнения выпускной квалификационной работы приведен в таблице 1.

Таблица 1 – График выполнения этапов ВКР

Наименование этапа	Срок выполнения этапа	Результат выполнения этапов	Примечание руководителя (отметка о выполнении этапа
Ознакомление с целью и задачами работы	01.02-6.02	Краткое эссе по теме ВКР	Выполнено
Сбор литературных источников	07.02-13.02	Список использованных источников	Выполнено
Анализ собранных источников литературы	14.02-20.02	Реферат о проблемно предметной области	Выполнено
Уточнение и обоснование актуальности цели и задач ВКР	21.02-28.02	Окончательная формулировка цели и задач ВКР	Выполнено
Решение первой задачи ВКР	29.02-10.03	Доклад и презентация по решению первой задачи	Выполнено
Решение второй задачи ВКР	11.03-25.03	Доклад и презентация по решению второй задачи	Выполнено
Решение третьей задачи ВКР	26.03-25.04	Доклад и презентация по третьей задаче ВКР	Выполнено
Решение четвертой задачи ВКР	26.04-15.05	Доклад и презентация по четвертой задаче ВКР	Выполнено
Подготовка доклада и презентации по теме ВКР	16.05-25.05	Доклад с презентацией по теме ВКР	Выполнено
Компоновка отчета по результатам решения задач ВКР	27.05-13.06	Отчет по результатам решения задач ВКР	Выполнено
Первичный нормоконтроль (Н/К)	14.06	Пояснительная записка, презентация ВКР	Выполнено

Окончание таблицы 1

Наименование этапа	Срок выполнения этапа	Результат выполнения этапов	Примечание руководителя (отметка о выполнении этапа
Предварительная защита результатов ВКР	18.06	Доклад с презентацией по теме ВКР	Выполнено
Вторичный нормоконтроль (Н/К)	26.06	Пояснительная записка, презентация ВКР	Выполнено
Итоговый нормоконтроль (Н/К)	28.06	Пояснительная записка, презентация ВКР	Выполнено
Защита ВКР	03.07	Пояснительная записка, доклад и презентация по результатам бакалаврской работы	

Руководитель ВКР _____ Р.В. Брежнев
подпись

Студент гр. КИ15-12Б _____ И.И. Павленко
подпись

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день большинство компаний используют высокотехнологическое оборудование в своей деятельности и работают в различных информационных системах. Для непрерывного функционирования и производительности информационных систем требуется серверное помещение, в котором будут расположено серверное и телекоммуникационное оборудование.

Согласно инструкции по проектированию зданий и помещений для электронно-вычислительных машин СН 512-78 [12], действующей по настоящее время, на здания и помещения, в которых размещены средства вычислительной техники, накладывается ряд требований, предусматривающих использование средств пожаротушения и сигнализации, устройств вентиляции и кондиционирования воздуха, а также устройств контроля параметров питающей электросети и бесперебойного электропитания.

Кроме необходимости применения указанных технических средств по их прямому назначению в настоящее время к серверным помещениям предъявляются дополнительные требования, связанные с их охраной от проникновения сторонних лиц с целью хищения информации или несанкционированного доступа к потокам данных, которые могут составлять тайну граждан, организаций либо государственную тайну.

Так как современное вычислительное оборудование создается с учетом работы в определенных микроклиматических условиях, то эксплуатация оборудования в условиях, отличных от рекомендованных заводом-изготовителем, может привести к выходу из строя оборудования и, как следствие, отсутствию гарантийного ремонта. Таким образом, система контроля состояния микроклимата северного помещения так же позволяет увеличить срок службы оборудования.

В настоящее время на рынке представлен ряд программно-аппаратных решений, реализованных такими производителями, как Uniping Server Solution, КСИТАЛ, ZPAS OVERSEE [9] и др. Целевой аудиторией существующих и пер-

спективных решений являются организации с высоким уровнем информатизации бизнес-процессов, что, как правило, требует значительных вычислительных ресурсов для функционирования баз данных и ресурсоемкого программного обеспечения. Такие организации могут обладать распределенной сетью серверных помещений, включающих от нескольких до сотен серверов.

Однако соотношения таких параметров систем мониторинга как стоимость, масштабируемость, простота в эксплуатации, обслуживании и ремонте, а также выбор элементной базы во многом являются определяющими факторами при формировании индивидуальных технических заданий.

В связи с этим разработка информационных систем мониторинга параметров серверных помещений с учетом индивидуальных потребностей заказчиков носит актуальный характер. В частности, актуальной является задача разработки системы мониторинга серверных помещений для ООО «Коммунальные информационные системы» в г. Красноярск, в рамках которой совместно с заказчиком разработано техническое задание.

Основываясь на актуальности, поставлена цель выпускной квалификационной работы: обеспечить мониторинг и контроль состояния оборудования серверного помещения для ООО «Коммунальные информационные системы» в г. Красноярск.

Для достижения поставленной цели были определены следующие задачи:

- обзор существующих систем мониторинга;
- выявление требований к проектируемой системе мониторинга;
- проектирование информационной системы;
- прототипирование системы мониторинга и внедрение на предприятии ООО «Коммунальные информационные системы».

1 Обзор существующих систем мониторинга

Перед тем как начать проектирование информационной системы, необходимо изучить основные определения и проанализировать существующие решения на рынке для выявления особенностей и недочетов в использовании подобных систем, что позволит наиболее полно сформировать технические и организационные требования.

1.1 Система мониторинга

Система мониторинга – это комплекс аппаратных и программных средств, обеспечивающих мгновенное отображение актуальной информации по состоянию оборудования информационной инфраструктуры для дальнейшего анализа и обнаружения возможных неисправностей и своевременного устранения проблем. Оповещения в системах мониторинга позволяют ответственным специалистам вовремя заметить неисправность в серверной комнате и незамедлительно принять меры по устранению этой неисправности. Подобный постоянный контроль приводит к минимизации сбоев и простоев в работе оборудования, позволяет поддерживать все ключевые сервисы в работоспособном состоянии и сохранять оптимальный уровень качества.

До появления автоматизированных систем мониторинга системные администраторы вынуждены были либо лично контролировать работу подведомственного оборудования, либо же не контролировать вообще ввиду нехватки рабочего времени. Такой подход включает в себя некоторые риски для организации в виде потери оборудования и, как следствие, части бюджета, или потери данных, что уже несет больший урон.

В настоящее время существует множество автоматических и полуавтоматических систем мониторинга, которые круглосуточно осуществляют контроль за оборудованием, что, несомненно, снимает большую нагрузку с системного администратора.

Наиболее типовыми задачами мониторинга серверных помещений являются:

– мониторинг климата и окружающей среды при работе компьютерного оборудования;

– мониторинг доступа в помещение и наличия движения в серверной комнате;

– мониторинг наличия электропитания;

– оповещение персонала в случае необходимости.

Система мониторинга должна постоянно отслеживать эти параметры и работать бесперебойно в течении длительного времени, быть простой в установке и настройке, а также не требовать частого сервисного обслуживания.

1.2 Обзор существующих аналогичных систем мониторинга

Рассмотрим две информационные системы мониторинга параметров помещений: информационную систему мониторинга параметров помещений UniPing server solution [2], систему GSM сигнализации и управления КСИТАЛ GSM [3].

1.2.1 Информационная система мониторинга параметров помещений UniPing server solution

Устройство UniPing server solution предназначено для сбора информации о параметрах окружающей среды и датчиках доступа в условиях серверной комнаты. Информация о состоянии датчиков передаётся через интерфейс Ethernet по протоколам HTTP (встроенный web сервер) и SNMP v1.

Внешний вид устройства UniPing server solution показан на рисунке 1.

Рисунок 1 – Внешний вид устройства UniPing server solution

Для подключения Ethernet используется разъём типа RJ-45, показанный на рисунке 2.

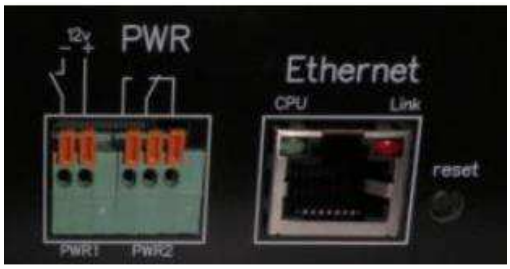


Рисунок 2 – Разъём типа RJ45 для подключения Ethernet

Разъём RJ-45 содержит два встроенных светодиода – CPU и Link. Светодиод CPU индицирует включение устройства и мигает при передаче пакетов от устройства. Светодиод Link индицирует наличие связи и мигает при приёме пакетов.

Для подключения бинарных датчиков используются 16 разъёмов, показанные на рисунке 3.

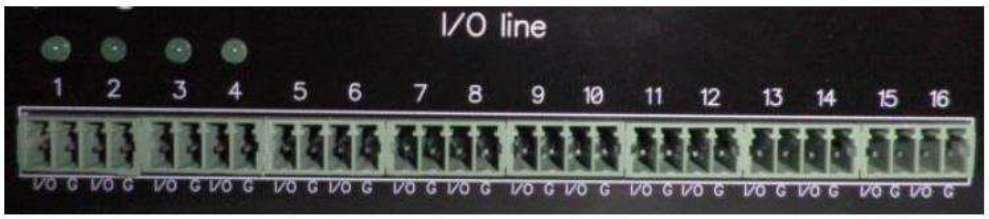


Рисунок 3 – Разъёмы для подключения бинарных датчиков

Эти разъёмы подключают IO линии с двумя состояниями: включено или выключено.

Каждая из IO линий может быть индивидуально сконфигурирована как вход или выход и содержит две клеммы: IO и GND.

Устройство UniPing server solution снабжено разъёмами для подключения специальных датчиков. Эти разъёмы показаны на рисунке 4.

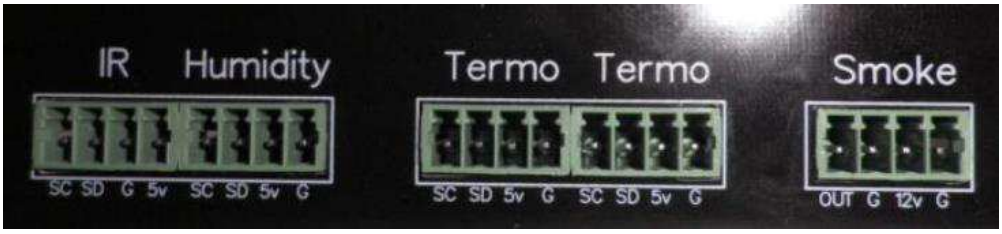
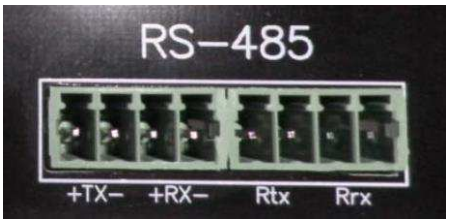


Рисунок 4 – Разъёмы для подключения специальных датчиков

Специальные датчики передают данные по цифровым шинам в специальном формате.

Для приемо-передачи данных в другие устройства в устройстве UniPing server solution предусмотрено использование интерфейсов RS482, RS485 и RS232.

На рисунке 5, *а* показан разъём RS485, а на рисунке 5, *б* – разъём RS232.



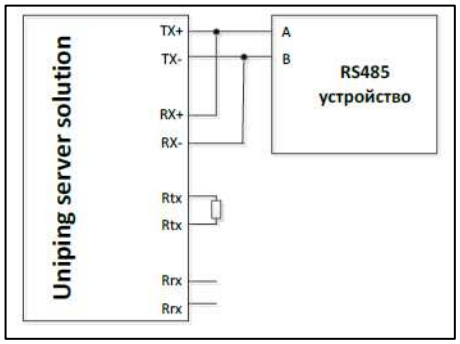
а *б*

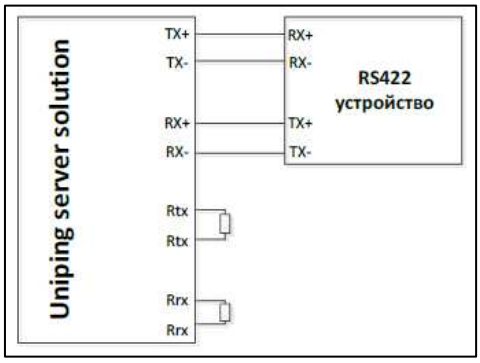
Рисунок 5 – Модели разъемов RS: *а* – RS485; *б* – RS232

Электрическая схема подсоединения разъёма RS485 показана на рисунке 6 *(а)*.

Устройство UniPing server solution также допускает подключение разъёма RS422, что проиллюстрировано на рисунке 6 *(б)*.

Работа с устройствами RS485/RS422 поддерживается в режиме конвертора интерфейсов, т.е. устройство отправляет принятые байты по интерфейсу RS485/RS422 в Ethernet в интерфейс RS485/RS422, а полученные из Ethernet в интерфейс RS485/RS422 данные при этом анализе протокола не поддерживаются.





а *б*

Рисунок 6 – Электрическая схема подсоединения разъёма: *а* – RS485; *б* – RS422

Работа с устройствами RS232 поддерживается в режиме конвертора интерфейсов, т.е. устройство отправляет принятые байты по интерфейсу RS232 в

Ethernet, а полученные из Ethernet в интерфейс RS232 данные при этом анализе протокола не поддерживаются.

Одновременное использование разъёмов RS485 и RS232 не допустимо.

Для дистанционного управления нагрузок устройства UniPing server solution используется разъём PWR, представленный на рисунке 7.

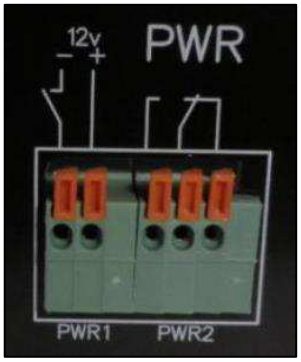


Рисунок 7 – Разъём PWR

Разъём PWR имеет два канал управления – PWR1 и PWR2. Канал PWR1 соединён с внутренним источником напряжения +12 В, а канал PWR2 представляет собой выведенные непосредственно на клеммник контакты реле.

Управление каналами PWR1 и PWR2 осуществляется по интерфейсам HTTP или SNMP v1.

Электрические параметры PWR1:

– напряжение на контактах при разомкнутом реле – 12 В;

– максимальный потребляемый ток – 0,7 А.

Электрические параметры PWR2:

– максимальный допустимый ток через контакты реле – 2 А;

– максимальное допустимое напряжение на контактах реле – 50 В.

К устройству UniPing server solution возможно подключение следующих типов датчиков:

– датчик температуры типа TS или WT – до 8 шт.;

– датчик влажности типа WS – 1 шт.;

– датчик дыма – 1 шт.;
– инфракрасный приёмопередатчик – 1 шт.;
– датчик наличия 220 В – до 16 шт.;
– датчик удара – до 8 шт.;
– датчик движения – до 16 шт.;
– датчик открытия/закрытия двери – до 16 шт.

Общее устройство системы показано на рисунке 8.

Рисунок 8 – Общая схема устройства системы

Исходя из проведенного обзора системы мониторинга, можно сделать вывод, что данная система обладает большой информативностью, так как содержит в себе разные датчики слежения. Позволяет получать информацию по sms и e-mail, что является существенным плюсом. Также данную систему можно применять для больших серверных помещений.

1.2.2 Система GSM сигнализации и управления КСИТАЛ GSM

Система GSM сигнализации и управления КСИТАЛ GSM используется для дистанционного контроля, охраны и управления стационарными объектами

с помощью сотового телефона. Данная система может быть использования для контроля, охраны и управления серверными помещениями.

Общий вид устройства КСИТАЛ GSM иллюстрируется рисунком 9.

Рисунок 9 – Общий вид устройства КСИТАЛ GSM

Система КСИТАЛ GSM обеспечивает:

– связь с 10 телефонными абонентами для рассылки SMS;

– связь с 10 телефонными абонентами для дозвона с голосовым сообщением;

– возможность передачи данных на любой сервер с фиксированным IP-адресом;

– перекрытие от 4 до 12 зон контроля для подключения шлейфов с датчиками;

– контроль оборудования посредством 3-х встроенных реле на ток 5 А и напряжение 220 В;

– измерение и передачу температуры в помещении с помощью встроенного термодатчика;

– подключение до 30 беспроводных термодатчиков (при установленном оборудовании РП433);

– возможность установки мастер-ключа;

– возможность установки дополнительных 6-ти ключей Touch Memory;

– возможность прослушивания звуковой обстановки помещения с помощью выносного микрофона;

– возможность программирования пользователем длинных (до 60 знаков) SMS, соответствующих срабатыванию зон контроля;

– возможность отправки SMS о постановке на контроль и о снятии с контроля;

– возможность отправки SMS при критическом падении температуры в помещении;

– возможность организации работы термостата в режиме день/ночь с автоматическим поддержанием температуры;

– возможность стабилизации температуры в помещении по дистанционному заданию;

– возможность обеспечения бесперебойного питания, контроль зарядки резервного 12 В аккумулятора, отправки SMS при пропадании напряжения 220 В и при критическом разряде резервного аккумулятора;

– возможность установки тревожной кнопки, позволяющей передавать тревожные SMS при снятой с контроля системе;

– автоматический ежедневный отчёт о работоспособности системы, в том числе по запросу;

– постановку системы на контроль как с помощью ключей Touch Memory, так и с помощью SMS;

– проводить диагностику неисправности шлейфов сигнализации (обрывы, замыкания) при постановке на контроль;

– возможность подключения до 4-х беспроводных блоков аппаратного расширения КСИТАЛ;

– подключение электропитания 12 В подключенных датчиков непосредственно от контроллера (сохраняется при пропадании 220 В);

– температурный диапазон работы от -25 до +50 град. С (при укомплектовании соответствующей SMS-картой).

Состав устройства КСИТАЛ GSM показан на рисунке 10.

Рисунок 10 – Состав устройства КСИТАЛ GSM

Датчики, входящие в состав устройства КСИТАЛ GSM, представлены на рисунке 11.

Рисунок 11 – Датчики, входящие в состав устройства КСИТАЛ GSM

Исходя из проведенного обзора системы мониторинга, можно сделать вывод, что данная система больше подходит для установки в доме, чем в серверном помещении, так как полностью зависит от GSM связи, а в свою очередь серверное помещение не гарантирует бесперебойной работы дан-ной системы.

Устройство КСИТАЛ GSM позволяет получать информацию и управлять датчиками только по sms, что является существенным минусом системы, но при этом в отличии от рассмотренной предыдущей системы она может функ-

ционировать при отключении электроэнергии, так как оснащена внутреннем аккумулятором.

1.3 Вывод к главе 1

Для удобства ниже представлена таблица 1, в которой отображены основные характеристики систем мониторинга и контроля и проведено сопоставление данных требований с возможностями рассмотренных выше систем. Данные представлены в таблице 2, где обозначение «+» означает, что данная функция имеется в системе, а знак «-» показывает, что данная функция отсутствует в рассматриваемой системе.

Таблица 2 – Сравнительный анализ возможностей систем мониторинга

	UniPing server solution	КСИТАЛ GSM
Встроенный аккумулятор для системы уведомлений	-	+
Встроенный GSM модем	-	+
Ethernet порт	+	-
Web application	+	-
Phone application	-	+
Кол-во датчиков	Ограничено	Ограничено
Уведомления по Email	+	-
Уведомление по SMS	+	+
Поддержка СМС команд	-	+
Встроенный журнал событий	+	-

Так как система мониторинга планируется использоваться в серверном помещении с небольшим количеством серверов, то сделан вывод о целесообразности разработки собственной информационной системы мониторинга и контроля, стоимость которой будет ниже. Помимо минимизации затрат, в раз-

рабатываемой системе будут реализованы все пожелания заказчика, представленные в техническом задании (Приложение А), так как рассматриваемые системы не могут удовлетворить потребности заказчика в полной мере.

2 Проектирование и разработка программно-аппаратной системы

2.1 Разработка структуры устройства

Структура устройства представляет собой совокупность элементарных звеньев объекта и связей между ними. Под элементарным звеном понимают часть объекта, системы управления и т. д., которая реализует элементарную функцию. Структурная схема предназначена наглядно показывать основные функциональные узлы устройства, их назначение и участие в общем процессе.

Общая структурная схема будет иметь следующие блоки:

1. Блок питания — это устройство, которое используется для преобразования напряжения, необходимого для работы системы, из напряжения общей электросети;
2. Датчик температуры — это устройство, которое позволяет отслеживать изменения температуры в среде и в случае критических изменений подавать сигнал;
3. Инфракрасный датчик движения — это электронный датчик, который измеряет инфракрасный свет, излучаемый объектами в его поле зрения;
4. Считыватель электронного ключа — это устройство, которое считывает параметры идентификации для последующего предоставления или отказа доступа в помещение;
5. ИК-передатчик — это устройство, которое позволяет совершать действия на расстояния совместно с инфракрасным приемником;
6. Модуль с двумя реле — это электромеханическое устройство, которое позволяет механическим способом замыкает цепь нагрузки с помощью электромагнита;
7. Микроконтроллер — это устройство, которое позволяет управлять электронными устройствами;
8. Сетевой модуль — автоматически конфигурируемый Ethernet-контроллер;

9. LCD-дисплей — один из видов экранов;

10. SD-card — это устройство, которое позволяет хранить на себе данные;

11. Web-application — веб-страница для отображения данных получаемых с датчиков;

12. Mobile-application — мобильное приложение для отображения данных получаемых с датчиков.

2.2 Выбор оборудования

В выборе основного оборудования нужно полностью отталкиваться от выбранного микроконтроллера. В связи с этим на первом шаге был выбран именно он.

В данной системе было решено использовать микроконтроллер Arduino, так как он полностью сможет удовлетворить потребности разрабатываемой системы, простой в использование и так же имеется большое количество полезной и русскоязычной информации.

На данный момент существует огромное количество моделей Arduino, которые отличаются между собой функциональностью, ценой и новизной модели. Мы рассмотрели самые популярные виды, а именно:

Arduino Mega — одна из самых мощных плат в линейке Arduino. Имеет память 256 Кб, которой хватит на 99,9% проектов, 54 цифровых входов/выходов и 16 аналоговых входов, представлена на рисунке 12.

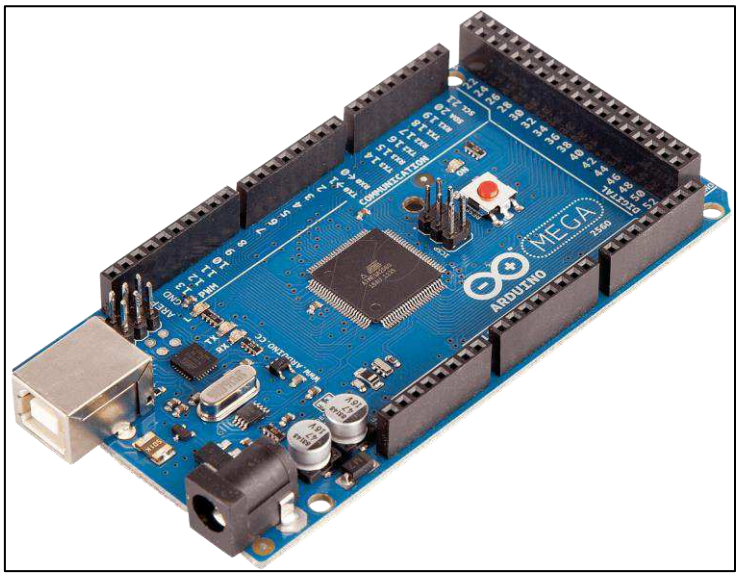


Рисунок 12 – Микроконтроллер Arduino Mega

Arduino Uno — урезанная версия модели Arduino Mega, наиболее распространённая плата из семейства arduino, имеет память 32 Кб, 14 цифровых входов/выходов и 6 аналоговых входов. Данная плата показана на рисунке 13. Немного, по сравнению с Mega, но для многих проектов хватает.

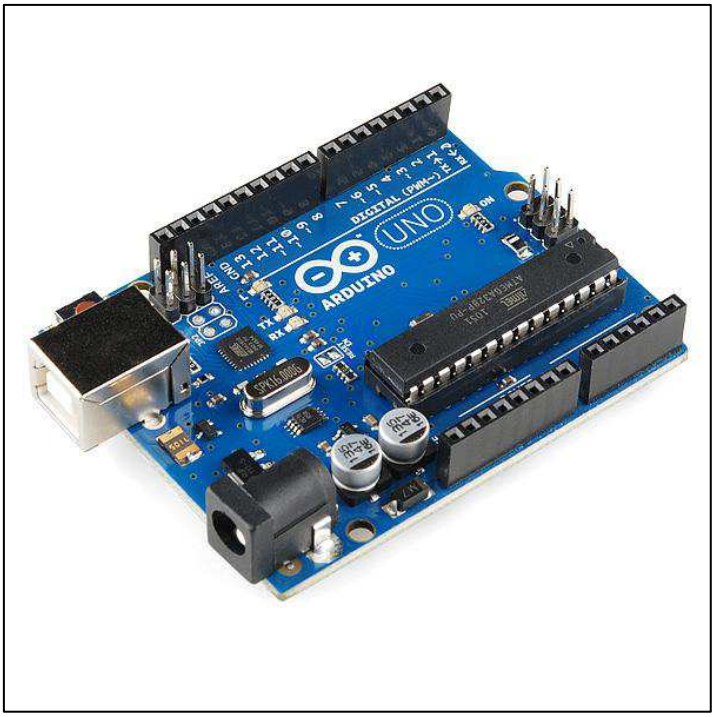


Рисунок 13 – Микроконтроллер Arduino Uno

Arduino Nano — имеет 14 цифровых входов/выходов и 8 аналоговых входов и память равная 32 Кб, немного лучше чем Arduino Uno, главное преимущество это малый размер. Данная плата показана на рисунке 14.

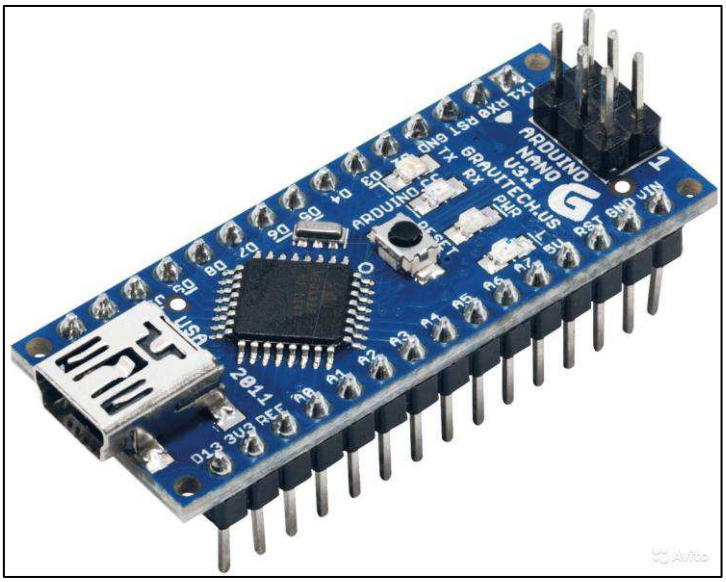


Рисунок 14 – Микроконтроллер Arduino Nano

Arduino Pro Mini — самая слабая плата, имеет память 16 Кб, 14 цифровых входов/выходов и 4 аналоговых входа. Также в данной плате, показанной на рисунке 15, отсутствует программатор, в предыдущих он уже встроен.

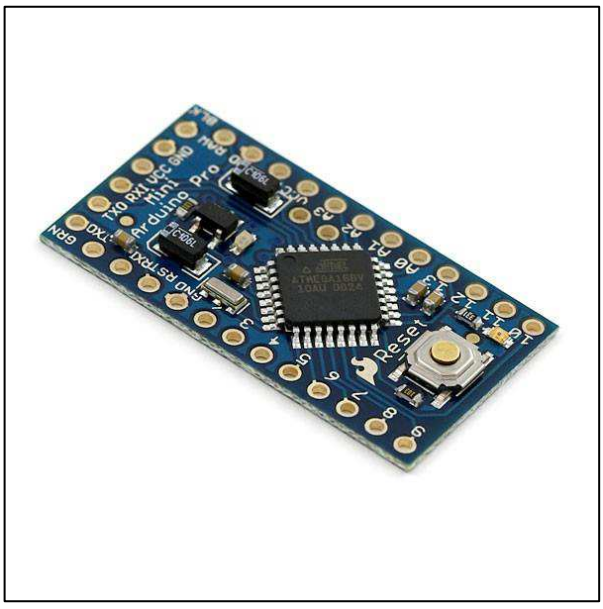


Рисунок 15 – Микроконтроллер Arduino Pro Mini

Для реализации нашего проекта было решено использовать Arduino Mega, так как в будущем возможны расширение функциональности системы мониторинга серверного помещения, благодаря техническим характеристикам данной платы это будет возможно сделать.

Остальное необходимое оборудование подбиралось исходя из совместимости с выбранным микроконтроллером, полное описание и наименование отражено в приложении (Приложение Б).

2.3 Анализ функциональных требований

Диаграмма вариантов использования представляет собой схему взаимодействия пользователя с системой, которая показывает связь между пользователем и различными случаями использования, в котором пользователь участвует. Диаграмма вариантов использования может идентифицировать различные типы пользователей системы и различные варианты использования и часто может сопровождаться другими типами диаграмм.

Общие компоненты:

– Акторы: пользователи, которые взаимодействуют с системой. Актор может быть человеком, организацией или внешней системой, взаимодействующей с приложением или системой. Они должны быть внешними объектами, которые производят или потребляют данные;

– Прецедент: конкретная последовательность действий и взаимодействия между участниками и системой. Прецедент также можно назвать сценарием;

– Цели: конечный результат большинства случаев использования.

Успешная диаграмма должна описывать действия и варианты, используемые для достижения цели.

Для построения взаимосвязей между акторами и прецедентами в языке UML имеется несколько стандартных видов отношений:

– Отношение ассоциации (association relationship) - служит для обозначения специфической роли актера при его взаимодействии с отдельным вариантом использования;

– Отношение включения (include relationship) между двумя вариантами использования - указывает на то, что заданное поведение для одного варианта использования включается в качестве составного фрагмента в последовательность поведения другого варианта использования;

– Отношение расширения (extend relationship) - определяет взаимосвязь базового варианта использования с другим вариантом использования, функциональное поведение которого задействуется базовым не всегда, а только при выполнении дополнительных условий;

– Отношение обобщения (generalization relationship) между вариантами использования применяется в том случае, когда необходимо отметить, что дочерние варианты использования обладают всеми особенностями поведения родительских вариантов [6].

При построении диаграммы вариантов использования хотелось показать, какие основные действия сможет совершать пользователь и сама система мониторинга. Общая диаграмма вариантов использования со стороны пользователя представлена на рисунке 16.

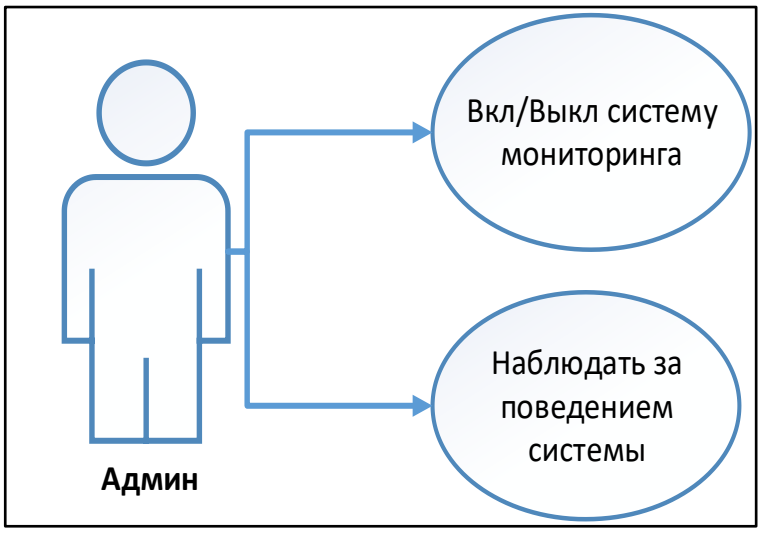


Рисунок 16 – Use-case диаграмма (со стороны пользователя)

Конкретизация вариантов использования:

1) Вкл/Выкл систему мониторинга:

– основное действующее лицо: админ;

– другие участники прецедента: отсутствуют;

– связи с другими вариантами использования: отсутствуют.

Краткое описание: данный вариант использования позволяет пользователю включить или выключить систему мониторинга.

2) Наблюдать за поведением системы:

– основное действующее лицо: админ;

– другие участники прецедента: отсутствуют;

– связи с другими вариантами использования: отсутствуют.

Краткое описание: данный вариант использования позволяет пользователю наблюдать за действия происходящими внутри системы мониторинга.

На рисунке 17 показаны функции, которые выполняются внутри системы без участия пользователя.

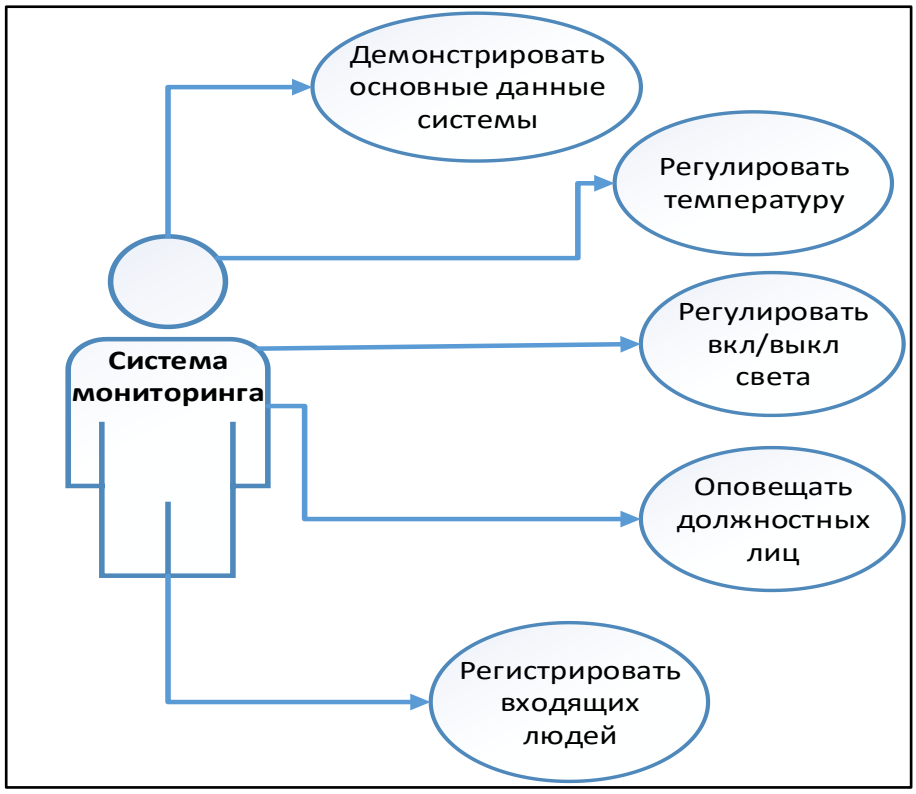


Рисунок 17 – Use-case диаграмма (процессы внутри системы)

Конкретизация вариантов использования:

1) Демонстрировать основные данные системы:

– основное действующее лицо: система мониторинга;

– другие участники прецедента: отсутствуют;

– связи с другими вариантами использования: отсутствуют.

Краткое описание: данный вариант использования позволяет АИС демонстрировать основные параметры, собранные системой мониторинга.

2) Регулировать температуру

– основное действующее лицо: система мониторинга;

– другие участники прецедента: отсутствуют;

– связи с другими вариантами использования: отсутствуют.

Краткое описание: данный вариант использования позволяет АИС регулировать работу кондиционера, при повышении/снижении температуры выше/ниже среднего значения.

3) Регулировать вкл/выкл света

– основное действующее лицо: система мониторинга;

– другие участники прецедента: отсутствуют;

– связи с другими вариантами использования: отсутствуют.

Краткое описание: данный вариант использования позволяет АИС включать/отключать свет при присутствии/отсутствии персонала в серверном помещении соответственно.

4) Оповещать должностных лиц

– основное действующее лицо: система мониторинга;

– другие участники прецедента: отсутствуют;

– связи с другими вариантами использования: отсутствуют.

Краткое описание: данный вариант использования позволяет АИС выполнять функцию оповещения должностных лиц в случае:

- Выхода значения температуры в серверном помещении за критическое значение описанном в П.3 технического задания;

- Проникновения в помещение посторонних лиц;

5) Регистрировать входящих людей

– основное действующее лицо: система мониторинга;

– другие участники прецедента: отсутствуют;

– связи с другими вариантами использования: отсутствуют.

Краткое описание: данный вариант использования позволяет АИС регистрировать входящих лиц в серверное помещение.

2.4 Разработка ИС

Основные требования к разрабатываемой ИС изложены в техническом задание (Приложение А).

Задача модуля температуры и влажности — мониторинг и регулировка температуры/влажности в серверном помещении, а также формирование сообщения для дальнейшего оповещения ответственных лиц при достижении критических значений.

Для реализации данной задачи были использованы:

- Датчик температуры и влажности DHT22, показан на рисунке 18;
- Провода для макетных плат (мама-папа).

Датчик состоит из двух частей емкостного датчика температуры и гигрометра. Первый используется для измерения температуры, второй используется для измерения влажности воздуха. Находящийся внутри чип может выполнять аналого-цифровые преобразования и выдавать цифровой сигнал, который считывается посредством микроконтроллера.

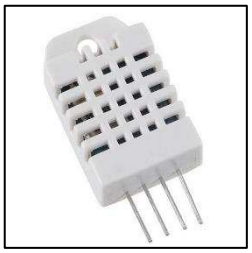


Рисунок 18 – Датчик температуры и влажности

Логика работы модуля температуры и влажности продемонстрирована на рисунке 19.

```
#include  <Wire.h>
#include "DHT.h" //Подключаем библиотеку термодатчик.
#define DHTTYPE DHT21 // DHT 21 (AM2301) //Здесь выбираем какой у нас датчик.
DHT dht(DHTPIN, DHTTYPE); // Объявляем датчик
// Проверяем находится ли температура в норме или вышла за установленные пределы
// Поднимаем и опускаем соответствующие флаги
// Опускаем флаг, разрешаем повторную отпраку сообщения на почту) при приходе температуры в норму
void alarmT(){
  if (t > t_max) {t_alarm_max=true;} else {t_alarm_max=false; send_t_alarm_max=false;}
  if (t < t_min) {t_alarm_min=true;} else { t_alarm_min = false ;  send_t_alarm_min = false ;}
 }
// То же самое для влажности
void  alarmH () {
  if  (h > h_max)  {h_alarm_max = true;}  else  {h_alarm_max = false;  send_h_alarm_max = false ;}
  if  (h  < h_min) {h_alarm_min = true;}  else  {h_alarm_min = false;  send_h_alarm_min = false ;}
 }
// Читаем данные из датчика DHT, Отправляем данные в Blynk
// и в случаи аварии отправляем сообщение на почту
void readDHT(){
  h = dht.readHumidity();
  t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit
  // Если данные считаны то строки ниже будут выполнены
  if (isnan(h) || isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  Blynk.virtualWrite(VPIN_LABEL_TEMP, t);  // Отправляем данные в Blynk
  Blynk.virtualWrite(VPIN_LABEL_HUMID, h); // Отправляем данные в Blynk
    alarmT(); // Проверяем аварии для Температуры
    alarmH(); // Проверяем аварии для Влажности

/ * Если (флаг аварии и не было ли отправки сообщения об аварии)
 * то отправим сообщение и поднимим флаг о том что мы уже отправили сообщение, для защиты от спама
на электронную почту */
    if (t_alarm_max && !send_t_alarm_max)
    {emailSEND(TextDHT("Осторожно! Превышена норма температуры", String(t)));
send_t_alarm_max=true;}
    if (t_alarm_min && !send_t_alarm_min)
    {emailSEND(TextDHT("Осторожно! Температура ниже нормы", String(t))); send_t_alarm_min=true;}
    if (h_alarm_max && !send_h_alarm_max)
    {emailSEND(TextDHT("Осторожно! Превышена норма влажности", String(h))); send_h_alarm_max=true;}
    if (h_alarm_min && !send_h_alarm_min)
    {emailSEND(TextDHT("Осторожно! Влажность ниже нормы", String(h))); send_h_alarm_min=true;}
  }
```

Рисунок 19 – Логика работы модуля температуры и влажности

Задача модуля освещенности — включение/отключение света при присутствии/отсутствии персонала в серверном помещении соответственно.

Для реализации данной задачи были использованы:

– инфракрасный датчик движения показан на рисунке 20;

– лампа накаливания;

– модуль с двумя реле;

– Провода для макетных плат (мама-папа).

На датчике установлены два переменных резистора, один регулирует чувствительность приборы, а другой отвечает за время продолжительности работы.

Рисунок 20 – Инфракрасный датчик движения

Логика работы модуля освещенности показана на рисунке 21.

```
void PIRsensor(){ // Функция считывания сигналов с датчика движения PIR
  //Serial.println(analogRead(PIN_PIR));
  //считываем состояние датчика если сигнал выше 300 значит поднимаем флаг statePIR,
иначе опускаем statePIR
  if (analogRead(PIN_PIR)>300){statePIR = HIGH;}
  else                        {statePIR = LOW;}
  //если есть движение включить Освещение иначе выключить Освещение
  if (statePIR == HIGH) {digitalWrite(PIN_RELE_SVET, HIGH);}
  else                  {digitalWrite(PIN_RELE_SVET, LOW);}
}
```

Рисунок 21 – Работа модуля освещенности

Блок-схема работы модуля освещенности представлена на рисунке 22.

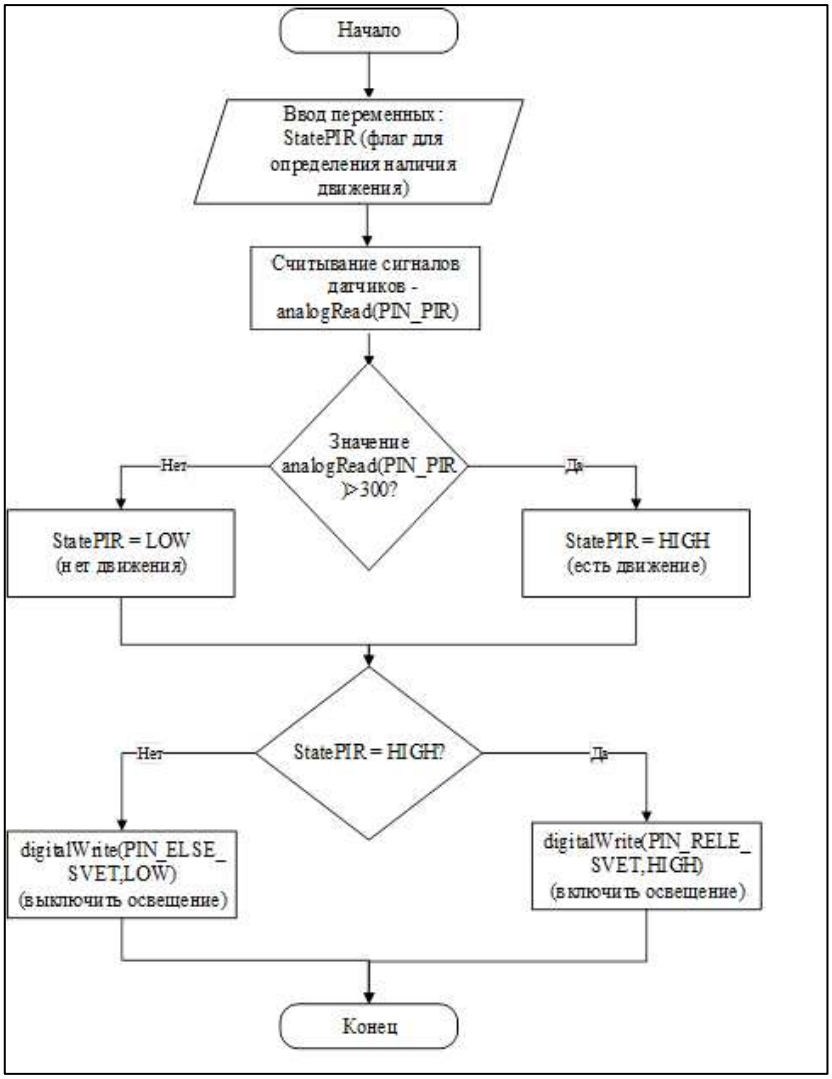


Рисунок 22 – Блок-схема работы модуля освещенности

Задача модуля контроля доступа — регистрация входящих в помещение лиц.

RFID (радиочастотная идентификация) — это метод обеспечения передачи, записи и хранения данных при помощи радиосигналов. Каждая RFID-система включает в себя считыватель/ридер и RFID-метку, в которой хранятся данные. Метки состоят из двух частей – интегральной схемы и антенны. Интегральная схема позволяет хранить и обрабатывать данные, антенна – принимать и передавать информацию. Для реализации данной задачи были использованы:

- RFID считыватель MFRC-522 показан на рисунке 23;
- RFID метка;
- Провода для макетных плат (мама-папа).

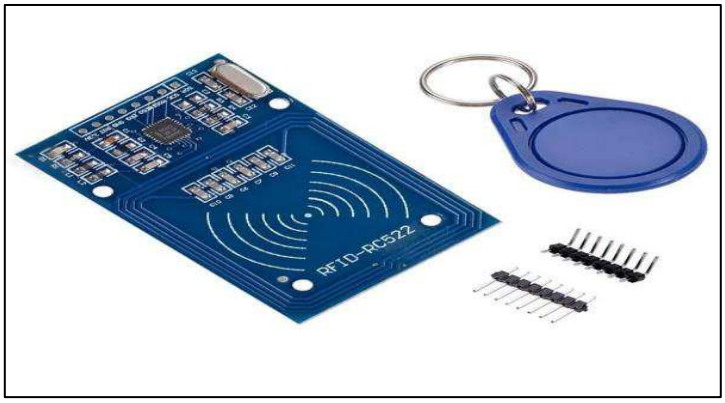


Рисунок 23 – RFID считыватель/метка

Логика работы RFID считывателя объемна, поэтому полное ее описание показано в Приложение И. На рисунке 24 показана основная часть логики.

```
// Инициализация считывателя RFID меток
void setupMFRC522(){
  SPI.begin();                // Инициализируем интерфейс SPI
  mfrc522.PCD_Init();         // Инициализируем модуль MFRC522
  mfrc522.PCD_DumpVersionToSerial();   // Выводим версию прошивки модуля на мо-
нитор серийного порта
// Выводим сообщение о том, что модуль готов к считыванию и ожидает метку
  Serial.println(F("Please, place RFID-label over the reader"));
}
// Чтение карт и ключей с RFID метками
// Логика такая провели ключем открыли, что бы закрыть нужно провести ключем еще
раз
void keyRFID(){
     digitalWrite(PIN_RELE_LOCK, StateKeyRFID); // Управляем реле в зависимости
от состояния замка
    Blynk.virtualWrite(VPIN_LOCK, StateKeyRFID); // шлем данные в таблицу Blynk
в зависимости от состояния замка

  // Ожидаем метку, и пока ее нет, прерываем дальнейшее выполнение этой функции
(keyRFID)
 if ( ! mfrc522.PICC_IsNewCardPresent()) {
  return;
 }
// Пытаемся прочитать метку, и пока это не получилось, прерываем дальнейшее вы-
полнение этой функции (keyRFID)
 if ( ! mfrc522.PICC_ReadCardSerial()) {
  return;
 }
```

Рисунок 24 – Логика работы RFID считывателя

Задача модуля стационарной визуализации — вывод на LCD экран данных с параметров датчиков в серверном помещении.

LCD экран оборудован меню, в котором при нажатии кнопки доступен просмотр каждого параметра по отдельности. При автономном режиме происходит автоматическое переключение меню.

Меню LCD экрана:

- экран 1, данные параметра датчика температуры DHT22 (влажность, температура);
- экран 2, данные параметра освещенности (свет включен/отключен);
- экран 3, данные со считывателя RFID (открыта/закрыта дверь).

Для реализации данной задачи были использованы:

- LCD дисплей показан на рисунке 25;
- Преобразователь интерфейса LCD в I2C;
- Провода для макетных плат (мама-папа).

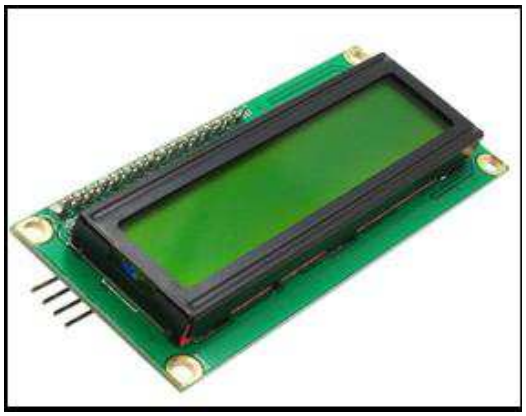


Рисунок 25 – LCD экран

Логика работы модуля стационарной визуализации показана на рисунке 26.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>      //Библиотека для LCD экрана
LiquidCrystal_I2C lcd(0x27,16,2);   // Объявляем экран по адресу 0x27 на шине
I2C // и 2 строки по 16 символов

// Прототип функции, дя того что бы компилятор знал что такая функция есть
void LCDmenu();

// Функция отвечающая за переключение экранов
void ScreenSwitching(){
  MenuPos++;
  if(MenuPos > 2){MenuPos=0;}
  LCDmenu();
}
// Функция вывода информации на LCD экран
void LCDmenu(){
static int8_t LastMenuPos=100;
switch (MenuPos) {
    case 0:
      //выполняется, когда MenuPos равно 0
        if (LastMenuPos!=MenuPos){ // Если в прошлый раз был другой экран, то
сотрем все и выведем новый экран
          lcd.clear();
          lcd.setCursor(0, 0); // Устанавливаем курсор в начало 1 строки
          lcd.print(F("Hum = % ")); // Выводим текст
          lcd.setCursor(7, 0); // Устанавливаем курсор на 7 символ
          lcd.print(h, 1); // Выводим на экран значение влажности
          lcd.setCursor(0, 1); // Устанавливаем курсор в начало 2 строки
          lcd.print(F("Temp = \1C ")); // Выводим текст, \1 - значок градуса
          lcd.setCursor(7, 1); // Устанавливаем курсор на 7 символ
          lcd.print(t,1); // Выводим значение температуры
        }else{    //   Иначе только обновим информацию в нужных местах
          lcd.setCursor(7, 0); // Устанавливаем курсор на 7 символ
          lcd.print(h, 1); // Выводим на экран значение влажности
          lcd.setCursor(7, 1); // Устанавливаем курсор на 7 символ
          lcd.print(t,1); // Выводим значение температуры
        }
      break;
    case 1:
      //выполняется когда  MenuPos равно 1
        lcd.clear();
        lcd.setCursor(0, 0); // Устанавливаем курсор в начало 1 строки
        if(statePIR == HIGH){lcd.print(F("SVET VKL"));}
        else                 {lcd.print(F("SVET OTKL"));}
      break;
    case 2:
      //выполняется когда  MenuPos равно 2
        lcd.clear();
        lcd.setCursor(0, 0); // Устанавливаем курсор в начало 1 строки
        if(!StateKeyRFID){
        lcd.print(F("CLOSED"));
        }
        else if (StateKeyRFID){
        lcd.print(F("OPEN"));
        }
      break;
    default:
```

Рисунок 26 – Логика работы модуля стационарной визуализации

Задача модуля удаленной визуализации — демонстрация основных параметров помещения на веб-странице и в мобильном приложении «Blynk».

Для реализации данной задачи используется Ethernet shield W5100 и сетевой кабель.

Ethernet Shield, показанный на рисунке 27, дает возможность подключать Arduino к интернету. Shield расширяет возможности Arduino и позволяет отсылать и принимать данные из любой точки мира, где есть интернет.

Web-сервер создается в Ethernet shield с помощью команды

Ethernet.begin (arduino_mac, device_ip, dns_ip, gateway_ip, subnet_mask).

Где:

- arduino_mac – мак адресс ардуино;
- device_ip – ip адресс ардуино;
- dns_ip - ip адресс dns;
- gateway_ip - шлюз IPv4;
- subnet_mask – маска подсети IPv4.

Веб-страница создается в программе Arduino IDE в виде html кода.

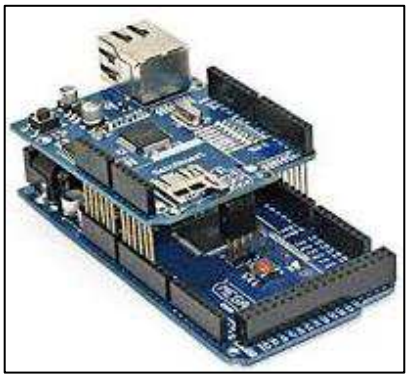


Рисунок 27 - Ethernet shield W5100 и Arduino Mega

Логика формирования веб-страницы показана на рисунке 28.

```
//выводим HTML страницу
        if (c == '\n' && currentLineIsBlank) {
          client. println (F("HTTP/1.1 200 OK")); //заголовочная информация
          client. println (F("Content-Type: text/html"));
          client. println (F("Connection: close"));
          client. println (F("Refresh: 5")); //автоматическое обновление каждые 5 сек
          client. println ();
          client. println (F("<!DOCTYPE HTML>")); //HTML тип документа
          client. println (F("<html>")); //открытие тега HTML
          client. println (F("<head> ")); //открытие тега Head
          client. println (F("<meta http-equiv='Content-Type' content='text/html ; char-
set=utf-8'/> "));
          client. print (F("<title> Microcompleks </title>")); //название страницы
          client. println (F("</head>")); //заголовочная информация
          client. println (F("<body>"));
          client. print (F("<H1>Микрокомплекс </H1>")); //заголовк на странице
          client. println (F(" <br>")); //перенос на след. строчку
          client. println (F("<hr/>")); //линия====================================
          client. println (F("Температура = ")); //Температура с DHT 11
          client. println (t); //переменная температуры
          client. println (F(" *C"));
          if(t_alarm_max) {client.println(F("Осторожно! Превышена норма температуры"));}
          else if (t_alarm_min){client.println(F("Осторожно! Температура ниже нормы"));}
          client. println (F("<br> ")); //перенос на след. строчку
          client. println (F("Влажность = ")); //Влажность с DHT 11
          client. println (h); //переменная влажности
          client. println (F(" %\t"));

          if(h_alarm_max) {client.println(F("Осторожно! Превышена норма влажности"));}
          else if (h_alarm_min){client.println(F("Осторожно! Влажность ниже нормы"));}
          client. println (F("<br> ")); //перенос на след. строчку
          client. println (F("<hr/>")); //линия====================================
          if(statePIR == HIGH){client.println(F("Свет в помещении включен"));}

          else if (statePIR == LOW){client.println(F("Свет в помещении выключен"));}
                    client. println (F("<br> ")); //перенос на след. строчку
          client. println (F("<hr/>")); //линия====================================
                  }
```

Рисунок 28 – Логика формирования веб-страницы

Логика остальных программных модулей отражена в приложениях:

- Логика взаимодействия с сервером – Приложение М;

- Логика работы часов реального времени – Приложение К;
- Логика работы с sd картой – Приложение Ж;
- Логика работы с моб. приложением Blynk – Приложение Г;
- Логика инициализации компонентов системы – Приложение В;
- Логика работы кнопки переключения меню LCD экрана – Приложение Д;
- Логика подключения системы к сети – Приложение Е;
- Логика работы ИК-передатчика – Приложение Н.

Перед сборкой макета микрокомплекса была реализована общая схема подключения. Пояснительная записка к схеме подключения (Приложение Б).

Для наглядности была построена модель состояний программно-аппаратного комплекса, которая показывает, как объект переходит из одного состояния в другое. Модель состояний служат для моделирования динамических аспектов системы. Модель состояний представлена на рисунке 29.

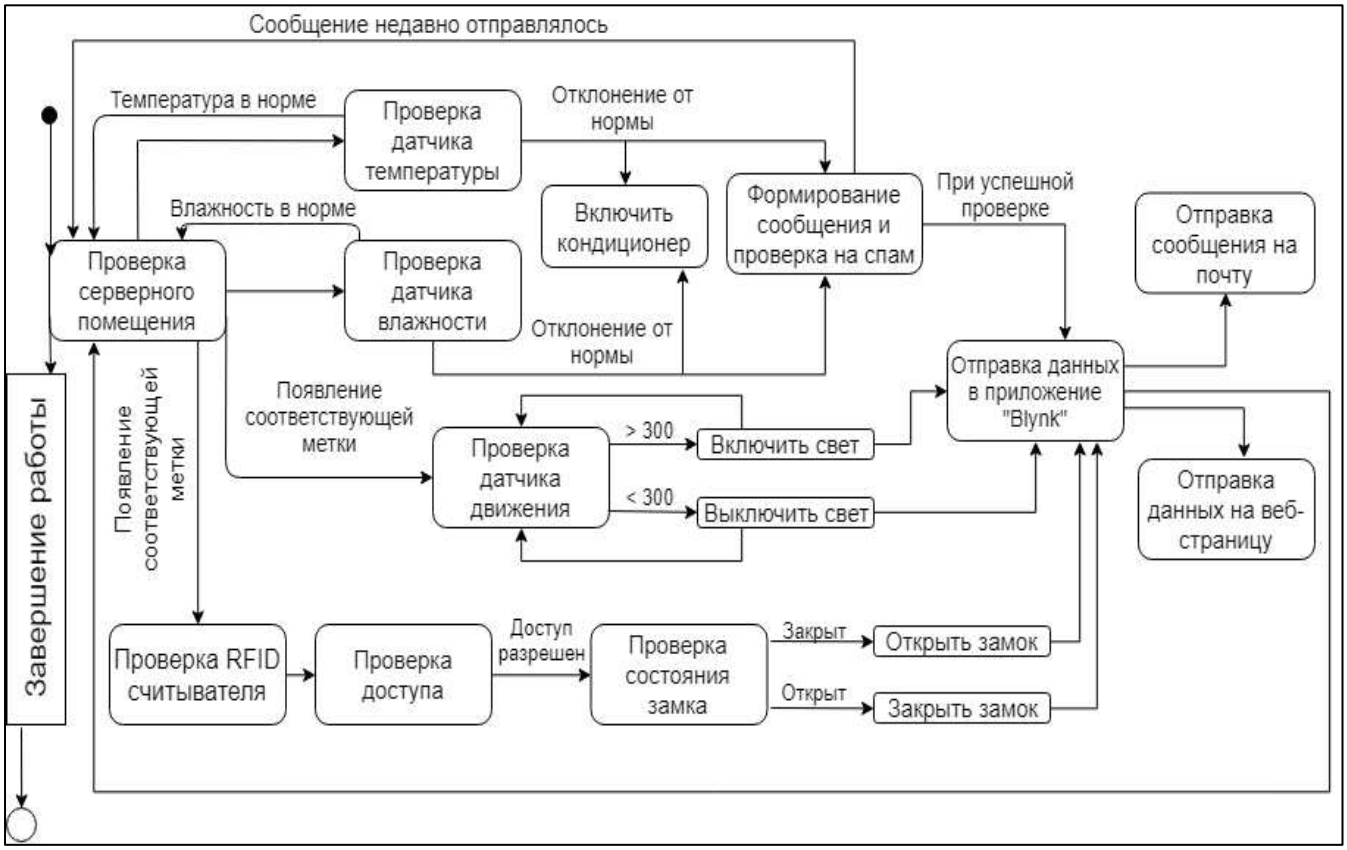


Рисунок 29 – Модель состояний программно-аппаратного комплекса

Модель компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. Во многих средах разработки модуль или компонент соответствует файлу. Пунктирные стрелки, соединяющие модули, показывают отношения взаимозависимости, аналогичные тем, которые имеют место при компиляции исходных текстов программ. На рисунке 30 показана модель компонентов разрабатываемой системы.

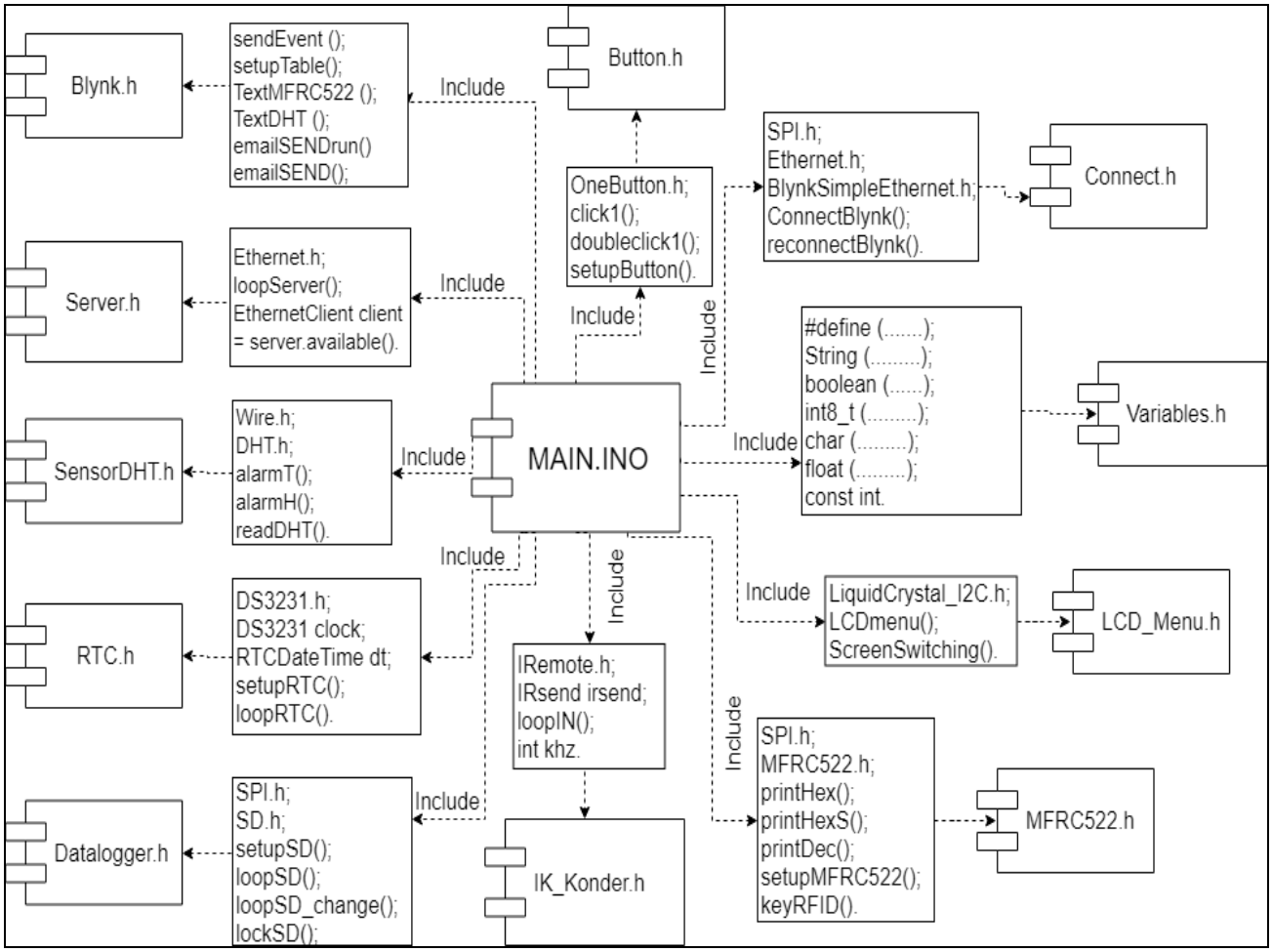


Рисунок 30 – Модель компонентов

Так же была построена архитектура комплекса, которая содержит графические изображения датчиков, устройств, процессов и связей между ними. Данная диаграмма показана на рисунке 31.

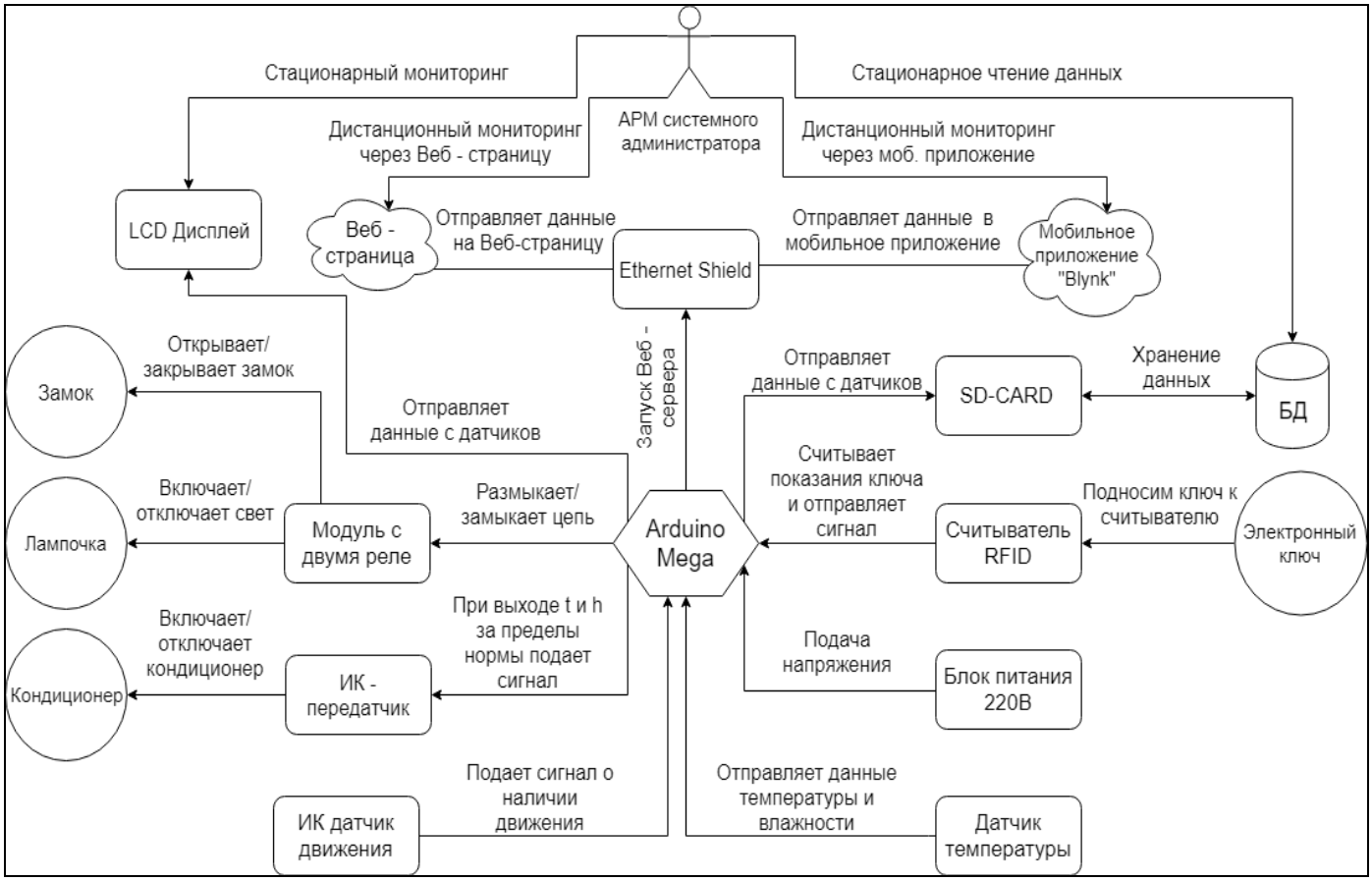


Рисунок 31 – Архитектура комплекса

Пояснительная записка к архитектуре комплекса представлена на рисунке 32.

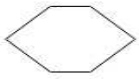
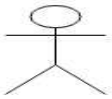
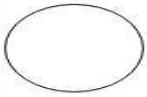
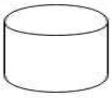
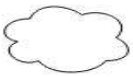
	Устройство которое организует выполнение всех процессов системы
	Рабочее место сотрудника
	Датчики подключаемые к устройству
	Сторонние компоненты которые взаимодействуют с системой
	База данных
	Удаленное хранение и визуализация обработанных системой данных

Рисунок 32 – Пояснительная записка к архитектуре комплекса

2.5 Вывод к главе 2

Итогом данной главы можно считать создание системы мониторинга, схема подключения датчиков и модулей продемонстрирована на рисунке 33. Пояснение к схеме подключения представлено в приложении Б.

Результат работы системы в качестве дистанционного мониторинга через веб-страницу представлен на рисунках 34-35.

Результат работы системы в качестве дистанционного мониторинга через мобильное приложение «Blynk» представлен на рисунке 36.

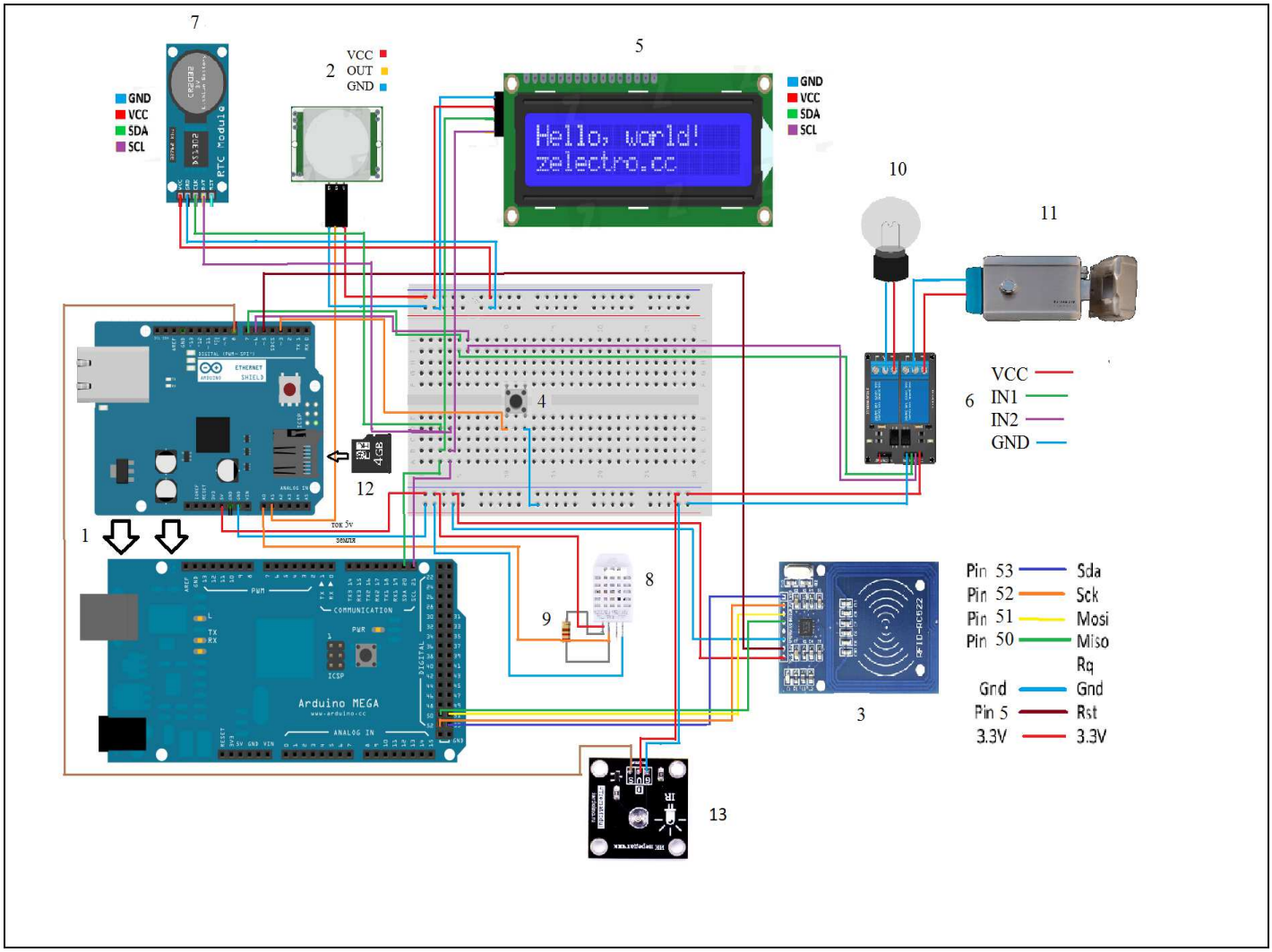


Рисунок 33 – Схема подключения датчиков и модулей

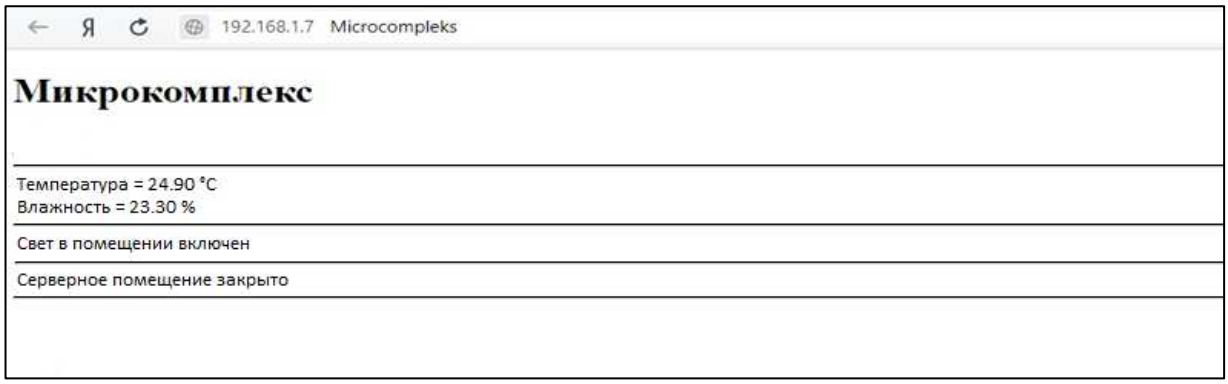


Рисунок 34 – Демонстрация параметров помещения на веб-странице

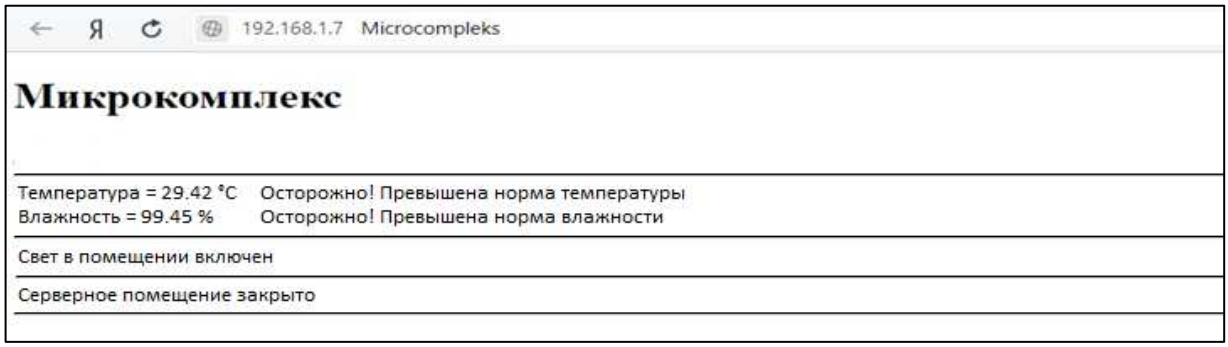


Рисунок 35 – Демонстрация параметров помещения с отклонениями от нормы

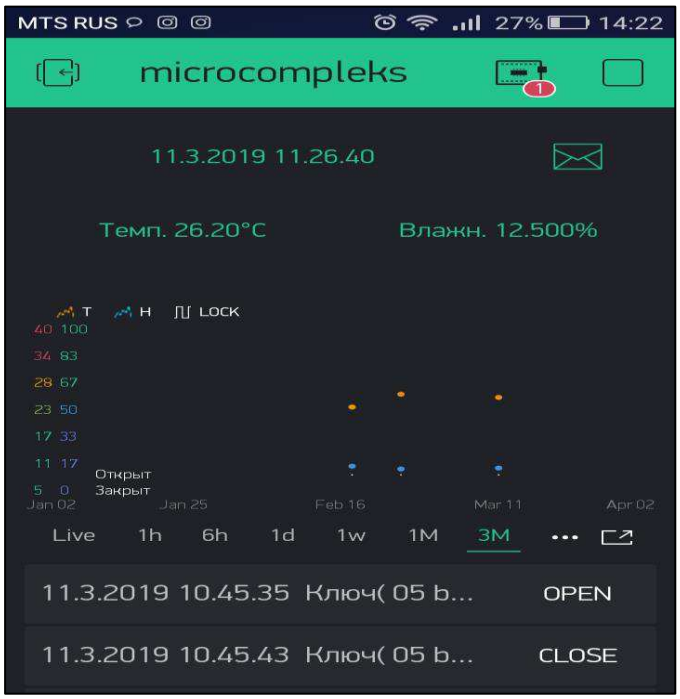


Рисунок 36 – Демонстрация параметров помещения в мобильном приложении «Blynk»

ЗАКЛЮЧЕНИЕ

В результате была достигнута цель бакалаврской работы, а именно обеспечить мониторинг и контроль состояния оборудования серверного помещения для ООО «Коммунальные информационные системы» в г. Красноярск.

Для достижения поставленной цели были решены следующие задачи:

- Обзор существующих систем мониторинга.
- Выявление требований к проектируемой системе мониторинга.
- Проектирование информационной системы.
- Прототипирование системы мониторинга и внедрение на предприятии ООО «Коммунальные информационные системы».

В результате действий, описанных в первой главе, был проведен анализ предметной области, что позволило выявить требования к разрабатываемой информационной системе.

Результатом второй главы стало решение задачи прототипирование системы мониторинга. В рамках решения данной задачи была определена структура устройства, было выбрано основное оборудование, которое будет использоваться в системе мониторинга, проведен анализ функциональных требований и построены диаграмма состояний программно-аппаратного комплекса, диаграмма компонентов и диаграмма развертывания.

При разработке было выявлено что, выбор использованного оборудования полностью себя оправдал и позволил разработать полноценную систему мониторинга, соответствующую выявленным требованиям.

Данная система мониторинга была внедрена на предприятии ООО «Коммунальные информационные системы» о чем свидетельствует акт об использовании результатов проектирования в рамках бакалаврской работы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Диаграмма прецедентов [Электронный ресурс] : / – Режим доступа: https://ru.wikipedia.org/wiki/

2. Диаграмма компонентов [Электронный ресурс] : / – Режим доступа: https://ru.wikipedia.org/wiki/

3. Диаграмма развертывания [Электронный ресурс] : / – Режим доступа: https://ru.wikipedia.org/wiki/

4. Модель реализации [Электронный ресурс] : / – Режим доступа: https://sites.google.com/site/anisimovkhv/learning/pris/lecture/tema15

5. Разработка структурной схемы устройства [Электронный ресурс] : / – Режим доступа: https://studfiles.net/preview/5154328/

6. Use-case diagrams [Электронный ресурс] : / – Режим доступа: https://www.uml-diagrams.org/use-case-diagrams-examples.html

7. Component diagram [Электронный ресурс] : / – Режим доступа: https://www.tutorialspoint.com/ uml/uml_component_diagram.htm

8. Deployment diagram [Электронный ресурс] : / – Режим доступа: https://www.tutorialspoint.com/uml/uml_deployment_diagram.htm

9. Solution Control Systems [Электронный ресурс] : / – Режим доступа: http://www.soliton.com.ua/index.html

10. Кситал GSM [Электронный ресурс] : / – Режим доступа: https://ksytal.ru/produkcziya/katalog-tovarov/ksital-gsm

11. Гаврилов А.В. Использование современных CASE-средств структурного проектирования при обучении студентов по направлению подготовки «прикладная информатика» / А.В. Гаврилов [Текст] – Открытое образование. 2015, №4, с.22-27.

12. Строительные нормы. Инструкция по проектированию зданий и помещений для электронно-вычислительных машин СН 512-78 [Электронный ресурс] : / – Режим доступа: https://ksytal.ru/produkcziya/katalog-tovarov/ksital-gsm

13. СТО 4.2–07–2014 Система менеджмента качества. Общие требования

к построению, изложению и оформлению документов учебной деятельности. – Введ. 30.12.2013. – Красноярск : СФУ, 2013. – 60 с.

14. Arduino Mega 2560 [Электронный ресурс]: / – Режим доступа: http://arduino.ru/Hardware/ArduinoBoardMega2560

15. Arduino Nano [Электронный ресурс] : / – Режим доступа: http://arduino.ru/Hardware/ArduinoBoardNano

16. Arduino [Электронный ресурс] : / – Режим доступа: https://www.arduino.cc

17. Программирование Arduino [Электронный ресурс] : / – Режим доступа: http://arduino.ru/Reference

18. Функции Arduino [Электронный ресурс] : / – Режим доступа: http://codius.ru/articles. html

19. ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы [Электронный ресурс] : / – Режим доступа: http://docs.cntd.ru/document/gost-34-602-89

ПРИЛОЖЕНИЕ А

ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА РАЗРАБОТКУ ИНФОРМАЦИОННОЙ СИСТЕМЫ МОНИТОРИНГА ПАРАМЕТРОВ СЕРВЕРНОГО ПОМЕЩЕНИЯ

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

на разработку информационной системы мониторинга параметров серверного помещения

на ____ листах

СОГЛАСОВАНО

Руководитель организации _____ Кувшинов Александр Альбертович
подпись

М.П.

Разработчик _____ Павленко Игорь Игоревич
подпись

Дата составления: «___»_____ 2019

СОДЕРЖАНИЕ

1. Общие сведения

1.1. Наименование микрокомплекса

1.1.1. Полное наименование микрокомплекса

Микрокомплекс на базе микроконтроллер для мониторинга параметров серверного помещения

1.1.2. Краткое наименование микрокомплекса

«МБММПСП»

1.2. Область применения.

Применяется для контроля параметров серверных помещений на предприятии.

1.3. Основания для проведения работ

Основанием для разработки «МБММПСП» является задание на выпускную квалификационную работу.

1.4. Наименование организаций – Заказчика и Разработчика

1.4.1. Заказчик

Заказчик: «ООО Коммунальные информационные системы»

Адрес фактический: г. Красноярск, ул. Академгородок 50/45

2. Назначение и цели создания микрокомплекса

2.1. Назначение микрокомплекса

«МБММПСП» предназначена для контроля следующих параметров серверного помещения:

- контроль температуры и влажности;
- контроль доступа;
- контроль освещённости.

1.2. Цели создания микрокомплекса

Микрокомплекс создается с целью:

- Отслеживания параметров микроклимата помещения
- Оповещения ответственных лиц в случае нарушения климатических условий и несанкционированного доступа в серверное помещение

2.3. Решаемые задачи

- Составление структурной схемы микрокомплекса
- Разработка спецификаций и узлов микрокомплекса
- Создание основного программного модуля
- Создание макета микрокомплекс

2.4. Ожидаемый результат

Повышение срока эксплуатации оборудования за счёт стабилизации температуры в помещении

3. Характеристика объектов автоматизации

Объектом автоматизации является помещение для размещения серверов «ООО Коммунальные информационные системы». Основным направлением деятельности компании является разработка, внедрение и сопровождение автоматизированных информационных систем различного назначения.

Серверное помещение характеризуется следующими параметрами:

- Площадь – $4{,}5м^2$;
- Количество серверов – 8 шт;
- Среднее тепловыделение – 2800 Вт.
- Температура воздуха - $20^0 \pm 2\ ^oC$ (не более 25 ^oC) СН 512-78 (п.3)
- Относительная влажность воздуха - 20-70 % (не более 75 % в холодный период, в теплый для 25 °С - не более 65 %, для 24 °С и ниже - не более 70 %) СН 512-78 (п.3)

4. Требования к микрокомплексу

4.1. Требования к микрокомплексу в целом

4.1.1. Требования к структуре и функционированию микрокомплекса

Микрокомплекс на базе микроконтроллера для мониторинга параметров серверного помещения должен быть централизованным.

Микрокомплекс на базе микроконтроллера для мониторинга параметров серверного помещения должен состоять из следующих модулей и их функций:

1) Модуль температуры и влажности

Выполняет функцию мониторинга и регулировки температуры в серверном помещении. При повышении/снижении температуры выше/ниже среднего значения, должен регулировать работу кондиционера.

2) Модуль освещенности

Выполняет функцию включения/отключения света при присутствии/отсутствии персонала в серверном помещении соответственно.

3) Модуль контроля доступа

Выполняет функцию регистрации входящих в помещении лиц.

4) Модуль Визуализации стационарный

Выполняет функцию демонстрации основных параметров помещения на устройстве отображения визуальной информации LCD экран.

5) Модуль Визуализации удаленный

Выполняет функцию демонстрации основных параметров помещения на веб-странице и в мобильном приложении «Blynk».

6) Модуль хранения данных

Выполняет функцию хранения данных с датчиков на sd-card.

4.1.2. Требования к надежности

4.1.2.1. Состав показателей надежности для микрокомплекса в целом

Уровень надежности должен достигаться согласованным применением организационных, организационно-технических мероприятий и программно-аппаратных средств.

Надежность должна обеспечиваться за счет:

- Дополнительной установки аккумулятора для микрокомплекса, обеспечивающего работу в автономном режиме не менее 20 минут;
- Наличия энергонезависимой памяти.
- своевременного выполнения процессов администрирования МБММПСП;
- соблюдения правил эксплуатации и технического обслуживания программно-аппаратных средств;
- предварительного обучения пользователей и обслуживающего персонала.

4.1.2.2. Перечень аварийных ситуаций, по которым регламентируются требования к надежности

Под аварийной ситуацией понимается аварийное завершение процесса работы МБММПСП при внешних воздействиях на микрокомплекс: удары, затопления и т.д.

4.1.2.3. Требования к надежности технических средств и программного обеспечения

К надежности оборудования предъявляются следующие требования:

- в качестве компонентов микрокомплекса должны использоваться средства с повышенной надежностью;
- микрокомплекс должен иметь возможность восстановления в случаях сбоев электропитания, за счет перехода на резервное питание.

К надежности электроснабжения предъявляются следующие требования:

- с целью повышения отказоустойчивости микрокомплекса необходима обязательная комплектация источником резервного питания с возможностью автономной работы системы не менее 20 минут.

4.1.3. Требования к безопасности

Аппаратное обеспечение микрокомплекса должно соответствовать требованиям пожарной безопасности в производственных помещениях по ГОСТ 12.1.004-91. «ССБТ. Пожарная безопасность. Общие требования».

Должно быть обеспечено соблюдение общих требований безопасности в соответствии с ГОСТ 12.2.003-91. «ССБТ. Оборудование производственное. Общие требования безопасности» при обслуживании системы в процессе эксплуатации.

4.2. Требования к лингвистическому обеспечению.

Программы для работы микрокомплекса должна разрабатываться в среде программирования «Arduino IDE» с использованием языка программирования C++.

5. Состав и содержание работ

Состав и содержание работ представлены в таблице 3.

Таблица 3 – Состав и содержание работ.

Наименование работ	**Срок сдачи**
Создание технического задания	21.06.2018
Составление структурной схемы микрокомплекса	20.07.2018
Разработка спецификаций и узлов микрокомплекса	30.09.2018
Создание основного программного модуля	08.11.2018
Сборка макета микрокомплекса	11.01.2019
Тестирование и отладка микрокомплекса	15.03.2019

ПРИЛОЖЕНИЕ Б

Пояснительная записка к схеме подключения

Таблица 4 – Пояснительная записка к схеме подключения.

№	Наименование	Пояснение
1	Arduino Mega2560 + Ethernet Shield	Платформа для работы с модулями и датчиками. Ethernet используется для подключения к интернету.
2	Инфракрасный датчик движения пироэлектрический сенсор	Используется для включения/выключения света
3	Считыватель RFID MFRC-522.	Используется для осуществления пропускного режима (светодиод используется как заменитель замка)
4	Кнопка тактовая	Используется для переключения пунктов меню на LCD экране
5	LCD дисплей 1602 символьный на HD44780 с подсветкой, зелёный	Используется для отображения параметров, передаваемых датчиками (имеет 3 пункта меню)
6	Модуль с двумя реле для Arduino.	Используется в связке с ИК датчиком движения и рфид считывателем для переключения света(вкл/выкл), а также открытия/закрытия электромеханического замка
7	Прецезионные часы реального времени на DS3231	Позволяет считывать текущее время и дату, для записи данных в sd-card
8	Цифровой датчик температуры и влажности DHT22	Используется для измерения температуры и влажности в помещении
9	Резистор с сопротивлением 10 кОм	
10	Лампочка накаливания	Включение/отключение света
11	Электромеханический замок	Открытие/закрытие двери
12	Micro SD-card	Используется для хранения параметров в БД
13	ИК – передатчик	Используется для управления кондиционером

ПРИЛОЖЕНИЕ В

Скетч инициализации компонентов системы

```
#include "Variables.h"  // Подключаем вкладку с Переменными

/*  Подключаем вкладку с Функциями отвечающими за подключение
 *  Тут подключаются Библиотеки для работы с Интернетом и приложением Blynk
 *  Введите Авторизационный ключ Blynk
 */
#include "Connect.h"
BlynkTimer timer;       // Создаем объект BlynkTimer с именем timer
#include "RTC.h"        // Подключаем вкладку с таймером реального времени
#include "BlynkCOD.h"   // Подключаем вкладку с кодом функций Blynk
#include "Datalogger.h" // Подключаем вкладку Записи на SD карту
#include "SensorDHT.h"  // Подключаем вкладку Датчика температуры и влажности
#include "Modul_mfrc522.h" // Подключаем вкладку считывателя RFID меток
#include "LCD_Menu.h"   // Подключаем вкладку LCD экрана
#include "Button.h"     // Подключаем вкладку Физической кнопки
#include "Server.h"     // Подключаем вкладку Сервера
#include "IK_Konder.h"     // Подключаем вкладку Сервера

// Функция считывания сигналов с датчика движения PIR
void PIRsensor(){
    //считываем состояние датчика если сигнал выше 300 значит поднимаем флаг
statePIR, иначе опускаем statePIR
  if (analogRead(PIN_PIR)>300){statePIR = HIGH;}
  else                        {statePIR = LOW;}
  //если есть движение включить Освещение иначе выключить Освещение
  if (statePIR == HIGH) {digitalWrite(PIN_RELE_SVET, HIGH);}
  else                  {digitalWrite(PIN_RELE_SVET, LOW);}
}
//Блок при включении ардуино инициализация всей периферии
void setup() {

// Настройка входов и выходов
  pinMode(PIN_RELE_SVET, OUTPUT);  // Реле Освещения
  pinMode(PIN_RELE_LOCK, OUTPUT);  // Реле Замок
  setupButton(); // Кнопка переключения экранов (Button.h)

// инициализируем сериал монитор
  Serial.begin(9600);
  while (!Serial);       // Ждем пока не инициализируется монитор серийного порта
  Serial.println(F("Start"));

// Инициализация Часов Реального Времени (RTC.h)
  setupRTC();

// Инициализация lcd  (LCD_Menu.h)
  lcd.init();
  lcd.backlight(); // Включаем подсветку

// Инициализируем датчик Температуры и Влажности (SensorDHT.h)
  Serial.println(F("DHTxx test!"));
  dht.begin();

// Инициализация SD карты (Datalogger.h)
  setupSD();
```

```
  delay(1000);

// Инициализируем модуль MFRC522 (Modul_mfrc522.h)
  setupMFRC522();
  delay(1000);
// Вызываем функцию подключения к Blynk (Connect.h)
  reconnectBlynk();
  delay(1000);
  setupTable(); //Стереть таблицу при старте

//запускаем сервер (Server.h)
  server.begin();

// Настраиваем таймеры
 timer.setInterval(60000L, reconnectBlynk); // Пере подключение Blynk при обрыве соединения
 timer.setInterval(2000, readDHT);     // Читаем температуру каждые 2 секунды
 ID_Timer_keyRFID = timer.setInterval(500, keyRFID);   // Читаем RFID метки и открываем закрываем замок
 ID_Timer_ScreenSwitching =  timer.setInterval(2000, ScreenSwitching);     // Переключаем меню экранов LCD
 timer.disable(ID_Timer_ScreenSwitching);  // Выключаем переключение экранов LCD
 timer.setInterval(50, [](){button1.tick();}); // Проверяем не нажата ли кнопка
 timer.setInterval(2000, LCDmenu);     // Обновляем данные на LCD экране
 timer.setInterval(1000, PIRsensor);   // Читаем состояние от Датчика движения
 timer.setInterval(1000, loopServer);  // Обрабатываем запросы от сервера
 timer.setInterval(1000, loopRTC);     // Читаем время с модуля RTC DS3231
 timer.setInterval(60000, loopSD);     // Сохраняем данные на SD карту
 timer.setInterval(1000, emailSENDrun);// Отправляем сообщение если требуется
}//setup()

void loop() {
  if (Blynk.connected()) // Если есть подключение к серверу Blynk
  { Blynk.run();} // Тик Blynk
  timer.run(); // Тик таймера
}//loop
```

ПРИЛОЖЕНИЕ Г

Скетч работы с мобильным приложением «Blynk»

```
int rowIndex = 0; // Начальный индекс строки таблицы

/* При открытии или закрытии замка нужно составить строку Дата+Время+ Ключ (00 00 00 00) Состояние */

// Составляем строку и отправляем в виджет Таблица
void sendEvent(String UID, String Value) {
  String Name = DataTime;
  Name += "  Ключ(" + UID;
  Name += ")  Замок";

  // Добовляем строку в таблицу
  Blynk.virtualWrite(VPIN_TABLE, "add", rowIndex, Name, Value);

  //Выделение последней добавленной строки в таблице
  Blynk.virtualWrite(VPIN_TABLE, "pick", rowIndex);
  rowIndex++;
}

// Тут настройка таблице при инициализации Ардуино
void setupTable()
{
  //Стереть таблицу при старте
  Blynk.virtualWrite(VPIN_TABLE, "clr");
}

// Код Mail сообщений

// Собираем строку о событии от считывателя RFID меток, для отправки по электронной почте
String TextMFRC522 (String UID, String Value){
    String Text="";
    Text += Data;
    Text += ",";
    Text += Time;
    Text += ",Ключ," + UID;
    Text += ",";
    Text += Value;
    return   Text;
}

// Собираем строку о событии от DHT, для отправки по электронной почте
String TextDHT (String Val, String X){
    String Text="";
    Text += Data;
    Text += ",";
    Text += Time;
    Text += ",";
    Text += X;
    Text += ",";
    Text += Val;
    return   Text;
}
```

```
// *** Внимание: вы можете отправлять только одно сообщение в течение 15 секунд! ***
/*
 *  Если есть письма ожидающие отправки (waitSEND=true)
 *  и с последнего отправленного письма прошло больше 15 секунд
(stateSEND_Mail_Timer=false)
 *  Отправим сообщение
 *  Поднимем флаг (stateSEND_Mail_Timer=true)
 *  Опустим флаг (waitSEND=false)
 *  Заведем таймер который опустит флаг stateSEND_Mail_Timer через 15 секунд
 *// Отправка сообщения
void emailSENDrun(){
#ifdef USE_MAIL_SEND
  if (waitSEND && !stateSEND_Mail_Timer){
    stateSEND_Mail_Timer=true; // Поднимем флаг (stateSEND_Mail_Timer=true)
    waitSEND = false;          // Опустим флаг (waitSEND=false)
    Blynk.email(MAIL, Subject, Message); // Отправим сообщение

 // Заведем таймер который опустит флаг stateSEND_Mail_Timer через 15 секунд
    timer.setTimeout(15000, [] () {stateSEND_Mail_Timer=false;});
  }
#endif
}

/*
 * Если нет ожидающих отправки сообщений то Сотрем данные которые хранились в строке
Message
 * иначе добавим символ переноса строки и добавим новый текст в строку с сообщением
 */
// Собираем строку для отправки и отправляем сообщение
void emailSEND(String Text) {

#ifdef USE_MAIL_SEND

  if (waitSEND == false){
    waitSEND = true;
    Message = "<br>"; // Стираем старые данные
    Message += Text;
  }
  else {
    waitSEND = true;
    Message += "<br>";// Добавляем новые данные
    Message += Text;
  }
  Serial.println(Message); // Выведем составленную строку в Сериал монитор
  emailSENDrun(); // Вызовем функцию отправки сообщения

#endif
}
```

ПРИЛОЖЕНИЕ Д

Скетч работы кнопки переключения меню LCD экрана

```
#include "OneButton.h" // Подключаем библиотеку с обработкой кнопок

OneButton button1(PIN_BUTTON_MENU, true); // Объявляем экземпляр класса

// При однократном нажатии вызовем переключение экрана
void click1() {
  Serial.println("ScreenSwitching");
  ScreenSwitching(); // Функция переключения экранов
} // click1

// При двойном клике включим или выключим автоматическое переключение экранов
void doubleclick1() {
  static boolean stateScreenSwitching = false; // Флаг состояния автоматического пе-
реключения экранов
 // благодаря static изменение значение будет сохранятся

  stateScreenSwitching = !stateScreenSwitching;     // Инвертировать флаг
  if (stateScreenSwitching){timer.enable(ID_Timer_ScreenSwitching); Seri-
al.println("ScreenSwitching AUTO ON");}
  else                      {tim-
er.disable(ID_Timer_ScreenSwitching);Serial.println("ScreenSwitching AUTO OFF");}
} // doubleclick1

void setupButton() {
  button1.attachClick(click1); // объявляем по одному клику выполнять функцию click1
  button1.attachDoubleClick(doubleclick1); // объявляем по двойному клику выполнять
функцию doubleclick1
}
```

ПРИЛОЖЕНИЕ Е

Скетч подключения к сети

```
#define BLYNK_PRINT Serial
#include <SPI.h>
#include <Ethernet.h>
#include <BlynkSimpleEthernet.h>

// Mac-адрес должен отличаться для каждого устройства в локальной сети
byte arduino_mac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };
IPAddress device_ip  (192, 168,   10,  224);
IPAddress dns_ip     (192, 168,   10,   1);
IPAddress gateway_ip (192, 168,   10,   1);
IPAddress subnet_mask(255, 255, 255,   0);

// Авторизиционный ключ Blynk прихоит на почту из приложения
// Blynk Auth Code
#define AUTH          "akls3e41dk1349sdajwqe123jfsdf"

//#define USE_LOCAL_SERVER // Раскоментируйте если нужен локальный сервер Blynk

#ifdef USE_LOCAL_SERVER
      #define SERVER          IPAddress (192, 168, 10, 9) // Свой IP пишите
      #define SERVER          "Blynk.dlinkddns.com" // Имя своего хоста (Пример:
"Blynk.dlinkddns.com")

#endif

//          Функции для подключения к Blynk          //
void ConnectBlynk()
{
#ifdef USE_LOCAL_SERVER // Если используются локальный сервер

  Ethernet.begin (arduino_mac, device_ip, dns_ip, gateway_ip, subnet_mask);
  // Проверьте наличие оборудования Ethernet
  if (Ethernet.hardwareStatus() == EthernetNoHardware) {
    Serial.println(F("Ethernet shield was not found.  Sorry, can't run without hard-
ware. :("));
    while (true) {
      delay(1); // ничего не делайте, нет смысла работать без оборудования Ethernet
    }
  }
  // Проверка подключения Кабеля Ethernet
  if (Ethernet.linkStatus() == LinkOFF) {
    Serial.println("Ethernet cable is not connected.");
  }
  else{
    Serial.println(Ethernet.localIP());
    Blynk.config(AUTH, SERVER, 8080);
    Blynk.connect(1000UL);
  }

  #else // Иначе (не используется локальный сервер) конектимся к "blynk-cloud.com"
Ethernet.begin (arduino_mac, device_ip, dns_ip, gateway_ip, subnet_mask);
  // Проверка подключения Кабеля Ethernet
  if (Ethernet.hardwareStatus() == EthernetNoHardware) {
```

```
    Serial.println(F("Ethernet shield was not found.  Sorry, can't run without hard-
ware. :("));
    while (true) {
      delay(1); // ничего не делайте, нет смысла работать без оборудования Ethernet
    }
  }
  // Проверка подключения Кабеля Ethernet
  if (Ethernet.linkStatus() == LinkOFF) {
    Serial.println("Ethernet cable is not connected.");
  }
  else{
    Serial.println(Ethernet.localIP());
    Blynk.config(AUTH, "blynk-cloud.com", 80);
    Blynk.connect(1000UL);
  }

#endif
}//ConnectBlynk()


// Реконектимся если обрыв связи
void reconnectBlynk() {
  if (!Blynk.connected()){
    BLYNK_LOG("Disconnected now");
    ConnectBlynk();
    if (Blynk.connected()){BLYNK_LOG("Reconnected");}
    else                  {BLYNK_LOG("Not reconnected");}
  }
}//reconnectBlynk()
```

ПРИЛОЖЕНИЕ Ж

Скетч работы SD-Card

```
#include <SPI.h>
#include <SD.h>

// Инициализация SD карты
void setupSD() {
  Serial.print(F("Initializing SD card..."));

  // проверьте, присутствует ли карта и может ли она быть инициализирована:
  if (!SD.begin(SDCARD_CS)) {
    Serial.println(F("Card failed, or not present"));
      // return; больше ничего не делай:
  }
  Serial.println(F("card initialized."));
}

/*
 * Две функции
 * loopSD() // Сохраням данные о температуре и влажности
 * loopSD_change() // Сохраням данные о температуре и влажности только если они из-
менились
* //Настраиваем таймеры
 * примеры таймеров:
 * timer.setInterval(60000, loopSD);            // Сохраняем данные на SD карту
 * timer.setInterval(60000, loopSD_change);     // Сохраняем данные на SD карту
 */

// Сохраням данные о температуре и влажности
void loopSD() {
    // создать строку для сборки данных в журнал:
  String dataString = "";
  dataString += Data;
  dataString += ",";
  dataString += Time;
  dataString += ",";
  dataString += "Humi ";
  dataString += ",";
  dataString += String(h);
  dataString += ",";
  dataString += "Temp ";
  dataString += ",";
  dataString += String(t);


// Открыть файл. обратите внимание, что одновременно может быть открыт только один
файл,
// таким образом, вы должны закрыть этот, прежде чем открывать другой.
  File dataFile = SD.open("loght.txt", FILE_WRITE);

  // если файл доступен, напишите в него:
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    // печать на последовательный порт:
    Serial.print("DataLogger  ");
```

```
    Serial.println(dataString);
  }
  // если файл не открыт, появится сообщение об ошибке:
  else {
    Serial.println(F("error opening loght.txt"));
  }

}//loopSD()

// Сохраням данные о температуре и влажности только если они изменились
void loopSD_change() {
    // make a string for assembling the data to log:
  String dataString = "";
  static float last_h=0;
  static float last_t=0;

  if (last_h != h && last_t != t){
  last_h = h;
  last_t = t;

  dataString += Data;
  dataString += ",";
  dataString += Time;
  dataString += ",";
  dataString += "Humi ";
  dataString += ",";
  dataString += String(h);
  dataString += ",";
  dataString += "Temp ";
  dataString += ",";
  dataString += String(t);


// Открыть файл. обратите внимание, что одновременно может быть открыт только один
файл,
// таким образом, вы должны закрыть этот, прежде чем открывать другой.
  File dataFile = SD.open("loght.txt", FILE_WRITE);

  // если файл доступен, напишите в него:
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    // печать на последовательный порт:
    Serial.print("DataLogger  ");
    Serial.println(dataString);
  }
  // если файл не открыт, появится сообщение об ошибке:
  else {
    Serial.println(F("error opening loght.txt"));
  }
 }
}

// Сохраняем данные об открытии или закрытии двери
void lockSD(String UID, String Value) {

  // создать строку для сборки данных в журнал:
  String dataString = "";

  dataString += Data;
```

```
  dataString += ",";
  dataString += Time;
  dataString += ",";
  dataString += "Key ";
  dataString += ",";
  dataString += UID;
  dataString += ",";
  dataString += Value;


// Открыть файл. обратите внимание, что одновременно может быть открыт только один
файл,
// таким образом, вы должны закрыть этот, прежде чем открывать другой.
  File dataFile = SD.open("lock.txt", FILE_WRITE);

  // если файл доступен, напишите в него:
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    // печать на последовательный порт:
    Serial.print("DataLogger  ");
    Serial.println(dataString);
  }
  // если файл не открыт, появится сообщение об ошибке:
  else {
    Serial.println(F("error opening lock.txt"));
  }

}
```

ПРИЛОЖЕНИЕ 3

Скетч работы LCD дисплея

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>      //Библиотека для LCD экрана

LiquidCrystal_I2C lcd(0x27,16,2);   // Объявляем экран по адресу 0x27 на шине I2C
                                    // и 2 строки по 16 символов

// Прототип функции, для того что бы компилятор знал, что такая функция есть
void LCDmenu();

// Функция отвечающая за переключение экранов
void ScreenSwitching(){
  MenuPos++;
  if(MenuPos > 2){MenuPos=0;}
  LCDmenu();
}

// Функция вывода информации на LCD экран
void LCDmenu(){
static int8_t LastMenuPos=100;
switch (MenuPos) {
    case 0:
      //выполняется, когда MenuPos равно 0

        if (LastMenuPos!=MenuPos){ // Если в прошлый раз был другой экран, то сотрем
все и выведем новый экран
          lcd.clear();
          lcd.setCursor(0, 0); // Устанавливаем курсор в начало 1 строки
          lcd.print(F("Hum = % ")); // Выводим текст
          lcd.setCursor(7, 0); // Устанавливаем курсор на 7 символ
          lcd.print(h, 1); // Выводим на экран значение влажности
          lcd.setCursor(0, 1); // Устанавливаем курсор в начало 2 строки
          lcd.print(F("Temp = \1C ")); // Выводим текст, \1 - значок градуса
          lcd.setCursor(7, 1); // Устанавливаем курсор на 7 символ
          lcd.print(t,1); // Выводим значение температуры
        }else{    //   Иначе только обновим информацию в нужных местах
          lcd.setCursor(7, 0); // Устанавливаем курсор на 7 символ
          lcd.print(h, 1); // Выводим на экран значение влажности
          lcd.setCursor(7, 1); // Устанавливаем курсор на 7 символ
          lcd.print(t,1); // Выводим значение температуры
        }
      break;
    case 1:
      //выполняется, когда MenuPos равно 1
        lcd.clear();
        lcd.setCursor(0, 0); // Устанавливаем курсор в начало 1 строки
        if(statePIR == HIGH){lcd.print(F("SVET VKL"));}
        else                 {lcd.print(F("SVET OTKL"));}
      break;
    case 2:
      //выполняется, когда MenuPos равно 2
        lcd.clear();
        lcd.setCursor(0, 0); // Устанавливаем курсор в начало 1 строки
        if(!StateKeyRFID){
        lcd.print(F("CLOSED"));
```

```
        }
        else if (StateKeyRFID){
        lcd.print(F("OPEN"));
        }
      break;
    default:
      // выполняется, если не выбрана ни одна альтернатива
        lcd.clear();
        lcd.setCursor(0, 0); // Устанавливаем курсор в начало 1 строки
        lcd.print(F("MenuPos Error")); // Выводим текст
      }
  LastMenuPos=MenuPos;
}
```

ПРИЛОЖЕНИЕ И

Скетч работы RFID считывателя

```
#include <SPI.h>
#include <MFRC522.h>          // Подключаем библиотеку для работы с модулем RC522

// Создаем объект для работы с библиотекой MFRC522 и сообщаем ей пины подключения
модуля
MFRC522 mfrc522(MFRC522_SS_PIN, MFRC522_RST_PIN);

// Card UID: 05 B6 97 BB
// Card UID: 50 27 9E 7C
// Читаем при помощи DumpInfo_My

// Вспомогательная подпрограмма для вывода массива байтов в виде шестнадцатеричных
значений в Сериал монитор.
void printHex(byte *buffer, byte bufferSize) {
  for (byte i = 0; i < bufferSize; i++) {
    Serial.print(buffer[i] < 0x10 ? " 0" : " ");
    Serial.print(buffer[i], HEX);
  }
}

// Вспомогательная подпрограмма для вывода массива байтов в виде строки.
String printHexS(byte *buffer, byte bufferSize) {
  String UID;
  for (byte i = 0; i < bufferSize; i++) {
    if(buffer[i] < 0x10){UID+= " 0";}
    else                {UID+= " ";}
    UID+= String(buffer[i], HEX);
  }
  return UID;
}

// Вспомогательная подпрограмма для вывода массива байтов в виде значений dec в Се-
риал монитор.
void printDec(byte *buffer, byte bufferSize) {
  for (byte i = 0; i < bufferSize; i++) {
    Serial.print(buffer[i] < 0x10 ? " 0" : " ");
    Serial.print(buffer[i], DEC);
  }
}

// Инициализация считывателя RFID меток
void setupMFRC522(){
  SPI.begin();                                      // Инициализируем интерфейс SPI
  mfrc522.PCD_Init();                               // Инициализируем модуль MFRC522
  mfrc522.PCD_DumpVersionToSerial();  // Выводим версию прошивки модуля на монитор
серийного порта
// Выводим сообщение о том, что модуль готов к считыванию и ожидает метку
  Serial.println(F("Please, place RFID-label over the reader"));
}

// Чтение карт и ключей с RFID метками
// Логика такая провели ключем открыли, что бы закрыть нужно провести ключем еще раз
void keyRFID(){
```

```
    digitalWrite(PIN_RELE_LOCK, StateKeyRFID); // Управляем реле в зависимости от состояния замка
    Blynk.virtualWrite(VPIN_LOCK, StateKeyRFID); // шлем данные в таблицу Blynk в зависимости от состояния замка

  // Ожидаем метку, и пока ее нет, прерываем дальнейшее выполнение этой функции (keyRFID)
 if ( ! mfrc522.PICC_IsNewCardPresent()) {
  return;
 }

// Пытаемся прочитать метку, и пока это не получилось, прерываем дальнейшее выполнение этой функции (keyRFID)
 if ( ! mfrc522.PICC_ReadCardSerial()) {
  return;
 }
        //    !!!!!ВАЖНО!!!!!!    //
// Весь код ниже будет исполнен только если две предыдущих проверки будут пройдены
 // Выводим в сериал монитор метку UID
    Serial.println(F("Card UID"));
    Serial.print(F("In hex: "));
    printHex(mfrc522.uid.uidByte, mfrc522.uid.size);
    Serial.println();

    Serial.print(F("In dec: "));
    printDec(mfrc522.uid.uidByte, mfrc522.uid.size);
    Serial.println();

// В цикле for проверяем ключ на совпадения с хранящимися в базе данных
    int flagCheck=0;
   for (byte j = 0; j < countKey; j++) {
        for (byte i = 0; i < 4; i++) {
          if (uidCard[j][i] != mfrc522.uid.uidByte[i])
          {flagCheck++; i=4;}
        }
   }

// Если число не совпавших ключей меньше общего количества ключей, то значит хотя бы один ключ совпал
  if (flagCheck<countKey){
  StateKeyRFID=!StateKeyRFID; // Меняем состояние флага
  if (StateKeyRFID){
      // Если true то открыто
      Serial.println(F("OPEN"));
      sendEvent(printHexS(mfrc522.uid.uidByte, mfrc522.uid.size), F("OPEN") ); // Отправка в Blynk
      emailSEND(TextMFRC522(printHexS(mfrc522.uid.uidByte, mfrc522.uid.size), F("OPEN") )); // Отправка письма
      lockSD(printHexS(mfrc522.uid.uidByte, mfrc522.uid.size), F("OPEN")); // Запись в лог

    }else{
      // Если false то закрыто
      Serial.println(F("CLOSE"));
      sendEvent(printHexS(mfrc522.uid.uidByte, mfrc522.uid.size), F("CLOSE") ); // Отправка в Blynk
      emailSEND(TextMFRC522(printHexS(mfrc522.uid.uidByte, mfrc522.uid.size), F("CLOSE") )); // Отправка письма
      lockSD(printHexS(mfrc522.uid.uidByte, mfrc522.uid.size), F("CLOSE")); // Запись в лог
```

```
    }
  }
      digitalWrite(PIN_RELE_LOCK, StateKeyRFID); // Управляем реле в зависимости от
состояния замка
    Blynk.virtualWrite(VPIN_LOCK, StateKeyRFID); // шлем данные в таблицу Blynk в
зависимости от состояния замка

// Нужно для чтения карт не чаще чем раз в 3 секунды
  timer.disable(ID_Timer_keyRFID);  // Отключаем опрос по таймеру функции keyRFID
  timer.setTimeout(3000, [](){      // Через 3 секунды
    timer.enable(ID_Timer_keyRFID); // Включаем опрос по таймеру функции keyRFID
    });

} //keyRFID()
```

ПРИЛОЖЕНИЕ К

Скетч работы часов реального времени

```
#include <Wire.h>
#include <DS3231.h>

DS3231 clock;   // Объявляем экземпляр класса
RTCDateTime dt; // Объявляем структуру с именем dt в нее буду сохранятся все данные от модуля RTC

// Инициализация DS3231
void setupRTC(){

  Serial.println(F("Initialize DS3231"));
  clock.begin();

  // Задать время компиляции эскиза
  //clock.setDateTime(__DATE__, __TIME__);

  // Отключить сигналы тревоги и сбросить аварийные сигналы для этого примера, так как сигналы тревоги поддерживаются батареей
  // В нормальных условиях настройки должны быть сброшены после включения питания и перезапуска микроконтроллера
  clock.armAlarm1(false);
  clock.armAlarm2(false);
  clock.clearAlarm1();
  clock.clearAlarm2();
}

void loopRTC(){
  dt = clock.getDateTime(); // Читаем данные из модуля RTC в структуру dt

  // Заполняем строки
  Data = String(dt.day) + "." + String(dt.month) + "." + String(dt.year);
  Time = String(dt.hour) + "." + String(dt.minute) + "." + String(dt.second);
  DataTime = Data + " " + Time;

  Blynk.virtualWrite(VPIN_LABEL_TIME, DataTime); // Отправляем время в виджет Blynk
  Serial.println(DataTime);                      // Выводим время в Сериал Монитор

Serial.print("Raw data: ");
Serial.print(dt.year);   Serial.print("-");
Serial.print(dt.month);  Serial.print("-");
Serial.print(dt.day);    Serial.print(" ");
Serial.print(dt.hour);   Serial.print(":");
Serial.print(dt.minute); Serial.print(":");
Serial.print(dt.second); Serial.println("");

}
```

ПРИЛОЖЕНИЕ Л

Скетч работы датчика температуры и влажности

```
#include <Wire.h>
#include "DHT.h" //Подключаем библиотеку термодатчик.

#define DHTTYPE DHT21 // DHT 21 (AM2301) //Здесь выбираем какой у нас датчик.

DHT dht(DHTPIN, DHTTYPE); // Объявляем датчик

// Проверяем находится ли температура в норме или вышла за установленные пределы
// Поднимаем и опускаем соответствующие флаги
// Опускаем флаг, разрешаем повторную отпраку сообщения на почту) при приходе темпе-
ратуры в норму
void alarmT(){
  if (t > t_max) {t_alarm_max=true;} else {t_alarm_max=false;
send_t_alarm_max=false;}
  if (t < t_min) {t_alarm_min=true;} else {t_alarm_min=false;
send_t_alarm_min=false;}
}

// То же самое для влажности
void alarmH(){
  if (h > h_max) {h_alarm_max=true;} else {h_alarm_max=false;
send_h_alarm_max=false;}
  if (h < h_min) {h_alarm_min=true;} else {h_alarm_min=false;
send_h_alarm_min=false;}
}

// Читаем данные из датчика DHT
// Отправляем данные в Blynk
// и в случаи аварии отправляем сообщение на почту
void readDHT(){
  h = dht.readHumidity();
  t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

  // Если данные считаны то строки ниже будут выполнены
  if (isnan(h) || isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  Blynk.virtualWrite(VPIN_LABEL_TEMP, t);  // Отправляем данные в Blynk
  Blynk.virtualWrite(VPIN_LABEL_HUMID, h); // Отправляем данные в Blynk

    alarmT(); // Проверяем аварии для Температуры
    alarmH(); // Проверяем аварии для Влажности

/*
 * Если (флаг аварии и не было ли отправки сообщения об аварии)
 * то отправим сообщение и поднимим флаг о том что мы уже отправили
 * сообщение для защиты от спама на электронную почту
 */
    if (t_alarm_max && !send_t_alarm_max)
    {emailSEND(TextDHT("Осторожно! Превышена норма температуры", String(t)));
send_t_alarm_max=true;}
```

```
    if (t_alarm_min && !send_t_alarm_min)
    {emailSEND(TextDHT("Осторожно! Температура ниже нормы", String(t)));
send_t_alarm_min=true;}

    if (h_alarm_max && !send_h_alarm_max)
    {emailSEND(TextDHT("Осторожно! Превышена норма влажности", String(h)));
send_h_alarm_max=true;}

    if (h_alarm_min && !send_h_alarm_min)
    {emailSEND(TextDHT("Осторожно! Влажность ниже нормы", String(h)));
send_h_alarm_min=true;}
}
```

ПРИЛОЖЕНИЕ М

Скетч взаимодействия с сервером

```
EthernetServer server(80);

void loopServer(){
  EthernetClient client = server.available();

    if (client){ //если запрос оканчивается пустой строкой
     boolean currentLineIsBlank = true; //ставим метку об окончании запроса (дослов-
но: текущая линия чиста)
    while (client.connected()) { //пока есть соединение с клиентом
     if (client.available()) { //если клиент активен
        char c = client.read(); //считываем посылаемую информацию в переменную "c"

     if (newInfo && c == ' '){
          //если переменная новой информации = 1 и "c", в которой записан запрос,
равен пустой строке
     newInfo = 0; //то обнуляем переменную поступления новой информации
        }

     if (c == '$'){ // если переменная "c", несущая отправленный нам запрос, содер-
жит символ $
          // "$" подразумевает разделение получаемой информации (символов)
     newInfo = 1; //то пришла новая информация, ставим метку новой информации в 1
        }

        //Проверяем содержание URL - присутствует $1 или $2
        if (newInfo == 1){ //если есть новая информация
          Serial.println (c);
          if (c == '1'){ //и "c" содержит 1
            Serial.println (F("Включить"));
            digitalWrite (PIN_RELE_SVET, HIGH); //то зажигаем светодиод
          }

          if (c == '2'){ //если "c" содержит 2
            Serial.println (F("Выключить"));
            digitalWrite (PIN_RELE_SVET, LOW); //гасим светодиод
          }
        }

        //если "c" равен символу новой строки, то начинаем новую строку
        if (c == '\n') {currentLineIsBlank = true;}

        //иначе, если "c" не равен символу возврата курсора на начало строки, то по-
лучаем символ на текущей строке
        else if (c != '\r') {currentLineIsBlank = false;}

        //выводим HTML страницу
        if (c == '\n' && currentLineIsBlank) {
          client. println (F("HTTP/1.1 200 OK")); //заголовочная информация
          client. println (F("Content-Type: text/html"));
          client. println (F("Connection: close"));
          client. println (F("Refresh: 5")); //автоматическое обновление каждые 5
сек
          client. println (); //Это не ошибка, так должно быть
          client. println (F("<!DOCTYPE HTML>")); //HTML тип документа
```

```
          client. println (F("<html>")); //открытие тега HTML
          client. println (F("<head> ")); //открытие тега Head
          client. println (F("<meta http-equiv='Content-Type' content='text/html ;
charset=utf-8'/> "));
          client. print (F("<title> Microcompleks </title>")); //название страницы
          client. println (F("</head>")); //заголовочная информация
          client. println (F("<body>"));
          client. print (F("<H1>Микрокомплекс </H1>")); //заголовк на странице
          client. println (F(" <br>")); //перенос на след. строчку
          client. println (F("<hr/>")); //линия==========
          client. println (F("Температура = ")); //Температура с DHT 11
          client. println (t); //переменная температуры
          client. println (F(" *C"));

   if(t_alarm_max) {client.println(F("Осторожно! Превышена норма температуры"));}
   else if (t_alarm_min){client.println(F("Осторожно! Температура ниже нормы"));}

          client. println (F("<br> ")); //перенос на след. строчку
          client. println (F("Влажность = ")); //Влажность с DHT 11
          client. println (h); //переменная влажности
          client. println (F(" %\t"));

   if(h_alarm_max) {client.println(F("Осторожно! Превышена норма влажности"));}
   else if (h_alarm_min){client.println(F("Осторожно! Влажность ниже нормы"));}

          client. println (F("<br> ")); //перенос на след. строчку
          client. println (F("<hr/>")); //линия==================

   if(statePIR == HIGH){client.println(F("Свет в помещении включен"));}
   else if (statePIR == LOW){client.println(F("Свет в помещении выключен"));}

          client. println (F("<br> ")); //перенос на след. строчку
          client. println (F("<hr/>")); //линия==================

   if(!StateKeyRFID){client.println(F("Серверное помещение закрыто"));}
   else if (StateKeyRFID){client.println(F("Серверное помещение открыто"));}

          client. println (F("</body>"));
          client. println (F("</html>")); //закрываем тег HTML
          client. println (F("<br> ")); //перенос на след. строчку
          client. println (F("<hr/>")); //линия================
          break; //выход
        }
      }
    }
    delay (1); //время на получение новых данных
    client. stop(); //закрываем соеднение
  }
}
```

ПРИЛОЖЕНИЕ Н

Скетч работы ИК-передатчика

```
#include <IRremote.h>

IRsend irsend;

void loopIN(){

  int khz = 38; // задаем частоту 38 кГц
  unsigned int irSignal_on[] = {9000, 4500, 560, 560, 560, 560, 560, 1690, 560, 560,
560, 560, 560, 560, 560, 560, 560, 560, 560, 560, 560, 1690, 560, 1690, 560, 560, 560, 1690,
560, 1690, 560, 1690, 560, 1690, 560, 1690, 560, 560, 560, 560, 560, 560, 560, 1690,
560, 560, 560, 560, 560, 560, 560, 560, 560, 1690, 560, 1690, 560, 1690, 560, 560,
560, 1690, 560, 1690, 560, 1690, 560, 1690, 560, 39416, 9000, 2210, 560}; // код для
включения кондиционера на температуру 20 градусов и 40 влажности

  unsigned int irSignal_of[] = {8000, 4300, 500, 400, 450, 450, 560, 1690, 560, 560,
560, 560, 560, 460, 560, 560, 460, 560, 560, 1590, 560, 1590, 560, 560, 560, 1690,
560, 1690, 560, 1690, 560, 1690, 560, 1690, 560, 560, 560, 560, 560, 560, 560, 1690,
560, 560, 560, 560, 560, 560, 560, 560, 560, 1690, 560, 1690, 560, 1690, 560, 560,
560, 1690, 560, 1690, 560, 1690, 560, 1590, 460, 39416, 8000, 2210, 460}; // код для
выключения кондиционера

  if (t > t_max) {irsend.sendRaw(irSignal_on, sizeof(irSignal_on) /
sizeof(irSignal_on[0]), khz);} // Включаем кондиционер при повышенной влажности
  else {irsend.sendRaw(irSignal_off, sizeof(irSignal_off) / sizeof(irSignal_off[0]),
khz);}
  delay(10000);

  if (h > h_max) {irsend.sendRaw(irSignal_on, sizeof(irSignal_on) /
sizeof(irSignal_on[0]), khz);} // Включаем кондиционер при повышенной влажности
  else {irsend.sendRaw(irSignal_off, sizeof(irSignal_off) / sizeof(irSignal_off[0]),
khz);}
  delay(50000);

 }
```

ПРИЛОЖЕНИЕ О

Скетч объявления переменных

```
//  Устройства работающие по шине I2C подключаем к следующим контактам
  //  I2C SCL A5(UNO) / 21(Mega)
  //  I2C SDA A4(UNO) / 20(Mega)

// Устройства подключенные по SPI
  #define MFRC522_RST_PIN    5    // Пин подключения вывода RST модуля MFRC522
  #define MFRC522_SS_PIN     53   // Пин подключения SDA(SS) вывода модуля MFRC522

  // SPI W5100 SD MFRC522 MOSI  51
  // SPI W5100 SD MFRC522 MISO  50
  // SPI W5100 SD MFRC522 SCK   52

  #define SDCARD_CS 4     // CS пин SD карты

// Прочие устройства
#define DHTPIN            A0    // Вывод к которому подключен датчик DHT
#define PIN_RELE_SVET     7  // Реле Освещеня
#define PIN_RELE_LOCK     6  // Реле Замок
#define PIN_PIR           A1  // Вход Датчика PIR
#define PIN_BUTTON_MENU   3  // Кнопка переключения экранов

// База данных виртуальных пинов

  #define VPIN_LABEL_TIME   V0  // Виджет Lable Часов
  #define VPIN_LABEL_TEMP   V1  // Виджет Lable Температура
  #define VPIN_LABEL_HUMID  V2  // Виджет Lable Влажность
  #define VPIN_LOCK         V5  // Виджет График
  #define VPIN_TABLE        V10 // Виджет Таблица

// База данных глобальных переменных

// Данные времени
String Data;
String Time;
String DataTime;

boolean statePIR = 0;       //переменная для хранения состояния датчика движения
int8_t MenuPos = 0;         //переменные для меню
boolean newInfo = 0; //переменная для новой информации Управление с сайта

// Данные от датчика температуры и влажности
// Влажность
  float h;        // Переменная для хранения данных с датчика
  int h_min = 20; // Минимально допустимый уровень
  int h_max = 70; // Максимально допустимый уровень
  boolean  h_alarm_min=false; // Флаг аварии
  boolean  h_alarm_max=false; // Флаг аварии
  boolean  send_h_alarm_min=false; // Флаг для однократного(защита от спама) сообще-
ния
  boolean  send_h_alarm_max=false; // Флаг для однократного(защита от спама) сообще-
ния
// Температура
  float t;        // Переменная для хранения данных с датчика
```

```
  int t_min = 18; // Минимально допустимый уровень
  int t_max = 25; // Максимально допустимый уровень
  boolean  t_alarm_min=false; // Флаг аварии
  boolean  t_alarm_max=false; // Флаг аварии
  boolean  send_t_alarm_min=false; // Флаг для однократного(защита от спама) сообщения
  boolean  send_t_alarm_max=false; // Флаг для однократного(защита от спама) сообщения

// База ID таймеров
// ID таймеры для включения и выключение функции которая опрашивается по интервальному таймеру
  int ID_Timer_keyRFID;         // keyRFID() Чтение карт и ключей с RFID метками
  int ID_Timer_ScreenSwitching; // ScreenSwitching() Функция отвечающая за переключение экранов

// База данных Отправки Mail сообщений
  // Если закомментировать не будет отправки сообщений
  #define USE_MAIL_SEND

  /* Установите большее значение, чтобы разрешить отправку длинных сообщений */
  #define BLYNK_MAX_SENDBYTES 1000

  char MAIL[] = "arduino@gmail.com"; // Почта для получения критических сообщений
  char Subject[] = "Subject: Microcomplex";
  String Message = " "; // Переменная для хранения текста сообщения

  boolean stateSEND_Mail_Timer=false; // Флаг разрешающий отправку сообщения
  boolean waitSEND=false;             // Флаг ожидание отправки сообщения

// База данных Ключей

  boolean StateKeyRFID = false; // Открыт ли сейчас замок RFID меткой

  // Читаем UID ключей при помощи скетча DumpInfo_My
  // Количество ключей
  const int countKey=2;

  // Пишем UID ключей в HEX формате
  byte uidCard[countKey][4] = {
  {0x05, 0xB6, 0x97, 0xBB},
  {0x50, 0x27, 0x9E, 0x7C}
  };

  // Массив имен
  String NameUID[countKey] = {
    "Петя",
    "Вася"
  };
```

ПРИЛОЖЕНИЕ П

Акт о внедрении

АКТ
о принятии к внедрению результатов дипломной работы
студента направления 09.04.02 – «Информационные системы и технологии»
ИКИТ СФУ

Павленко Игорь Игоревич
(имя, фамилия, отчество студента)

Разработка информационной системы мониторинга параметров серверного помещения для ООО «Коммунальные информационные системы»
(тема выпускной квалификационной работы)

1. Наименование предложения, разработанного в рамках дипломного проектирования:

Разработана система мониторинга для возможности отслеживания параметров микроклимата помещения и оповещения ответственных лиц в случае нарушения климатических условий и несанкционированного доступа в серверное помещение

2. Наименование организации, где осуществлено внедрение результатов дипломной работы: «ООО Коммунальные информационные системы»

3. Эффект от внедрения:

В результате внедрения ожидается повышение срока эксплуатации серверного оборудования за счет стабилизации температуры в помещении, а также упрощение работы системного администратора за счет дистанционного контроля за помещением.

4. Форма внедрения:

Разработанный комплекс внедрен на предприятии полностью

Руководитель предприятия
(учреждения, отдела) _____ (подпись) Кувшинов А.А. (расшифровка подписи)

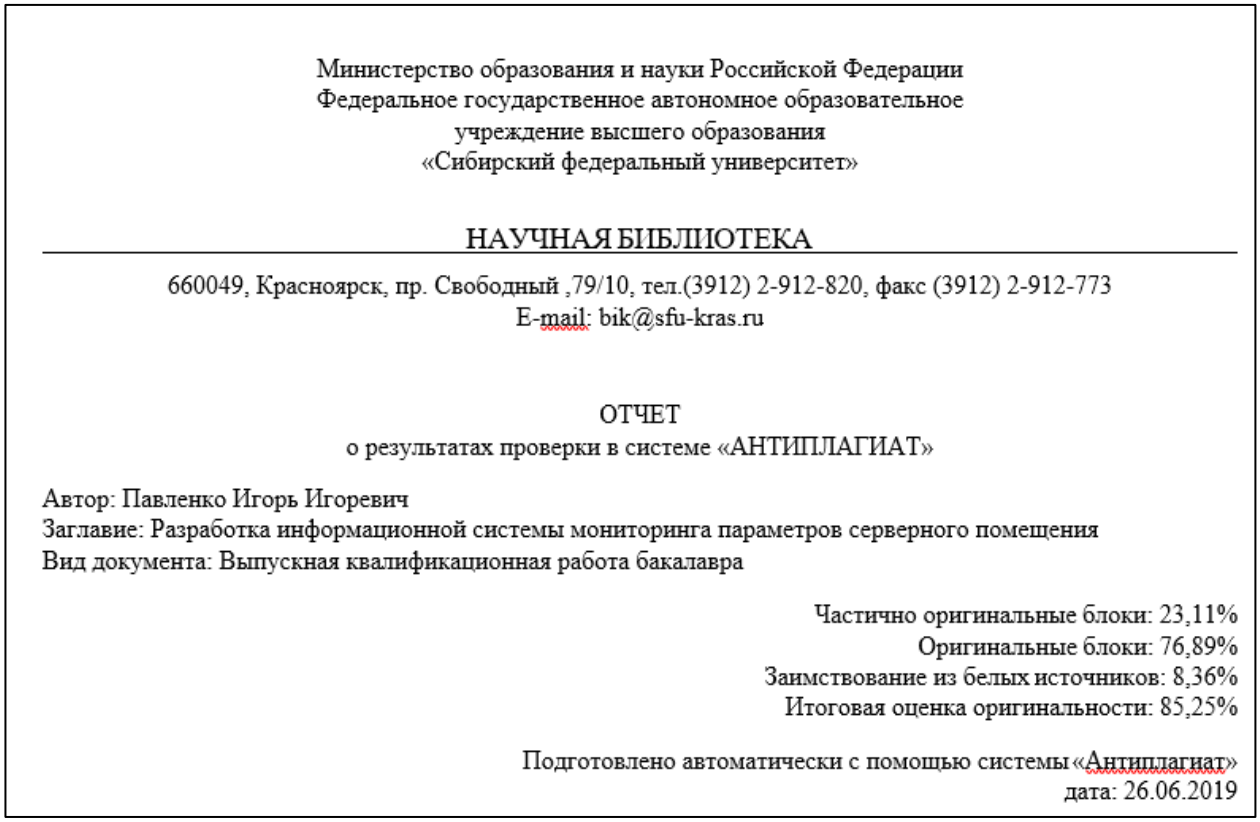
М.П. предприятия

«24» 10 2019 г.

Рисунок П.1 – Акт о внедрении

ПРИЛОЖЕНИЕ Р

Отчет «Антиплагиат»



Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Сибирский федеральный университет»

НАУЧНАЯ БИБЛИОТЕКА

660049, Красноярск, пр. Свободный ,79/10, тел.(3912) 2-912-820, факс (3912) 2-912-773
E-mail: bik@sfu-kras.ru

ОТЧЕТ
о результатах проверки в системе «АНТИПЛАГИАТ»

Автор: Павленко Игорь Игоревич
Заглавие: Разработка информационной системы мониторинга параметров серверного помещения
Вид документа: Выпускная квалификационная работа бакалавра

Частично оригинальные блоки: 23,11%
Оригинальные блоки: 76,89%
Заимствование из белых источников: 8,36%
Итоговая оценка оригинальности: 85,25%

Подготовлено автоматически с помощью системы «Антиплагиат»
дата: 26.06.2019

Рисунок Р.1 – Отчет о результатах проверки в системе «Антиплагиат»

ПРИЛОЖЕНИЕ С

Плакаты презентации

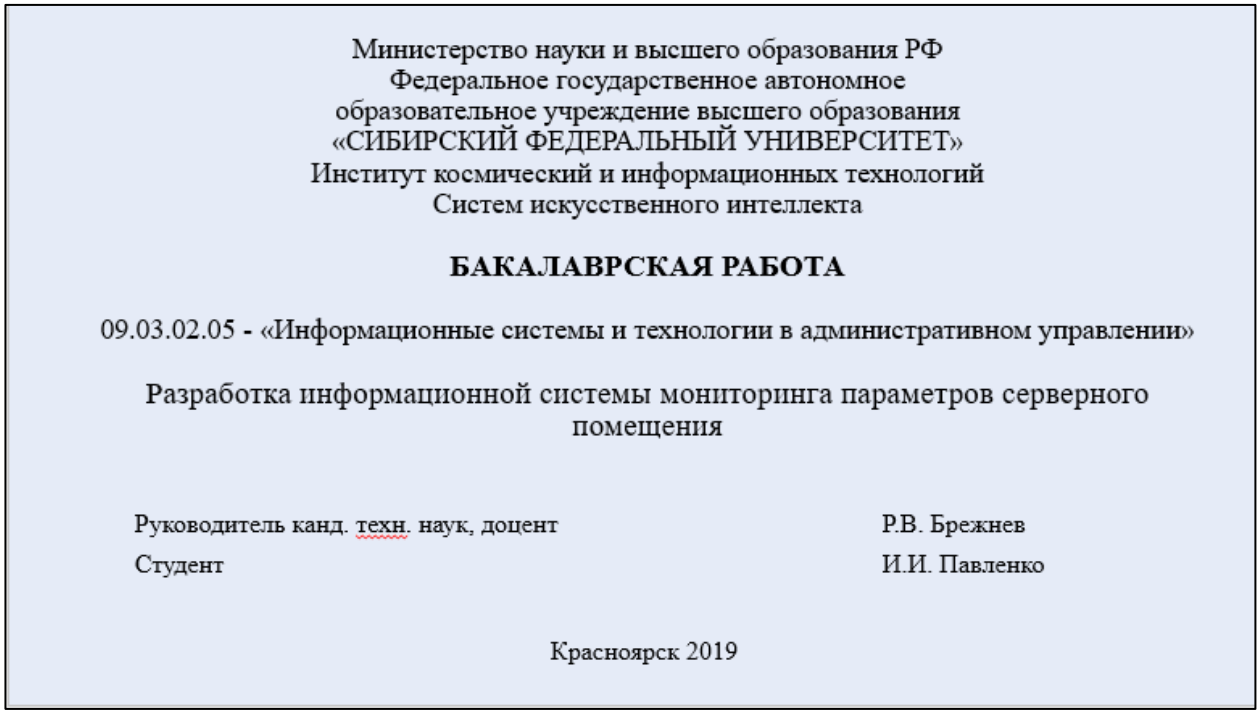


Рисунок С.1 – Плакат презентации № 1

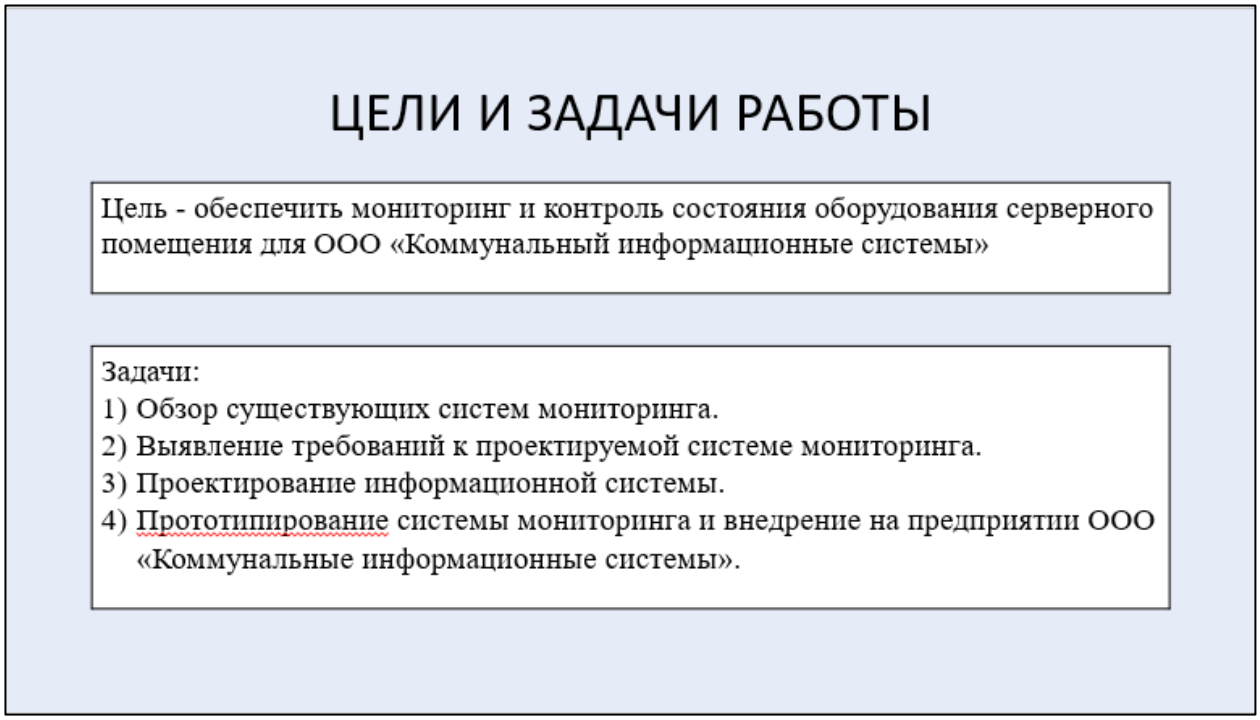


Рисунок С.2 – Плакат презентации № 2

СИСТЕМА МОНИТОРИНГА

Наиболее типовыми задачами мониторинга серверных помещений являются:

- мониторинг микроклимата в помещении;
- мониторинг доступа в помещение;
- мониторинг наличия электропитания;
- оповещение персонала в случае необходимости.

Рисунок С.3 – Плакат презентации № 3

ОБЗОР СУЩЕСТВУЮЩИХ СИСТЕМ МОНИТОРИНГА

UniPing server solution

Кситал GSM

Рисунок С.4 – Плакат презентации № 4

Рисунок С.5 – Плакат презентации № 5

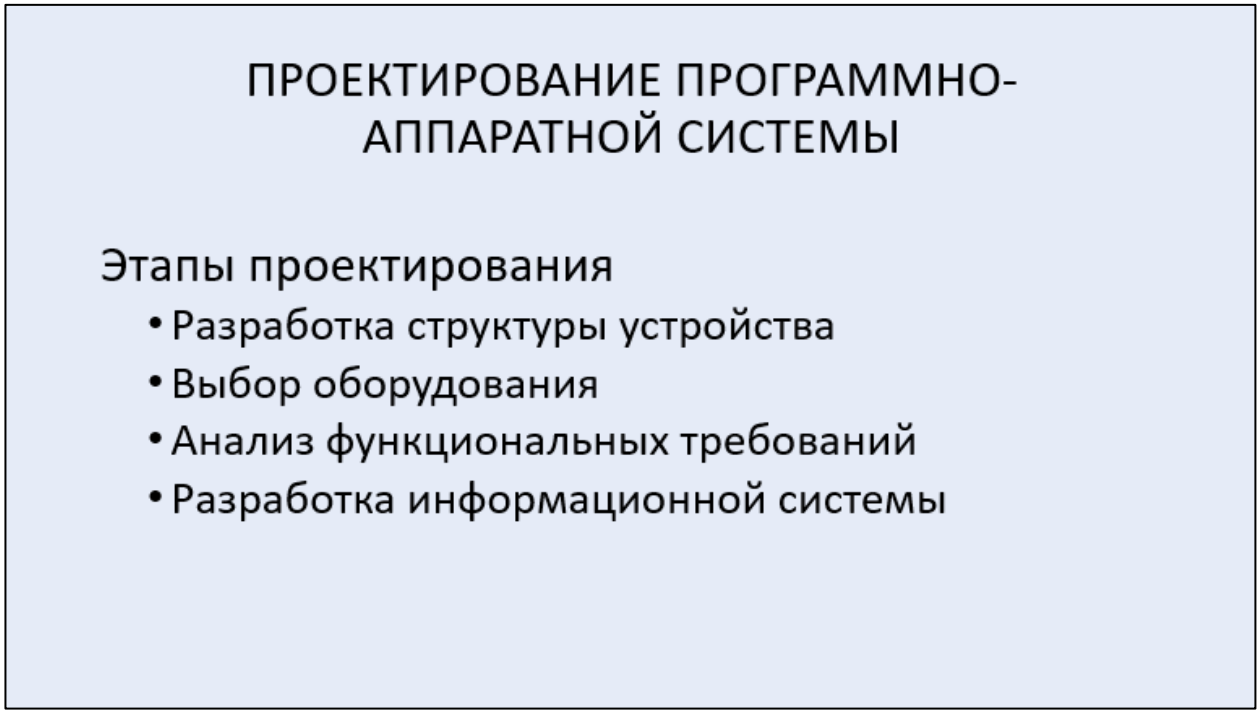


Рисунок С.6 – Плакат презентации № 6

Рисунок С.7 – Плакат презентации № 7

Рисунок С.8 – Плакат презентации № 8

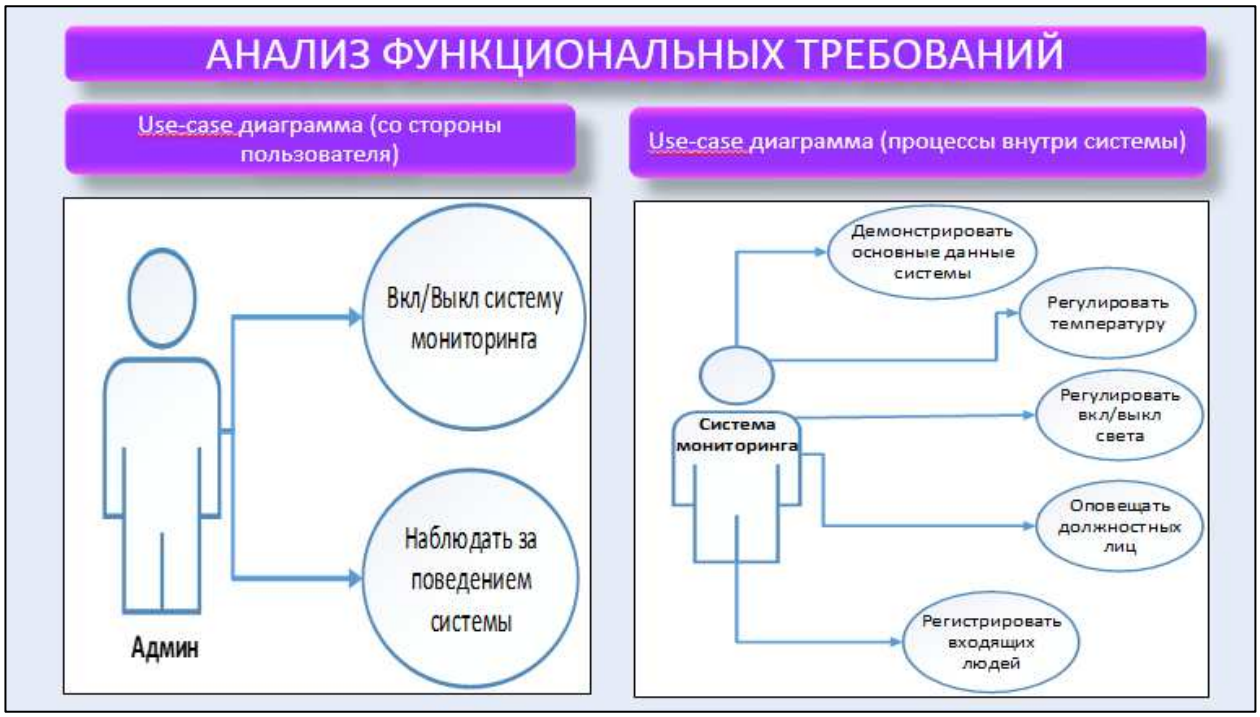


Рисунок С.9 – Плакат презентации № 9

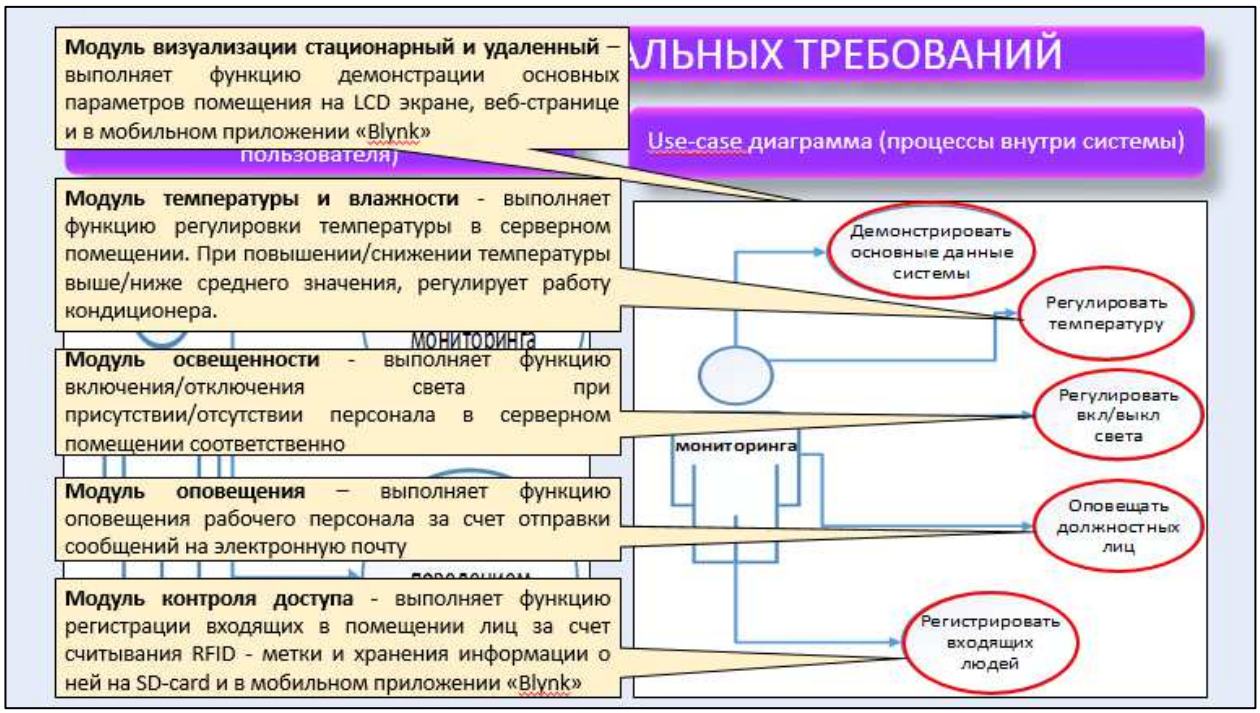


Рисунок С.10 – Плакат презентации № 10

РАЗРАБОТКА ИС

ТРЕБОВАНИЯ К СТРУКТУРЕ И ФУНКЦИОНИРОВАНИЮ ИС

Основные требования к разрабатываемой ИС изложены в техническом задании (Приложение А).

В разрабатываемой ИС предлагается выделить следующие функциональные модули:

- Модуль температуры и влажности
- Модуль освещенности
- Модуль контроля доступа
- Модуль Визуализации стационарный
- Модуль Визуализации удаленный

Рисунок С.11 – Плакат презентации № 11

СХЕМА ПОДКЛЮЧЕНИЯ ДАТЧИКОВ И МОДУЛЕЙ

1) Arduino Mega2560 + Ethernet Shield
2) Инфракрасный датчик движения
3) Считыватель RFID MFRC-522
4) Кнопка тактовая
5) LCD дисплей
6) Модуль с двумя реле
7) Прецезионные часы реального времени
8) Цифровой датчик температуры и влажности
9) Резистор
10) Лампочка накаливания
11) Электромеханический замок
12) Micro SD-card
13) ИК – передатчик

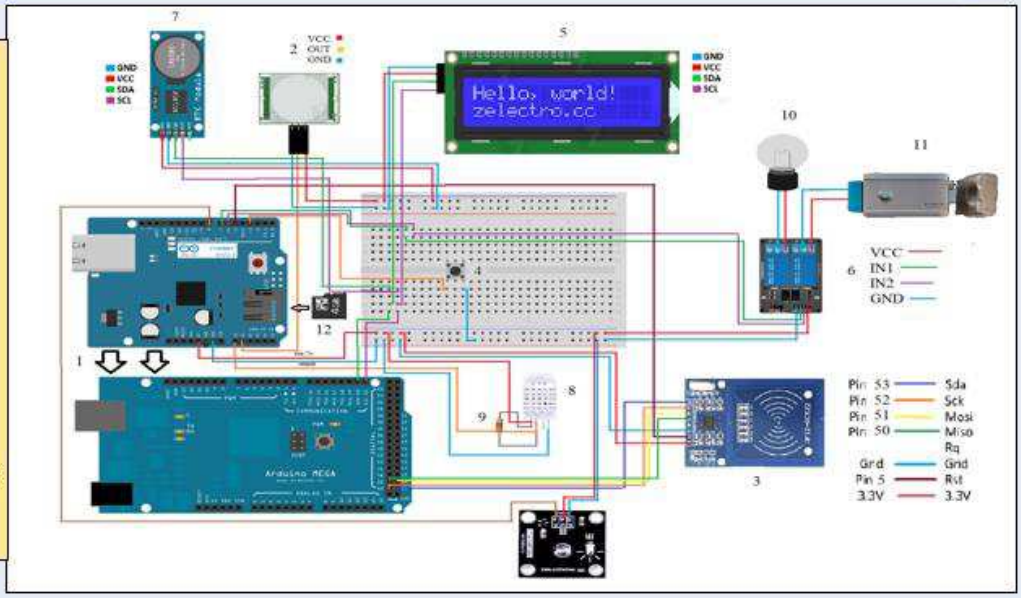


Рисунок С.12 – Плакат презентации № 12

Рисунок С.13 – Плакат презентации № 13

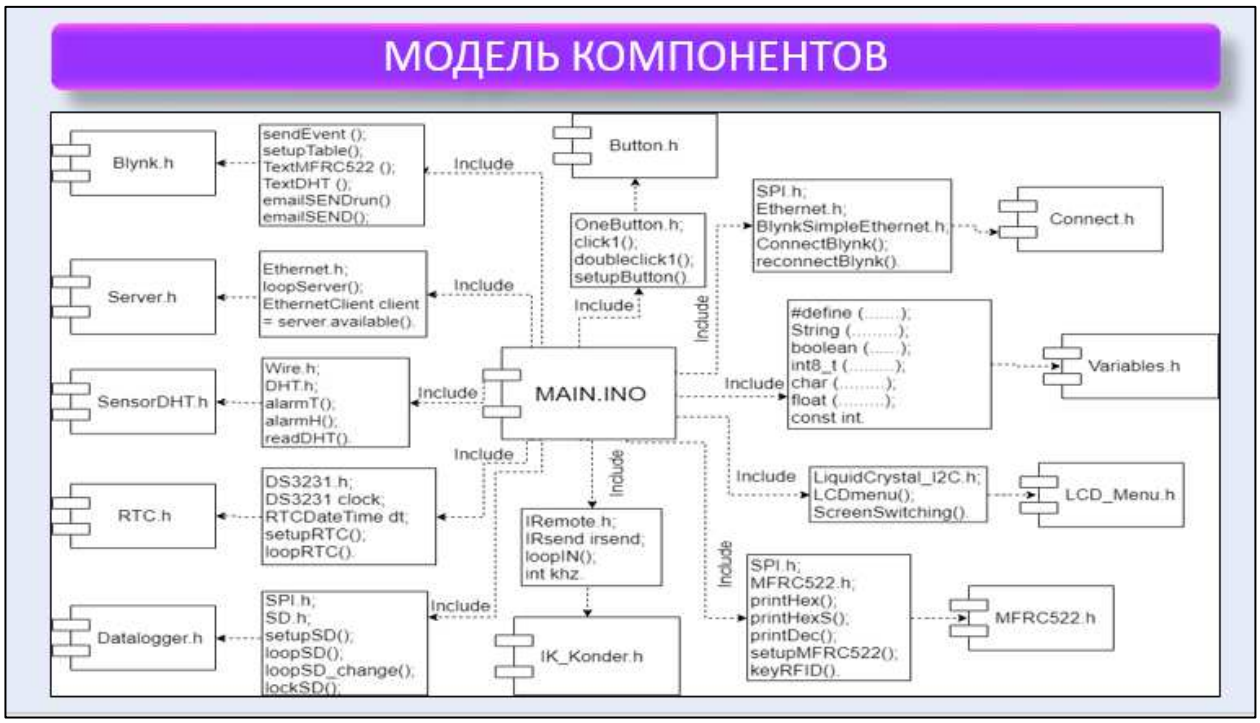


Рисунок С.14 – Плакат презентации № 14

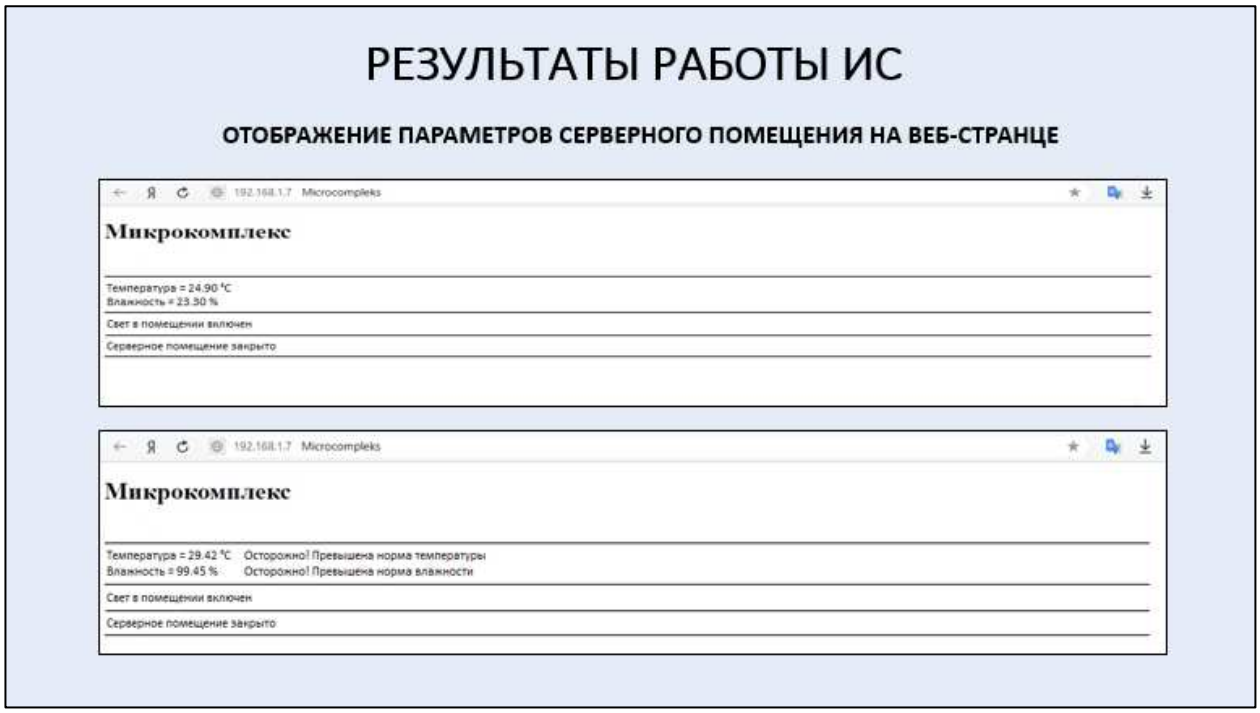


Рисунок С.15 – Плакат презентации № 15

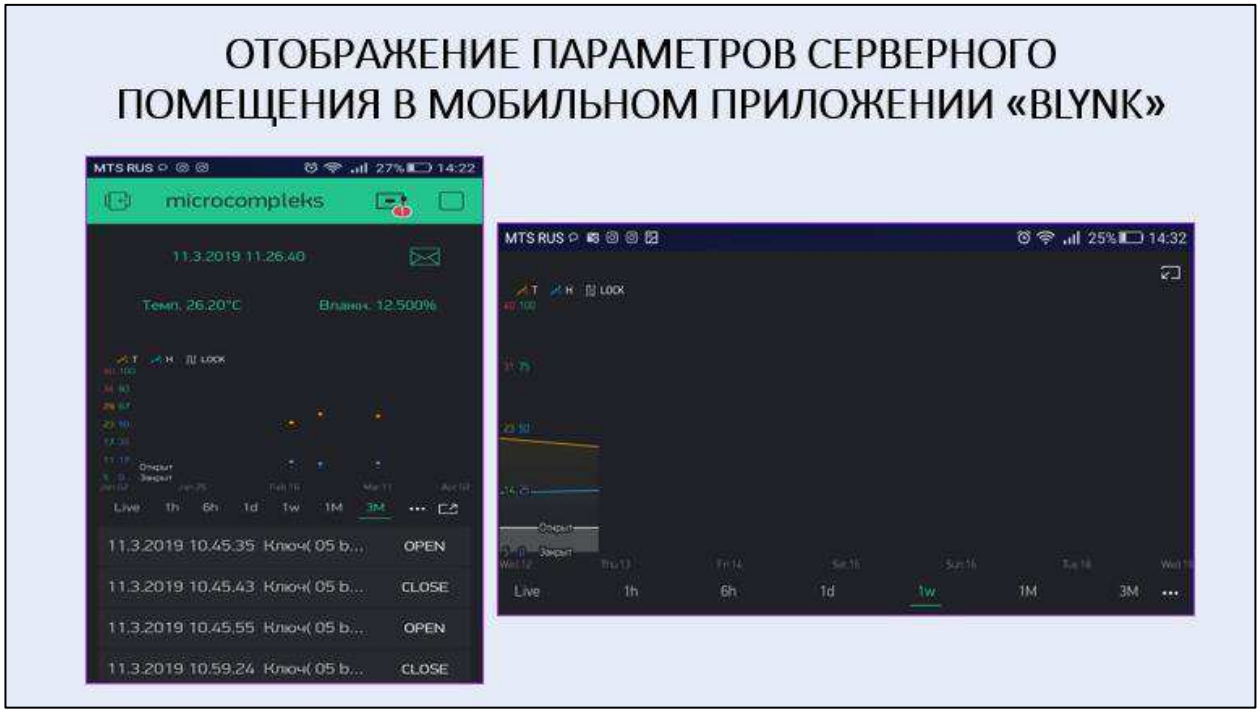


Рисунок С.16 – Плакат презентации № 16

ЗАКЛЮЧЕНИЕ

1. Выполнен **обзор и анализ** существующих систем мониторинга:
   1. Выявлены и проанализированы **требования к ИС**;
   2. Разработано **техническое задание**.
2. В ходе **проектирования** разрабатываемой ИС:
   1. Разработана **структура устройства;**
   2. Выбрано оборудования для разработки;
   3. Построены **модель состояний программно-аппаратного комплекса, модель компонентов** и **архитектура комплекса;**
   4. Разработана полноценная **система мониторинга,** соответствующая выявленным требованиям.

Рисунок С.17 – Плакат презентации № 17

АКТ О ВНЕДРЕНИИ

АКТ

о принятии к внедрению результатов дипломной работы
студента направления 09.04.02 – «Информационные системы и технологии»
ИКИТ СФУ

Павленко Игорь Игоревич
(имя, фамилия, отчество студента)

Разработка информационной системы мониторинга параметров серверного помещения для ООО «Коммунальные информационные системы»
(тема выпускной квалификационной работы)

1. Наименование предложения, разработанного в рамках дипломного проектирования:

Разработана система мониторинга для возможности отслеживания параметров микроклимата помещения и оповещения ответственных лиц в случае нарушения климатических условий и несанкционированного доступа в серверное помещение

2. Наименование организации, где осуществлено внедрение результатов дипломной работы: «ООО Коммунальные информационные системы»

3. Эффект от внедрения:

В результате внедрения ожидается повышение срока эксплуатации серверного оборудования за счет стабилизации температуры в помещении, а также упрощение работы системного администратора за счет дистанционного контроля за помещением.

4. Форма внедрения:

Разработанный комплекс внедрен на предприятии полностью

Руководитель предприятия
(должность, отдела)
М.П. предприятия

(подпись) (расшифровка подписи)

20 г.

Рисунок С.18 – Плакат презентации № 18

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Г.М. Цибульский
подпись
«____»_____2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 — Информационные системы и технологии

Разработка информационной системы мониторинга параметров серверного помещения

Руководитель _____ доцент, канд. техн. наук Р.В. Брежнев
подпись, дата

Выпускник _____ И.И. Павленко
подпись, дата

Красноярск 2019