МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра дискретной математики и информационных технологий

РЕАЛИЗАЦИЯ СИСТЕМЫ МОНИТОРИНГА ТЕХНОЛОГИЧЕСКОГО ПРОЦЕССА С ИСПОЛЬЗОВАНИЕМ ИНТЕРНЕТА УМНЫХ ВЕЩЕЙ

БАКАЛАВРСКАЯ РАБОТА

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Лаптева Юрия Владиславовича

Научный руководитель
доцент, к. ф.-м. н. _____ А. Д. Панфёров

Заведующий кафедрой
к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

За последние годы технологии продвинулись далеко вперед. Были открыты новые направления как в промышленности, так и в ИТ сфере. Одной из ключевых концепций, позволивших расширить возможности технологических процессов является Интернет вещей.

Интернет вещей (IoT) - концепция, объединения различных устройств посредством как беспроводных технологий, так и через интернет. Устройства могут обмениваться информацией как при участии человека, так и без какого-либо вмешательства. Развитие концепции Интернета вещей произошло благодаря широкому распространению беспроводных сетей, смартфонов, ноутбуков.

Применение устройств из категории Интернета вещей так же получило распространение и в производстве. Были изобретены системы удаленного мониторинга с различными возможностями, требуемыми в конкретной отрасли.

Устройства из категории Интернет вещей можно найти и в ювелирном деле. Так появились устройства, способные точно и быстро вырезать необходимые заготовки из ювелирного воска по загруженным удаленно с ноутбука 3D моделям, лазерные установки, способные как выжигать сложные элементы на металле, так и прожигать его. Так же существуют контролеры для работы с муфельными печами. Данные устройства позволяют отслеживать и корректировать температуру внутри изолированного короба.

Муфельные печи появились в конце XVII - начале XVIII вв. и представляли собой контейнер (муфель), устойчивый к высоким температурам и имеющий непроницаемую для отвода тепла структуру.С увеличением масштабов производства необходимость таких конструкций значительно возросло.

В современном мире муфельные печи получили системы контроля температуры и времени работы. На рынке представлено широкий ассортимент систем мониторинга таких как: механический контролеры температуры печи, автоматические регуляторы муфельной печи.

На данный момент не существует гибко настраиваемых систем мониторинга для муфельных печей с возможностью дистанционного управления через WEB интерфейс.

Целью данной выпускной квалификационной работы является разработка системы мониторинга технологического процесса со следующими функци-

оналом: поддержание определенной температуры в течении установленного периода времени(так называемых "температурных полок"), сбор, хранение и показ статистики о температуре и времени ее поддержания при работе устройства, удаленное создание и редактирование температурных полок.

Для реализации данной цели было необходимо решить следующие задачи:

- изучить общие положения концепции Интернета вещей;
- изучить представленные на рынке системы мониторинга для муфельных печей;
- разработать конструкцию муфельной печи, способную сохранять целостность при высоких температурах;
- изучить электронные компоненты, способные работать длительное время под нагрузками;
- продумать и разработать клиентскую и серверную часть приложения;

1 Общие сведения о предметной области

В данном разделе будут рассмотрены общие концепция Интернета умных вещей, история появления, развитие данной концепции. Также будут рассмотрены различные конфигурации систем мониторинга для муфельных печей и их основные различия.

1.1 Интернет умных вещей

Интернет вещей (IoT) - концепция, объединения различных устройств посредством как беспроводных технологий, так и через интернет. Устройства могут обмениваться информацией как при участии человека, так и без какого-либо вмешательства. Развитие концепции Интернета вещей произошло в 2008 - 2009 годах благодаря широкому распространению беспроводных сетей, смартфонов, ноутбуков.

Термин «Интернет вещей» появился в 1999 году когда сотрудник Procter n Gamble Кевин Эштон предложил использовать радиочастотные метки для оптимизации логистики корпорации. Для изучения данной концепции в Массачусетском технологическом институте был создан Центр автоматической идентификации.

Одним из главных направлений в развитии Интернета вещей можно считать Промышленный Интернет вещей. Это система подключенных компьютерных сетей и подключенных к ним производственных объектов с различными встроенными датчиками и специализированным программным обеспечением, позволяющим не только собирать и анализировать данные, но и удаленно управлять устройствами. Также в таких системах может использоваться полностью автоматическое управление без участия человека.

1.1.1 Как устроен Промышленный Интернет вещей

При внедрении концепции Промышленного Интернета вещей на оборудование устанавливаются различные датчики, управляемые электронно механизмы, контроллеры и различные интерфейсы для взаимодействия человека с машинами. В результате появляется возможность собирать и анализировать получаемые данные. Обработанные данные поступают на сервера, откуда доступны для просматривания и анализа уже людьми.

Такой сбор данных помогает предотвращать внеплановые простои, поломки оборудования, сбои в управлении, тем самым позволяя предприятию

функционировать более эффективно.

Уже сейчас появляются различные цифровые экосистемы, которые позволяют предприятиям объединять различные участки рынка, что в свою очередь позволяет комплексно подходить к анализу и грамотнее распределять производственные ресурсы.

1.2 Системы мониторинга для муфельных печей и их применение

В данном разделе будут рассмотрены основные конфигурации систем мониторинга для муфельных печей. их ключевые особенности, их плюсы и минусы, и их применение.

Базовой функцией для каждой системы мониторинга для муфельных печей можно считать поддержание определенной температуры в течении заданного времени. Все описанные ниже системы способны удерживать так называемые температурные полки, но отличаются количеством их установок.

На рынке существует несколько видов систем мониторинга для муфельных печей:

- механический контроллер температуры печи;
- автоматический регулятор муфельной печи;
- ПИД-контроллер-программатор для муфельной печи;

1.2.1 Механический контроллер температуры печи

Данный класс системы мониторинга предназначен для поддерживания заданной температуры на протяжении всего времени, выставленного на таймере. Механический контроллер представляет собой термопару, подключенную к простому контроллеру, зачастую с возможностью выставления лишь одной поддерживаемой температуры на выставленное время. К плюсам данного класса устройств можно отнести:

- простота в использовании;
- долговечность;

К минусам таких устройств можно отнести:

- обязательное присутствие человека;
- зачастую невозможность установки больше чем одной температурной полки;
- невозможность передачи информации на удаленные устройства;

На рисунке 1 представлен общий вид класса данных устройств. Данный вид

Рисунок 1 – Общий вид механических контроллеров для муфельной печи

контроллеров невозможно отнести к Интернету вещей, так как процесс не автоматизирован и требует постоянного участия человека.

1.2.2 Автоматический регулятор муфельной печи

Данный класс системы мониторинга является улучшенной версией предыдущего класса устройств. Автоматический регулятор муфельной печи имеет уже микроконтроллер для более точного измерения температуры. Также данная система имеет возможность выставления нескольких необходимых температурных значений. К плюсам данного класса устройств можно отнести:

- возможность выставления нескольких необходимых темпертурных значений;
- проведение высокоточных температурных измерений по всей площади камеры;
- отсутствие необходимости нахождения рядом человека;

К минусам таких устройств можно отнести:

- невозможность составления прогнозов и недопущение повторения ошибок;
- невозможность передачи информации на удаленные устройства;

На рисунке 2 представлен общий вид класса данных устройств. Данный класс систем мониторинга можно частично отнести к Интернету вещей, так как процесс автоматизирован и не требует вмешательства человека, но для реализации поставленной в рамках выпускной квалификационной работы бакалавра уели не подходит, так как не имеет возможности интеграции с удаленными серверами.

Рисунок 2 – Общий вид автоматических регуляторов муфельной печи

1.2.3 ПИД-контроллер-программатор для муфельной печи

Данный класс систем мониторинга является самым востребованным на сегодняшний день. Работа данных устройств строится отслеживании состояния муфельной печи и образование немедленных соответствующих сигналов на управляющие устройства. Зачастую, данные устройства имеют на борту 2 дисплея, показывающие как установленную температуру, так и текущую. К плюсам данного класса устройств можно отнести:

- немедленное реагирование на текущий процесс с возможностью устранения неточностей;
- оценка отклонения, происходящих на всем периоде работы;
- отсутствие необходимости нахождения рядом человека;
- наличие инструментов для удаленной работы с устройствами;

К минусам таких устройств можно отнести:

- программирование таких устройств достаточно трудоемкое занятие, так как нужно обладать как специальными знаниями, так и специализированными инструментами

На рисунке 3 представлен общий вид класса данных устройств. Данный класс систем полностью соответствует концепции Промышленного Интернета вещей.

Для реализации цели выпускной квалификационной работы было принято решение использовать одноплатный микрокомпьютер, подходящий для реализации концепции данного типа устройств.

Рисунок 3 – Общий вид ПИД-контроллеров-программаторов для муфельной печи

2 Сведения о конфигурации системы мониторинга технологического процесса

В данном разделе будут рассмотрены основные компоненты, использованные в рамках выпускной квалификационной работы, и общая схема подключения.

2.1 Однопалатный компьютер PrangePi One

Orange Pi One — одноплатный микрокомпьютер с четырех-ядерным процессором ARM Cortex-A7 H3. Имеет четырех-ядерный процессор Cortex-A7 с частотой 1.2ГГц, видео ядро Mali 400MP2 (600 МГц). В Orange Pi One доступно 512 МБ DDR3 ОЗУ. С каждой стороны платы находится по чипу производства Samsung объемом по 256 МБ. Данный одноплатный микрокомпьютер имеет слот карты памяти, поддерживаемый до 64 Гб, используемый в качестве основной памяти.Есть порт Ethernet со скоростью до 100 Мб/с. Главной особенностью одноплатных компьютеров является наличие 40 пинов, которыми можно програмно управлять, что и было использовано в дальнейшем.

Orange Pi One поддерживает широкий набор операционных систем: Android, Ubuntu, Armbian, ArchLinux, Gentoo, OpenSUSE, Kali, Fedora, Rasberry Pi Image. Мною была выбрана операционная система Armbian, так как данная ОС отличается быстротой работы и возможностью взаимодействия с GPIO пинами.

Данный элемент выполняет функцию гибко настраиваемого управляющего контроллера, подключенного посредством Ethernet к WI-FI роутеру для взаимодействия с микрокомпьютером из любой точки мира. Он подает сигналы, включающие и выключающие твердотельное реле SSR-25DA, сбор информации с термопары К-типа и модуля преобразователя термопары MAX6675, для вывода информации о текущей температуре внутри муфельной печи и заданной в WEB приложении требуемой температуры. На микрокомпьютере находится вся серверная и клиентская части приложения.

На официальном сайте микрокомпьютеров Orange Pi есть вся необходимая информация по запуску и настройке устройства. Для полноценной работы с микрокомпьютером необходимы следующие элементы:

— TransFlash-карта (далее TF-карта) класса 10 для записи на нее образа операционной системы, устанавливаемой на микрокомпьютер. Класс по-

казывает, как быстро работает карта. 10 класс является самым быстрым у TF-карт, он подходит даже для для записи Full HD в формате RAW, просмотра фильмов и прохождения игр на планшете или ноутбуке. Минимальный требуемый размер TF-карты – 8 гигабайтов;

- Ethernet кабель для подключения одноплатного микрокомпьютера к WI-FI роутеру;
- лабораторный блок питания, выставленный на 5V и 2A для питания Orange PI;

На рисунке 4 представлен одноплатный компьютер Orange Pi One. В

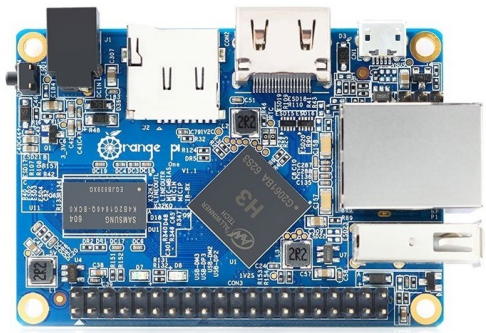

Рисунок 4 – Одноплатный компьютер Orange Pi One

выпускной квалификационной работе мною были использованы следующие GPTO выходы для следующих электронных компонентов:

- для твердотельного реле SSR-25DA были задействованы пины 39 и 40;
- для термопары К-типа и модуля преобразователя термопары MAX6675 были задействованы пины 17, 20, 21, 22,23;
- для дисплея SPI 1602 LCD были использованы 2, 3, 5, 6;

В дальнейшем будут рассмотрено функциональное назначение каждого из использованных GPIO выходов.

На рисунке 5 можно посмотреть GPIO выходы одноплатного микрокомпьютера Orange Pi One.

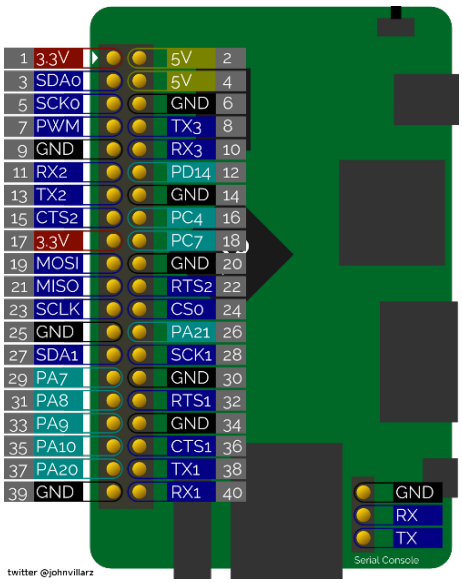

Рисунок 5 – Одноплатный компьютер Orange Pi One

2.2 Твердотельное реле SSR-25DA

Твердотельное реле SSR-25DA - полупроводниковое реле, предназначено для бесконтактной коммутации осветительных приборов, нагревательных элементов и других устройств с переменным напряжением питания от 24 до 480 В. Данный прибор не требует пайки, так как имеет винтовые разъемы, для подключения проводов. Твердотельные реле имеет много преимуществ перед электромеханическими реле: низкий уровень помех за счет встроенного детектора нуля, возможность управлять напрямую с выводов контроллера, гораздо больший эксплуатационный ресурс, высокое быстродействие, отсутствие механической контактной группы, бесшумность, небольшие размеры, гальваническая развязка от управляющей цепи и высокая надежность. Из удобств стоит отметить возможность подключения без паяльника, так как все разъемы выполнены под винтовой зажим. Реле SSR можно без опасения подключать напрямую к выводам микроконтроллеров и одноплатных компьютеров, так как в этом электронном реле есть гальваническая развязка с силовым блоком.

Данное реле было использовано в выпускной квалификационной работе для взаимодействия с нагревательным элементом внутри муфельной печи. Оно выполняет функцию отключения и включения тока внутри цепи.

На рисунке 6 изображено твердотельное реле SSR-25DA.

Рисунок 6 – Твердотельное реле SSR-25DA

2.3 Термопара К-типа и модуль аналого-цифрового преобразования MAX6675

Термопара К-типа представляет собой соединение из хромеля (сплава хрома и никеля с примесями кремния, меди, марганца, кобальта) и алюминия. Принцип действия термопары основан на том, что нагревание или охлаждение контактов между проводниками, отличающимися химическими или физическими свойствами, сопровождается возникновением термоэлектродвижущей силы. Данный вид термопар предназначен для замера температур до 1000 градусов. Выходным сигналом сенсора К — типа служит постоянное напряжение, пропорционально зависящее от температуры в точке состыковки контактов термопары. Так как выходные сигналы термопары аналоговые, было принято решение использовать аналого-цифровой преобразователь MAX6675 с копменсацией холодного спая.

На модуле MAX6675 установлена одна микросхема MAX6675ISA фирмы Maxim Integrated Products конденсатор и пару разъемов. Микросхема имеет 12 битный АЦП, SPI интерфейс и точность микросхема 0,25 градусов.

Для подключения модуля MAX6675 к микроконтроллеру используется шина SPI, для этого на модуле выведены пять контактнов, назначение каждого вывода приведено ниже.

— GND - минусовое питание модуля;
— VCC - плюсовое питание модуля;
— SCK - тактовые импульсы;
— SC - вывод интерфейса SPI;
— SO - вывод интерфейса SPI;

На рисунке 7 представлена термопара К-типа и модуль аналого-цифрового

преобразования MAX6675.

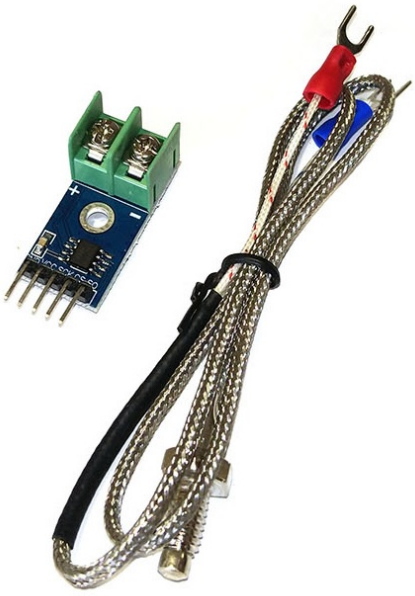

Рисунок 7 – Термопара К-типа и модуль аналого-цифрового преобразования MAX6675

2.4 Дисплей I2C 1602 LCD

Дисплей I2C 1602 LCD - это жидкокристаллический, текстовый, двух строчный, 16 знакомест в каждой строке, цифровой I2C индикатор с подсветкой. Каждое знакоместо имеет разрешение 8 х 5 точек. Общее количество точек экрана 1280 пикселей. В память устройства встроено 192 знака, еще 8 знаков может определить сам пользователь. Дисплей основан на контроллере HD44780 и предназначен для отображения любой текстовой информации. Благодаря дополнительно установленному I2C модулю расширения портов на микросхеме PCF85741, диплей стал занимать гораздо меньше портов управления GPIO на одноплатном микрокомпьютере Orange Pi One.

Для подключения модуля I2C 1602 LCD к микроконтроллеру используется шина I2C, для этого на модуле выведены четыре контактна, назначение каждого вывода приведено ниже.

— GND - минусовое питание модуля;
— VCC - плюсовое питание модуля;
— SCL - тактовые импульсы;
— SDA - вывод интерфейса I2C;

2.5 Общая схема подключения

Для корректной работы свей системы в целом и отдельных ее элементов необходимо правильно подключить все компоненты из которых состоит система мониторинга технологического процесса.

На рисунку 8 показана общая схема подключения всех электрических модулей.

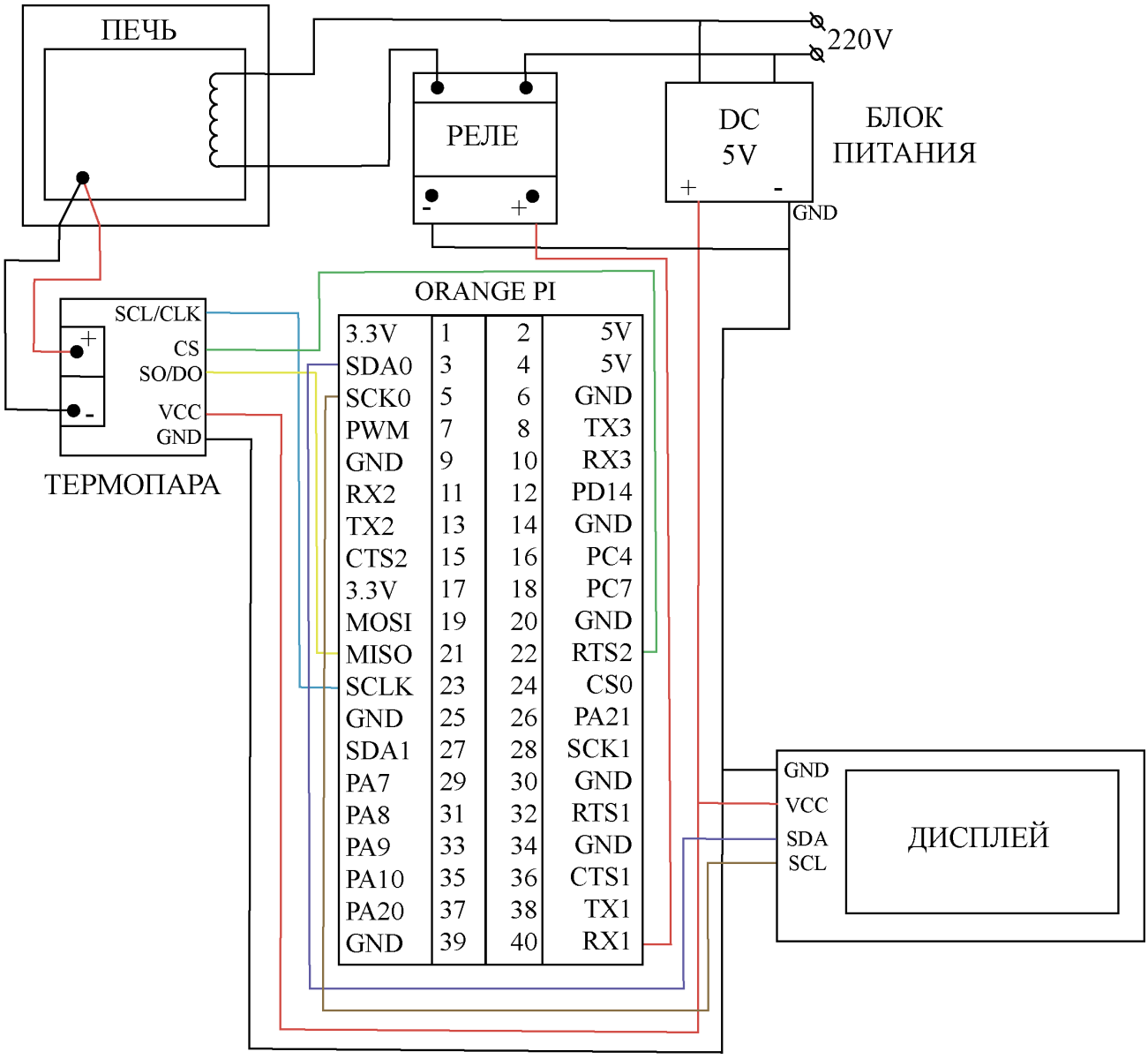

Рисунок 8 – Общая схема подключения электрических компонентов

Подключение всех электрических компонентов производилось при помощи соединительных проводов.

3 Теоретические сведения об используемых в разработке технологиях

В реализованной версии системы мониторинга технологического процесса клиентской частью приложения является WEB интерфейс, написанный на языках HTML и JavaScript. Для серверной части приложения был использован язык Python. Обе части приложения находятся на одноплатном микрокомпьютере Orange Pi One. Клиентская часть позволяет как наблюдать за процессами, происходящими внутри муфельной печи на расстоянии, так и настраивать температурные режимы. Серверная часть отвечает за обработку запросов, происходящих на клиентской части приложения, их обработку и своевременное взаимодействие с электрическими составляющими системы.

В данном разделе описываются технологии, примененные в рамках выпускной квалификационной работы бакалавра.

3.1 Клиент-серверная архитектура

Архитектура разработанного приложения представляет собой WEB-приложение

WEB-приложение – это клиент-серверное приложение, в котором браузер выступает клиентом, а сервером – WEB-сервер. Основная часть приложения, находится на стороне WEB-сервера, который обрабатывает полученные запросы в соответствии с бизнес-логикой продукта и формирует ответ, отправляемый пользователю. На этом этапе в работу включается браузер, именно он преобразовывает полученный ответ от сервера в графический интерфейс, понятный пользователю.

3.2 Серверная часть приложения

WEB-сервер был написан на языке программирования Python, так как операционная система Armbian отлично подходит для работы с данным языком и позволяет взаимодействовать с GPIO выходами. Также на сервере в формате JSON хранятся записи о температурных режимах.

3.2.1 Python

Python - высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью. Данный язык ориентирован на повышение читаемости кода и качества его написания.Python является объектно ориентированным языком.

Python применяется в различных сферах. Данный язык хорошо подходит как для маленьких разработок, так и для крупных проектов. Данная особенность обусловлена тем, что:

- язык гибкий и просто масштабируемый. Данный язык позволет разработчикам адаптировать высокоуровневую логику приложения, что позволяет легко расширять сложные приложения по мере необходимости;
- язык имеет большое количество готовых библиотек для решения конкретных типов задач. Ниже представлены библиотеки, которые были использованы для в выпускной квалификационной работе:
  - spidev - библиотека для работы с внрешними устройствами по протоколу SPI;
  - smbus - библиотека для работы с внешними устройствами по протоколу I2C;
  - pyA20 - библиотека для взаимодействия с GPIO выходами одноплатных микрокомпьютеров;
  - Flask - библиотека для разработки серверной части приложений;
- поддержка модульности;
- кроссплатформенность;

Практически каждая организация, занимающаяся разработкой программного обеспечения, так или иначе использует Python как для решения долговременных, стратегических задач проектирования, так и для решения краткосрочных тактических задач, таких как тестирование и системное администрирование. Такие известные компании, как Google и Intel, Cisco и Hewlett-Packard, используют язык Python, выбрав его за гибкость, простоту использования и обеспечиваемую им высокую скорость разработки. Он позволяет создавать эффективные и надежные проекты, которые легко интегрируются с программами и инструментами, написанными на других языках.

На Python были написаны скрипты для работы с одноплатным компьютером Orange Pi One для приема и обработки информации, поступающей с клиентской части приложения, а также для работы с электрическими частями устройств.

Причинами выбора данного языка программирования можно считать:

- операционная система изначально поддерживала работу с данным языком программирования;

— для Python существует множество библиотек и фреймворков, позволяющих упростить выполнение поставленных задач;

3.2.2 Протокол SPI

SPI (Serial Perpheral Interface) - последовательный переферийный протокол обмена данными. Этот протокол был разработан компанией Motorola и в настоящие время используется многими производителями. Он предназначен для связи микроконтроллеров между собой, микроконтроллеров и переферийных устройств.

Для передачи данных в SPI используются три линии:

— MISO - по этой линии ведущий (Master) принимает данные от ведомого (Slave);
— MOSI - по этой линии ведущий отправляет данные ведомому;
— SCK - Эта линия служит для передачи тактового сигнала ведомому устройству;

На рисунке 9 показано, как ведущее устройство взаимодействует с ведомым.

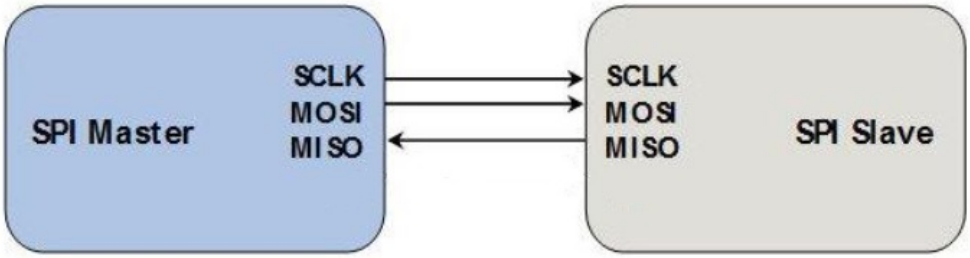

Рисунок 9 – Взаимодействие ведущего и ведомого устройства

В данной выпускной квалификационной работе бакалавра протокол SPI был использован для взаимодействия с аналого-цифровым преобразователем MAX6675, для считывания температуры с термопары. Использование протокола SPI обусловлено технологическими особенностями данного устройства.

3.2.3 Протокол I2C

I2C (Inter-Integrate Circuit) - последовательная шина данных для связи различных интегральных схем. Данный протокол использует двунаправленные линии связи SDA и SCL для соединения низкоскоростных периферийных компонентов с микроконтроллером.

Данный протокол разработан фирмой Philips в начале 1980-х как 8-битная шина внутренней связи для создания управляющей электроники.

Для передачи данных в протоколе I2c используются две линии связи:

— SDA - последовательная линия данных;

— SCL - последовательная линия тактирования;

На рисунке 10 показано как по данному протоколу используя две линии связи передаются данные.

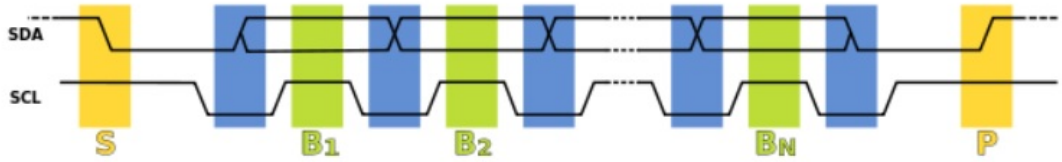

Рисунок 10 – Взаимодействие ведущего и ведомого устройства

На рисунке11 показано, как ведущее устройство взаимодействует с ведомым.

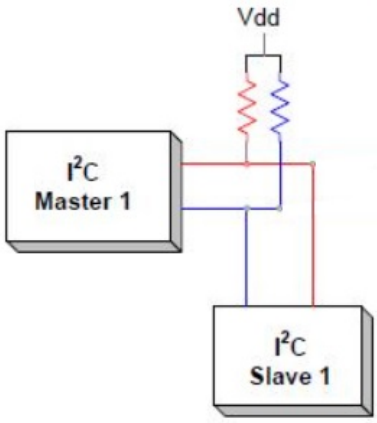

Рисунок 11 – Взаимодействие ведущего и ведомого устройства

В данной выпускной квалификационной работе использование протокола I2C было связано с тем, что дисплей I2C 1602 LCD взаимодействует с одноплатным контроллером посредством данного протокола.

3.2.4 Библиотека pyA20

Данная библиотека предоставляет методы для управления выводами GPIO для Orange Pi One. Она написана для платы A20-OLinuXino-MICRO, но также может использоваться и с другими платами. В функционале данной библиотеки доступны следующие методы:

— init() - произвести инициализацию модуля;
— getcfg() - считать из устройства конфигурацию GPIO;
— setcfg() - установить собственную конфигурацию GPIO;
— input() - возвращает текущее значение на выходе GPIO;
— output() - установить значение на выходе GPIO;
— pullup() - установить подтягивающеее напряжение для логической еденицы или логического нуля;

Для установки различных значений для выходов GPIO доступны следующие контакты:

— HIGH - логическая еденица;
— LOW - логический ноль;
— OUTPUT - выход логической еденийы
— PULLUP - тяга логической еденицы
— PULLDOWN - тяга логического нуля

Ниже приведен простой пример использования данной библиотеки для вывода на выход GPIO PG7 логической еденицы:

```
1  from pyA20.gpio import gpio
2  from pyA20.gpio import port
3
4  gpio.init()
5  gpio.setcfg(port.PG7, gpio.OUTPUT)
6
7  gpio.output(port.PG7, gpio.HIGH)
```

В данном примере мы сначала из библиотеки pyA20 импортируем необходимые классы:

```
1 from pyA20.gpio import gpio
2 from pyA20.gpio import port
```

Затем мы инициализируем модуль:

```
1  gpio.init()
```

Дальше мы назначаем контакт PG7 как управляемый для выхода и подаем на него сигнал:

```
1 gpio.setcfg(port.PG7, gpio.OUTPUT)
2 gpio.output(port.PG7, gpio.HIGH)
```

При использовании данной библиотеки каждый выход GPIO микроконтроллера имеет определенные названия представленные на рисунке 12.

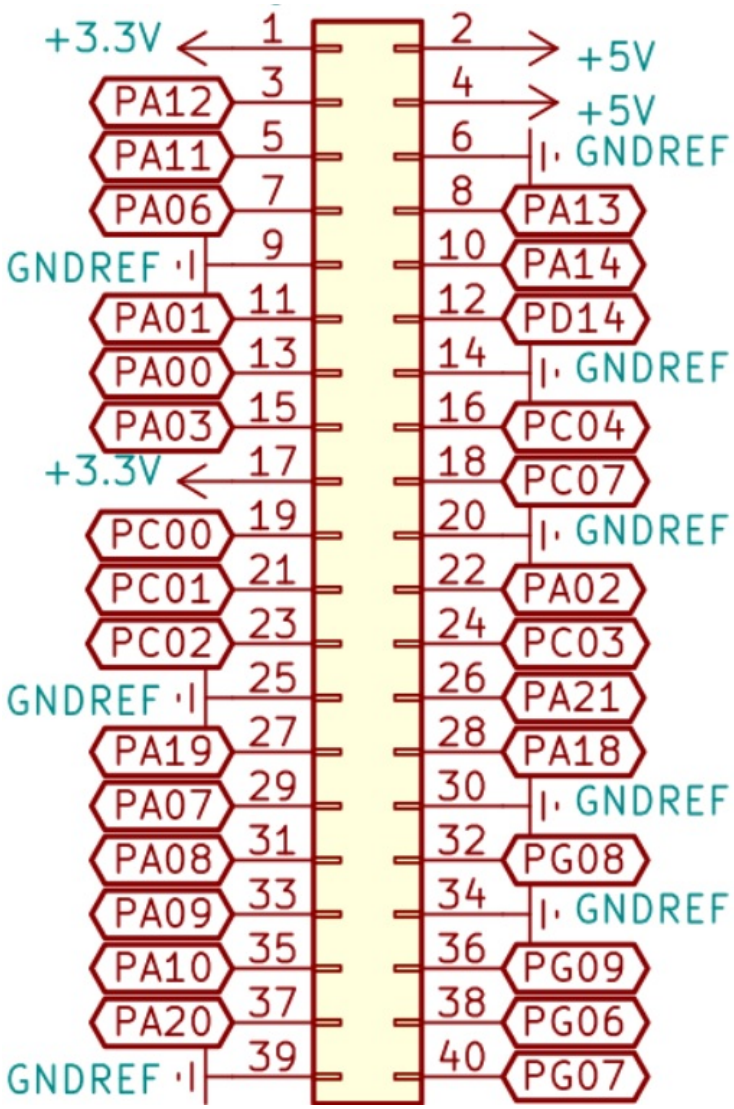

Рисунок 12 – Названия пинов микрокомпьютера Orange Pi One при работе с библиотекой pyA20

Данная библиотека была использована в выпускной квалификационной работе для взаимодействия с твердотельным реле. Плюсовой контакт реле был установлен в порт PG7, а минусовой на землю.

3.2.5 Библиотека Flask

Flask - фреймфорк для создания WEB-приложений на языке Python. Flask отличается от других фреймворков тем, что позволяет разработчикам полностью контролировать свои приложения. Он поставляется с ядром, которое включает в себя весь необходимый функционал для работы WEB-приложений.

Во Flask многие вещи предварительно сконфигурированы, на основе

общей базовой конфигурации. Например, шаблоны и статические файлы сохранены в подкаталогах в пределах исходного дерева.

Flask имеет две основные особенности. Это подсистема маршрутизации, отладки и интерфейс WEB-сервера(WSGI) взяты из мощной библиотеки для веб-приложений под названием Werkzeug. Поддержка шаблонов обеспечивается при помощи популярного шаблонизатора Jinja2.

В Flask нет встроенной поддержки для доступа к базам данных, проверки веб-форм, аутентификации пользователей или других задач высокого уровня. Эти и многие другие ключевые сервисы, которые необходимы большинству веб-приложений, доступны через расширения, которые интегрируются с основными пакетами. Программист может выбирать расширения, которые лучше всего подходят для проекта, или же писать свои собственные.

Ниже приведен базовый пример использования данной библиотеки.

```
1 from flask import Flask
2 app = Flask(name)
3
4 @app.route("/")
5 def hello():
6   return "Hello World!"
7
8 if name == "main":
9   app.run()
```

Сначала мы импортируем Flask класс. Экземпляр этого класса будет нашим WSGI приложением. Первым аргументом является имя модуля приложения. Если вы используете один модуль (как в данном примере), вы должны использовать name, потому что в зависимости от того, было ли это начато как приложение или как импорт модуля, название будет другим (’main’ по сравнению с реальным именем импорта).

```
1 from flask import Flask
```

Далее мы создаем экземпляр этого класса. Мы передаем ему имя модуля или пакета. Это необходимо, так как Flask не знает, где искать шаблоны, статические файлы, и так далее.

```
1 app = Flask(name)
```

Затем мы используем route(). Декоратор говорит Flask, что URL должен вызывать нашу функцию.

```
1
2 @app.route("/")
```

Функция задает имя, которое также используется для создания URL-адресов для этой функции, и возвращает сообщение, что мы хотим отобразить в браузере пользователя.

```
1 def hello():
2         return "Hello World!"
```

Использую run() функцию для запуска локального сервера с нашим приложением. Условие name == «main» означает, что сервер работает только в том случае, если скрипт выполняется непосредственно из Python интерпретатора и не используется в качестве импортированного модуля.

```
1 if name == "main":
2         app.run()
```

Данная библиотека была использована в выпускной квалификационной работе для создания WEB-сервера.

3.2.6 JavaScript Object Notation

JSON (JavaScript Object Notation) – формат для хранения и обмена данными. Файлы JSON представляют собой более простую альтернативу формату м аналогичными функциями XML.

Существует два основных элемента объекта JSON: ключ и значения.

- ключи должны быть строками. Они содержат последовательность символов, которые заключены в кавычки;
- значения являются допустимым типом данных JSON. Они могут быть в форме массива, объекта, строки, логического значения, числа или значения null;

Ниже представлен пример JSON объекта, хранящегося на сервере и использующийся для хранения значений, записанных из клиентской части, где ключ - номер цикла прокаливания, а значение состоит из массива, где первое значение - требуемая температура, а второе - время цикла работы устройства.

```
{
   "1":[
      100,
      30
   ],
   "2":[
      200,
      30
   ]
}
```

3.3 Клиентская часть приложения

Клиентская часть WEB-приложения представляет собой сайт. Данный сайт имеет начальную страницу на которой из имеется вывод текущей температуры внутри муфельной печи, страницу текущего состояния, которая в реальном времени ввиде графика отображает процессы, происходящие внутри муфельной печи, страницу режимов работы муфельной печи.

3.3.1 Javascript

Изначально, данный зык программирования был создан для анимирования HTML страниц. Впоследствии, данный язык стал применяться в реализации сценариев WEB-страниц, таких как отправление запросов на сервер, обработку и вывод принимаемых данных с сервера. JavaScript запускается на стороне клиента, который может использоваться для создания и программирование того, как будет вести себя WEB-страница при наступлении каких-либо событий. К основным архитектурным чертам можно отнести:

- динамическая типизация;
- слабая типизация;
- автоматическое управление памятью;
- прототипное программирование;
- функции как объекты первого класса;

Для выпускной квалификационной работы был выбран данный язык в качестве взаимодействия с серверной частью приложения, обработка полученных данных.

3.3.2 HTML CSS

HTML (HyperText Markup Language) — стандартный язык разметки документов во Всемирной паутине. Большинство веб-страниц создаются при помощи языка HTML (или XHTML). Язык HTML интерпретируется браузерами и отображается в виде документа в удобной для человека форме. HTML представляет собой нобор так называемых «тегов», описывающих структуру документа. Основные теги следующие:

- основные: html, head, title, body;
- структурны: div, span;
- текстовые: p, ul, li, h1-h6, b;
- таблицы: table, tr, td, th;
- ссылки: a;
- мультимедиа:img, object;
- формы: forom, input, label, select, option;
- специальные: script, link, meta

CSS (Cascading Style Sheets) — формальный язык описания внешнего вида документа, написанного с использованием языка разметки. Преимущественно используется как средство описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но может также применяться к любым XML-документам, например, к SVG или XUL.

В выпускной квалификационной работе были использованы язык разметки язык стилей для создания WEB страниц.

4 Разработка приложения

В данном разделе будут описаны скрипты, используемые в серверной и клиентской частях приложения. А также будет описана структура HTML документов.

4.1 Серверная часть приложения

WEB-сервер был разработан на языке программирования Python. При разработке использовались библиотеки Flask - для создания WEB сервера, smbus - для работы с дисплеем I2C 1602a LCD посредством шины I2C, spidev - для работы с аналого-цифорвым преобразователем MAX6675 посредством шини SPI, time - для корректной работы с временем, pyA20 - для работы с выходами GPIO.

Объявление импортированных библиотек выглядит следующим образом:

```
from flask import Flask, request, url_for, redirect, render_template
import spidev
import time
import smbus
from pyA20.gpio import gpio
from pyA20.gpio import port
```

После подключения всех библиотек объявляем константы, необходимые для работы с дисплеем посредством шины I2C.

Подключаем дисплей I2C 1602A LCD по схеме к одноплатному микрокомпьютеру запускаем утилиту i2cdetect для определения адреса шины. Получем адрес 0x27.

Создаем константу:

```
I2C_ADDR  = 0x27
```

Далее создаем константы, определяющие максимальное количество символов в строке. В моем случае это 16.

```
LCD_WIDTH = 16
```

Создадим две константы обозначения отправления либо данных, либо команд.

```
LCD_CHR = 1 # Mode - Sending data
LCD_CMD = 0 # Mode - Sending command
```

Далее создадим константы адресов первой и второй строчки дисплея.

```
LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line
```

Далее мы открываем для считывания и записи I2C интерфейс командой:

```
bus = smbus.SMBus(0)
```

После предварительных настроек создаем экземпляр класса Flask. Мы передаем ему имя модуля. Это необходимо, так как Flask не знает, где искать шаблоны, статические файлы, и так далее.

```
app = Flask(temperature_controller)
```

Опишем роугинг, созданный для открытия HTML страниц.

```
@app.route("/", methods=['POST', 'GET'])
def index():
    return render_template("index.html")
```

В данной части при GET запросе на стороне клиента будет отправлена стартовая страница сайта.

```
@app.route("/modes", methods=['POST', 'GET'])
def modes():
    if request.method == 'POST':
        return redirect(url_for('modes'))
    return render_template("modes.html")
```

В данной части при POST или GET запросе на стороне клиента будет отправлена страница режимов работы сайта.

```
@app.route("/graph", methods=['POST', 'GET'])
def graph():
    if request.method == 'POST':
        return redirect(url_for('graph'))
    return render_template("graph.html")
```

В данной части кода при POST или GET запросе на стороне клиента будет отправлена страница текущего состояния сайта.

Следующей разработанной функцией является функция для взятия температуры с термопары. мной был объявлен декоратор route() со значением

temperature. При отправке со стороны клиента по адресу с /temperature, ответом от сервера будет текущая температура внутри муфельной печи или, если произошел обрыв термопары, то ответом будет «Термопара не подключена».

```
@app.route("/temperature", methods=['POST', 'GET'])
def temperature():
    temp = temperatureCheck()
    if temp <= 1:
        return ("Термопара не подключена")
    else:
        return (str(temp))
```

В данной части кода я вызываю написанную мной функцию temperatureCheck() которая выглядит следующим образом:

```
def temperature_check():
    spi = spidev.SpiDev()
    spi.open(0,0)
    resp = spi.readbytes(2)
    temp = ((resp[1] + resp[0]*256)/8)*0.25
    return temp
```

Значение температуры мы берем из протокола SPI по 2 байта и преобразуем данные значения в градусы Цельсия.

Следующая функция необходима для считывания установленных температурных полок из полученного с клиентской части JSON объекта. Так же она записывает данные из JSON объекта в файл для последующего воспроизведения температурных установок на микрокомпьютере и запускает функцию работы с дисплеем и термопарой. Далее, она загружает страницу «текущего состояния» для отслеживания корректной работы устройства.

```
@app.route("/start_record", methods=['POST', 'GET'])
def start_record():
    if request.method == 'POST':
        f = open('temperature_mode.txt', 'w')
        i = 1
        for data in request.get_json():
            if i == 2:
                i = 1
                f.write(data + '\n')
            else:
```

```
                i += 1
                f.write(data + ' ')
        f.close()
    start_working()
    return render_template("graph.html")
```

В данной части кода вызывается функция StartWorking(), которая запускает работу дисплея а также считывает первую температурную полку из созданного файла и вызывает функцию работы с конкретной температурной полкой. Данная функция выглядит следующим образом:

```
def start_working():
    lcd_byte(0x01, LCD_CMD)
    records = []
    with open("temperature_mode.txt", "r") as f:
        for line in f.readlines():
            records.append(line)
    while (records):
        cur_mode(records.pop(0))
```

Функция работы с текущей температурной полкой выглядит следующим образом:

```
def cur_mode(line):
    lst = line.split()
    temperature = int(lst[0])
    time_set = int(lst[1])

    lcd_init()

    signal = port.PG7
    gpio.init()
    gpio.setcfg(signal, gpio.OUTPUT)

    while (time_set):
        minute = 60
        while (minute):
            cur_temperature = temperature_check()
            lcd_string("SET" + "{}".format(temperature),LCD_LINE_1)
            lcd_string("CUR" + "{}".format(cur_temperature),LCD_LINE_2)
            if cur_temperature < temperature:
                gpio.output(signal, 1)
```

```
            else:
                gpio.output(signal, 0)
                minute = minute - 1
            time.sleep(1)
        time_set = time_set - 1
```

В данной функции мы сначала объявляем две переменные, которые в дальнейшем будут выводится на дисплей.

```
    lst = line.split()
    temperature = int(lst[0])
    time_set = int(lst[1])
```

Следующим нашим действием мы вызываем функцию lcdInit, которая инициализирует настройки дисплея.

```
def lcd_init():
    # Initialise display
    lcd_byte(0x33,LCD_CMD) # 110011 Initialise
    lcd_byte(0x32,LCD_CMD) # 110010 Initialise
    lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
    lcd_byte(0x0C,LCD_CMD) # 001100 Display On,Cursor Off, Blink Off
    lcd_byte(0x28,LCD_CMD) # 101000 Data length, number of lines, font size
    lcd_byte(0x01,LCD_CMD) # 000001 Clear display
    time.sleep(E_DELAY)
```

Далее мы инициализируем порт PG7 для взаимодействия с твердотельным реле.

```
    signal = port.PG7
    gpio.init()
    gpio.setcfg(signal, gpio.OUTPUT)
```

Дальше мы создаем цикл, в котором обрабатываем время, которое было передано для текущей температурной настройки. Так же проверяем каждую секунду изменения температуры функцией temperatureCheck() если температура ниже необходимой. подаем сигнал на твердотельное реле, если же температура достаточная или же выше требуемой, останавливаем подачу сигнала. Также данная часть кода отправляет в функцию печати на дисплей - lcdString, которая будет описана позже, значения текущей температуры и температуры заданной температурной полкой.

```
    while (time_set):
        minute = 60
        while (minute):
            cur_temperature = temperature_check()
            lcd_string("SET " + "{}".format(temperature),LCD_LINE_1)
            lcd_string("CUR " + "{}".format(cur_temperature),LCD_LINE_2)
            if cur_temperature < temperature:
                gpio.output(signal, 1)
            else:
                gpio.output(signal, 0)
                minute = minute - 1
            time.sleep(1)
        time_set = time_set - 1
```

Функция вывода необходимых строк выглядит следующим образом:

```
def lcd_string(message,line):
    # Send string to display

    message = message.ljust(LCD_WIDTH," ")

    lcd_byte(line, LCD_CMD)

    for i in range(LCD_WIDTH):
        lcd_byte(ord(message[i]),LCD_CHR)
```

Данная функция на вход получает строку, которую нужно вывести и номер строки (1 или 2) для вывода на дисплей.

Далее Использую run() функцию для запуска локального сервера с моим приложением. Условие name == «main» означает, что сервер работает только в том случае, если скрипт выполняется непосредственно из Python интерпретатора и не используется в качестве импортированного модуля.

```
if temperature_controller == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)
```

Полный код приведен в Приложении А, листинг «server.py»