Муниципальное автономное общеобразовательное учреждение

«Средняя общеобразовательная школа №19»

**Система автоматизированного температурно-пропускного контроля (САТПК)**

творческий проект

|  |
| --- |
| Обучающиеся: Чечнева Анна Артёмовна,  Деревяшкина Анна Игоревна, 11Б кл.  Руководители: Романов Сергей Юрьевич, Можанов Станислав Александрович учителя информатики и ИКТ  Куратор: Филонова Е. В. |

п. Пироговский, 2021г.

ОГЛАВЛЕНИЕ

**Аннотация**…...……………………………………………...………………..…...4**Введение**…………………………………………………………………...…..….6

**Глава 1. Назначение системы**……………………….……………...………8-11

1.1.Система контроля и управления доступом…………………...................8

1.2. История Arduino……….............................................................................9

1.3. Telegram Bot……………………………....................................................9

1.4. Wi-Fi...........................................................................................................10

Вывод к первой главе……………………………………..............................11

**Глава 2. Система автоматизированного температурно-пропускного контроля**……………………………………………………………….......…12-18

2.1 Предпроектное исследование. Анкетирование…………………………12

2.2. Выбор материалов и оборудования…………………………...………...12

2.3. ARDUINO. Общие сведения.….........................................................…...12

2.4. Технологическая карта САТПК……………………......……………..…16

Вывод ко второй главе……………………………….….…………………....16

**Заключение**………………………….………..………………….…………..….17

**Список используемых интернет -источников**…………………..…….....…19

**Приложения**………………………………………..…….……….……………..20

Приложение 1. Результаты предпроектного анкетирования……………….…21

Приложение 2. Таблица расчёта стоимости макета...........................................23

Приложение 3. Код, отвечающий за работу инфракрасного датчика температуры MLX90614 и модуля датчика расстояния HC-SR04...................24

Приложение 4. Код, отвечающий за работу модуля датчика расстояния HC-SR04, светодиодной RGB матрицы, аналогового сервомотора постоянного вращения.................................................................................................................27

Приложение 5. Код, отвечающий за работу Wi-Fi-модуля………………...…31

Приложение 6. Макетная и принципиальная схемы соединения элементов..35

**Аннотация проекта**

**Название проекта:** Система автоматизированного температурно-пропускного контроля (САТПК).

**Предметная область:** информационные технологии.

**Участники проекта:** учащиеся 11-го класса.

**Цели и задачи проекта:**

- автоматизировать процесс замера температуры;

- ускорить проход учащихся в здание школы;

- сократить риски потенциального заражения учителей и всего персонала;

- создать модель САТПК на базе Arduino;

- провести предпроектное исследование;

- проанализировать полученные данные;

- изучить принцип работы платформы Arduino;

- собрать макет САТПК, производящего замер температуры и пропуск человека;

- написать программное обеспечение для работы САТПК на базе платформы Arduino;

- провести тестирование макета САТПК, производящего замер температуры и пропуск человека.

**Описание проекта:** создание модели системы автоматизированного температурно-пропускного контроля (САТПК) на базе Arduino для увеличения скорости температурно-пропускного контроля в образовательном учреждении и снижения риска заражения COVID-19.

**Информационные технологии:** Microsoft office 365, Arduino IDE.

**Ключевые слова:**САТПК, Arduino, СКУД, температурно-пропускной контроль, датчики, сервомотор.

**Авторы проекта:** Деревяшкина Анна Игоревна и Чечнева Анна Артёмовна – ученицы 11Б класса МАОУ СОШ №19 п. Пирогово.

**Project annotation**

**Project name:** Automated temperature control system (ATCS).

**Subject area:** Information technology.

**Project participants:** 11th grade students.

**Aim and objectives of the project:**

- to make the process of temperature measurement automatic;

- to speed up the students’ admission into the school building;

- to reduce the risks of potential teachers and all staff’s infection;

- to create an ATCS model based on Arduino;

- to conduct a pre-project research;

- to analyze the received data;

- to learn how the Arduino works;

- to make a model of ATСS for the temperature and human admission measurement;

- to write software for ATСS operation based on Arduino;

- to test the ATСS model measuring the temperature and human admission.

**Project description:** creating a model of an Automated temperature control system (ATCS) based on Arduino to increase the speed of temperature control in an educational institution and reduce the risk of COVID-19 infection.

**Information technology:** Microsoft office 365, Arduino IDE.

**Keywords:** ATCS, Arduino, ACS, temperature control, sensors, servo.

**Project authors:** Derevyashkina Anna Igorevna and Chechneva Anna Artemovna - pupils of the 11B grade, school 19, Pirogovskiy.

**Введение**

**Актуальность:** За всю свою историю человечество не раз сталкивалось с пандемиями и эпидемиями различных болезней, начиная от бубонной чумы в 15 веке и заканчивая лихорадкой Эбола 2014 года. Сегодня мы в очередной раз столкнулись с проблемой новой болезни и повсеместного её распространения. С марта 2020 года в России было установлено множество ограничительных мер по профилактике заражения COVID-19. Одной из таки мер стал контроль температуры людей в общественных местах, в том числе и в учебных учреждениях.

**Проблема:** В нашей школе нет возможности поставить специальные рамы для измерения и контроля температуры, поэтому учителя и охрана ежедневно стоят на входе и проводят замеры. Данный проект необходим в связи с острой эпидемиологической обстановкой в стране и реализовать его следует до конца апреля 2021 года.

Исходя из проблемы, ставим перед собой следующие **цели**:

- автоматизировать процесс замера температуры;

- ускорить проход учащихся в здание школы;

- сократить риски потенциального заражения учителей и всего персонала.

- создать модель САТПК на базе Arduino

**Задачи:**

1. Провести предпроектное исследование.
2. Проанализировать полученные данные.
3. Придумать и выбрать лучший вариант САТПК, производящего замер температуры и пропуск человека.
4. Придумать и выбрать варианты работы устройства.
5. Изучить принцип работы платформы Arduino.
6. Собрать макет САТПК, производящего замер температуры и пропуск человека.
7. Написать программное обеспечение для работы САТПК на базе платформы Arduino.
8. Провести тестирование макета САТПК, производящего замер температуры и пропуск человека.
9. При наличии недочётов – исправить их.

**Методы исследования:**

1. *Эмпирические:* наблюдение; анкетирование; опрос; тестирование; фотографирование; измерение; сравнение.
2. *Экспериментально-теоретические:*эксперимент; анализ; моделирование; логический.
3. *Математический метод:* статистические; визуализация данных, количественная обработка данных; качественная обработка данных.
4. *Теоретические:* изучение и обобщение; формализация; анализ и синтез.

**Практическая значимость:** Наш проект является достаточно актуальным, ведь эпидемиологическая обстановка не лучшая, и контролировать её в школе не менее важно. В нашей школе обучается свыше двух тысяч человек. Большинство учеников, приходит незадолго до начала первого урока, в результате чего собираются целые очереди. Наше устройство способно решить данную проблему, как в нашей школе, так и в других нуждающихся в этой системе учреждениях.

**Продукт проекта:** макет системы автоматизированного температурно-пропускного контроля (САТПК) со встроенным Wi-Fi-модулем.

**Глава 1. Назначение системы**

**1.1 Система контроля и управления доступом**

**СКУД** — это одна из наиболее эффективных систем, способных предотвратить проникновение любых нежелательных людей на всю охраняемую территорию. Системы контроля доступа, способны помочь и обеспечить целостность и защиту не только для материальных ценностей и важной информации, но и гарантируют безопасность для персонала и посетителей. Позволяют держать двери закрытыми и при этом, не мешают никакой ежедневной деятельности сотрудников. СКУД отслеживают перемещения всех сотрудников в офисе и разрешают вести учет и фиксировать отработанное сотрудниками время и всегда выявлять нарушения трудовой дисциплины. Программное обеспечение систем, позволяет получить данные о прогулах, опоздавших, либо ушедших с работы раньше времени, не только ежедневно, но и в любой отрезок времени.

Система контроля и управления доступом (СКУД)— это совокупность технических средств и организационных мероприятий, позволяющих контролировать доступ к объектам СКУД и отслеживающих перемещение людей по охраняемой территории. В настоящее время, СКУД признаны одним из наиболее эффективных методов решения задач комплексной безопасности для объектов.

Глядя на неуклонный рост интереса к СКУД и перспективу широкого их применения в ближайшем будущем, не следует забывать, что СКУД лишь упрощает процесс идентификации, экономит время и повышает эффективность работы служб безопасности предприятия, но, при этом, все равно требует контроля со стороны человека. От уровня вероятных угроз и поставленных перед системой задач, зависит необходимость подбора оптимального соотношения между людьми и техническими ресурсами системы. Установка системы контроля доступа, позволит не только поднять уровень общей безопасности, но и сократить издержки затрат на ее обеспечение, поскольку СКУД не требуют большого количества персонала для обслуживания, экономичны в потреблении электроэнергии. [7]

**1.2 История Arduino**

Основателями компании, которая начала создавать платы Arduino, являются итальянцы Массимо Банци, Девида Куартиллье, Тома Иго, Джанлука Мартино и Девида Меллиса. Такой была первоначальная команда создателей. А название они позаимствовали у итальянского бара, который, в свою очередь, был назван в честь короля Италии. [4]

Многие устройства для начинающих на основе Arduino не требуют серьезных знаний в технике или программировании. Arduino называют аппаратно-программной платформой. Она изначально создана компанией Arduino Software и представляет собой плату с контактами для подключения дополнительных компонентов.

Технические характеристики зависят от модели используемого микроконтроллера. [6]

**1.3 Telegram Bot**

**Боты** – это специальные программы, выполняющие различные функции и упрощающие жизнь их пользователей. Написанные для платформы Telegram, они предназначены для выполнения самых разных функций: от получения новостей до поиска информации и даже торговли акциями. Главное задачей бота является автоматический ответ после введенной ему пользователем команды. При этом, работая непосредственно через интерфейс Telegram, программа имитирует действия живого пользователя, за счет чего пользование таким ботом удобно и понятно.

Боты делятся на несколько направлений:

* *Чат-боты.* Представляют из себя простейший чат, имитирующий общение на заданную пользователем тематику.
* *Боты-информаторы.* Отдельный вид ботов, главная цель которых — информирование пользователя о тех или иных событиях (новости, мероприятия, публикации и т.п.).
* *Игровые боты.*Боты, в которых можно поиграть в различные игры. По большей части, это текстовые версии разных игр.
* *Боты-ассистенты.*Боты, разработанные различными онлайн-сервисами как дополнение к основной веб-версии.

Благодаря своему принципу работы, боты стали карманными помощниками, пользоваться которыми можно даже не покидая мессенджер. Они предоставляют возможность решать элементарные задачи при помощи мгновенных команд, при все эти программы не нуждаются в установке и не занимают отдельное место в памяти вашего девайса. [11]

**1.4 Wi-Fi**

**Wi-Fi** — технология беспроводной локальной сети с устройствами на основе стандартов [IEEE 802.11](https://ru.wikipedia.org/wiki/IEEE_802.11).

Обычно схема сети Wi-Fi содержит не менее одной [точки доступа](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D1%81%D0%BF%D1%80%D0%BE%D0%B2%D0%BE%D0%B4%D0%BD%D0%B0%D1%8F_%D1%82%D0%BE%D1%87%D0%BA%D0%B0_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%B0) и не менее одного клиента. Также возможно подключение двух клиентов в режиме [точка-точка (Ad-hoc)](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D1%81%D0%BF%D1%80%D0%BE%D0%B2%D0%BE%D0%B4%D0%BD%D0%B0%D1%8F_ad-hoc-%D1%81%D0%B5%D1%82%D1%8C), когда точка доступа не используется, а клиенты соединяются посредством [сетевых адаптеров](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%B0%D0%B4%D0%B0%D0%BF%D1%82%D0%B5%D1%80) «напрямую». Точка доступа передаёт свой идентификатор сети ([SSID](https://ru.wikipedia.org/wiki/SSID)) с помощью специальных сигнальных пакетов на скорости 0,1 Мбит/с каждые 100 мс. Поэтому 0,1 Мбит/с — наименьшая [скорость передачи данных](https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D0%BE%D1%80%D0%BE%D1%81%D1%82%D1%8C_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) для Wi-Fi. Зная SSID сети, клиент может выяснить, возможно ли подключение к данной точке доступа. При попадании в зону действия двух точек доступа с идентичными SSID приёмник может выбирать между ними на основании данных об уровне сигнала. Стандарт Wi-Fi даёт клиенту полную свободу при выборе критериев для [соединения](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B5%D0%B4%D0%B8%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5). Более подробно принцип работы описан в официальном тексте стандарта.

Однако стандарт не описывает всех аспектов построения беспроводных локальных сетей Wi-Fi. Поэтому каждый производитель оборудования решает эту задачу по-своему, применяя те подходы, которые он считает наилучшими с той или иной точки зрения. [5]

**Вывод к первой главе**

Выяснив, что в нашей школе используется система контроля и управления доступом, и изучив ее принцип работы, мы обнаружили, что она не оснащена термодатчиками. В связи с тем, что школа не может приобрести специальное оборудование, бесконтактно измеряющее температуру и уменьшающее риск заражения учителей, мы решили создать систему автоматизированного температурно-пропускного контроля (САТПК) на базе платформ Arduino, оснащенную Wi-Fi-модулем, передающим в Telegram-бот данные о проходе в здание учащегося.

**Глава 2. Система автоматизированного температурно-пропускного контроля**

**2.1 Предпроектное исследование. Анкетирование**

Мы провели анкетирование среди учителей и персонала нашей школы, чтобы выяснить насколько комфортно и быстро они проводят замеры температуры учащихся. Проанализировав данные, собранные в ходе анкетирования, наша команда обнаружила, что использование бесконтактных термометров не занимает слишком много времени, но всё равно приводит к созданию очередей, что повышает вероятность заражения.

**2.2 Выбор материалов и оборудования**

Для создания САТПК, производящего замер температуры и пропуск человека, мы решили усовершенствовать систему контроля и управления доступом(СКУД) нашей школы, внедрив в нее САТПК, используя при этом наиболее простую, широко распространённую и многофункциональную платформу Arduino.

Платформа Arduino создавалась, чтобы практически любой пользователь смог создать свое интересное и полезное устройство. Arduino выполнена на популярных микропроцессорах Amtel и ATMEGA, кодирование программы может расширяться на платформе С++. Оболочка программы является легкой в применении для начинающих программистов, однако имеет достаточную гибкость для работы профессионалов. Не маловажным фактором является приемлемая цена Arduino. [6]

**Корпус модели САТПК**

Корпус модели САПТК был создан из деревянной фанеры толщиной 7мм, поролона, окрашен экологичной краской на водной основе.

**2.3 ARDUINO. Общие сведения.**

**Arduino Nano**

Arduino Nano – это одноплатный контроллер с открытыми начальными кодами, который возможно использовать во множестве различных приложений. Это самый простой и наиболее дешевый вариант из микроконтроллеров для различных любителей, студентов и профессиональных разработчиков проектов на основе микроконтроллера.

В платах Arduino используются либо микроконтроллеры Atmel AVR, либо микроконтроллер Atmel ARM, а в некоторых из версий имеется интерфейс USB. Обладают шестью или более выводами аналоговых входов и четырнадцатью или более выводами цифровых входов и выходов, использующихся для возможности подключения к микроконтроллеру датчиков, различных приводов и иных периферийных схем.

***Преимущества***

* Цена. Arduino Nano возможно купить менее чем за 1000 руб.
* Кроссплатформенность. Программное обеспечение Arduino осуществляет работу на большинстве известных программ Windows, Macintosh OS X, Linux, являясь открытым приложением, работающим на Java.
* Простая среда программирования. Программная оболочка является достаточно простой в применении для новичков, но весьма гибкой для большинства продвинутых пользователей, чтобы оптимально быстро достичь нужного вам результата. Особенно комфортно в образовательной среде, где студенты достаточно легко разберутся с платформой, а преподаватели смогут разработать учебный курс.
* Открытый исходный код. Язык может расширяется с помощью C++ библиотек, значительно более продвинутых, там специалисты могут самостоятельно создать свой собственный эксклюзивный инструментарий для Arduino на основе инновационного компилятора AVR C.
* Открытые спецификации и схемы оборудования. Arduino основан на микроконтроллерах Atmel ATMEGA8 и ATMEGA168. Схемы модулей публикуются под лицензией Creative Commons, из-за этого опытные схемотехники могли создавать свои собственные версии модуля. Даже весьма неопытные пользователи смогут делать макетную версию данного модуля, чтобы понимать, каким же образом он осуществляет работу и экономит деньги. [3]

**MTR-SERVO-FS90R, Аналоговый сервомотор постоянного вращения**

Аналоговый сервомотор с крутящим моментом 1.5 кг/см при напряжении питания 6В и свободным вращением на 360°, может использоваться как мотор, который не требует для работы дополнительного драйвера. Идеально подходит для робототехники. [10]

**Troyka-RGB Led 4x4, Светодиодная RGB матрица 4×4 [на основе LEDs SK6812]**

Светодиодная RGB-матрица подойдёт для индикации состояния устройств, пригодится при создании точечной подсветки и украсит рекламные конструкции. На квадратном модуле со стороной в один дюйм находится сразу 16 трёхцветных светодиодов. Каждый диод обладает собственным контроллером и может менять цвет индивидуально.

Светодиодный модуль имеет на борту буфер преобразования логических уровней. Это позволяет модулю работать как с сигналами на уровне 5 вольт, так и на уровне в 3.3 вольта. [9]

**Инфракрасный датчик температуры MLX90614**

Измеряет 2 вида температур - температуру окружающей среды и температуру объекта на расстоянии. Инфракрасные термометры работают на основе явления, называемого излучением черного тела. Все, что находится при температуре выше абсолютного нуля, имеет внутри движущиеся молекулы. Чем выше температура, тем быстрее движутся молекулы. По мере движения молекулы излучают инфракрасное излучение - тип электромагнитного излучения ниже видимого спектра света. По мере того как они нагреваются, они излучают больше инфракрасного излучения и даже начинают излучать видимый свет. Инфракрасные термометры обнаруживают и измеряют это излучение. [2]

**Модуль Wi-Fi ESP8266 (ESP-01)**

ESP-01 — плата-модуль WiFi на базе популярного чипсета ESP8266EX. На борту платы находится микросхема Flash-памяти объёмом 2 МБ, чип ESP8266EX, кварцевый резонатор, два индикаторных светодиода и миниатюрная антенна из дорожки на верхнем слое печатной платы в виде змейки. Flash-память необходима для хранения программного обеспечения. При каждом включении питания, ПО автоматически загружается в чип ESP8266EX.

По умолчанию модуль настроен на работу через «AT-команды». Управляющая плата посылает команды — Wi-Fi модуль выполняет соответствующую операцию.

Внутри чипа ESP8266 прячется целый микроконтроллер, который является самодостаточным устройством. Прошивать модуль можно на разных языках программирования. [1]

**Модуль датчика расстояния HC-SR04 для Arduino**

Датчик расстояния Arduino является прибором бесконтактного типа, и обеспечивает высокоточное измерение и стабильность. Диапазон дальности его измерения составляет от 2 до 400 см. На его работу не оказывает существенного воздействия электромагнитные излучения и солнечная энергия. В комплект модуля с HC SR04 Arduino также входят ресивер и трансмиттер. [8]

**2.4 Технологическая карта САТПК**

Так как Wi-Fi-модуль выступает в роли отдельной платы, прямое подключение всех компонентов системы невозможно. Поэтому была использована система «**Ведущий — ведомый**» ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *master/slave*), соединяющая две платы Arduino Nano и Wi-Fi-модуль: плата "ведущий" получает данные с **инфракрасного датчика температуры**, анализирует их и передает на две платы: "ведомую", задачей которой является зажигание двух **светодиодных RGB матриц 4×4** и приведение в действие макета шлагбаума за счет работы **аналогового сервомотора постоянного вращения**, и **Wi-Fi-модуля**, передающего сообщение в Telegram-бот.

Если температура соответствует допустимым значениям (34.5С-37,3С), в действие приводятся **светодиодные RGB матрицы 4×4**, загорающиеся зелёным цветом, а **аналоговый сервомотор постоянного вращения** поднимает шлагбаум, который держится до момента, пока **модуль датчика расстояния** не засечет дистанцию более 15 см, подразумевающее отсутствие препятствий под шлагбаумом. В Telegram-бот приходит сообщение "Температура в норме!!!"

Если температура не соответствует допустимым значениям (34.5С-37,3С), в действие приводятся **светодиодные RGB матрицы 4×4**, загорающиеся красным цветом, а **аналоговый сервомотор постоянного вращения** не поднимает шлагбаум. В Telegram-бот приходит сообщение "!!!Заберите ребёнка домой!!!"

**Вывод ко второй главе**

Используя знания о датчиках и платформе Arduino, мы собрали макет системы автоматизированного температурно-пропускного контроля (САТПК) со встроенным Wi-Fi-модулем. Процесс сборки и приобретения необходимых деталей показал, что данная модель является бюджетной и доступной для широкого распространения.

**Заключение**

Начиная с сентября 2020 года, наша команда обнаружила проблему пропуска детей в школу, из-за чего и взялась за разработку эффективного решения и создание концепции нового и актуального элемента СКУД школы. Свою работу мы начали с опроса персонала школы, из которого выяснили, что время прохода одного ребенка составляет 5-7 секунд, что приводит к образованию толпы у входа и способствует потенциальному распространению инфекции.

Изучив материал на тему создания макетов и принцип работы программной среды Arduino, мы принялись за подбор необходимых электронных компонентов, таких как датчики температуры и расстояния, сервомотор постоянного вращения, RGB-матрицы, Wi-Fi-модуль. Так же мы изучили литературу на тему работы и создания Telegram-ботов, что пригодилось нам для реализации визуальной передачи данных с макета на подключенное устройство.

По завершению изучения теоретического материала, мы начали создавать макет системы автоматизированного температурно-пропускного контроля (САТПК) со встроенным Wi-Fi-модулем и провели множество тестов на различных возрастных группах, доказывающих эффективную работу и скорость замера. Создав данное устройство, нам удалось автоматизировать процесс замера температуры. За счёт увеличения скорости, очереди стали меньше, а значит и риск потенциального заражения снизился. Так же нам удалось реализовать передачу данных о проходе человека в здание и о его температуре на привязанное устройство. Стоит отметить, что нам пока не удалось создать базу данных для полноценного контроля проходящих через САТПК, но модель имитации базы данных на n-количество пользователей создана и активно дорабатывается.

Таким образом нам удалось достичь всех поставленных целей и задач, а также создать действующий макет системы автоматизированного температурно-пропускного контроля, который является универсальной и экономически выгодной системой. В данный момент наша команда занимается подключением RFID-модуля, благодаря которому будет происходить идентификация входящего в здание сотрудника или учащегося и запись температуры тела и времени прохода в базу данных. В будущем, мы планируем внедрить САТПК в систему школы, что облегчит организацию пропускного режима и снимет лишнюю нагрузку с сотрудников.

**Список используемых интернет-источников**

1. Амперка [Электронный ресурс] Модуль Wi-Fi ESP8266 (ESP-01): подключение, распиновка, прошивка в IDE. Режим доступа: [www.wiki.amperka.ru](http://www.wiki.amperka.ru) свободный.
2. АрдуиноПлюс.ру [Электронный ресурс]/ Инфракрасный датчик температуры MLX90614. Режим доступа: [www.arduinoplus.ru](http://www.arduinoplus.ru) свободный.
3. АрдуиноПлюс.ру [Электронный ресурс]/ Контроллер Arduino Nano для начинающих. Режим доступа: [www.arduinoplus.ru](http://www.arduinoplus.ru) свободный.
4. АрдуиноПлюс.ру [Электронный ресурс]/ Что такое Ардуино. Режим доступа: [www.arduinoplus.ru](http://www.arduinoplus.ru) свободный.
5. ВикипедиЯ. Свободная библиотека [Электронный ресурс]/WIFI. Режим доступа: www.[ru.wikipedia.org](https://ru.wikipedia.org/wiki/Wi-Fi) свободный.
6. Электросам. Ру [Электронный ресурс]/ Arduino Uno. Устройство и применение. Режим доступа:www.electrosam.ru свободный.
7. Allbest [Электронный ресурс]/ Погост А. А. Системы контроля и управления доступом. 2012. Режим доступа: www.knowledge.allbest.ru свободный.
8. ARDUINOMASTER [Электронный ресурс]/ Ультразвуковой датчик расстояния Ардуино HC-SR04. Режим доступа: [www.arduinomaster.ru](http://www.arduinomaster.ru) свободный.
9. CHIPDIP [Электронный ресурс Troyka-RGB Led 4x4, Светодиодная RGB матрица 4×4 [на основе LEDs SK6812]. Режим доступа: www.chipdip.ru свободный.
10. CHIPDIP [Электронный ресурс]/ MTR-SERVO-FS90R, Аналоговый сервомотор постоянного вращения, 360°, 1.5 кг/см, 6В. Режим доступа: [www.chipdip.ru](http://www.chipdip.ru) свободный.
11. Shark Develop [Электронный ресурс]/ Боты в Telegram что это такое и как они работают. Режим доступа: [www.sharkdevelop.com](https://sharkdevelop.com/boty-v-telegram/) свободный.

**Приложения**

Приложение 1

Результаты предпроектного анкетирования

Диаграмма 1

Диаграмма 2

Диаграмма 3

Приложение 2

Таблица

Расчет стоимости макета САТПК

|  |  |  |
| --- | --- | --- |
| Компоненты макета | Стоимость компонента за шт./ м/ м2 в рублях | |
| у местных поставщиков | на международной торговой площадке Aliexpress |
| Плата Arduino Nano | 339 | 250 |
| MTR-SERVO-FS90R, Аналоговый сервомотор постоянного вращения | 97 | 74 |
| Troyka-RGB Led 4x4, Светодиодная RGB матрица 4×4 | 470 | отсутствует |
| Инфракрасный датчик температуры MLX90614 | 765 | 500 |
| Модуль Wi-Fi ESP8266 (ESP-01) | 159 | 120 |
| Модуль датчика расстояния HC-SR04 для Arduino | 99 | 60 |
| Соединительные провода | 40 | 20 |
| Фанера (7мм) | 55 | отсутствует |
| Итог: | 2932 | 2329 |

Приложение 3

Программный код 1

**Код, отвечающий за работу инфракрасного датчика температуры MLX90614 и модуля датчика расстояния HC-SR04***// Подключение библиотек*  
#include <Wire.h>  
#include <Servo.h>  
#include <Adafruit\_MLX90614.h>   
Adafruit\_MLX90614 mlx = Adafruit\_MLX90614();  
*// Создаем объект*  
Servo;   
   
void setup()   
{  
*// Инициализируем взаимодействие по последовательному порту*  
 Serial.begin(115200);  
*// инициализация датчика температуры*  
 mlx.begin();  
*//Определяем вводы и выводы*  
 pinMode(trigPin, OUTPUT);  
 pinMode(echoPin, INPUT);  
 delay(1000);  
}  
  
void loop()   
{  
 *// Инициализация переменных*  
 int a[1000];  
 int x=0;  
 float sum=0;  
 float t=0;  
 int c=0;  
 int duration, cm;  
 float SRt2=0;  
 float val = mlx.readObjectTempC();   
 *// Подтягивание значений (Смещение диапазона значений)*  
 val = map(val, -70, 380, -65.2, 384.8);  
 delay (500);  
 *// Выполнять пока температура больше 30С*   
 while (val>30)   
 {   
 val = mlx.readObjectTempC();   
 val = map(val, -70, 380, -65.2, 384.8);   
 *// Сумма замеренной температуры*  
 t=t+val;  
  *// Количество замеров*   
 x++;  
 *// Генерация короткого импульса длительностью 2-5 микросекунд*  
 digitalWrite(trigPin, LOW);  
 delayMicroseconds(2);  
 *// Выставив высокий уровень сигнала, ждем около 10 микросекунд. В этот момент датчик будет посылать сигналы с частотой 40 КГц*  
 digitalWrite (trigPin, HIGH);  
 delayMicroseconds(10);  
 *// Время задержки акустического сигнала на эхолокаторе.*  
 digitalWrite(trigPin, LOW);  
 *// Преобразование времени в расстояние*  
 duration = pulseIn(echoPin, HIGH);  
 cm = duration / 58;  
 *// Вычисление расстояния*  
 c=c+cm;  
 }  
 x=x-1;  
 *// Расчёт средних значений температуры и расстояния в период замера*  
 float SRt=t/x;  
 int SRR=c/x;  
 *// Корректировка температуры в зависимости от расстояния*  
 if (SRR==3)  
 {  
 SRt2=SRt+0.6;   
 }  
 if (SRR==4)  
 {  
 SRt2=SRt+1;   
 }  
 if (SRR==5)  
 {  
 SRt2=SRt+1.7;   
 }  
 if (SRR==6)  
 {  
 SRt2=SRt+2.2;   
 }  
 if (SRR==7)  
 {  
 SRt2=SRt+3;  
 }  
 if (SRR==8)  
 {  
 SRt2=SRt+3.4;  
 }  
 if (SRR==9)  
 {  
 SRt2=SRt+3.9;   
 }  
 if (SRR==10)  
 {  
 SRt2=SRt+4;  
 }  
   
*// Вывод результата температуры и передача вердикта в монитор порта*  
 if (SRt2>37.4)  
 {  
 Serial.println ("N");  
 }  
 if (34.5<SRt2 && SRt2<37.4)  
 {  
 Serial.println ("Y");  
 }  
}

Приложение 4

Программный код 2

**Код, отвечающий за работу модуля датчика расстояния HC-SR04, светодиодной RGB матрицы, аналогового сервомотора постоянного вращения***// Подключение библиотек*  
#include <Wire.h>  
#include <Servo.h>  
#include <Adafruit\_MLX90614.h>   
Adafruit\_MLX90614 mlx = Adafruit\_MLX90614();  
#include <Adafruit\_NeoPixel.h>  
*// Создаем объект*  
Servo servo;   
  
*// номер пина, к которому подключена RGB-матрица*  
#define MATRIX\_PIN 4  
*// количество светодиодов в матрице*  
#define LED\_COUNT 16  
   
*// создаём объект класса Adafruit\_NeoPixel*  
Adafruit\_NeoPixel matrix = Adafruit\_NeoPixel(LED\_COUNT, MATRIX\_PIN, NEO\_GRB + NEO\_KHZ800);  
  
*// константы для выводов*  
#define PIN\_TRIG 9  
#define PIN\_ECHO 8  
  
 long duration, cm;  
*// Контакт, к которому прикреплен светодиод*  
 const int ledPin = 13;   
*//Переменная для чтения входящих в последовательный порт данных*  
 int incomingByte;   
  
void setup()   
{  
*// Инициализируем взаимодействие по последовательному порту*  
 Serial.begin(115200);  
*//Определяем вводы и выводы*  
 pinMode(PIN\_TRIG, OUTPUT);  
 pinMode(PIN\_ECHO, INPUT);  
*// инициализация RGB-матрицы*  
 matrix.begin();   
*// инициализация датчика температуры*  
 mlx.begin();  
*// Указываем объекту класса Servo, что серво присоединен к пину 10*  
 servo.attach(10);   
*// Выставляем начальное положение*   
 servo.write(0);   
 delay(1000);  
}  
  
void loop()   
{  
 *// Генерация короткого импульса длительностью 2-5 микросекунд*  
 digitalWrite(PIN\_TRIG, LOW);  
 delayMicroseconds(5);  
 digitalWrite(PIN\_TRIG, HIGH);  
 *// Выставив высокий уровень сигнала, ждем около 10 микросекунд. В этот момент датчик будет посылать сигналы с частотой 40 КГц*  
 delayMicroseconds(10);  
 digitalWrite(PIN\_TRIG, LOW);  
 *// Время задержки акустического сигнала на эхолокаторе*  
 duration = pulseIn(PIN\_ECHO, HIGH);  
 *// Преобразование времени в расстояние*  
 cm = (duration / 2) / 29.1;  
 *// Задержка между измерениями для корректной работы скетча*  
 delay(250);  
   
 *// Нахождение последних данных в последовательном порту*  
 if (Serial.available() > 0)   
 {   
 incomingByte = Serial.read();  
   
 *// Нахождение данных в последовательном порту соответсвующих нормальной температуре*  
 if (incomingByte == 'Y')   
 {   
  *// Заполняем RGB-матрицу по сегментам «бегущий огонь» зелёного цвета*  
 colorWipe(matrix.Color(0, 255, 0), 50);  
  *// Поворачиваем серво на 90 градусов*  
 servo.write(20);   
 delay(2000);   
   
 while (cm<15)  
  *// выполнять проверку расстояния до тех пор, пока расстояние меньше 15см, после поворачиваем серво на 90 градусов*  
 {   
  *// Задержка между измерениями для корректной работы скетча*  
 delay (1000);  
 *// Генерация короткого импульса длительностью 2-5 микросекунд*  
 digitalWrite(PIN\_TRIG, LOW);  
 delayMicroseconds(5);  
 digitalWrite(PIN\_TRIG, HIGH);  
  *// Выставив высокий уровень сигнала, ждем около 10 микросекунд. В этот момент датчик будет посылать сигналы с частотой 40 КГц*  
 delayMicroseconds(10);  
 digitalWrite(PIN\_TRIG, LOW);  
  *// Время задержки акустического сигнала на эхолокаторе*  
 duration = pulseIn(PIN\_ECHO, HIGH);  
  *// Преобразование времени в расстояние*  
 cm = (duration / 2) / 29.1;  
 }  
 *// Поворачиваем серво на 90 градусов*  
 servo.write(-20);   
 *// заполняем RGB-матрицу по сегментам «бегущий огонь» белого цвета*  
 colorWipe(matrix.Color(0, 0, 0), 50);  
 }  
   
 *// Нахождение данных в последовательном порту соответсвующих нормальной температуре*  
 if (incomingByte == 'N')   
 {   
  *// заполняем RGB-матрицу по сегментам «бегущий огонь» красного цвета*  
 colorWipe(matrix.Color(255, 0, 0), 50);  
 delay(1000);   
  *// заполняем RGB-матрицу по сегментам «бегущий огонь» белого цвета*  
 colorWipe(matrix.Color(0, 0, 0), 50);  
 }  
 }  
}  
  
*// функция заполнения каждого сегмента*  
void colorWipe(uint32\_t c, uint8\_t wait)  
{  
 for (uint16\_t i = 0; i < matrix.numPixels(); i++)   
 {  
  *// заполняем текущий сегмент выбранным цветом*  
 matrix.setPixelColor(i, c);  
 matrix.show();  
  *// ждём*  
 delay(wait);  
 }  
}

Приложение 5

Программный код 3

Код, отвечающий за работу Wi-Fi-модуля

#include <string.h>

#include <ESP8266WiFi.h>

#include <ESP8266HTTPClient.h>

#include <ArduinoJson.h>

char TelegrammBOTtoken[50] = "1908978488:AAEw2fsL2HfTqTOESkrd7Crw7oGj7f5yh5c";

*//активация бота по ключу*

String Messange = "Сообщение !";

String IncomeMsg = "";

int LastMSG = 0;

int AskMSG = 0;

uint32\_t LastChatID = 0;

int incomingByte = 0;

void setup()

{

  Serial.begin(9600);

  WiFi.mode(WIFI\_STA);                //WiFi.begin("Galaxy S999ac", "zcrq4231");

  WiFi.begin("U310", "QwertytyU310");

*// параметры подключения к сети Wi-Fi*

  while (WiFi.status() != WL\_CONNECTED) {

    delay(5000);

    Serial.print(".");                                  *// пока не подключится выводить ". . . ."*

  }

    IncomeMsg = GetInitData();

*// присвоение последнего сообщения для парсинга*

  LastMSG = AskMSG;                      *//проверка нового сообщения*

  SendMessage(LastChatID, "ESP-модуль старт: " + IncomeMsg);

*//проверка Wi-Fi-модуля на работоспособность*

  delay(5000);

*//ESP.deepSleep(0, RF\_NO\_CAL);*

*// открываем порт Serial*

  Serial.begin(115200);

}

void loop() {

   if (Serial.available() > 0) {                      *//если есть доступные данные в Serial*

*// считываем байт*

        incomingByte = Serial.read();

        if(incomingByte == 'N') {

         SendMessage(LastChatID, "!!!Заберите ребёнка домой!!!");

*//отправка сообщения в telegram*

         }

         if(incomingByte == 'Y') {

         SendMessage(LastChatID, "Температура в норме!!!");

*//отправка сообщения в telegram*

         }

    }

    IncomeMsg = GetInitData();

    if (AskMSG != LastMSG) {

*// Если есть новое сообщение в телеграм боте, то отправляем его исходному адресату*

      LastMSG = AskMSG;

      SendMessage(LastChatID, "Ваше сообщение: " +   IncomeMsg);

      SendMessage(LastChatID, "Температура: " + Srt);

      }

    delay(1000);

    }

String GetInitData() {

  StaticJsonDocument<30> doc;

*// Создаем и наполняем json для последующей отправки на сервер*

  doc["method"] = "getUpdates";

  doc["offset"] = -1;                                *//Для проверки последнего сообщения*

  DynamicJsonDocument Answer(1532);

  deserializeJson(Answer, telegramPOST([doc.as](http://doc.as/)<String>()));

*// Парсим JSON-содержимое ответа сервера*

  AskMSG = Answer["result"][0]["message"]["message\_id"];

  LastChatID = Answer["result"][0]["message"]["chat"]["id"];

  return Answer["result"][0]["message"]["text"];

}

String telegramPOST(String jsonarr) {

*// Функция для работы с телеграм ботом*

  std::unique\_ptr<BearSSL::WiFiClientSecure>client(new BearSSL::WiFiClientSecure);

  client->setInsecure();

  HTTPClient https;

  String token = TelegrammBOTtoken;

  if (https.begin(\*client, "<https://api.telegram.org/bot>" + token + "/")) {    *// HTTPS*

    https.addHeader("Content-Type", "application/json");

    https.POST(jsonarr);

    String Answer = https.getString();

    https.end();

    return Answer;

  }

}

uint32\_t chatid() {

  StaticJsonDocument<30> doc;

*// Создаем и наполняем json для последующей отправки на сервер*

  doc["method"] = "getUpdates";

  doc["offset"] = -1;                              *//Для проверки последнего сообщения*

  DynamicJsonDocument Answer(1532);

  deserializeJson(Answer, telegramPOST([doc.as](http://doc.as/)<String>()));

*// Парсим JSON-содержимое ответа сервера*

  return Answer["result"][0]["message"]["chat"]["id"];         *//id*

}

String UserName() {

  StaticJsonDocument<30> doc;

*// Создаем и наполняем json для последующей отправки на сервер*

  doc["method"] = "getUpdates";

  doc["offset"] = -1;                   *//Для проверки последнего сообщения*

  DynamicJsonDocument Answer(1532);

  deserializeJson(Answer, telegramPOST([doc.as](http://doc.as/)<String>()));

*// Парсим JSON-содержимое ответа сервера*

  return Answer["result"][0]["message"]["from"]["first\_name"];  *//id*

}

void SendMessage(uint32\_t chatid, String message) {

  StaticJsonDocument<100> doc;

*// Создаем и наполняем json для последующей отправки на сервер*

  doc["method"] = "sendMessage";

  doc["chat\_id"] = chatid;

  doc["text"] = message;

  DynamicJsonDocument Answer(1532);

  deserializeJson(Answer, telegramPOST([doc.as](http://doc.as/)<String>()));

*// Парсим JSON-содержимое ответа сервера*

  analogWrite(LED\_BUILTIN, 0);

}

Приложение 6

Схема1

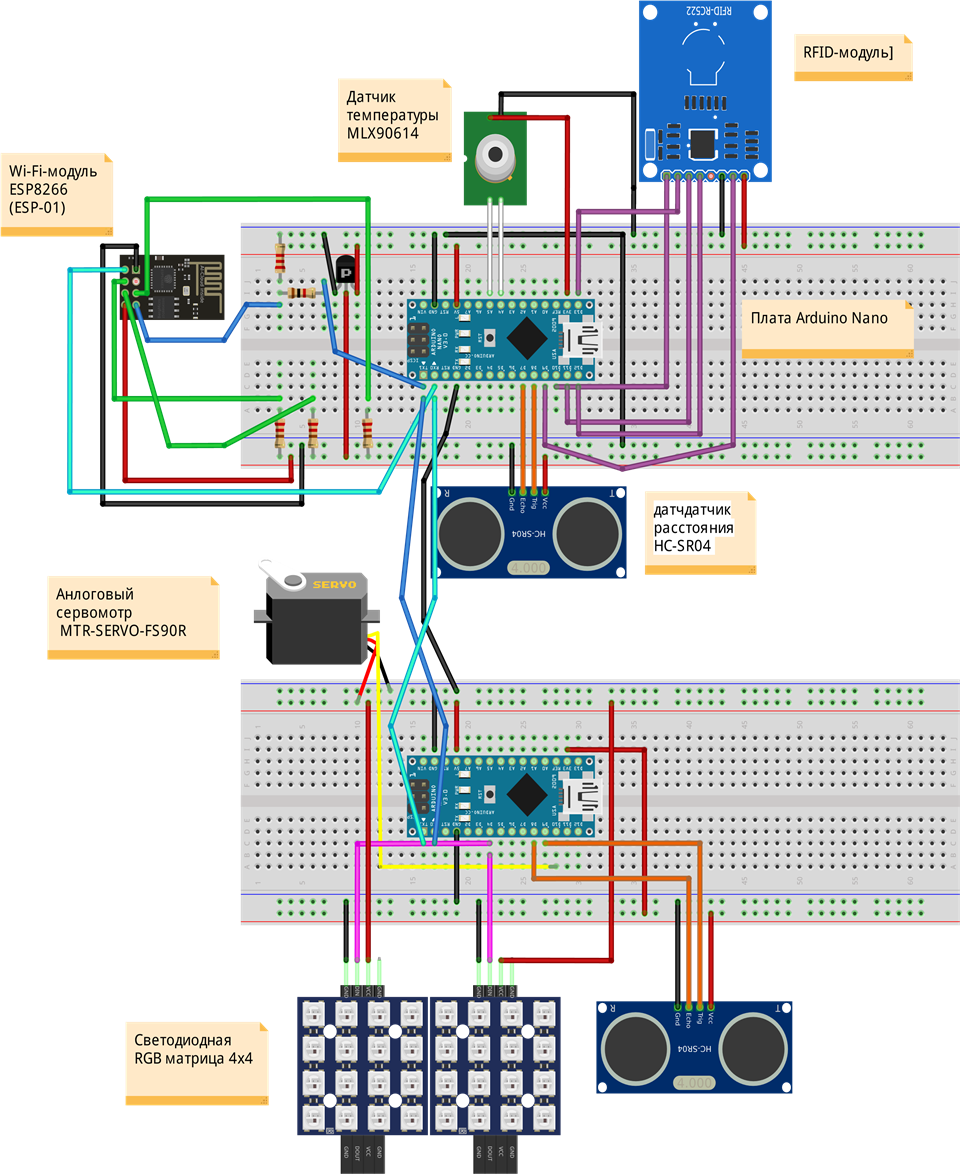
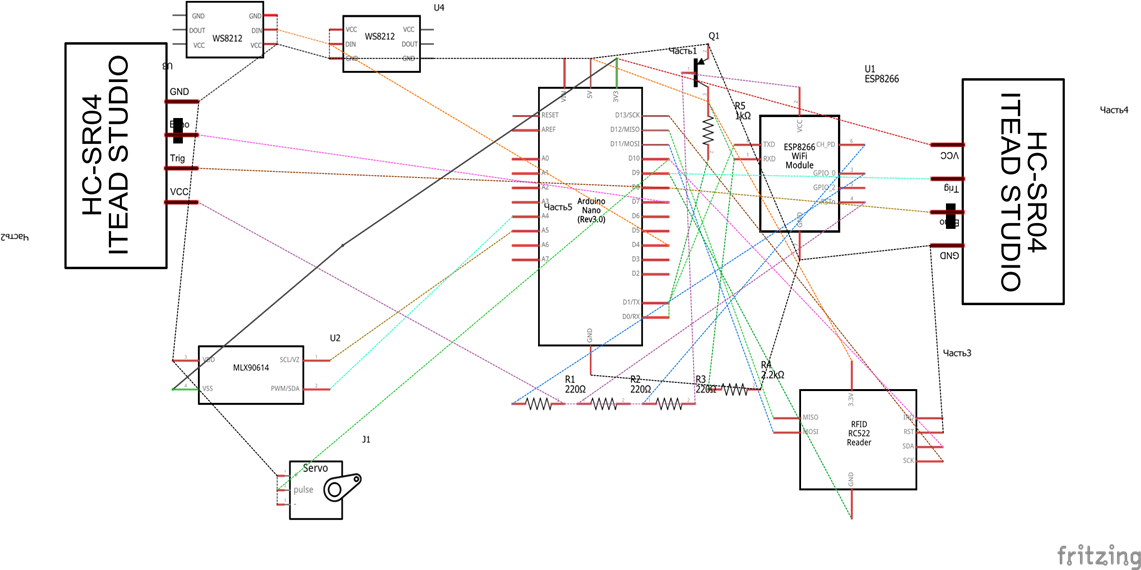
Схема макетной платы

Схема 2

Принципиальная схема