МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра физики открытых систем

Разработка системы дистанционного управления освещением

название темы выпускной квалификационной работы полужирным шрифтом

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студента 4 курса 4041 группы
направления 09.03.02 «Информационные системы и технологии»
код и наименование направления (специальности)
института физики
наименование факультета, института, колледжа
Тюлюкина Валерия Андреевича
фамилия, имя, отчество

Научный руководитель
доцент кафедры ФОС,
к.ф.-м.н. _____ М.К. Куровская
должность, уч. степень, уч. звание | дата, подпись | инициалы, фамилия

Заведующий кафедрой
д.ф.-м.н., профессор _____ А.А. Короновский
должность, уч. степень, уч. звание | дата, подпись | инициалы, фамилия

Саратов 2023 год

Введение. Благодаря современным технологическим достижениям все больше сфер жизни общества подвергаются автоматизации и внедрению информационных систем. Начиная от первых электронных вычислительных машин, созданных в середине 20-го века и заканчивая автоматизацией ведения бухгалтерского учета на предприятиях. Современный мир невозможно представить без колоссального количества информационных сетей и процессов, протекающих в них.

Бытовые информационные сети существуют уже не одно десятилетие и описаны многими стандартами за авторством различных производителей. Например, стандарт X-10, разработанный в 1975 году шотландской компанией Pico Electronics, описывает принципы связи между различными бытовыми устройствами с использованием домашней электросети. Однако, многие из этих стандартов обладают существенными недостатками. Сети, созданные по упомянутому выше стандарту X-10, например, имеют крайне низкую скорость передачи команд ввиду своих конструктивных особенностей.

Объектом исследования в данной работе является так называемый Интернет вещей (англ. Internet of things) — концепция сети передачи данных между бытовыми устройствами, оснащенными теми или иными встроенными средствами приема-передачи информации. данная концепция была предложена в 1999 году Кевином Эштоном, основателем исследовательской группы Auto-ID Labs при Массачусетском технологическом институте. В 2004 году в журнале Scientific American опубликована статья «The Internet of Things» [1], подробно описывающая многие аспекты данной концепции. В этой статье представлен так называемый «Internet-0», который, по мнению авторов, больше подходит для использования в домашних сетях Интернета вещей.

Целью данной работы является разработка и создание работоспособного макета системы дистанционного управления источниками света, на основе которого в дальнейшем возможно создание полноценного продукта. Однако, так как результатом работы является макет системы, степень детализации узлов представляется возможным оставить низкой. Таким образом, данная работа представляет собой один из этапов жизненном цикле прдукта — этап проектирования. В частности, для достижения поставленных целей выполнены функциональный, конструкторский, схемотехнический и программно-алгоритмический этапы проектирования.

Практическая значимость данной работы заключается в том, что полученный в результате макет может послужить основой для развития полноценной системы управления освещением, пригодной для применения как в бытовых целях, так и на промышленных объектах. Основным недостатком существующих систем подобного типа является их стоимость. Нижняя граница рыночной стоимости системы управления умным домом в России составляет около 100 тыс. рублей, тогда как верхняя зачастую превышает 1 млн. рублей. Более того, многие продукты, присутствующие на рынке, проблематично установить своими руками для обычного человека, т.е. присутствует потребность в найме монтажной группы, что дополнительно повышает стоимость системы.

Основной причиной выбора именно системы управления ИС в качестве предмета исследования послужила простота разработки, позволяющая максимально абстрагироваться от технических особенностей конечных приборов и алгоритмов обмена данными, реализованных в них. Для достижения поставленной цели были сформулированы и решены следующие задачи:

1. Подобрать наиболее подходящую топологию сети и механизмы связи между её частями.
2. Разработать и реализовать электротехнические схемы конечных устройств.
3. Разработать ПО для устройства управления и встроенного ПО для конечных устройств.
4. Использовать в макете компоненты, имеющие оптимальное соотношение «цена-качество» и максимизировать количество пригодных для использования в полноценной версии продукта компонентов.

В процессе выполнения работы был произведен анализ отечественной и зарубежной литературы, построение модели разрабатываемой системы. Также на некоторых этапах работы осуществлялось абстрагирование от реальных физических параметров используемых компонентов.

Практическая часть. В начале разработки была создана функциональная схема системы. Согласно государственному стандарту 2.701-2008, функциональная схема является документом, разъясняющим процессы, протекающие в отдельных функциональных цепях или изделия в целом. [2]. С целью упрощения сборки, при создании демонстрационного макета количество конечных устройств сокращено до одного. На приведенной на рисунке 1 схеме изображена версия разрабатываемой системы с одним устройством и тремя конечными

устройствами.

Данная система может быть разделена на две основных составляющих: электротехническую и программную. Электротехническая часть включает в себя все разрабатываемые и используемые электротехнические компоненты, методы их соединения и физические параметры. Программная часть же включает в себя встраиваемое ПО для модулей ESP-01 и управляющую графическую оболочку для управляющего устройства.

Электротехническая часть. Для описания созданного демонстрационного макета необходимо две принципиальных электрических схемы — схема конечного устройства и схема маршрутизатора. Однако, так как маршрутизатор состоит из одного компонента, модуля ESP-01, подключенного к источнику питания, его схема не приводится.

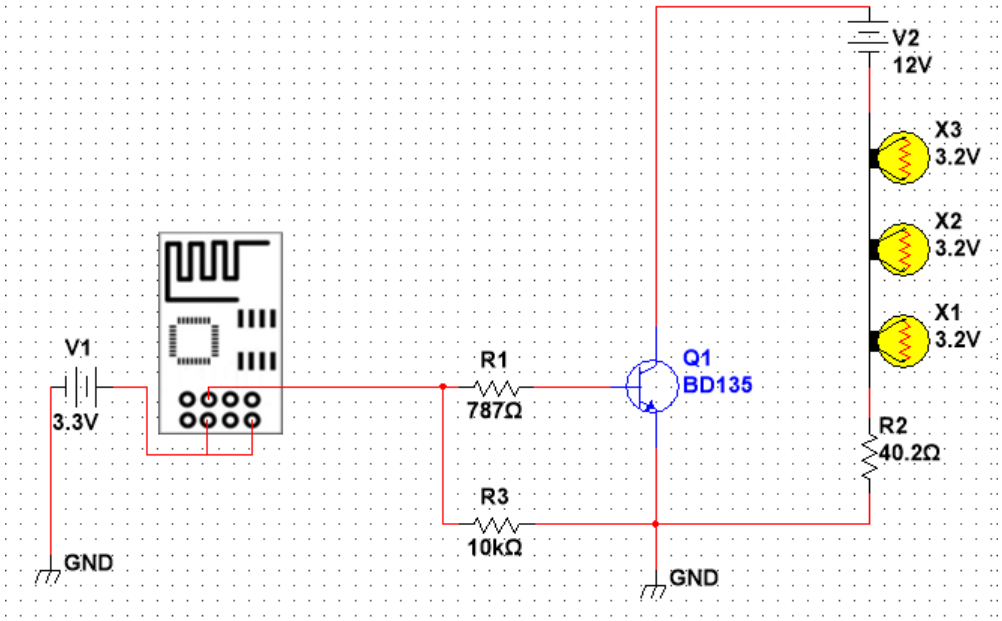

Рисунок 1 – Схема принципиальная электротехническая конечного устройства

В качестве ИС в макете используются три последовательно подключенных светодиода SMD2835, в качестве ключа же используется транзистор N-P-N типа КТ815А, включенный по схеме с общим эмиттером и работающий в режиме ключа. При использовании транзистора в режиме ключа, существуют два состояния, называемые состоянием отсечки и насыщения. В состоянии отсечки напряжение «база-эмиттер» U_{be} равно нулю [3], отсутствует и ток базы, что соответствует ключу в запертом состоянии. Противоположное состояние называется состоянием насыщения, при нём транзистор открыт полностью и сопротивление участка «коллектор-эмиттер» столь мало, что при включении

транзистора без нагрузки, он перегорает мгновенно.

В используемой схеме присутствуют два резистора — R1 и R2. Резистор R1 является базовым резистором транзистора Q1 и предназначен для подачи нужного тока на базу транзистора.

Приложение для устройства на базе Android. Управление системой производится с помощью устройства на базе операционной системы Android 11. Основная причина такого выбора - широкая распространенность устройств, использующих данную операционную систему, а также доступность средств разработки и отладки ПО. При этом стоит учитывать, что эти средства поддерживаются разработчиком ОС Android. Для разработки приложения был выбран язык C# с пакетом Xamarin Forms.

Как и многие другие программные пакеты для языка C#, пакет Xamarin Forms работает с использованием ООП. Каждая страница в решении является экземпляром того или иного класса, взаимодействующим с другими страницами, обработчиками данных и иными объектами.

При запуске приложения изначально пользователь видит страницу MainPage. Задача этой страницы в том, чтобы запрашивать имя пользователя и пароль для доступа к управлению системой. На этапе макета данный функционал реализован только частично — запрашиваемые данные сохраняются только локально в кеше приложения в формате текстового файла.

На данный момент функционал этой страницы состоит в следующем: при её появлении на экране выполняется метод OnAppearing(), реализующий следующий алгоритм:

1. Вызывается метод UpdateFileList(), получающий список файлов, находящихся в кеше приложения.
2. Содержимому страницы приравнивается StackLayout mainContent, содержащий все элементы управления, размещаемые на странице. Внешний вид страницы представлен на рисунке 5.
3. Кнопке tryLogIn присваивается обработчик события, работающий следующим образом:

   3.1 Создается словарь records, в который помещаются все записи о ранее зарегистрированных пользователях, получаемые с помощью метода processFile().

   3.2 В зависимости от того, присутствует ли в словаре введенное имя

пользователя и сопоставлено ли оно с введенным паролем, выбирается один из трех вариантов действий:

3.2.1 При наличии записи, в которой имя пользователя и пароль совпадают с введенными, на экран выводится экземпляр страницы DevicesPage, заранее объявленный в теле класса MainPage.

3.2.2 При наличии записи, в которой совпадает с введенным имя пользователя, но не совпадает пароль, выводится сообщение о некорректно введенном пароле.

3.2.3 При отсутствии записи, ключом которой является введенное имя пользователя, выводится сообщение с предложением зарегистрировать нового пользователя.

Листинг 1: Переопределенный метод OnAppearing() класса MainPage

```
protected override void OnAppearing()
{
        UpdateFileList();
        Content = mainContent;
        tryLogIn.Clicked += async (sender, args) =>
        {
                done = false;
                Dictionary<string, string> records = processFile("App4Data");
                if (records.TryGetValue(loginEntry.Text, out string value)){
                        if (value == passwordEntry.Text)
                        {
                        IReadOnlyList<Page> l = Navigation.NavigationStack;
                        Device.BeginInvokeOnMainThread(() => {
                                if (!done){
                                        Navigation.PushAsync(mp);
                                        done = true;
                                }
                        });
                        this.OnDisappearing();
                        }
                        else
                        {
```

```
                    await DisplayAlert("Предупреждение",
                    "Неверный пароль!", "Повторить попытку");
                }
            }
        else
        {
            bool res = await DisplayAlert("Предупреждение",
            "Пользователь с таким именем не найден!"+
            "Желаете зарегестрироваться?",
            "Да", "Нет");
            if (res)
            {
            object temp = new object();
            SaveFile("{loginEntry.Text}|{passwordEntry.Text}");
            }
            else
            {
            App.Current.Quit();
            }
        }
};
        }
```

MenuPage представляет собой экземпляр класса страницы навигации, на которой размещены три вкладки - «Подключенные устройства» (DevicesPage), «Настройки» и «О проекте», каждая из которых в свою очередь является страницей, содержащей контент. Сама страница MenuPage не отображается в приложении, однако её существование позволяет поддерживать навигацию между разделами.

DevicesPage - страница и одноименный с ней класс, наследуемый от класса ContentPage, на которой выводятся все доступные подключения при условии, что Android-устройство находится в предварительно заданной целевой WiFi-сети. Для обеспечения этого требования достаточно жестко зафиксировать в коде SSID WiFi-сети, в которой предполагается работа. Информация о подключении находится на странице SettingsPage.

Стандарт IEEE 802.11, WiFi, не позволяет напрямую узнать количество и IP-адреса узлов, находящихся в сети, а, следовательно, и напрямую обратиться к ним. В связи с этим был разработан следующий алгоритм (листинг 1):

1. При каждом появлении на экране страницы DevicesPage вызывается переопределенный метод OnAppearing, в котором создается экземпляр класса StackLayout, в который добавляется кнопка «Обновить список устройств» и еще один StackLayout, называющийся btnsl, предназначенный для хранения информации об устройствах в виде кнопок — экземпляров класса Button. btnsl инициируется списком известных при прошлом появлении страницы устройств, при первом появлении страницы этот список пуст.
2. При нажатии этой кнопки, в сеть отправляется широковещательный запрос с текстом foreignlanguageenglish«AQ», после чего приложение переходит в режим ожидания.
3. Конечные устройства при получении данного сообщения отправляют по адресу отправителя сообщение, содержащее три поля: имя устройства, его состояние («ВКЛ»/«ВЫКЛ»)и IP-адрес.
4. При получении ответного сообщения от конечного устройства, приложение создает кнопку, надпись на которой состоит из имени устройства и его состояния, после чего данная кнопка добавляется в btnsl.

   При нажатии на кнопку, по соответствующему IP-адресу отправляется пакет с текстом «inv». Действия конечного устройства описаны в разделе 2.2.2
5. Предыдущий пункт повторяется до тех пор, пока не будут обработаны все входящие сообщения от конечных устройств.

На рисунке 6 изображен внешний вид страницы DevicesPage после того, как была нажата кнопка «Обновить список устройств» при условии, что в сети находится только одно конечное устройство с именем First.

Листинг 2: Переопределенный метод OnAppearing() класса DevicesPage

```
protected override void OnAppearing()
{
  StackLayout sl = new StackLayout(); //StackLayout для содержимого
  Label header = new Label
  {
    Text = "Устройства в данной сети",
```

```
  FontSize = Device.GetNamedSize(NamedSize.Large, typeof(Label)),
  HorizontalOptions = LayoutOptions.Center
};
Button b1 = new Button { Text = "Обновить список устройств" };
sl.Children.Add(header);
sl.Children.Add(b1);
StackLayout btnsl = savedData;//StackLayout для кнопок-устройств.
//Копирует предыдущее состояние
this.Content = sl;
sl.Children.Add(btnsl);
b1.Clicked += (sender, args) => //Задаем обработчик события
{
    btnsl.Children.Clear();
    byte[] recArray = new byte[64];
    ConnectionData conn = new ConnectionData();
    sl.Children.Add(header);
    var data = Encoding.UTF8.GetBytes("ABCD");
    conn.udpClient.Send(data, data.Length, "255.255.255.255", 8888);
    while (conn.udpClient.Available != 0)
    {
      var from = new IPEndPoint(IPAddress.Any, 0);
      byte[] recvBuffer = conn.udpClient.ReceiveAsync().Result.Buffer;
      try
      {
        string[] tmp = splitPack(recvBuffer);
        tmp[1] = tmp[1] == "0" ? "Выкл" : "Вкл";
        Button b = new Button { Text = tmp[0] + " " + tmp[1] };
        b.Clicked += (sender, args) =>
        {
          byte[] data1 = Encoding.UTF8.GetBytes("inv");
          byte[]adrB = Encoding.UTF8.GetBytes(tmp[2]);
          //IPAddress adr = new IPAddress(adrB);
          conn.udpClient.Send(data1, data1.Length, tmp[2], 8888);
          byte[] recvBuffer2 = conn.udpClient.ReceiveAsync().Result.Buffer;
```

```
            string[] tmp2 = splitPack(recvBuffer2);
            tmp2[1] = tmp2[1] == "0" ? "Выкл" : "Вкл";
            b.Text = tmp2[0] + " " + tmp2[1];
          };
          btnsl.Children.Add(b);
          int c1 = 100000;
          while (c1 > 0){c1--;}
        }
        catch (Exception e) { }
        int c = 1000000000;
        while (c > 0){c--}
      }
  };
  savedData = btnsl;
}
```

Прошивки модулей ESP-01. Микроконтроллеры ESP8266, в отличие от процессоров Android-устройств, не способны работать с языком C#. Языком, используемым для разработки ПО, предназначенного для данных микроконтроллеров, является C. C.

В программном обеспечении для ESP-01, выступающего в роли точки доступа реализованы четыре обработчика событий:

1. Обработчик подключения устройства к сети.
2. Обработчик отключения устройства от сети.
3. Обработчик запросов поиска сети.
4. Обработчик запросов поиска сети, отвечающий за мигание встроенного светодиода в течение первых 10 секунд после запуска устройства.

Отдельного рассмотрения требует программный код, предназначенный для микроконтроллера в составе конечного устройства. Метод $\mathrm{loop}()$ представляет собой бесконечный цикл, выполняемый модулем ESP-01, предварительно настроенным в качестве абонента сети WiFi. В процессе выполнения данного метода, представленного в листинге 3, в отладочной конфигурации макета происходит следующее:

1. Выставляется нужное состояние вывода микроконтроллера, подключенного к транзистору.

2. Если переменная, отвечающая за состояние конечного устройства не равна нулю, что соответствует состоянию «Включено», сообщение об этом выводится по SPI на компьютер, где считывается строенным в среду разработки прослушивателем SPI-порта.
3. В случае, когда микроконтроллер имеет необработанное сообщение, проверяется его содержимое.

   Если в поле данных полученного сообщения находится код «AQ», то по SPI на компьютер передается сообщение «BROADCAST PACKAGE». В ином случае, микроконтроллер изменяет значение переменной status на противоположное и отправляет по SPI пакет с текстом «command».

   Вне зависимости от типа сообщения, микроконтроллер отправляет на управляющее устройство пакет, содержащий три поля: имя устройства, значение переменной status и свой IP-адрес.

Заключение. Данная работа нацелена на разработку принципов построения и создание макета такой системы с использованием беспроводных технологий. Теоретическое исследование показало, что наиболее подходящей топологией для такой системы является топология «звезда». В качестве протокола обмена сообщениями между устройствами был использован протокол UDP, ввиду его простоты, влекущей за собой низкую потребность в вычислительных мощностях. В соответствии с поставленными задачами, в процессе исследования был разработан полностью функционирующий макет системы дистанционного управления источниками света. Для этого были разработаны:

— Функциональная схема разрабатываемой системы.
— Электрические принципиальные схемы для маршрутизатора и конечных устройств.
— Программное обеспечение для конечных и устройств, управляющего устройства и маршрутизатора.
— Макет разрабатываемой системы

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Kevin Ashton. That ‘Internet of Things’ Thing. In the real world, things matter more than ideas. (англ.). RFID Journal (22 июня 2009).

2 ГОСТ 2.701-2008. ЕСКД. Схемы. Виды и типы. Общие требования к выполнению, 25.12. 2008

3 Физические основы электроники: метод. указания к лабораторным работам / сост. В. К. Усольцев. — Владивосток: Изд-во ДВГТУ, 2007. — 50 с.:ил.