Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ
«МИФИ»

ИНСТИТУТ ЛАЗЕРНЫХ И ПЛАЗМЕННЫХ ТЕХНОЛОГИЙ
КАФЕДРА №31 ПРИКЛАДНАЯ МАТЕМАТИКА

Чернявский Андрей Дмитриевич

Отчет

по проектной практике на тему:

ТЕМА ПРОЕКТА

Разработка бинарных часов с индикацией, имеющих синхронизацию времени по сети

Руководитель проекта

_____ М.А. Чмыхов

«___» _____ 2021 г.

Москва, 2021 г.

Содержание

Аннотация

Цель проекта – описать процесс создания прототипа часов на платформе Arduino с бинарной индикацией и синхронизацией по сети.

Для достижения цели проекта ставим следующие задачи:

- Уяснить состав аппаратной части электронного конструктора Arduino;
- Определить способ представления информации о времени на панели индикации;
- Определить способ синхронизации часов;
- Определить компоненты, с использованием которых можно было бы реализовать часы с синхронизацией;
- Составить принципиальную схему;
- Подобрать (разработать) программный код для работы компонентов схемы.

1 Используемые инструменты

1.1 История создания Arduino

Arduino представляет собой комбинацию аппаратной и программной частей для простой разработки электроники.

Arduino создавалось преподавателями для обучения школьников и студентов электротехнике, программированию, радиоэлектронике, системам автоматизации. Идея имела успех и проект пошел дальше. Благодаря открытой архитектуре любой желающий может производить микроконтроллеры, дополнять модельный ряд, писать программы. Все схемы и исходный код программ есть в открытом доступе.

После того как Arduino получило более широкое распространение, многие производители электроники начали выпускать собственные платы на базе микроконтроллеров Arduino.

Название конструктора отсылает нас к средневековому королю Италии Ардуину, поскольку начинала проект команда из пяти итальянцев.

1.2 Возможности Arduino

С технической точки зрения, Arduino умеет принимать и отправлять сигналы в соответствии с инструкциями в прошивке. Это позволяет получать и обрабатывать информацию с сенсоров и передавать команды исполнительным механизмам или другим устройствам. Например, микроконтроллер может получать данные с датчиков температуры, давления, влажности и выводить сводную информацию на дисплей и управлять светом, моторами, приводами.

В сети часто встречаются проекты умного дома на базе данного конструктора.

1.3 Аппаратная часть Arduino

Аппаратная часть включает в себя большое количество видов плат Arduino со встроенными программируемыми микроконтроллерами, а также дополнительные модули.

Классические платы спроектированы для монтажа в стопки через штыревые разъёмы, то есть без пайки контактов. Порты ввода-вывода микроконтроллеров оформлены в виде штыревых линеек. Это главный плюс для начинающих - не нужно использовать паяльник и ничего паять. На платах сделаны удобные контакты, которые можно соединять удобными перемычками с любыми сторонними модулями, дисплеями, сенсорами и др.

Микроконтроллеры отличаются наличием предварительно прошитого в них загрузчика. С помощью этого загрузчика пользователь загружает свою программу в микроконтроллер без использования традиционных отдельных аппаратных программаторов. Загрузчик соединяется с компьютером через интерфейс USB или с помощью отдельного переходника **UART-USB**.

Существует целая линейка микроконтроллеров. Поскольку наша задача не является сложной в вычислительном плане, мы воспользуемся недорогой платой **Arduino Nano v3.0**.

1.4 Программная часть Arduino

Программная часть состоит из бесплатной оболочки (IDE – Integrated development environment). В этой среде удобно писать скетчи и загружать их на микроконтроллер. В среде разработки уже предустановлено большое количество примеров и дополнительных библиотек.

Язык программирования называется Arduino C и представляет собой язык C++.

Среда разработки включает текстовый редактор, транслятор, средства автоматизации сборки, отладчик.

2 Принципиальная схема часов с синхронизацией по сети

2.1 Подход к решению задачи

Для реализации часов с синхронизацией нам необходимы следующие ключевые компоненты:

- Генератор импульсов с более-менее приемлемой точностью, к примеру, +/- 1 секунда в час, то есть в относительном выражении 10^{-4};
- Способ синхронизации, в качестве которого мы выбираем сигнал навигационной системы;
- Устройство отображения, представляющее каждый разряд (часы, минуты, секунды) в двоичном виде, то есть имеющее минимум два состояния;
- Схема управления.

2.2 Генератор импульсов

В качестве генератора импульсов в типовых решениях используется кварцевый генератор, отличающийся приемлемой стабильностью тактовых колебаний, особенно при условии стабилизации температуры резонатора.

2.3 Синхронизация с сервером точного времени

Стабильность кварцевого резонатора для использования в часах не является достаточной. Период собственных колебаний кварца зависит от ряда факторов, прежде всего, от температуры резонатора. Поэтому часам требуется периодическая синхронизация.

При синхронизации времени по сети мы получаем данные точного времени не непосредственно от спутника, а от одного из многочисленных серверов точного времени (NTP-server).

Многие службы имеют потребность в доступе к сигналам точного времени , в частности, сервис «умный дом».

При этом запрашивается текущее время на сервере и используют его для установки своих собственных часов.

Существует несколько уровней NTP серверов, где первый уровень подключен к атомным часам, а второй и третий уровни серверов распределяют на себя нагрузку по актуальным запросам из интернета.

Передача информации о времени осуществляется по протоколу сетевого времени (NTP) - это стандарт для синхронизации часов в компьютерных системах через сети связи на основе пакетов. Протокол был специально разработан для обеспечения надежной информации о времени через сети с переменным временем прохождения пакетов.

2.4 Индикация

При двоичном представлении используются только два состояния индикатора – 0 (ноль) и 1 (единица). В качестве устройство отображения используем светодиоды.

Мы будем представлять дату не в полностью двоичном формате, а в более привычном для нас двоично-десятичном, когда каждый разряд представляется своей колонкой светодиодов.

Всего мы насчитали, что необходимо 20 светодиодов:

- необходимо 6 (шесть) колонок светодиодов – по две колонки на единицы и десятки часов, минут и секунд;
- для каждого младшего разряда для 10 возможных состояний достаточно будет 4 индикаторов, итого 3 x 4 = 12 штук;
- для старших разрядов секунд и минут с максимальным числом состояний 6 берем по 3 индикатора (2 x 3 = 6), для старшего разряда часов – 2 индикатора (три состояния).

В приведенном примере включенный светодиод означает единицу, выключенный – ноль. Каждая горизонтальная линия отображает степень двойки снизу-вверх, от нулевой до третьей. Поэтому при чтении информации с часов надо просуммировать значения колонок с включенными светодиодами.

В примере на рис. 1 часы отображают время 23:59:17 в 24-часовом формате.

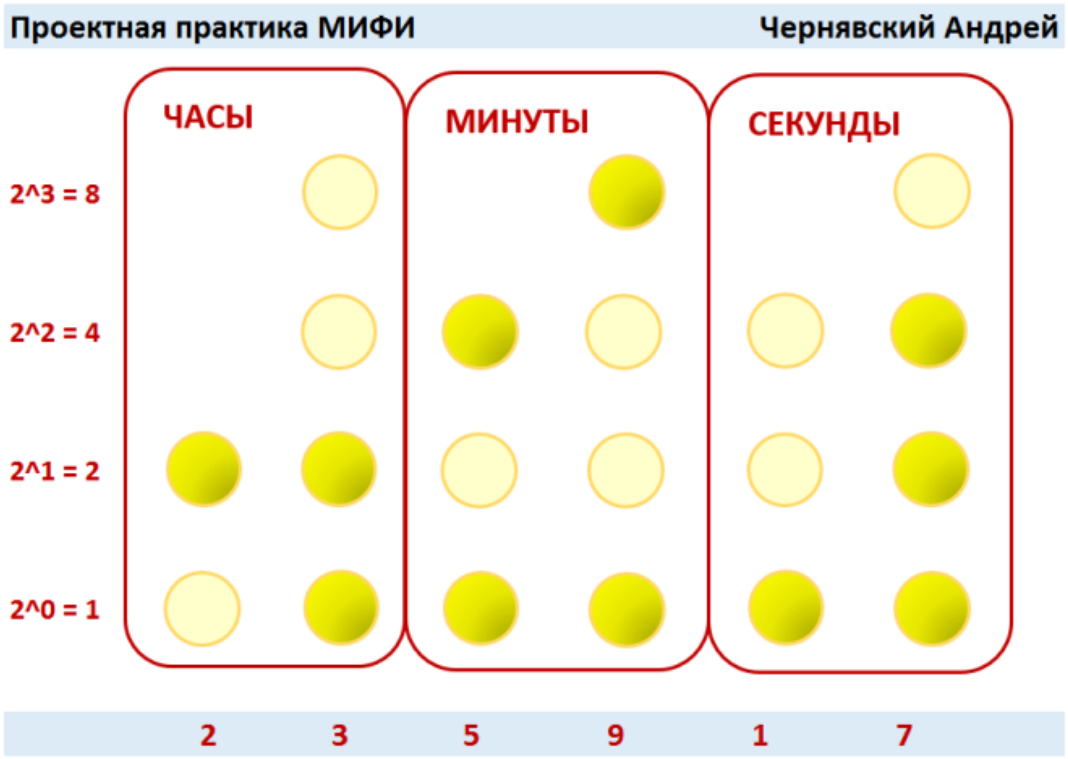

Рис. 1: Схема индикации

3 Реализация проекта

3.1 Генератор импульсов

В качестве источника импульсов используем популярную микросхему DS1307 от Dallas Semiconductors. Точность часов зависит от точности кварцевого резонатора и точности соответствия между ёмкостной нагрузкой схемы тактового генератора и внутренней ёмкостью кварцевого резонатора. Дополнительная погрешность будет вноситься дрейфом частоты кварцевого резонатора, происходящим из-за температурных перепадов. Вообще говоря, данный модуль считается не считается точным – есть и более совершенные модели, в которых сильно снижены температурные колебания за счет термостата.

Как указано в описании модуля, «Часы реального времени с последовательным интерфейсом DS1307 – это малопотребляющие полные двоично-десятичные часы-календарь, включающие 56 байтов энергонезависимой статической ОЗУ. Адреса и данные передаются последовательно по двухпроводной двунаправленной шине. Часы-календарь отсчитывают секунды, минуты, часы, день, дату, месяц и год. Последняя дата месяца автоматически корректируется для месяцев с количеством дней

меньше 31, включая коррекцию високосного года. Часы работают как в 24-часовом, так и в 12-часовом режимах с индикатором AM/PM»

Литиевая батарея емкостью 4,8 А/ч способна поддерживать модуль в течение 10 лет (при комнатной температуре).

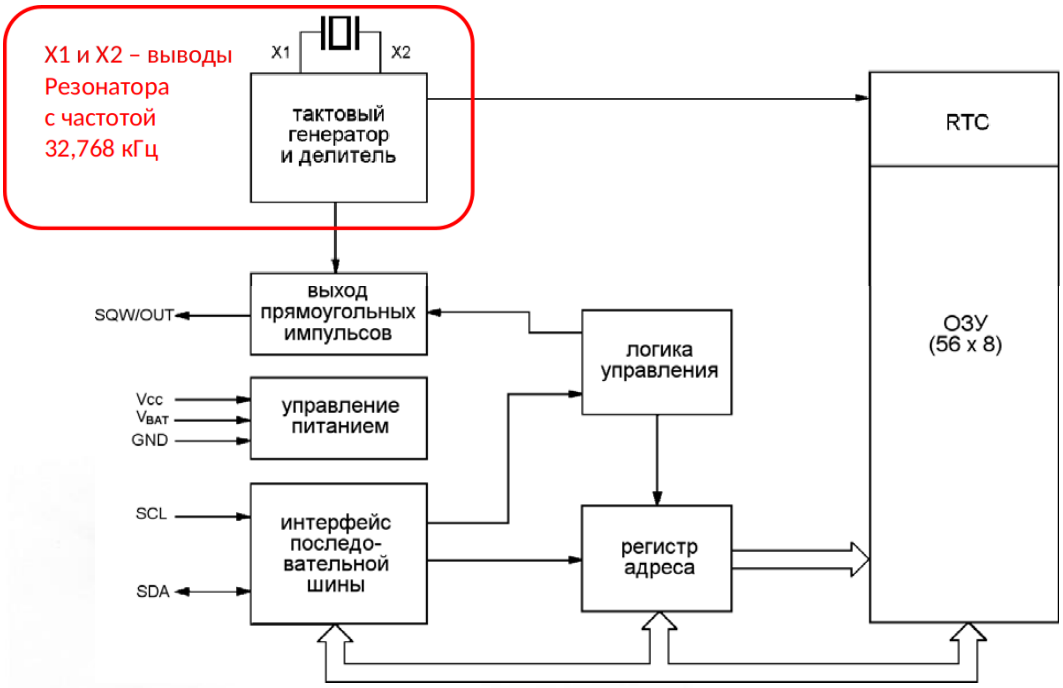

Рис. 2: Схема часов реального времени RTC DS1307

Выше приведена блок-схема модуля, в нашей задаче нас интересуют:

- VCC и GND – питание +5В и земля соответственно;
- SCL – Serial Clock Input - вход последовательных синхроимпульсов) – используется для синхронизации данных по последовательному интерфейсу;
- SDA – Serial Data Input/Output serial - вход/выход последовательных данных) – вывод входа/выхода для двухпроводного последовательного интерфейса

3.2 WiFi модуль

Для подключения к сети Интернет мы будем использовать беспроводное соединение WiFi.

В качестве модуля связи выбираем популярный компактный и недорогой модуль ESP8266.

Модуль создан для использования в умных розетках, mesh-сетях, IP-камерах, беспроводных сенсорах, носимой электронике и так далее.

Рис. 3: Схема работы NTP сервера

Модуль имеет металлический экран, защищающий микросхемы от внешних наводок, тем самым обеспечивает более стабильную работу, свою керамическую антенну и разъём для внешней антенны.

Модуль поддерживает Wi-Fi протоколы 802.11 b/g/n с WEP, WPA, WPA2.

3.3 Индикаторы на светодиодах

Для панели индикации можно использовать обычные китайские светодиоды на 3 V.

Подключать светодиоды необходимо через резистор сопротивлением около 200 Ом, поскольку светодиод обладает малым внутренним сопротивлением и при прямом подключении может перегореть.

3.4 Принципиальная схема

Принципиальная схема отражает схему соединения основных компонентов.

Для черчения под Arduino удобно использовать бесплатную программу (среду разработки) Fritzing. С использованием данной программы можно сформировать макет реальной печатной платы. Программа включает набор готовых компонентов, в частности, макетные и монтажные платы (в том числе Arduino), целый набор ана-

логовых и цифровых микросхем, любые радиодетали: конденсаторы, транзисторы, резисторы, светодиоды, батарейки, кнопки.

Схема доступна для рисования, как в окне «Макетная плата», так и в окне «Принципиальная схема» простым перетаскиванием нужных компонентов на рабочее поле. При выборе окна «Печатная плата» можно приступить к разводке проводников и размещению элементов.

Результат работы экспортируется в pdf-файл или jpeg.

3.5 Макетная (монтажная) плата

Как говорилось, удобство Arduino как обучающего конструктора состоит в использовании типовых элементов, соединяемых между собой без пайки.

Концепция беспаечной макетной платы позволяет многократно использовать одни и те же компоненты обучающего конструктора.

Однако, если мы хотим долгое время использовать устройство, то нам необходимо пропаивать контакты во избежание их окисления. Тогда следует воспользоваться обычной платой с пайкой.

Лицевая сторона макетной платы Обратная сторона (видны соединения и цветные шины питания по бокам)

3.6 Стоимость компонентов

Можно обозначить стоимость компонентов: основная плата Nano стоит около 500 рублей, часы RTC - 100 рублей, приемник GPS – 500 рублей, макетная плата – 100 рублей, светодиоды - по 10 рублей за штуку плюс прочее. Всего набор не дороже 2-х тысяч рублей с возможностью неоднократного использования всех компонентов.

4 3D моделирование в Fusion 360

В ходе реализации проекта была разработана 3D модель часов с использованием программы Fusion 360.

При этом для наглядности компоненты часов разделены на 2 блока:

- Блок управления, включающий плату Arduino, часы реального времени RTC и модуль WiFi для синхронизации через Интернет;

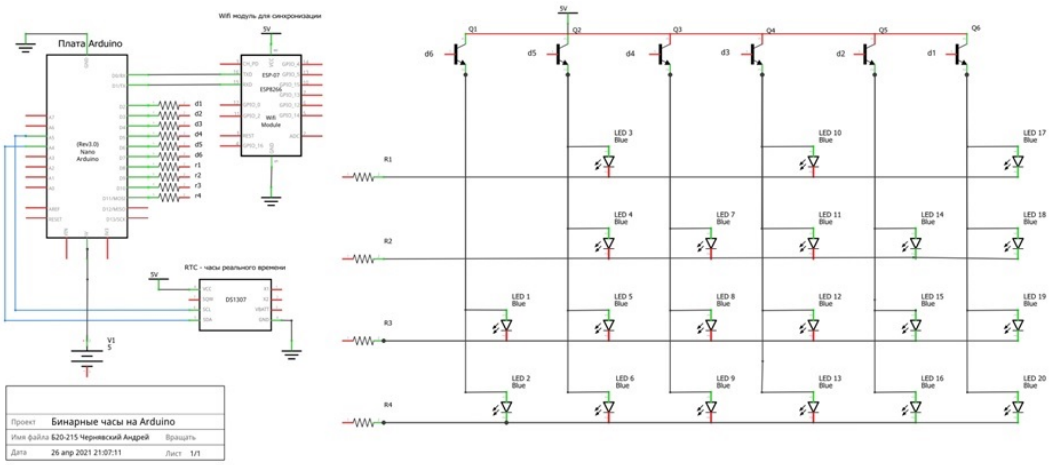

Рис. 4: Принципиальная схема

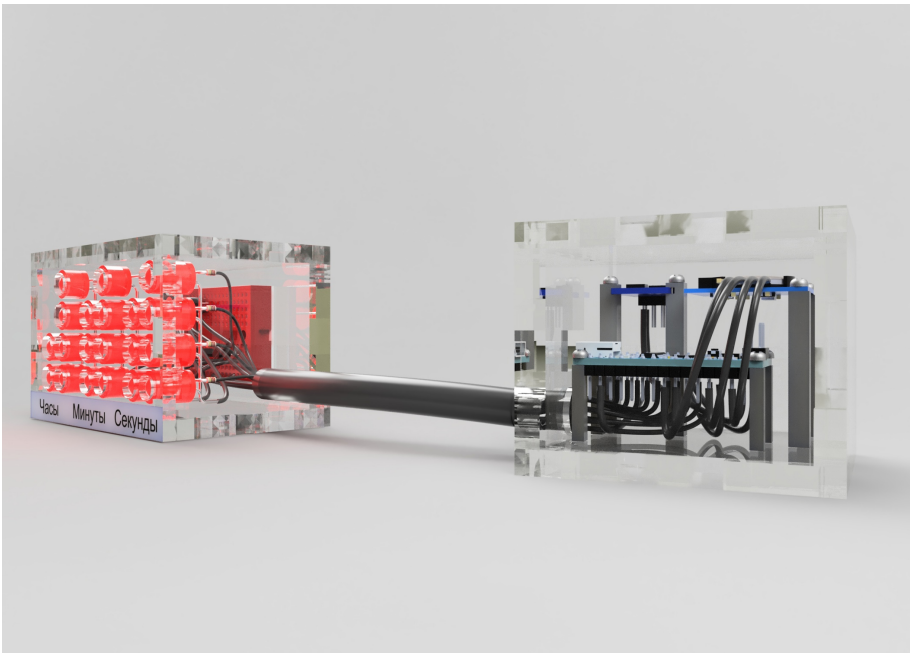

Рис. 5: 3D модель бинарных часов

- Панель индикации, включающая макетную плату и 20 светодиодов;

5 Программирование

5.1 Общие положения

Программирование микроконтроллеров Arduino осуществляется на языке программирования C++. Этот язык является низкоуровневым, поэтому считается сложным и имеет высокий порог вхождения. Но для программирования Arduino используется упрощенная версия C++.

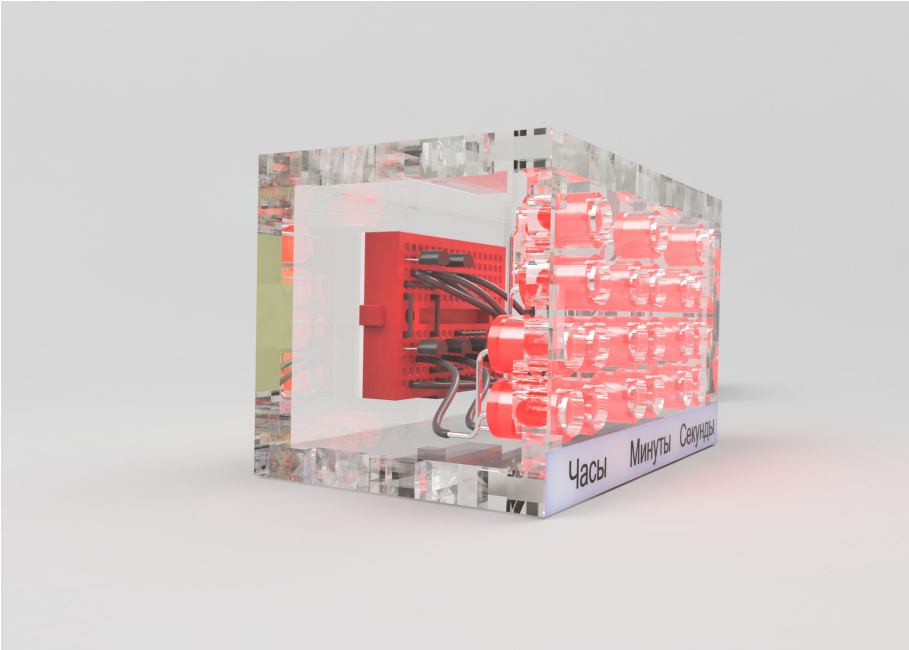

Рис. 6: 3D модель панели индикации

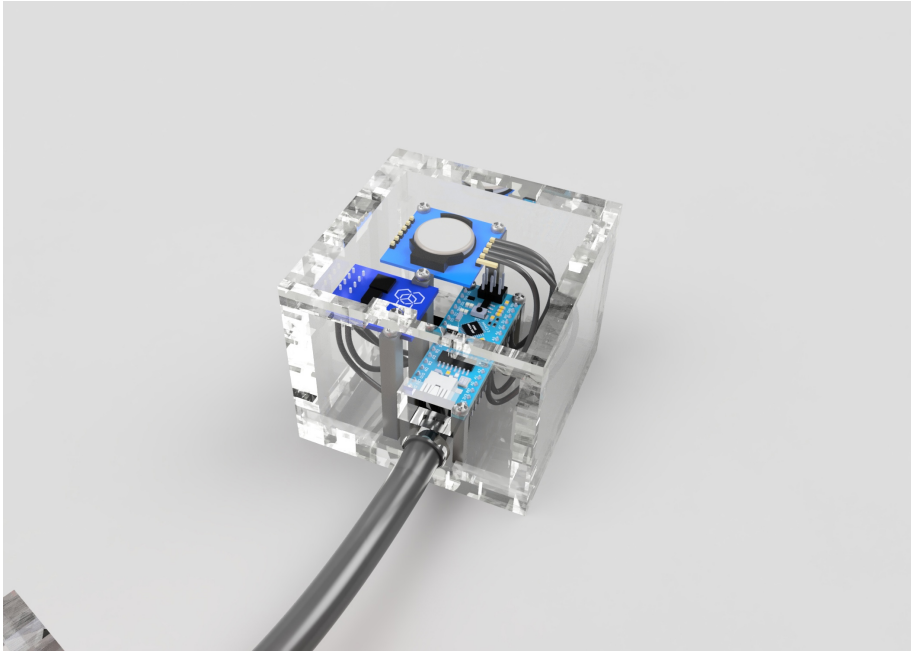

Рис. 7: 3D модель блока управления

bynary_clock.ino | GPS | ReadMe.adoc | RTC_1307

```
: Author: andrey_chernyavsky
: Email: {
  AuthorEmail
}
: Date: 04 / 01 / 2021
: Revision: version#
: License: Public Domain

#include <Time.h>                    // Подключаем библиотеки для работы
#include <Wire.h>
#include <DS1307RTC.h>
void setup()  {
  setTime(12, 00, 0, 01, 1, 2021);  /* Устанавливаем первоначальное время в формате: Часы, минуты, секунды,
                                       день, месяц, год – 12:00 1-го января 2021 года */

  setSyncProvider(RTC.get);          /* Метод get() читает RTC и возвращает полученную дату в POSIX формате */

  RTC.set(now());                    /* метод set() - записывает дату в RTC принимая в качестве параметра дату
                                       в POSIX формате */
}
void loop()
{
  delay(100);                        // ставим задержку внутри цикла в 100 миллисекунд
}

```

Рис. 8: Код RTC

Также для упрощения разработки прошивок существует множество функций, классов, методов и библиотек, благодаря чему работать с этими микроконтроллерами достаточно удобно.

5.2 Обработка сигналов часов реального времени

Для работы с модулем часов реального времени нам потребуются библиотеки DS1307RTS Time. Наша несложная задача заключается в получении информации о времени из модуля RTC.

5.3 Получение информации о точнм времени с NTP сервера

Для получения информации о времени с сервера NTP следует воспользоваться ESP8266WiFi.h.

5.4 Формирование сигналов для бинарной индикации

Последняя задача будет посложнее предыдущих, потому что потребуется преобразование десятичного представления в двоично-десятичное, то есть каждое десятичное число требуется преобразовать в двоичное для представления 4-мя индикаторами (светодиодами).

bynary_clock.ino | DS1307 | GPS | GPS_synch | RTC_1307 | WiFi

```
#include <NTPClient.h>
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
const char *ssid     = "YOUR_SSID";
const char *password = "YOUR_PASS";
const long utcOffsetInSeconds = 3600;
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};
// Определение NTP-клиента для получения времени
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", utcOffsetInSeconds);
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while ( WiFi.status() != WL_CONNECTED )  {
    delay ( 500 );
    Serial.print ( "." );
  }
  timeClient.begin();
}
void loop()
{
```

Рис. 9: Код GPS

5.5 Ссылка на репозиторий GitHub

Программный код, использованный при выполнении проекта, размещен в репозитории GitHub по адресу: `https://github.com/Andrey621/clock_with_binary_indication`

6 Заключение

В результате исполнения проекта были решены следующие задачи:

- Получено представление о составе аппаратных средств конструктора Arduino;
- Уяснен способ компоновки электронных элементов с использованием макетной платы Arduino;
- Уяснен порядок взаимодействия компонентов с использованием языка Arduino C.
- созданы варианты бинарных часов с индикацией и синхронизацией времени по GPS, а также по сети Интернет (через NTP сервер);
- разработана принципиальная схема для обоих вариантов синхронизации;
- уяснен состав аппаратно-программных средств конструктора Arduino;
- подобраны Arduino-совместимые компоненты для реализации проекта в обоих вариантах;

bynary_clock.ino | DS1307 | GPS | GPS_synch | RTC_1307

```
// подключение библиотек
#include <Wire.h>
#include "RTClib.h"
#include <TimerOne.h>

int temp,inc,hours1,minut,add=11; // определение типа переменных
/* присваиваем номерным выводам платы имя константы dx, компилятор заменит имя dx на номер вывода */
#define d1 12
#define d2 11
#define d3 10
#define d4 9
#define d5 8
#define d6 7
#define r1 6
#define r2 5
#define r3 4
#define r4 3
int HOUR,MINUT,SECOND; //переменные для хранения значения часов, минут и секунд
void callback()
{
  digitalWrite(13, digitalRead(13) ^ 1);
  count++; // инкремент, увеличение значения счетчика на единицу
  if(count>=7) // значение старшего разряда в секундах не может превышать 6
  count=1;
  switch(count%7) { // остаток от деления счетчика на 7
    case 1:
    Clear(d1);
    temp=SECOND%10; // получаем цифру младшего разряда секунд
    show(temp);
    digitalWrite(d1, LOW);
    break;
    case 2:
    Clear(d2);
    temp=SECOND/10; // получаем цифру старшего разряда секунд
```

Рис. 10: Код индикации, часть 1

```
      show(temp);
      digitalWrite(d2, LOW);
      for (int i = 0; i < 10000; i++)
      {
      }
      break;
    case 3:
      Clear(d3);
      temp = MINUT % 10; // получаем цифру младшего разряда минут
      show(temp);
      digitalWrite(d3, LOW);
      for (int i = 0; i < 10000; i++)
      {
      }
      break;
    case 4:
      Clear(d4);
      temp = MINUT / 10; // получаем цифру старшего разряда минут
      show(temp);
      digitalWrite(d4, LOW);
      for (int i = 0; i < 10000; i++)
      {
      }
      break;
    case 5:
      Clear(d5);
      temp = HOUR % 10;
      show(temp);
      digitalWrite(d5, LOW);
      for (int i = 0; i < 10000; i++)
      {
      }
      break;
```

Рис. 11: Код индикации, часть 2

```
    case 6:
      Clear(d6);
      temp = HOUR / 10;
      show(temp);
      digitalWrite(d6, LOW);
      for (int i = 0; i < 10000; i++)
      {
      }
      break;
  }
}
/* функция для вывода информации на 4 светодиода одного десятичного разряда с использованием побитового сдвига вправо */
void show(int d)
{
  for (int i = 0; i < 1; i++)
  {
    digitalWrite(r4, !((temp >> 0) & 1));
    digitalWrite(r3, !((temp >> 1) & 1));
    digitalWrite(r2, !((temp >> 2) & 1));
    digitalWrite(r1, !((temp >> 3) & 1));
    // delay(1);
    for (int i = 0; i < 1000; i++);
  }
}
```

Рис. 12: Код индикации, часть 3

- отработана программная часть:

  получение данных с платы с платы часов реального времени RTC;

  синхронизация времени по данным сети GPS или одного из NTP серверов с подключением к сети Интернет с использованием WiFi модуля;

  управление работой светодиодов для двоично-десятичной индикации времени;

- разработана объемная модель часов с использованием программы Fusion 360;

- создан 10-секундный ролик, отражающий работу часов (смену состояний светодиодов) с 23:59:56 по 00:00:05

Список литературы

При работе над проектом были использованы следующие источники информации:

1. Виктор Петин. Проекты с использованием контроллера Arduino. СПб: БХВ-Петербург, 2015.
2. Улли Соммер. Программирование микроконтроллерных плат Arduino Freeduino. СПб: БХВ-Петербург, 2012.
3. Марк Геддес. 25 крутых проектов с Arduino. Москва: Эксмо, 2019.
4. Виктор Петин. 77 проектов для Arduino. Litres, 2019.
5. Блум Джереми. Изучаем Arduino: инструменты и методы технического волшебства. СПб: БХВ-Петербург, 2015
6. Саймон Монк. Программируем Arduino. СПб: Питер, 2017
7. Салахова А. А., Феоктистова О. А., Александрова Н. А., Храмова М. В. Arduino. Полный учебный курс. От игры к инженерному проекту. Лаборатория знаний, 2020
8. Лидия Слоун Клайн. Make: Fusion 360. 3D моделирование для мейкеров. BHV-СПб, 2021