Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных средств

ОТЧЁТ ПО ПРЕДДИПЛОМНОЙ ПРАКТИКЕ

на тему

«Электронная книга на базе ESP32»

|  |  |
| --- | --- |
| Выполнил: | Руководитель практики от |
| ст. гр. 850701 | университета: |
| Каленик С.А. | Порхун М.И. |
|  |  |
|  | Руководитель практики от |
|  | предприятия: |
|  | Никитин А.В. |

Минск 2022

CОДЕРЖАНИЕ

[Введение. постановка задачи 3](#_Toc75713398)

[1 СТРУКТУРА И ИСТОРИЯ ПРЕДПРИЯТИЯ 4](#_Toc75713399)

[2 Анализ задачи. Функциональная спецификация системы 7](#_Toc75713400)

[3 Предварительное проектирование системы 9](#_Toc75713401)

[3.1 Разбиение системы на модули 9](#_Toc75713402)

[3.2 Выбор соотношения между аппаратными программными средствами 11](#_Toc75713403)

[3.3 Построение структурной схемы аппаратной части системы 11](#_Toc75713404)

[3.4 Описание структурной схемы 12](#_Toc75713405)

[4 Проектирование аппаратных средств системы 13](#_Toc75713406)

[4.1 Выбор типа микроконтроллера 13](#_Toc75713407)

[4.2 Разработка принципиальной схемы системы 14](#_Toc75713408)

[4.2.1 Управление E-ink дисплеем с помощью ESP32 14](#_Toc75713409)

[4.3 Описание работы системы по принципиальной схеме 15](#_Toc75713410)

[5 Проектирование программного обеспечения 16](#_Toc75713411)

[5.1 Разработка схемы алгоритма работы системы и программы 16](#_Toc75713412)

[5.2 Описание алгоритма работы системы и программы 16](#_Toc75713413)

[6 Аппаратная реализация устройства 18](#_Toc75713414)

[6.1 Макетирование устройства 18](#_Toc75713415)

[6.2 Тестирование устройства 18](#_Toc75713416)

[Заключение 20](#_Toc75713417)

[Список использованных источников 21](#_Toc75713418)

[ПРИЛОЖЕНИЕ А (Обязательное) Схема электрическая структурная 22](#_Toc75713419)

[ПРИЛОЖЕНИЕ Б (Обязательное) Схема электрическая принципиальная 24](#_Toc75713420)

[ПРИЛОЖЕНИЕ В (Обязательное) Блок-схема алгоритма 26](#_Toc75713421)

[ПРИЛОЖЕНИЕ Г (Обязательное) Код программы 28](#_Toc75713422)

Введение

Электронная книга – это устройство, использующее экран на электронных чернилах для отображения текста. В отличие от обычных матричных экранов, текст не пропадает при подаче питания, а также не требуется свет для отображения. Благодаря этому уменьшается нагрузка на глаза и читателю приятней использовать электронную книгу, чем, например, телефон. Еще одной отличительной чертой данного типа устройств является низкое энергопотребление, поскольку электричество тратиться лишь на смену текста на экране, в отличие от обычных матричных экранов, в которых включенный экран постоянно потребляет много энергии [[1]](#wikiEreader).

В проекте рассматривается задача реализации электронной книги на базе модуля ESP32. Приведена структурная схема устройства и её описание, а также обобщённый алгоритм функционирования устройства. Проверка работоспособности устройства выполнена опытным путём.

# СТРУКТУРА И ИСТОРИЯ ПРЕДПРИЯТИЯ

ООО "ЭгоСекьюр" – отечественная компания-разработчик ПО. Специализируется на разработке комплексных программных решений категорий Endpoint Protection Platforms (EPP) и Mobile Device Management (MDM). EPP/MDM-решения, разрабатываемые компанией, используются для управления и защиты конечных (в т.ч. мобильных) устройств в составе корпоративных информационных сетей.

Решения EgoSecure ориентированы на сегмент малого и среднего бизнеса, обеспечивают контроль конечных устройств – внешних носителей данных, устройств связи, принтеров, оптических приводов и пр.; управление приложениями, шифрование и фильтрацию данных по типу файлов, а также защиту информации на мобильных устройствах и носителях. Все решения работают под единой консолью управления, что дает возможность централизованно управлять политиками безопасности.

Ниже перечислена часть возможностей осинового продукта компании:

* Запрет выгрузки и блокировки программы.
* Смена прав в автономном режиме (генерация кода доступа).
* Запись движения файлов, позволяющая выполнять законодательные и внутрикорпоративные требования.
* Синхронизация с ActiveDirectory и Novell eDirectory.
* Просмотр и экспорт отчетов, изменение логов: информации о разрешениях, доступе к устройствам, передаче данных, заблокированных попытках доступа и т.д.
* Протоколирование изменений и всех потоков данных.
* Создание списка разрешенных устройств.
* Поддержка пользователей без доступа к сети.
* Централизованное сканирование каждого устройства.
* Управления запросами пользователей.
* Настройка сообщений для пользователей.
* Применение режима «только для чтения».
* Установление различных прав доступа в зависимости от дня недели и времени суток.
* Создание MSI-пакета для установки агента.
* Интеграция с различными директориями, позволяющая избегать дублирования списка контроля доступа пользователей и групп пользователей.
* Ограничение переноса данных «белыми» и «черными» списками.
* Защита одним или несколькими паролями доступа к подробным логам с опциями фильтрации и сортировки.
* Интеграция со сторонними программами посредством XML-интерфейса.
* Поддержка распределенных сред

Компания основана в 2005 под именем cynapspro. Основной продукт компании был достаточно популярен в Германии, Австрии и Швейцарии.

В декабре 2011 года в InfoWatch приобрели Cynapspro и приступили к адаптации продукта для российского рынка. Для выпуска на российский рынок систему пришлось доработать — перевести интерфейс и документацию, а также исключить модуль полного шифрования дисков, который в оригинальном исполнении противоречил отечественному законодательству.

Смена бренда связана с изменением подходов к разработке решений класса endpoint protection. EgoSecure – это новый подход к безопасности конечных точек, когда заказчик получает весь необходимый защитный функционал «из одних рук». Решение построено по модульному принципу, при необходимости все дополнительные функциональные возможности можно легко активировать из панели управления. Заказчику просто нужно выбрать функции, которые наиболее полно соответствуют его требованиям к безопасности конечных точек. К существующим функциям решений EgoSecure добавится целый ряд новых за счет применения технологий InfoWatch и более тесной интеграции разработчиков.

На протяжении многих лет, основной продукт дополнялся все более новыми возможностями, а также улучшался имевшийся функционал. Например, были добавлены функции по контролю действий сотрудников в облаке. Также было улучшено шифрование и система контроля.

В 2018 году группа компаний InfoWatch продала принадлежащий ей контрольный пакет компании EgoSecure немецкой компании Matrix42, которая является ведущим поставщиком программного обеспечения для управления рабочим пространством.

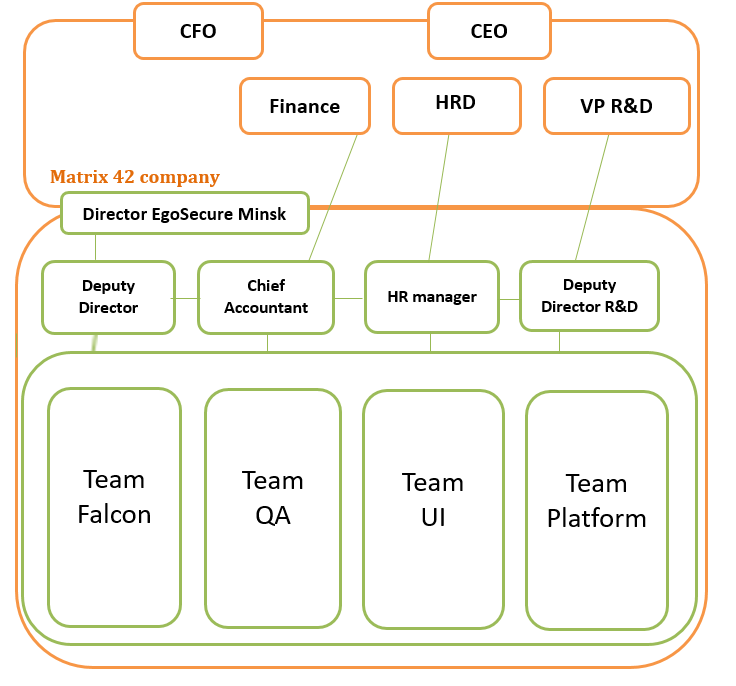


Рисунок 1.1 – Структура ООО «ЭгоСекьюр»

# Обзор аналогичных разработок

## The Open Book Project

The Open Book [2] — это совместимая с E-ink экранами плата на базе SAMD51, вдохновленная Adafruit PyBadge [3]. В дополнение к стандартным функциям он поддерживает различные периферийные устройства, предназначенные для создания доступного универсального устройства для чтения текста.

Основные характеристики:

– 4,2-дюймовый дисплей электронной бумаги с возможностью частичного обновления, управляемый по выделенной шине SPI;

– семь кнопок (клавиши направления, кнопка выбора и кнопки перелистывания страниц), доступные через сдвиговый регистр, а также восьмая кнопка сброса, подключенная к контакту прерывания на SAMD51;

– cлот MicroSD на основной шине SPI;

– светодиоды для индикации этапов зарядки;

– флэш-чип емкостью 2 МБ.

Как следует из названия, целью этой платы является создание доступного устройства с открытым исходным кодом для чтения книг. Эти периферийные устройства были выбраны с учетом этого варианта использования (экран для отображения слов; MicroSD для хранения книг для чтения; Flash-чип для хранения системных данных).

## ESP32 Based ePub Reader

Целью этого проекта стала сделать электронную книгу с поддержкой формата ePub на базе существующих аппаратных платформ, таких как EPDiy V6, M5 Epaper и Lilygo EPD [4].

Проект имеет ограниченную поддержку форматирования — CSS-содержимое файла ePub не анализируется, поэтому просто используются стандартные теги HTML, такие как <h1>, <b>, <i>.

Автор включил только 4 стиля шрифта — обычный, полужирный, курсив и полужирный курсив. Шрифты поддерживают только латинские символы и знаки припинания.

Файлы Epub довольно сложно анализировать. Несмотря на расширение файла epub, на самом деле это zip-архивы, содержащие несколько файлов. Чтобы прочитать файл, автор использует zip-библиотеку. Эта библиотека была модифицирована для работы на ESP32 с PSRAM.

## Электронная книга на STM32H750

В данном проекте автор решил с нуля разработать электронную книгу и поставил перед собой следующие задачи [5]:

– необходимо разработать электронную книгу с экраном 8 дюймов;

– поддержка формата FB2;

– считывание файлов с SD-карты;

– поддержка кириллицы;

– низкое энергопотребление;

– навигацию по пунктам меню сделать светодиодами. Имеется в виду, что отображенные на экране списки для выбора (файлы, список действий) будут статичными, и навигация по ним не будет заставлять перерисовывать экран. Выбор производится включением светодиода напротив списка;

– для управления книгой использовать 4 кнопки — «вперед(вверх)», «назад(вниз)», «меню», «функция»;

– различать на кнопках длинное и короткое нажатие. Кнопку «вперед(вверх)» разместить под пальцем левой руки, держащей книгу, чтобы для перехода на следующую страницу не требовалось перехватывать книгу (это я совсем под себя делаю);

– корпус из дерева.

Для создания книги он использовал следующие компоненты:

– экран E-ink — Waveshare 7.5inch E-ink raw display 800×480;

– микроконтроллер STM32H750VBT6;

– энергонезависимое хранилище для запоминания служебной информации. Их два — EEPROM — AT24C02D и FLASH — W25Q64.

# Анализ технического задания

В данном дипломном проекте разработана и реализована система для чтения электронных книг.

Система обладает следующими функциональными возможностями:

– показ текста книги на экран с возможностью смены страниц и расчёта всех страниц в зависимости от содержимого;

– возможность хранить и открывать книгу из постоянной памяти;

– сенсорное управление;

– главное меню с выбором книг из памяти;

– возможность хранить и открывать книгу из постоянной памяти.

Для выполнения дипломного проекта потребовалось приобрести следующие электрические компоненты:

– LILYGO T5-4.7 inch E-Paper ESP32;

– LILYGO T5-4.7 inch Capacitive Touch Cover;

– LILYGO T5-4.7 inch T-FH Interface RF Card Expansion Module;

– аккумулятор 18650.

Все вышеперечисленные электрические компоненты были приобретены в соответствии с условиями эксплуатации, которые указаны в техническом задании.

# Разработка структуры системы устройства электронная книга на базе Esp32

Структурная схема устройства представлена на рисунке 3.1.

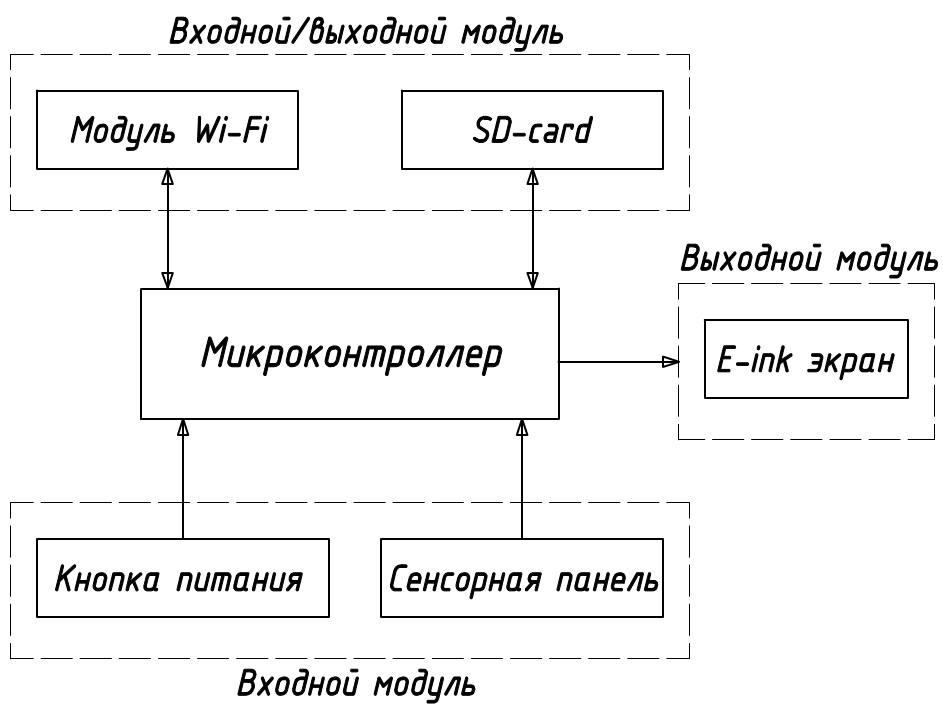


Рисунок 1 – Структурная схема электронной книги

Центральной частью схемы является микроконтроллер. Он обеспечивает взаимодействие всех элементов устройства. Микроконтроллер взаимодействует с Wi-Fi модулем для предоставления возможности загружать книги при помощи телеграмм-бота. Также микроконтроллер связан с модулем SD-карты для чтения и записи книг. На E-ink экране [[6]](#displayDataSheet) отображается текст книги, выбранной пользователем. Управление электронной книгой осуществляется при помощи сенсорной панели. Также для экономии заряда батареи используется кнопка питания для включения/выключения устройства.

# аппартано-Программная реализация системы устройства электронная книга на базе esp32

## Разработка алгоритма работы системы устройства электронная книга на базе ESP32

Обобщённый алгоритм работы системы приведён ниже:

1. На первом этапе устройство сканирует все файлы в памяти и добавляет их имена в список книг, если формат файла поддерживается;

2. Далее формируется главное меню устройства, в центре которого отображается список из 3 книг, а также указатели для перемещения по списку книг (указатель слева – отображение предыдущих 3 книг из общего списка, справа – следующие 3 книги). Список книг отображается в виде вытянутого по горизонтали прямоугольника, в центре которого указано название книги без учёта формата файла и пути к нему;

3. На следующем шаге пользователь должен при помощи нажатия на соответствующие пункты меню на экране выбрать книгу из отображаемых или сменить страницу для дальнейшего поиска и выбора нужной книги;

4. Далее, когда выбрана необходимая книга, по её полному пути в директории происходит чтение файла;

5. На следующем этапе происходит обработка текста входе которой определяется, сколько слов поместится в одну строку на экране. Определение происходит следующим образом:

5.1. За стартовый индекс начала строки принимается ноль;

5.2. Выбирается N следующих символов. Число N задаётся больше максимального количества символов в строке. Индекс конца строки равен сумме стартового индекса и N;

5.3. Производится проверка поместится ли текст, размещенный между стартовым и конечным индексами;

5.4. Если текст не помещается, тогда происходит смещение конечного индекса таким образом, чтобы сместить конечный индекс ровно на длину одного слова с конца. Далее происходит выполняется возвращение к пункту 5.3;

5.5. Сохранение индексов начала и конца итоговой строки;

5.6. Если индекс конца строки меньше, чем длинна текста книги, тогда стартовый индекс равен конечному. Переход к пункту 5.2;

6. Далее необходимо рассчитать итоговое количество страниц книги, которое равно общему количество строк, деленное на количество строк, которые можно отобразить на экране. Результат округляется в большую сторону;

7. Происходит отображение текста текущей страницы при помощи индексов, подсчитанных в 5 пункте. Также отображается номер текущей страницы относительно количества всех страниц в книге;

8. В случае нажатия на экран происходит переход на следующую или предыдущую страницу. Левая сторона сенсора отвечает за переход назад, а правая – вперёд;

9. В случае нажатия на текущий номер страницы или общее количество страниц происходит возвращение к пункту 1;

10. В случае нажатия на кнопку питания происходит отображение рисунка уведомляющего пользователя об включении/выключении устройства.

Заключение

В данной работе приведён процесс разработки электронной книги на базе микроконтроллера ESP32 с использованием экрана на электронных чернилах и сенсорной панели. Также приведена структурная схема с кратким описанием взаимодействия элементов в ней. Рассмотрен общий алгоритм функционирования устройства. Также выполнена проверка работоспособности устройства опытным путем.

Список использованных источников

1. Электронная книга (устройство) [Электронный ресурс]. – Режим доступа: <https://en.wikipedia.org/wiki/E-reader>
2. Техническое описание E-ink экрана [Электронный ресурс]. – Режим доступа: <https://github.com/Xinyuan-LilyGO/LilyGo-EPD47/blob/master/Display_datasheet.pdf>

ПРИЛОЖЕНИЕ А  
(Обязательное)  
Схема электрическая структурная

ПРИЛОЖЕНИЕ Б  
(Обязательное)  
Схема электрическая принципиальная

ПРИЛОЖЕНИЕ В  
(Обязательное)  
Блок-схема алгоритма

ПРИЛОЖЕНИЕ Г  
(Обязательное)  
Код программы

Файл OWM\_EPD47\_epaper.ino

#include <Arduino.h> // In-built

#include <esp\_task\_wdt.h> // In-built

#include "freertos/FreeRTOS.h" // In-built

#include "freertos/task.h" // In-built

#include "epd\_driver.h" // https://github.com/Xinyuan-LilyGO/LilyGo-EPD47

#include "esp\_adc\_cal.h" // In-built

#include <ArduinoJson.h> // https://github.com/bblanchon/ArduinoJson

#include <HTTPClient.h> // In-built

#include <WiFi.h> // In-built

#include <SPI.h> // In-built

#include <time.h> // In-built

#include <WiFiClientSecure.h> // In-built

#include <UniversalTelegramBot.h> // In-built

#include "owm\_credentials.h"

#include "forecast\_record.h"

#include "lang.h"

#include "Button2.h"

#define SCREEN\_WIDTH EPD\_WIDTH

#define SCREEN\_HEIGHT EPD\_HEIGHT

#define ROWS\_ON\_SCREEN 17

struct Page

{

int startRowPositions[ROWS\_ON\_SCREEN];

int endRowPositions[ROWS\_ON\_SCREEN];

int rows;

};

class PagesData

{

public:

void AddPage()

{

numberOfPages++;

if (numberOfPages % 5 == 1)

{

pages = (Page\*) realloc (pages, 5 \* ((numberOfPages / 5) + 1) \* sizeof(Page));

}

}

Page& operator[] (const int index)

{

return pages[index];

}

int GetNumberOfPages()

{

return numberOfPages;

}

protected:

Page\* pages = NULL;

int numberOfPages = 0;

};

WiFiClientSecure clientSecure;

UniversalTelegramBot bot(BOTtoken, clientSecure);

int botRequestDelay = 1000;

unsigned long lastTimeBotRan = 0;

enum tgMode {TG\_NONE, TG\_IN\_WAKE, TG\_IN\_SLEEP, TG\_UPDATE\_RANGE};

int tgCurMode = TG\_NONE;

#define BATT\_PIN 36

#define BUTTON\_1 39

#define BUTTON\_2 34

#define BUTTON\_3 35

Button2 btn1(BUTTON\_1);

Button2 btn2(BUTTON\_2);

//Button2 btn3(BUTTON\_3);

enum alignment {LEFT, RIGHT, CENTER};

#define White 0xFF

#define LightGrey 0xBB

#define Grey 0x88

#define DarkGrey 0x44

#define Black 0x00

#define autoscale\_on true

#define autoscale\_off false

#define barchart\_on true

#define barchart\_off false

//#define LONG\_LIFE\_MODE

int CurrentPage = 1;

PagesData pagesData;

String AllText = "One thing was certain, that the white kitten had had nothing to do with it:—it was the black kitten’s fault entirely. For the white kitten had been having its face washed by the old cat for the last quarter of an hour (and bearing it pretty well, considering); so you see that it couldn’t have had any hand in the mischief."\

"The way Dinah washed her children’s faces was this: first she held the poor thing down by its ear with one paw, and then with the other paw she rubbed its face all over, the wrong way, beginning at the nose: and just now, as I said, she was hard at work on the white kitten, which was lying quite still and trying to purr—no doubt feeling that it was all meant for its good."\

"But the black kitten had been finished with earlier in the afternoon, and so, while Alice was sitting curled up in a corner of the great arm-chair, half talking to herself and half asleep, the kitten had been having a grand game of romps with the ball of worsted Alice had been trying to wind up, " \

"and had been rolling it up and down till it had all come undone again; and there it was, spread over the hearth-rug, all knots and tangles, with the kitten running after its own tail in the middle."\

"“Oh, you wicked little thing!” cried Alice, catching up the kitten, and giving it a little kiss to make it understand that it was in disgrace. “Really, Dinah ought to have taught you better manners! You ought, Dinah, you know you ought!” she added, "\

"looking reproachfully at the old cat, and speaking in as cross a voice as she could manage—and then she scrambled back into the arm-chair, taking the kitten and the worsted with her, and began winding up the ball again. "\

"But she didn’t get on very fast, as she was talking all the time, sometimes to the kitten, and sometimes to herself."\

"Kitty sat very demurely on her knee, pretending to watch the progress of the winding, and now and then putting out one paw and gently touching the ball, as if it would be glad to help, if it might.";

boolean LargeIcon = true;

boolean SmallIcon = false;

boolean NewsMode = false;

boolean PermanentLock = false;

#define Large 20 // For icon drawing

#define Small 8 // For icon drawing

String Time\_str = "--:--";

String Date\_str = "-- --- ----";

int wifi\_signal, CurrentHour = 0, CurrentMin = 0, CurrentSec = 0, EventCnt = 0, vref = 1100;

//################ PROGRAM VARIABLES and OBJECTS ##########################################

#define max\_readings 24 // Limited to 3-days here, but could go to 5-days = 40

Forecast\_record\_type WxConditions[1];

Forecast\_record\_type WxForecast[max\_readings];

float pressure\_readings[max\_readings] = {0};

float temperature\_readings[max\_readings] = {0};

float humidity\_readings[max\_readings] = {0};

float rain\_readings[max\_readings] = {0};

float snow\_readings[max\_readings] = {0};

long SleepDuration = 30; // Sleep time in minutes, aligned to the nearest minute boundary, so if 30 will always update at 00 or 30 past the hour

int WakeupHour = 8; // Don't wakeup until after 07:00 to save battery power

int SleepHour = 23; // Sleep after 23:00 to save battery power

int UpdateRange = 1; // Update screen after 1 min

long StartTime = 0;

long SleepTimer = 0;

long Delta = 30; // ESP32 rtc speed compensation, prevents display at xx:59:yy and then xx:00:yy (one minute later) to save power

unsigned long lastScreenUpdate = 0;

//fonts

#include "opensans8b.h"

#include "opensans10b.h"

#include "opensans12b.h"

#include "opensans18b.h"

#include "opensans24b.h"

GFXfont currentFont;

uint8\_t \*framebuffer;

void handleNewMessages(int numNewMessages)

{

Serial.println("handleNewMessages");

Serial.println(String(numNewMessages));

for (int i=0; i<numNewMessages; i++)

{

// Chat id of the requester

String chat\_id = String(bot.messages[i].chat\_id);

if (chat\_id != CHAT\_ID)

{

bot.sendMessage(chat\_id, "Пользователь не автаризован!", "");

continue;

}

// Print the received message

String text = bot.messages[i].text;

Serial.println(text);

String from\_name = bot.messages[i].from\_name;

if (text == "/start")

{

String welcome = "Привет, " + from\_name + ".\n";

welcome += "Вы можете использовать следующие команды для управления устройством:\n\n";

welcome += "/update для обновления экрана \n";

welcome += "/WakeupHour для установки времени пробуждения устройства \n";

welcome += "/SleepHour для установки времени засыпания устройства \n";

welcome += "/UpdateRange для установки времени обновления экрана устройства \n";

bot.sendMessage(chat\_id, welcome, "");

}

else if (text == "/update")

{

bot.sendMessage(chat\_id, "Экран обновлен", "");

UpdateView();

}

else if (text == "/WakeupHour")

{

bot.sendMessage(chat\_id, "Введите желаемое время пробуждения(0-23):", "");

tgCurMode = TG\_IN\_WAKE;

}

else if (text == "/SleepHour")

{

bot.sendMessage(chat\_id, "Введите желаемое время засыпания(0-23):", "");

tgCurMode = TG\_IN\_SLEEP;

}

else if (text == "/UpdateRange")

{

bot.sendMessage(chat\_id, "Введите желаемое время обновление экрана(1-60):", "");

tgCurMode = TG\_UPDATE\_RANGE;

}

else if (tgCurMode == TG\_IN\_WAKE || tgCurMode == TG\_IN\_SLEEP || tgCurMode == TG\_UPDATE\_RANGE)

{

for (int counter = 0; counter < text.length(); counter++)

{

if (!isDigit(text[counter]))

{

bot.sendMessage(chat\_id, "Введено некорректное значение!\nПопробуйте еще раз.", "");

return;

}

}

int result = text.toInt();

int startOfRange = tgCurMode == TG\_UPDATE\_RANGE ? 1 : 0;

int endOfRange = tgCurMode == TG\_UPDATE\_RANGE ? 60 : 23;

if (result > endOfRange || result < startOfRange)

{

bot.sendMessage(chat\_id, "Число не должно выходить за диапазон " + String(startOfRange)

+ "-" + String(endOfRange) + "!\nПопробуйте еще раз.", "");

return;

}

switch (tgCurMode)

{

case TG\_IN\_WAKE:

WakeupHour = result;

bot.sendMessage(chat\_id, "Время пробуждения изменено на " + String(WakeupHour), "");

break;

case TG\_IN\_SLEEP:

SleepHour = result;

bot.sendMessage(chat\_id, "Время засыпания изменено на " + String(SleepHour), "");

break;

case TG\_UPDATE\_RANGE:

UpdateRange = result;

bot.sendMessage(chat\_id, "Время обновления экрана изменено на " + String(UpdateRange), "");

break;

default:

bot.sendMessage(chat\_id, "Произошла неизвестная ошибка", "");

break;

}

tgCurMode = TG\_NONE;

}

}

}

void BeginSleep()

{

epd\_poweroff\_all();

UpdateLocalTime();

SleepTimer = (SleepDuration \* 60 - ((CurrentMin % SleepDuration) \* 60 + CurrentSec)) + Delta; //Some ESP32 have a RTC that is too fast to maintain accurate time, so add an offset

esp\_sleep\_enable\_timer\_wakeup(SleepTimer \* 1000000LL); // in Secs, 1000000LL converts to Secs as unit = 1uSec

Serial.println("Awake for : " + String((millis() - StartTime) / 1000.0, 3) + "-secs");

Serial.println("Entering " + String(SleepTimer) + " (secs) of sleep time");

Serial.println("Starting deep-sleep period...");

esp\_deep\_sleep\_start(); // Sleep for e.g. 30 minutes

}

boolean SetupTime()

{

configTime(gmtOffset\_sec, daylightOffset\_sec, ntpServer, "time.nist.gov"); //(gmtOffset\_sec, daylightOffset\_sec, ntpServer)

setenv("TZ", Timezone, 1); //setenv()adds the "TZ" variable to the environment with a value TimeZone, only used if set to 1, 0 means no change

tzset(); // Set the TZ environment variable

delay(100);

return UpdateLocalTime();

}

uint8\_t StartWiFi()

{

Serial.println("\r\nConnecting to: " + String(ssid));

IPAddress dns(8, 8, 8, 8); // Use Google DNS

WiFi.disconnect();

WiFi.mode(WIFI\_STA); // switch off AP

WiFi.setAutoConnect(true);

WiFi.setAutoReconnect(true);

WiFi.begin(ssid, password);

WiFi.setSleep(false);

if (WiFi.waitForConnectResult() != WL\_CONNECTED)

{

Serial.printf("STA: Failed!\n");

WiFi.disconnect(false);

delay(500);

WiFi.begin(ssid, password);

}

if (WiFi.status() == WL\_CONNECTED)

{

wifi\_signal = WiFi.RSSI(); // Get Wifi Signal strength now, because the WiFi will be turned off to save power!

Serial.println("WiFi connected at: " + WiFi.localIP().toString());

}

else Serial.println("WiFi connection \*\*\* FAILED \*\*\*");

return WiFi.status();

}

void StopWiFi()

{

WiFi.disconnect();

WiFi.mode(WIFI\_OFF);

Serial.println("WiFi switched Off");

}

void PrintInfoInBot()

{

bot.sendMessage(CHAT\_ID, "Произошло обновление текущих метеоданных", "");

}

void UpdateView()

{

epd\_poweron(); // Switch on EPD display

epd\_clear(); // Clear the screen

fillRect(0, 0, SCREEN\_WIDTH, SCREEN\_HEIGHT, White); // clear framebuffer

DisplayInfo();

edp\_update(); // Update the display to show the information

epd\_poweroff\_all(); // Switch off all power to EPD

// #ifdef LONG\_LIFE\_MODE

// if (StartWiFi() == WL\_CONNECTED && SetupTime() == true)

// #else

// if (SetupTime() == true)

// #endif

// {

// bool WakeUp = false;

// if (WakeupHour > SleepHour)

// WakeUp = (CurrentHour >= WakeupHour || CurrentHour <= SleepHour);

// else

// WakeUp = (CurrentHour >= WakeupHour && CurrentHour <= SleepHour);

// if (WakeUp)

// {

// byte Attempts = 1;

// bool RxWeather = false;

// bool RxForecast = false;

// WiFiClient client; // wifi client object

// while ((RxWeather == false || RxForecast == false) && Attempts <= 2)

// { // Try up-to 2 time for Weather and Forecast data

// if (RxWeather == false) RxWeather = obtainWeatherData(client, "weather");

// if (RxForecast == false) RxForecast = obtainWeatherData(client, "forecast");

// Attempts++;

// }

// Serial.println("Received all weather data...");

// if (RxWeather && RxForecast)

// { // Only if received both Weather or Forecast proceed

// #ifdef LONG\_LIFE\_MODE

// StopWiFi(); // Reduces power consumption

// #endif

// epd\_poweron(); // Switch on EPD display

// epd\_clear(); // Clear the screen

// DisplayWeather(); // Display the weather data

// PrintInfoInBot(); // Print main info in telegram bot

// edp\_update(); // Update the display to show the information

// epd\_poweroff\_all(); // Switch off all power to EPD

// }

// }

// }

// #ifdef LONG\_LIFE\_MODE

// BeginSleep();

// #endif

}

void DisplayInfo()

{

setFont(OpenSans8B);

drawString(SCREEN\_WIDTH / 2, 4, String(CurrentPage) + " / " + String(pagesData.GetNumberOfPages()), CENTER);

drawLine(0, 18, SCREEN\_WIDTH, 18, Black);

drawLine(0, 19, SCREEN\_WIDTH, 19, Black);

setFont(OpenSans12B);

int xOffSet = 5;

int yOffset = 22;

for (int currentRow = 0; currentRow < pagesData[CurrentPage - 1].rows; currentRow++)

{

if (currentRow == pagesData[CurrentPage - 1].rows - 1)

yOffset += 5;

drawString(xOffSet, yOffset, AllText.substring(pagesData[CurrentPage - 1].startRowPositions[currentRow], pagesData[CurrentPage - 1].endRowPositions[currentRow]), LEFT);

yOffset += 30;

Serial.println(AllText.substring(pagesData[CurrentPage - 1].startRowPositions[currentRow], pagesData[CurrentPage - 1].endRowPositions[currentRow]));

}

}

void CalculateAllPositions()

{

setFont(OpenSans12B);

int xOffSet = 5;

int yOffset = 0;

String CurrStr;

const int SymbolsInRow = SCREEN\_WIDTH / 12;

int StartPosition = 0;

char \* data;

int x1, y1; //the bounds of x,y and w and h of the variable 'text' in pixels.

int w, h;

int xx, yy;

bool NeedEndCycle = false;

int CurrentRow = 1;

int CurrentReadingPage = 0;

while (true)

{

if (CurrentRow == 1)

{

pagesData.AddPage();

CurrentReadingPage = pagesData.GetNumberOfPages() - 1;

}

pagesData[CurrentReadingPage].startRowPositions[CurrentRow - 1] = StartPosition;

if (StartPosition + SymbolsInRow > AllText.length() - 1)

{

CurrStr = AllText.substring(StartPosition);

NeedEndCycle = true;

}

else

CurrStr = AllText.substring(StartPosition, StartPosition + SymbolsInRow);

do

{

if (!NeedEndCycle)

{

do

{

CurrStr.remove(CurrStr.length() - 1);

}

while (CurrStr[CurrStr.length() - 1] != ' ');

}

data = const\_cast<char\*>(CurrStr.c\_str());

xx = xOffSet;

yy = yOffset;

get\_text\_bounds(&currentFont, data, &xx, &yy, &x1, &y1, &w, &h, NULL);

if (xx > SCREEN\_WIDTH && NeedEndCycle)

NeedEndCycle = false;

}

while (xx > SCREEN\_WIDTH);

StartPosition += CurrStr.length();

if (NeedEndCycle)

{

pagesData[CurrentReadingPage].endRowPositions[CurrentRow - 1] = StartPosition;

pagesData[CurrentReadingPage].rows = CurrentRow;

break;

}

pagesData[CurrentReadingPage].endRowPositions[CurrentRow - 1] = StartPosition - 1;

if (++CurrentRow > ROWS\_ON\_SCREEN)

{

CurrentRow = 1;

pagesData[CurrentReadingPage].rows = ROWS\_ON\_SCREEN;

}

}

}

void buttonPressed(Button2 &b)

{

Serial.println("Button attached to pin " + String(b.getAttachPin()));

bool NeedUpdate = true;

if (b.getAttachPin() == BUTTON\_1)

{

if (++CurrentPage > pagesData.GetNumberOfPages())

{

CurrentPage = pagesData.GetNumberOfPages();

NeedUpdate = false;

}

}

else if (b.getAttachPin() == BUTTON\_2)

{

if (--CurrentPage == 0)

{

CurrentPage = 1;

NeedUpdate = false;

}

}

Serial.println("Current page = " + String(CurrentPage));

if (NeedUpdate)

UpdateView();

}

void InitialiseSystem()

{

StartTime = millis();

Serial.begin(115200);

while (!Serial);

Serial.println(String(\_\_FILE\_\_) + "\nStarting...");

epd\_init();

framebuffer = (uint8\_t \*)ps\_calloc(sizeof(uint8\_t), EPD\_WIDTH \* EPD\_HEIGHT / 2);

if (!framebuffer) Serial.println("Memory alloc failed!");

memset(framebuffer, 0xFF, EPD\_WIDTH \* EPD\_HEIGHT / 2);

#ifndef LONG\_LIFE\_MODE

btn1.setPressedHandler(buttonPressed);

btn2.setPressedHandler(buttonPressed);

//btn3.setPressedHandler(buttonPressed);

//StartWiFi();

#endif

//clientSecure.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); // Add root certificate for api.telegram.org

}

void loop()

{

#ifndef LONG\_LIFE\_MODE

btn1.loop();

btn2.loop();

//btn3.loop();

// if (millis() > lastTimeBotRan + botRequestDelay)

// {

// int numNewMessages = bot.getUpdates(bot.last\_message\_received + 1);

// while(numNewMessages)

// {

// Serial.println("got response");

// handleNewMessages(numNewMessages);

// numNewMessages = bot.getUpdates(bot.last\_message\_received + 1);

// }

// lastTimeBotRan = millis();

// }

// if (millis() > lastScreenUpdate + UpdateRange \* 60000)

// {

// UpdateView();

// lastScreenUpdate = millis();

// }

#endif

}

void setup()

{

InitialiseSystem();

CalculateAllPositions();

UpdateView();

}

void DrawSegment(int x, int y, int o1, int o2, int o3, int o4, int o11, int o12, int o13, int o14) {

drawLine(x + o1, y + o2, x + o3, y + o4, Black);

drawLine(x + o11, y + o12, x + o13, y + o14, Black);

}

void DrawPressureAndTrend(int x, int y, float pressure, String slope) {

drawString(x + 25, y - 10, String(pressure, (Units == "M" ? 0 : 1)) + (Units == "M" ? "hPa" : "in"), LEFT);

if (slope == "+") {

DrawSegment(x, y, 0, 0, 8, -8, 8, -8, 16, 0);

DrawSegment(x - 1, y, 0, 0, 8, -8, 8, -8, 16, 0);

}

else if (slope == "0") {

DrawSegment(x, y, 8, -8, 16, 0, 8, 8, 16, 0);

DrawSegment(x - 1, y, 8, -8, 16, 0, 8, 8, 16, 0);

}

else if (slope == "-") {

DrawSegment(x, y, 0, 0, 8, 8, 8, 8, 16, 0);

DrawSegment(x - 1, y, 0, 0, 8, 8, 8, 8, 16, 0);

}

}

void DisplayStatusSection(int x, int y, int rssi) {

setFont(OpenSans8B);

DrawRSSI(x + 320, y, rssi);

DrawBattery(x + 100, y);

}

void DrawRSSI(int x, int y, int rssi) {

int WIFIsignal = 0;

int xpos = 1;

for (int \_rssi = -100; \_rssi <= rssi; \_rssi = \_rssi + 20) {

if (\_rssi <= -20) WIFIsignal = 30; // <-20dbm displays 5-bars

if (\_rssi <= -40) WIFIsignal = 24; // -40dbm to -21dbm displays 4-bars

if (\_rssi <= -60) WIFIsignal = 18; // -60dbm to -41dbm displays 3-bars

if (\_rssi <= -80) WIFIsignal = 12; // -80dbm to -61dbm displays 2-bars

if (\_rssi <= -100) WIFIsignal = 6; // -100dbm to -81dbm displays 1-bar

fillRect(x + xpos \* 8, y - WIFIsignal, 6, WIFIsignal, Black);

xpos++;

}

}

boolean UpdateLocalTime() {

struct tm timeinfo;

char time\_output[30], day\_output[30], update\_time[30];

while (!getLocalTime(&timeinfo, 5000)) { // Wait for 5-sec for time to synchronise

Serial.println("Failed to obtain time");

return false;

}

CurrentHour = timeinfo.tm\_hour;

CurrentMin = timeinfo.tm\_min;

CurrentSec = timeinfo.tm\_sec;

//See http://www.cplusplus.com/reference/ctime/strftime/

Serial.println(&timeinfo, "%a %b %d %Y %H:%M:%S"); // Displays: Saturday, June 24 2017 14:05:49

if (Units == "M") {

sprintf(day\_output, "%s, %02u %s %04u", weekday\_D[timeinfo.tm\_wday], timeinfo.tm\_mday, month\_M[timeinfo.tm\_mon], (timeinfo.tm\_year) + 1900);

strftime(update\_time, sizeof(update\_time), "%H:%M", &timeinfo); // Creates: '@ 14:05:49' and change from 30 to 8 <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

sprintf(time\_output, "%s", update\_time);

}

else

{

strftime(day\_output, sizeof(day\_output), "%a %b-%d-%Y", &timeinfo); // Creates 'Sat May-31-2019'

strftime(update\_time, sizeof(update\_time), "%r", &timeinfo); // Creates: '@ 02:05:49pm'

sprintf(time\_output, "%s", update\_time);

}

Date\_str = day\_output;

Time\_str = time\_output;

return true;

}

void DrawBattery(int x, int y) {

uint8\_t percentage = 100;

esp\_adc\_cal\_characteristics\_t adc\_chars;

esp\_adc\_cal\_value\_t val\_type = esp\_adc\_cal\_characterize(ADC\_UNIT\_1, ADC\_ATTEN\_DB\_11, ADC\_WIDTH\_BIT\_12, 1100, &adc\_chars);

if (val\_type == ESP\_ADC\_CAL\_VAL\_EFUSE\_VREF) {

Serial.printf("eFuse Vref:%u mV", adc\_chars.vref);

vref = adc\_chars.vref;

}

float voltage = analogRead(36) / 4096.0 \* 6.566 \* (vref / 1000.0);

if (voltage > 1 ) { // Only display if there is a valid reading

Serial.println("\nVoltage = " + String(voltage));

percentage = 2836.9625 \* pow(voltage, 4) - 43987.4889 \* pow(voltage, 3) + 255233.8134 \* pow(voltage, 2) - 656689.7123 \* voltage + 632041.7303;

if (voltage >= 4.20) percentage = 100;

if (voltage <= 3.20) percentage = 0; // orig 3.5

drawRect(x + 25, y - 14, 40, 15, Black);

fillRect(x + 65, y - 10, 4, 7, Black);

fillRect(x + 27, y - 12, 36 \* percentage / 100.0, 11, Black);

drawString(x + 85, y - 14, String(percentage) + "% " + String(voltage, 1) + "v", LEFT);

}

}

// Symbols are drawn on a relative 10x10grid and 1 scale unit = 1 drawing unit

void addcloud(int x, int y, int scale, int linesize) {

fillCircle(x - scale \* 3, y, scale, Black); // Left most circle

fillCircle(x + scale \* 3, y, scale, Black); // Right most circle

fillCircle(x - scale, y - scale, scale \* 1.4, Black); // left middle upper circle

fillCircle(x + scale \* 1.5, y - scale \* 1.3, scale \* 1.75, Black); // Right middle upper circle

fillRect(x - scale \* 3 - 1, y - scale, scale \* 6, scale \* 2 + 1, Black); // Upper and lower lines

fillCircle(x - scale \* 3, y, scale - linesize, White); // Clear left most circle

fillCircle(x + scale \* 3, y, scale - linesize, White); // Clear right most circle

fillCircle(x - scale, y - scale, scale \* 1.4 - linesize, White); // left middle upper circle

fillCircle(x + scale \* 1.5, y - scale \* 1.3, scale \* 1.75 - linesize, White); // Right middle upper circle

fillRect(x - scale \* 3 + 2, y - scale + linesize - 1, scale \* 5.9, scale \* 2 - linesize \* 2 + 2, White); // Upper and lower lines

}

void addrain(int x, int y, int scale, bool IconSize) {

if (IconSize == SmallIcon) {

setFont(OpenSans8B);

drawString(x - 25, y + 12, "///////", LEFT);

}

else

{

setFont(OpenSans18B);

drawString(x - 60, y + 25, "///////", LEFT);

}

}

void addsnow(int x, int y, int scale, bool IconSize) {

if (IconSize == SmallIcon) {

setFont(OpenSans8B);

drawString(x - 25, y + 15, "\* \* \* \*", LEFT);

}

else

{

setFont(OpenSans18B);

drawString(x - 60, y + 30, "\* \* \* \*", LEFT);

}

}

void addtstorm(int x, int y, int scale) {

y = y + scale / 2;

for (int i = 0; i < 5; i++) {

drawLine(x - scale \* 4 + scale \* i \* 1.5 + 0, y + scale \* 1.5, x - scale \* 3.5 + scale \* i \* 1.5 + 0, y + scale, Black);

if (scale != Small) {

drawLine(x - scale \* 4 + scale \* i \* 1.5 + 1, y + scale \* 1.5, x - scale \* 3.5 + scale \* i \* 1.5 + 1, y + scale, Black);

drawLine(x - scale \* 4 + scale \* i \* 1.5 + 2, y + scale \* 1.5, x - scale \* 3.5 + scale \* i \* 1.5 + 2, y + scale, Black);

}

drawLine(x - scale \* 4 + scale \* i \* 1.5, y + scale \* 1.5 + 0, x - scale \* 3 + scale \* i \* 1.5 + 0, y + scale \* 1.5 + 0, Black);

if (scale != Small) {

drawLine(x - scale \* 4 + scale \* i \* 1.5, y + scale \* 1.5 + 1, x - scale \* 3 + scale \* i \* 1.5 + 0, y + scale \* 1.5 + 1, Black);

drawLine(x - scale \* 4 + scale \* i \* 1.5, y + scale \* 1.5 + 2, x - scale \* 3 + scale \* i \* 1.5 + 0, y + scale \* 1.5 + 2, Black);

}

drawLine(x - scale \* 3.5 + scale \* i \* 1.4 + 0, y + scale \* 2.5, x - scale \* 3 + scale \* i \* 1.5 + 0, y + scale \* 1.5, Black);

if (scale != Small) {

drawLine(x - scale \* 3.5 + scale \* i \* 1.4 + 1, y + scale \* 2.5, x - scale \* 3 + scale \* i \* 1.5 + 1, y + scale \* 1.5, Black);

drawLine(x - scale \* 3.5 + scale \* i \* 1.4 + 2, y + scale \* 2.5, x - scale \* 3 + scale \* i \* 1.5 + 2, y + scale \* 1.5, Black);

}

}

}

void addsun(int x, int y, int scale, bool IconSize) {

int linesize = 5;

fillRect(x - scale \* 2, y, scale \* 4, linesize, Black);

fillRect(x, y - scale \* 2, linesize, scale \* 4, Black);

drawLine(x - scale \* 1.3, y - scale \* 1.3, x + scale \* 1.3, y + scale \* 1.3, Black);

drawLine(x - scale \* 1.3, y + scale \* 1.3, x + scale \* 1.3, y - scale \* 1.3, Black);

if (IconSize == LargeIcon) {

drawLine(1 + x - scale \* 1.3, y - scale \* 1.3, 1 + x + scale \* 1.3, y + scale \* 1.3, Black);

drawLine(2 + x - scale \* 1.3, y - scale \* 1.3, 2 + x + scale \* 1.3, y + scale \* 1.3, Black);

drawLine(3 + x - scale \* 1.3, y - scale \* 1.3, 3 + x + scale \* 1.3, y + scale \* 1.3, Black);

drawLine(1 + x - scale \* 1.3, y + scale \* 1.3, 1 + x + scale \* 1.3, y - scale \* 1.3, Black);

drawLine(2 + x - scale \* 1.3, y + scale \* 1.3, 2 + x + scale \* 1.3, y - scale \* 1.3, Black);

drawLine(3 + x - scale \* 1.3, y + scale \* 1.3, 3 + x + scale \* 1.3, y - scale \* 1.3, Black);

}

fillCircle(x, y, scale \* 1.3, White);

fillCircle(x, y, scale, Black);

fillCircle(x, y, scale - linesize, White);

}

void addfog(int x, int y, int scale, int linesize, bool IconSize) {

if (IconSize == SmallIcon) {

y -= 10;

linesize = 1;

}

for (int i = 0; i < 6; i++) {

fillRect(x - scale \* 3, y + scale \* 1.5, scale \* 6, linesize, Black);

fillRect(x - scale \* 3, y + scale \* 2.0, scale \* 6, linesize, Black);

fillRect(x - scale \* 3, y + scale \* 2.5, scale \* 6, linesize, Black);

}

}

void Sunny(int x, int y, bool IconSize, String IconName) {

int scale = Small, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

else y = y - 3; // Shift up small sun icon

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

scale = scale \* 1.6;

addsun(x, y, scale, IconSize);

}

void MostlySunny(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addsun(x - scale \* 1.8, y - scale \* 1.8, scale, IconSize);

addcloud(x, y, scale, linesize);

}

void MostlyCloudy(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addcloud(x, y, scale, linesize);

addsun(x - scale \* 1.8, y - scale \* 1.8, scale, IconSize);

}

void Cloudy(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addcloud(x + 15, y - 22, scale / 2, linesize); // Cloud top right

addcloud(x - 10, y - 18, scale / 2, linesize); // Cloud top left

addcloud(x, y, scale, linesize); // Main cloud

}

void Rain(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addcloud(x, y, scale, linesize);

addrain(x, y, scale, IconSize);

}

void ExpectRain(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addsun(x - scale \* 1.8, y - scale \* 1.8, scale, IconSize);

addcloud(x, y, scale, linesize);

addrain(x, y, scale, IconSize);

}

void ChanceRain(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addsun(x - scale \* 1.8, y - scale \* 1.8, scale, IconSize);

addcloud(x, y, scale, linesize);

addrain(x, y, scale, IconSize);

}

void Tstorms(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addcloud(x, y, scale, linesize);

addtstorm(x, y, scale);

}

void Snow(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addcloud(x, y, scale, linesize);

addsnow(x, y, scale, IconSize);

}

void Fog(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addcloud(x, y - 5, scale, linesize);

addfog(x, y - 5, scale, linesize, IconSize);

}

void Haze(int x, int y, bool IconSize, String IconName) {

int scale = Small, linesize = 5, Offset = 10;

if (IconSize == LargeIcon) {

scale = Large;

Offset = 35;

}

if (IconName.endsWith("n")) addmoon(x, y + Offset, scale, IconSize);

addsun(x, y - 5, scale \* 1.4, IconSize);

addfog(x, y - 5, scale \* 1.4, linesize, IconSize);

}

void CloudCover(int x, int y, int CCover) {

addcloud(x - 9, y + 2, Small \* 0.3, 2); // Cloud top left

addcloud(x + 3, y - 2, Small \* 0.3, 2); // Cloud top right

addcloud(x, y + 10, Small \* 0.6, 2); // Main cloud

drawString(x + 20, y, String(CCover) + "%", LEFT);

}

void Visibility(int x, int y, String Visi) {

float start\_angle = 0.52, end\_angle = 2.61, Offset = 8;

int r = 14;

for (float i = start\_angle; i < end\_angle; i = i + 0.05) {

drawPixel(x + r \* cos(i), y - r / 2 + r \* sin(i) + Offset, Black);

drawPixel(x + r \* cos(i), 1 + y - r / 2 + r \* sin(i) + Offset, Black);

}

start\_angle = 3.61; end\_angle = 5.78;

for (float i = start\_angle; i < end\_angle; i = i + 0.05) {

drawPixel(x + r \* cos(i), y + r / 2 + r \* sin(i) + Offset, Black);

drawPixel(x + r \* cos(i), 1 + y + r / 2 + r \* sin(i) + Offset, Black);

}

fillCircle(x, y + Offset, r / 4, Black);

drawString(x + 20, y, Visi, LEFT);

}

void addmoon(int x, int y, int scale, bool IconSize) {

if (IconSize == LargeIcon) {

fillCircle(x - 85, y - 100, uint16\_t(scale \* 0.8), Black);

fillCircle(x - 57, y - 100, uint16\_t(scale \* 1.6), White);

}

else

{

fillCircle(x - 28, y - 37, uint16\_t(scale \* 1.0), Black);

fillCircle(x - 20, y - 37, uint16\_t(scale \* 1.6), White);

}

}

void Nodata(int x, int y, bool IconSize, String IconName) {

if (IconSize == LargeIcon) setFont(OpenSans24B); else setFont(OpenSans12B);

drawString(x - 3, y - 10, "?", CENTER);

}

/\* (C) D L BIRD

This function will draw a graph on a ePaper/TFT/LCD display using data from an array containing data to be graphed.

The variable 'max\_readings' determines the maximum number of data elements for each array. Call it with the following parametric data:

x\_pos-the x axis top-left position of the graph

y\_pos-the y-axis top-left position of the graph, e.g. 100, 200 would draw the graph 100 pixels along and 200 pixels down from the top-left of the screen

width-the width of the graph in pixels

height-height of the graph in pixels

Y1\_Max-sets the scale of plotted data, for example 5000 would scale all data to a Y-axis of 5000 maximum

data\_array1 is parsed by value, externally they can be called anything else, e.g. within the routine it is called data\_array1, but externally could be temperature\_readings

auto\_scale-a logical value (TRUE or FALSE) that switches the Y-axis autoscale On or Off

barchart\_on-a logical value (TRUE or FALSE) that switches the drawing mode between barhcart and line graph

barchart\_colour-a sets the title and graph plotting colour

If called with Y!\_Max value of 500 and the data never goes above 500, then autoscale will retain a 0-500 Y scale, if on, the scale increases/decreases to match the data.

auto\_scale\_margin, e.g. if set to 1000 then autoscale increments the scale by 1000 steps.

\*/

void DrawGraph(int x\_pos, int y\_pos, int gwidth, int gheight, float Y1Min, float Y1Max, String title, float DataArray[], int readings, boolean auto\_scale, boolean barchart\_mode) {

#define auto\_scale\_margin 0 // Sets the autoscale increment, so axis steps up fter a change of e.g. 3

#define y\_minor\_axis 5 // 5 y-axis division markers

setFont(OpenSans10B);

int maxYscale = -10000;

int minYscale = 10000;

int last\_x, last\_y;

float x2, y2;

if (auto\_scale == true) {

for (int i = 1; i < readings; i++ ) {

if (DataArray[i] >= maxYscale) maxYscale = DataArray[i];

if (DataArray[i] <= minYscale) minYscale = DataArray[i];

}

maxYscale = round(maxYscale + auto\_scale\_margin); // Auto scale the graph and round to the nearest value defined, default was Y1Max

Y1Max = round(maxYscale + 0.5);

if (minYscale != 0) minYscale = round(minYscale - auto\_scale\_margin); // Auto scale the graph and round to the nearest value defined, default was Y1Min

Y1Min = round(minYscale);

}

// Draw the graph

last\_x = x\_pos + 1;

last\_y = y\_pos + (Y1Max - constrain(DataArray[1], Y1Min, Y1Max)) / (Y1Max - Y1Min) \* gheight;

drawRect(x\_pos, y\_pos, gwidth + 3, gheight + 2, Grey);

drawString(x\_pos - 20 + gwidth / 2, y\_pos - 28, title, CENTER);

for (int gx = 0; gx < readings; gx++) {

x2 = x\_pos + gx \* gwidth / (readings - 1) - 1 ; // max\_readings is the global variable that sets the maximum data that can be plotted

y2 = y\_pos + (Y1Max - constrain(DataArray[gx], Y1Min, Y1Max)) / (Y1Max - Y1Min) \* gheight + 1;

if (barchart\_mode) {

fillRect(last\_x + 2, y2, (gwidth / readings) - 1, y\_pos + gheight - y2 + 2, Black);

} else {

drawLine(last\_x, last\_y - 1, x2, y2 - 1, Black); // Two lines for hi-res display

drawLine(last\_x, last\_y, x2, y2, Black);

}

last\_x = x2;

last\_y = y2;

}

//Draw the Y-axis scale

#define number\_of\_dashes 20

for (int spacing = 0; spacing <= y\_minor\_axis; spacing++) {

for (int j = 0; j < number\_of\_dashes; j++) { // Draw dashed graph grid lines

if (spacing < y\_minor\_axis) drawFastHLine((x\_pos + 3 + j \* gwidth / number\_of\_dashes), y\_pos + (gheight \* spacing / y\_minor\_axis), gwidth / (2 \* number\_of\_dashes), Grey);

}

if ((Y1Max - (float)(Y1Max - Y1Min) / y\_minor\_axis \* spacing) < 5 || title == TXT\_PRESSURE\_IN) {

drawString(x\_pos - 10, y\_pos + gheight \* spacing / y\_minor\_axis - 5, String((Y1Max - (float)(Y1Max - Y1Min) / y\_minor\_axis \* spacing + 0.01), 1), RIGHT);

}

else

{

if (Y1Min < 1 && Y1Max < 10) {

drawString(x\_pos - 3, y\_pos + gheight \* spacing / y\_minor\_axis - 5, String((Y1Max - (float)(Y1Max - Y1Min) / y\_minor\_axis \* spacing + 0.01), 1), RIGHT);

}

else {

drawString(x\_pos - 7, y\_pos + gheight \* spacing / y\_minor\_axis - 5, String((Y1Max - (float)(Y1Max - Y1Min) / y\_minor\_axis \* spacing + 0.01), 0), RIGHT);

}

}

}

for (int i = 0; i < 3; i++) {

drawString(20 + x\_pos + gwidth / 3 \* i, y\_pos + gheight + 10, String(i) + "d", LEFT);

if (i < 2) drawFastVLine(x\_pos + gwidth / 3 \* i + gwidth / 3, y\_pos, gheight, LightGrey);

}

}

void drawString(int x, int y, String text, alignment align) {

char \* data = const\_cast<char\*>(text.c\_str());

int x1, y1; //the bounds of x,y and w and h of the variable 'text' in pixels.

int w, h;

int xx = x, yy = y;

get\_text\_bounds(&currentFont, data, &xx, &yy, &x1, &y1, &w, &h, NULL);

if (align == RIGHT) x = x - w;

if (align == CENTER) x = x - w / 2;

int cursor\_y = y + h;

write\_string(&currentFont, data, &x, &cursor\_y, framebuffer);

}

void fillCircle(int x, int y, int r, uint8\_t color) {

epd\_fill\_circle(x, y, r, color, framebuffer);

}

void drawFastHLine(int16\_t x0, int16\_t y0, int length, uint16\_t color) {

epd\_draw\_hline(x0, y0, length, color, framebuffer);

}

void drawFastVLine(int16\_t x0, int16\_t y0, int length, uint16\_t color) {

epd\_draw\_vline(x0, y0, length, color, framebuffer);

}

void drawLine(int16\_t x0, int16\_t y0, int16\_t x1, int16\_t y1, uint16\_t color) {

epd\_write\_line(x0, y0, x1, y1, color, framebuffer);

}

void drawCircle(int x0, int y0, int r, uint8\_t color) {

epd\_draw\_circle(x0, y0, r, color, framebuffer);

}

void drawRect(int16\_t x, int16\_t y, int16\_t w, int16\_t h, uint16\_t color) {

epd\_draw\_rect(x, y, w, h, color, framebuffer);

}

void fillRect(int16\_t x, int16\_t y, int16\_t w, int16\_t h, uint16\_t color) {

epd\_fill\_rect(x, y, w, h, color, framebuffer);

}

void fillTriangle(int16\_t x0, int16\_t y0, int16\_t x1, int16\_t y1,

int16\_t x2, int16\_t y2, uint16\_t color) {

epd\_fill\_triangle(x0, y0, x1, y1, x2, y2, color, framebuffer);

}

void drawPixel(int x, int y, uint8\_t color) {

epd\_draw\_pixel(x, y, color, framebuffer);

}

void setFont(GFXfont const &font) {

currentFont = font;

}

void edp\_update() {

epd\_draw\_grayscale\_image(epd\_full\_screen(), framebuffer); // Update the screen

}

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Обозначение* | | | | | *Наименование* | | *Дополнительные сведения* | | |
|  | | | | | Текстовые документы | |  | | |
|  | | | | |  | |  | | |
| БГУИР КП 1-40 02 02 017 ПЗ | | | | | Отчёт по практике | | 47 с. | | |
|  | | | | |  | |  | | |
|  | | | | | *Графические документы* | |  | | |
|  | | | | |  | |  | | |
| ГУИР 467849.001 Э1 | | | | | Схема электрическая структурная | | Формат А4 | | |
|  | | | | |  | |  | | |
| ГУИР 467849.002 Э3 | | | | | Схема электрическая принципиальная | | Формат А2 | | |
|  | | | | |  | |  | | |
| ГУИР 467849.004 ПД | | | | | Схема алгоритма работы | | Формат А4 | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  | | | | |  | |  | | |
|  |  |  |  |  | *БГУИР КП 1-40 02 02 017 ПЗ* | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| *Изм.* | *Л.* | *№ докум.* | *Подп.* | *Дата* | *Wi-Fi метеостанция*  *Ведомость курсового проекта* | *Лит* | | *Лист* | *Листов* |
| *Разраб.* | | *Каленик С.А.* |  |  | *T* | | 56 | 56 |
| *Пров.* | | *Порхун М.И.* |  |  | *Кафедра ЭВС,  гр.850701* | | | |
| *Т.контр.* | |  |  |  |
| *Н. Контр.* | |  |  |  |
| *Утв.* | |  |  |  |