

[This document contains detailed information about the project and was created as a go-to place to find answers to all of your HOWs and WHYs, so the software support and further project development is easier.]

[ August 2024 ]

[42 prague smart sigN.]

[technical documentation.]

[Table of contents.]

[SUMMARY.]

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | About the project  General program runtime description  Hardware description  Hardware decisions explained  Circuit diagrams and schematics  Graphics  Description of the program files  Description of the program functions  Description of the program libraries and their functions  Description of the program constants  Architectural decisions explained  Communication protocols  Power management  Safety considerations  The Don’ts of changing the program  Development tools  Microcontroller Flashing process  Suggestions for dealing with confidential information  Known bugs  Future development suggestions  Sources |

[ABOUT THE PROJECT.]

[**General description of the program run**.]

Let's take a look at one day of the 42 Prague Smart Sign‘s life. Let’s say that on this particular day there is an exam scheduled for 13:00 and it will take 3 hours.

It is still night and the Sign is showing the cluster number with the default icons from yesterday, while still being asleep. Time passes by. Now, it is 6:00 in the morning. The Sign wakes up to check if there are exams today. It goes to the Intra server and sees that there will be an exam starting at 13:00 and ending at 16:00. The Sign replaces the default icons with a note “The cluster will be reserved for an exam today at 13:00” while still displaying the cluster number. Since there is nothing more for the Sign to do, it sets its alarm clock for 12:00 (an hour before the exam) and goes to sleep.

It is 12 o'clock and the Sign wakes up again. It checks Intra to make sure the exam was not canceled during its sleep and replaces the cluster number with a big warning sign that says: “RESERVATION! The cluster is reserved for an exam. Please, vacate it in due time. You have XX minutes left”. Instead of XX it first says 50 minutes, then 25 minutes and finally 5 minutes left.

Finally, it is 13:00. The exam begins. The Sign changes the previous warning sign for a new one, saying “DO NOT ENTER! Exam in progress! The exam will end at 16:00”. At this point the Sign has nothing else to do, so again it sets its alarm clock for 16:00 and goes to sleep.

At 16 o'clock the Sign wakes up, checks Intra, finds no more exams for today, replaces the warning sign with a cluster number with the default icons, sets its alarm clock until the next scheduled wake up - in this case 18:00 - and goes back to sleep.

At 18 o'clock the Sign wakes up, checks Intra and finds no more exams. Its work for today is over. It goes to sleep to wake up again the next morning at 6.

[PROGRAM RUN STEP-BY-STEP.]

**НАЧАЛО РАБОТЫ ПРОГРАММЫ**

1. включение
2. инициализация Серийного порта

* вывод тире в серийный порт даёт время синхронизовать передачу данных с компьютером и избежать потерю важных данных

1. инициализация файловой системы

* при ошибке инициализации файловой системы не будут доступны следующие функции: восстановление последнего использованного чата Телеграм, значения Секрет и значение флага ОТА после переустановки программы, после отключения электропитания или после программного ресета (т.е. после всех случаев когда данные памяти RTC теряются); запись значения последнего использованного чата Телеграм, значения Секрет и значение флага ОТА

1. инациализация кнопок (отключено из-за багов)
2. инициализания ADC для замеров аккумулятора
3. инициализация SPI порта дисплея
4. проверка причины перезагрузки

* также восставливает значение последнего использованного чата Телеграм, значение Секрет и значение флага ОТА после переустановки программы, после отключения электропитания или после программного ресета (т.е. после всех случаев когда данные памяти RTC теряются)
* также уводит устройство в сон на 24 часа если срабатывает BROWN OUT детектор. Он срабатывает если заряда аккумулятора недостаточно для продолжения работы. В таком случае **[РАБОТА УСТРОЙСТВА ЗАКАНЧИВАЕТСЯ ЗДЕСЬ]**, пока аккумулятор ни будет заряжен.

1. проверка аккумулятора
   * + из-за технических особенностей устройства мы можем определить уровень заряда аккумулятора только когда он почти разряжен. Точные замеры аккумулятора можно проводить ориентировочно между 3% и 0% заряда аккумулятора. В показаниях ADC это соответствует 800 и 400.
     + берём 5 замеров уровня заряда и высчитываем их среднее значение
     + все показатели выше 800 означают, что аккумулятор достаточно заряжен и сообщать о разряженном аккумуляторе не требуется – выходим из функции
     + подключаемся к Wi-Fi чтобы сообщить о состоянии аккумулятора в Телеграм
     + все показатели ниже 400 означают полностью разряженный аккумулятор. Ни смотря на то, что не сработал BROWN OUT детектор в предыдущем шаге, с таким низким зарядом нельзя продолжать работу. Сообщаем о разряженном аккумуляторе в Телеграм, выводим на дисплей сообщение "Low battery" и уводит устройство в сон на 24 часа. В таком случае **[РАБОТА УСТРОЙСТВА ЗАКАНЧИВАЕТСЯ ЗДЕСЬ]**, пока аккумулятор ни будет заряжен.
     + показатели между 700 и 600 могут означать что устройство заряжается
     + если мы до сих пор не вышли из функции, но показатели ниже 800, значит аккумулятор уже разряжен, но ещё можно продолжить работать. Сообщаем о разряженном аккумуляторе в Телеграм, выводим на дисплей сообщение "Low battery" и продолжаем исполнение программы.
2. инициализация функции OTA (отключено из-за блокировки файерволом)
3. переход в режим OTA (отключено из-за блокировки файерволом)
4. делаем выбор в каком режиме продолжать работу: в режиме номера кластера (дефолтный режим) или в режиме экзамена
   * + Режим номера кластера отображает номер кластера + пиктограммы (в обычный день) или сообщение-предупреждение об экзамене (в день экзамена) или сообщения об ошибке (невозможность получить данные экзаменов, истёкший срок действия Секрета, низкий заряд аккумулятора). Этот режим активен 99% времени.
     + Режим экзамена включается 1 час заранее до начала экзамена, показывает предупреждение о скором начале экзамена, потом переключается в предупреждение о проходящем экзамене и после окончания экзамена переходит обратно в Режим номера кластера. Этот режим активен только в день экзамена, 1 час до начала экзамена + всё время проведения экзамена.
     + Двойная проверка флага состояния экзамена в этой функции необходима для перехода из одного режима в другой. Не менять!

**В РЕЖИМЕ НОМЕРА КЛАСТЕРА**

1. подключение к сети Wi-Fi
2. проверка входящих сообщений в Телеграм чате
   * + Через чат Телеграм может прийти новый Секрет, который будет полезен далее при запросе данных экзаменов
3. синхронизация времени, даты и летнего/зимнего режима времени с NTP сервером
   * + Без данных времени нельзя обеспечить надёжную работу устройства. Если поле нескольких попыток получить данные времени не удалось, на дисплей выводится ошибка а само устройство уводится в сон до следующего запланированного пробуждения. В таком случае **[РАБОТА УСТРОЙСТВА ЗАКАНЧИВАЕТСЯ ЗДЕСЬ]**, пока ни удастся получить данные времени при одном из будущих запланнированных пробуждений.
4. получаем данные экзаменов на актуальный день из Интры
   * + Без данных экзаменов нельзя обеспечить надёжную работу устройства. Если поле нескольких попыток получить данные экзаменов не удалось, на дисплей выводится ошибка а само устройство уводится в сон до следующего запланированного пробуждения. В таком случае **[РАБОТА УСТРОЙСТВА ЗАКАНЧИВАЕТСЯ ЗДЕСЬ]**, пока ни удастся получить данные экзаменов при одном из будущих запланнированных пробуждений.
     + заходим на сайт Интры
     + логинимся на сайте Интры
     + заходим на страницу расписания на сегодняшний день
     + читаем полученный HTML код до обнаружения данных об экзамене
     + очищаем данные от лишнего мусора
     + сверяем полученные данные с имеющимися данными в памяти
     1. если данные отличаются, то изменяем их на дисплее;
     2. если нет, то оставляем как есть
5. настраиваем время следующего включения
6. выключение питания дисплея
7. выключение

[GETTING READY TO MAINTAIN AND DEVELOP THE PROJECT.]

[How to Get Exams Info from Intra.]

The Smart Sign does it in the following 6 steps:

1. connects to Wi-Fi,
2. connects to the 42 Intra server,
3. asks the server for a temporary access token using the UID and the Secret,
4. retreives the temporary access token from the server response,
5. asks the server for exam information for a particular campus, a particular cluster, on a particalar date,
6. retreives the exam information from the server response.

For testing purposes, this process can be recreated on a computer, in Terminal using Curl:

1. enter these variables into the Terminal

**CLIENT\_ID=**put\_your\_42\_API\_app\_UID\_number\_here

**SECRET\_ID=**put\_your\_42\_API\_app\_Secret\_token\_here

2. ask the 42 Intra server for a temporary access token

**curl -X POST --data "grant\_type=client\_credentials&client\_id=${CLIENT\_ID}&client\_secret=${SECRET\_ID}" https://api.intra.42.fr/oauth/token**

3. copy the access token from the server response and enter it as a variable into the Terminal

**TKN=**put\_received\_access\_token\_here

4. ask the server to send you the information about exams in the cluster C3 and put it into a .json file. 56 is the ID of the 42 Prague campus. Curl does not like square brackets [ ] in its calls, so they need to be escaped with a backslash \.

**curl -H "Authorization: Bearer $TKN" "https://api.intra.42.fr/v2/campus/56/exams&filter\[location\]=C3" > c3\_exams.json**

If you want to filter the results down to the exact date, as the Smart Sign does, use the following call instead.

**curl -H "Authorization: Bearer $TKN" "https://api.intra.42.fr/v2/campus/56/exams?filter\[location\]=C3&range\[begin\_at\]=2024-07-12T05:00:00.000Z,2024-07-12T22:00:00.000Z" > c3\_exams1.json**

5. this command opens the .json file in the Terminal

**python -m json.tool < prague\_exams.json | grep "begin\_at" | tr -d " ," | awk -F '"begin\_at":' '{print("["++count"]:", $2)}'**

[example of the 42 server access token response as the smart sign sees it.]

HTTP/2 200

date: Thu, 11 Jul 2024 13:19:37 GMT

content-type: application/json; charset=utf-8

cache-control: no-store

etag: W/"77a2df7a4e20f5f76e6364d36bc76e8a"

pragma: no-cache

set-cookie: \_mkra\_stck=15e20a8020c702e70007eb1e185a06fb%3A1720703982.2018037; path=/; max-age=10; expires=Thu, 11 Jul 2024 13:19:47 -0000; HttpOnly

status: 200 OK

vary: Origin,Accept-Encoding

x-rack-cors: preflight-hit; no-origin

x-request-id: 3d153728-82b5-48a0-84e7-7c1f1efe598a

x-runtime: 0.076367

cf-cache-status: DYNAMIC

report-to: {"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v4?s=5%2Bb21KLqrzLETXPtKW2gerMAMrEPjiLAWT6eRUKeyuOVy3b5pvEr6Tc7D%2BMB%2BB4gqUHrTyXWaYy01CmZjQqUGReP7COyDKfBhKpl75Kwd%2FWrMWCVZD%2FkWhvM1iHF0V43hw%3D%3D"}],"group":"cf-nel","max\_age":604800}

nel: {"success\_fraction":0,"report\_to":"cf-nel","max\_age":604800}

server: cloudflare

cf-ray: 8a191610cec6bc03-FRA

{"access\_token":"03e4cb9b861dad6c49f2267cf97bd18a942507efa7840dc971008d264596cf89","token\_type":"bearer","expires\_in":6564,"scope":"public","created\_at":1720703340,"secret\_valid\_until":1722585613}

[example of the 42 server EXAM INFORMATION response as the smart sign sees it.]

[HOW TO DRAW ON THE DISPLAY.]

[SERVICE MESSAGES MEANING.]

[LIBRARIES AND THEIR USE.]

The project was built in Arduino IDE 1.8.19.

It uses board 'esp32' version 2.0.11

The libraries in bold are explicitly included in the project.

|  |  |  |
| --- | --- | --- |
| **Arduino.h** |  | String variables manipulations |
| **LittleFS** | 2.0.0 | stores data even without electricity (Telegram chat number, Secret, OTA flag value) |
| FS | 2.0.0 | dependency for the LittleFS library |
| **ArduinoOTA** | 2.0.0 | for the Over The Air update functionality |
| **WiFiUdp** | 2.0.0 | dependency for the ArduinoOTA library |
| **ESPmDNS** | 2.0.0 | dependency for the ArduinoOTA library |
| Update | 2.0.0 | dependency for the ArduinoOTA library |
| **time.h** |  | gets NTP Server date and time; deciphers UNIX timestamp for the SECRET expiration date |
| **stdio.h** |  | provides printf() function for the DEBUG macro |
| **stdint.h** |  | provides fixed-width integer types |
| **esp\_system.h** |  | allows to use ESP-IDF native functions |
| **esp\_sleep.h** |  | allows to use the Deep Sleep power-saving functionality |
| **driver/adc.h** |  | for battery charge measurements |
| **Wire** | 2.0.0 | for SPI reconfiguration in the ft\_display\_init function |
| SPI | 2.0.0 | dependency for the Wire library |
| **GxEPD2\_3C** | 1.5.2 | 3-coloured version of the GxEPD2 library for e-paper displays |
| **GxEPD2\_BW** | 1.5.2 | dependency for the GxEPD2\_3C library |
| Adafruit\_GFX\_Library | 1.11.8 | dependency for the GxEPD2\_3C library |
| Adafruit\_BusIO | 1.14.4 | dependency for the GxEPD2\_3C library |
| **Fonts/FreeSansBold24pt7b.h** |  | the fonts come from the Adafruit GFX library which gets called by the GxEPD2 library |
| **WiFi** | 2.0.0 | for Wi-Fi functionality |
| **WiFiClientSecure** | 2.0.0 | for secure HTTPS requests |
| **UniversalTelegramBot** | 1.3.0 | Telegram bot; for wireless SECRET update and low battery notifications |
| ArduinoJson | 6.21.3 | dependency for the UniversalTelegramBot library |

[BUGS AND SUGGESTIONS HOW TO FIX THEM.]

***Display does not work / does not draw an image.***

* you may have miscalculated the image coordinates and the image gets drawn outside of the display field of coordinates. Remember that setRotation also rotates the display field of coordinates. Remember that coordinates always point to the top left corner of an image, but for a text it is the bottom left pixel of the first character in the first line.
* you may have misaligned the image with the partial update window if you are using one. Remember that even when you draw an image in a partial update window, you use the display field of coordinates to place it; a partial update window does not have its own field of coordinates.
* the display driver memory may be full. Run your program and open the Serial monitor. When an image gets drawn on the display, in the Serial monitor it says “Updating xxxxxxxxxx” where xxxxxxxxxxxx is a very long number that can be different each program run. Among all the outputted messages find the “Updating” message that corresponds to your image being drawn. Now look higher and find the “Power on” message. Keep looking higher and find the “Power off” message, it should not be separated from your “Updating” message by other “Updating” messages. If you cannot find it, it means that this is a bug. To fix it, in your code use the function to force the display to power off before drawing the image. The display will power back on automatically.

***DEBUG\_PRINTF causes a compilation error.***

* the DEBUG\_PRINTF macro is designed in such a way, that every time it is called, it shall be given exactly 2 arguments. If you need to output only one argument, simply send an empty string as the second argument. An empty string looks like quotation marks with nothing between them (“”). Look for examples in the program code.

***DEBUG\_PRINTF does not output a message.***

* the DEBUG\_PRINTF macro cannot output String type variables natively. To do that, you need to explicitly cast the String variable into the C-style string with c\_str() command. Look for examples in the program code.

***Serial monitor is empty / outputs gibberish.***

* check the DEBUG macro in the 42-Prague-Smart-Sign.h header file. The DEBUG definition should not be commented out for the Serial output to work. Additionally, you can set the Core Debug Level to "Verbose" in the Arduino IDE Tools to get detailed information about the firmware processes.
* make sure that the baud rate in the Serial monitor is set to the same baud rate as in the ft\_serial\_init() function of the program.
* you may encounter such behaviour right after the software update. It is normal. Try closing and opening again the Serial monitor window. If that does not help, push the Reset ("R") button on the module.

***Serial monitor skips some messages / does not show some messages.***

* it is a common situation at the beginning of the program. Serial communication between the computer and the microcontroller needs time to stabilise and synchronise itself. ESP32-C3 USB Serial is especially prone to this issue. To overcome it, increase the delay inside of ft\_serial\_init() or add a few empty messages to be outputted after the Serial.begin() command. You may well try to implement both of the suggested solutions at the same time.

***Wi-Fi does not connect / reconnect without apparent reason.***

* thoroughly check your network SSID and password spelling. Surprisingly, it is a very widely spread cause. A single character written small instead of capital may easily prevent you from connecting.
* make sure not to use ft\_delay() in any of your functions responsible for connecting or reconnecting to Wi-Fi. The ft\_delay() function not only delays the program execution but also puts the microcontroller's inner Wi-Fi module to sleep. Using ft\_delay() in functions responsible for retrieving information from the Internet may result in unexpected behaviour. If you are not sure that using ft\_delay() is safe in your particular function, use delay() instead.

***OTA does not work. Cannot see the device in the ports list.***

* make sure that the Sign and your computer are connected to the same Wi-Fi network.
* try closing and reopening Arduino IDE.
* the school firewall may be blocking OTA connection. Ask your campus system administrator if it could be overcome.

***Adding multiple Strings together with the “+” command causes compilation error.***

* strangely, sometimes the compiler may not like it in one part of the code and be completely fine with it in another. The solution is to explicitly cast the variable after the first “+” command into String with the String(your\_variable\_or\_text) command. Understandably, it is strange to cast a String variable into String, but it works.

***WARNING: Skipping SSL Verification. INSECURE!***

* not a bug.
* this message appears when connecting to the Intra server and is caused by the following line in the intra\_interaction.cpp file: „client1.setInsecure();“.
* one one hand, it can be solved by getting and setting up a certificate for this connection. On the other hand, it does not affect the program run at all and can be ignored.

***setSocketOption(): fail on 0, errno: 9, "Bad file number"***

* a minor issue and does not necessarily indicate a problem with the program.
* this message may appear when the Smart Sign fails the first attempt to get a server response from the Intra server and goes for the second or third attempt.
* this error can occur when you try to set a socket option on a socket that has already been closed or is in the process of being closed. This can happen during the transition between closing the previous connection and opening a new one. As long as the SSL/TLS communication with the Intra server is functioning correctly after the reconnection, this error can generally be ignored.

***spiAttachMISO(): SPI Does not have default pins on ESP32C3!***

* not a bug.
* This message appears when the microcontroller assigns pins for the display SPI port. In this project we do not use the MISO pin (thus the „-1“ value defined for the SPI\_MISO\_PIN in the constants.h file).

***401 Unauthorized. Error! Server response came without the Access Token.***

* often happens when something is wrong with the Secret token authentication, commonly with the Secret token itself. Most likely, an extra character was added to your Secret token somewhere along the way. The character may even not to be visible in the Serial monitor. It may happen when you write to or read from the filesystem files. Try using trim() on the variable (e.g. your\_string\_variable.trim();), it will remove spaces and/or new line signs at the beginning and at the end of the string.
* rarely may happen due to the Intra server maintenance. There is no solution to it but to wait.

***Compilation error: “Section .dram0.bs 'Will Not Fit In Region Dram0\_0\_seg' Region.`Dram0\_0\_seg 'Overflowed by 9648 Bytes. Collect2: Error: LD Returned 1 Exit Status”***

* it means that the program takes more RAM space than it is available. DRAM stands for Data Random Access Memory and is used for data.
* This error may be caused for example by excessive use of global variables, large arrays, big buffers, etc.
* The most likely reason for this error in this project is the display buffer being too big. The ESP32 and the ESP32-S2 are especially prone to this issue. To overcome this problem the display buffer size should be reduced. It can be done in the display instantiation. Full display buffer instantiation: GxEPD2\_3C<GxEPD2\_750c\_Z08, GxEPD2\_750c\_Z08::HEIGHT> display(GxEPD2\_750c\_Z08(SPI\_SS\_PIN, DC\_PIN, RST\_PIN, BUSY\_PIN)); Reduced display buffer instantiation: GxEPD2\_3C<GxEPD2\_750c\_Z08, GxEPD2\_750c\_Z08::HEIGHT/2> display(GxEPD2\_750c\_Z08(SPI\_SS\_PIN, DC\_PIN, RST\_PIN, BUSY\_PIN));
* Less likely reason for this error is an excessive use of the global scope for data. To solve it reduce the number of global variables, use the file system to store the data instead of arrays.
* Software update fails while trying to connect to the microcontroller board
* Go into the "Tools" menu and change the Upload Speed to 115200. Sometimes the IDE automatically sets the Upload Speed to the highest value and your board may happen not to support it.

[External Information Sources.]

Документация к данным NTP сервера: <https://cplusplus.com/reference/ctime/tm/>

Страница дисплея **GDEY075Z08** <https://www.laskakit.cz/good-display-gdey075z08-7-5--800x480-epaper-cerveny-displej/>

Даташит дисплея **GDEY075Z08** <https://www.laskakit.cz/user/related_files/gdey075z08.pdf>  Пример кода для дисплея от LaskaKit <https://github.com/LaskaKit/Testcode_examples/blob/main/Displays/E-Paper/7-50/GDEY075Z08_GxEPD2/GDEY075Z08_GxEPD2.ino>

Драйвер дисплея называется UC8179, вот даташит <https://www.laskakit.cz/user/related_files/uc8179.pdf>

Библиотека для использования дисплея называется **GxEPD2**, вот страница <https://github.com/ZinggJM/GxEPD2>

Форум с вопросами по фичам и багам библиотеки **GxEPD2** с дисплеем **GDEY075Z08** <https://forum.arduino.cc/t/good-display-epaper-for-arduino/419657>

Страница обсуждения ошибки размера RAM памяти у ESP32 <https://github.com/espressif/arduino-esp32/issues/1163>

Инструкция к **XIAO ESP32C3** <https://wiki.seeedstudio.com/XIAO_ESP32C3_Getting_Started/>

Видео туториал по устранению ошибок с ArduinoOTA <https://www.youtube.com/watch?v=z_btZfxrS48>

Самая детальная инструкция по **использованию кнопок** с ESP32, которая только существует <https://esp32io.com/tutorials/esp32-button>