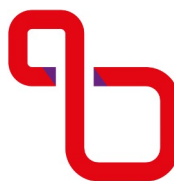




**Politechnika  
Śląska**



**UCZELNIA  
BADAWCZA**  
INICJATYWA DOSKONAŁOŚCI

# Raport

Rodzaj zajęć

## Projekt

Przedmiot

## Systemy Mikroprocesorowe i Wbudowane

Rok akademicki	Miasto	Kierunek	Semestr	Prowadzący	Grupa	Sekcja
2025/2026	G	Inf	5	GB	5	9
Planowany termin wykonywania ćwiczenia			Faktyczny termin wykonania ćwiczenia			
Data	Godzina		Data		Godzina	
26.01.2026	11:30-13:00		26.01.2026		11:30-13:00	
Numer ćwiczenia	Temat ćwiczenia					
	Projekt – Asystent Snu					

Skład sekcji		
	Imię i nazwisko	Uwagi
1	Piotr Szlejter	
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		

## **1. Temat Projektu:**

Asystent Snu – urządzenie mierzące czynniki wpływające na sen.

## **2. Linki do repozytorium:**

- <https://drive.google.com/drive/folders/1vB9gLUQ3DNj9q-B88YDbHVliXITGyozg?usp=sharing>

- <https://github.com/xxxsiwydymxxx/AsystentSnu.git>

## **3. Opis założeń projektu:**

- Urządzenie mierzące czynniki mające wpływ na sen.
- Urządzenie składa się z stacji bazowej, oraz nakładanego na palec, bezprzewodowego urządzenia – czujnika.
- Urządzenie mierzy: temperaturę, ciśnienie atmosferyczne, wilgotność powietrza, puls, poziom natężenia tlenu w krwi.
- Urządzenie wyświetla na bieżąco zmierzone wartości i zapisuje je na kartę microSD ze znacznikiem czasowym.
- Zestaw składa się z 2 urządzeń łączących się bezprzewodowo, zasilanie głównego urządzenia za pomocą kabla USB, urządzenia dodatkowego za pomocą akumulatora z możliwością podłączenia go do ładowania.
- Urządzenie dodatkowe zasilane przez akumulator przez czas przynajmniej 10 godzin.
- Urządzenie dodatkowe ma możliwość podłączenia do ładowania.
- Urządzenie dodatkowe ma możliwość wyłączenia się w celu nie zużywania energii.
- Urządzenie dodatkowe przesyła do urządzenia głównego rezultaty pomiarów pulsu i nasycenia tlenem krwi.

## **4. Analiza zadania:**

Potrzebne były 2 mikrokontrolery, pełniące funkcję jednostki głównej, przetwarzającej pomiary i komunikujące się ze sobą. Ze względu na energooszczędność względem Wi-fi, i to że urządzenia będą współpracować ze sobą w obrębie jednego pomieszczenia wybrane zostały urządzenia posiadające moduł BLE – Bluetooth low energy.

Czujniki zostały dobrane tak, aby w rozsądnej kwocie mierzyły jak najwięcej czynników mogących mieć wpływ na sen.

Urządzenie dodatkowe powinno być zasilane z baterii aby ograniczyć możliwość zerwania kabla w trakcie snu, gdy czujnik przymocowany jest do palca.

## **4. Wybrane podzespoły:**

### **a) urządzenie główne:**

- mikrokontroler: Raspberry Pi pico 2 w, wybrane z powodu niskiej ceny, możliwości Bluetooth / Wi-fi, wysokiej wydajności mikrokontrolera.

- czujniki: czujnik bme280: niska cena, możliwości pomiaru wilgotności, ciśnienia atmosferycznego oraz temperatury na jednym module.

- wyświetlacz OLED: Waveshare zgodny z sterownikiem ssd1306: wyświetlacz jest dość czytelny dla małej ilości danych, oraz był już dostępny w zasobach własnych.

- czytnik kart microSD: najtańszy model dostępny w sklepie Botland komunikujący się przez SPI.

- Moduł RTC: HW-111: moduł pozwoli na zapisywanie danych na kartę microSD ze znacznikiem czasowym.

### **b) urządzenie dodatkowe:**

- mikrokontroler: Seeed xiao NRF52840: wybrany ze względu na wbudowaną ładowarkę baterii, możliwości łączenia się przez bluetooth, oraz niski pobór mocy.

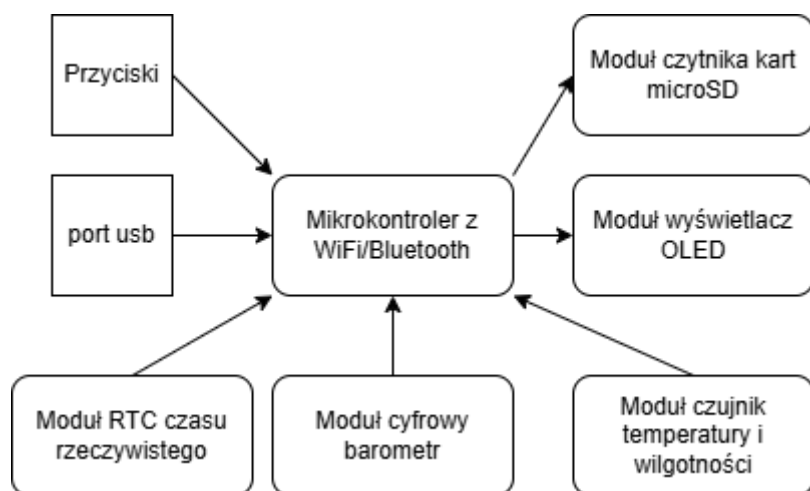
- czujnik tętna: M5stack heart MAX30100: wybrany ze względu na gotową obudowę i możliwość odczytu tętna i nasycenia tlenem krwi.

- akumulator li-pol akyga 900 mAh

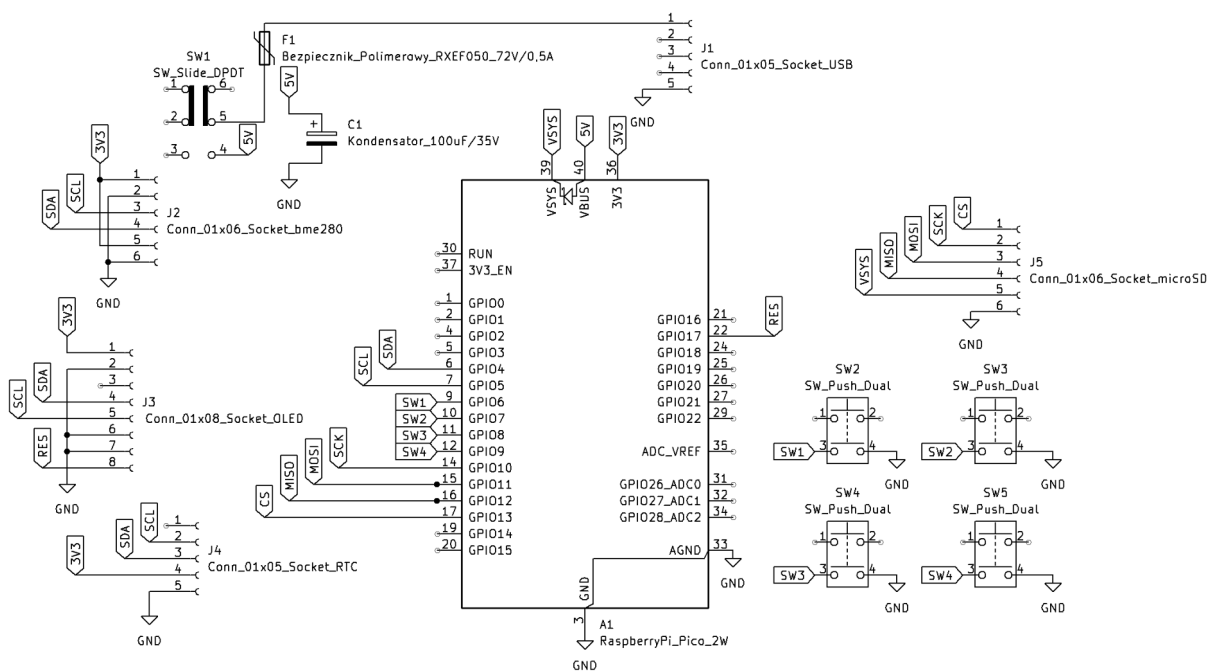
## 5. Specyfikacja wewnętrzna:

### a) Schematy blokowe i ideowe

- urządzenie główne schemat blokowy:



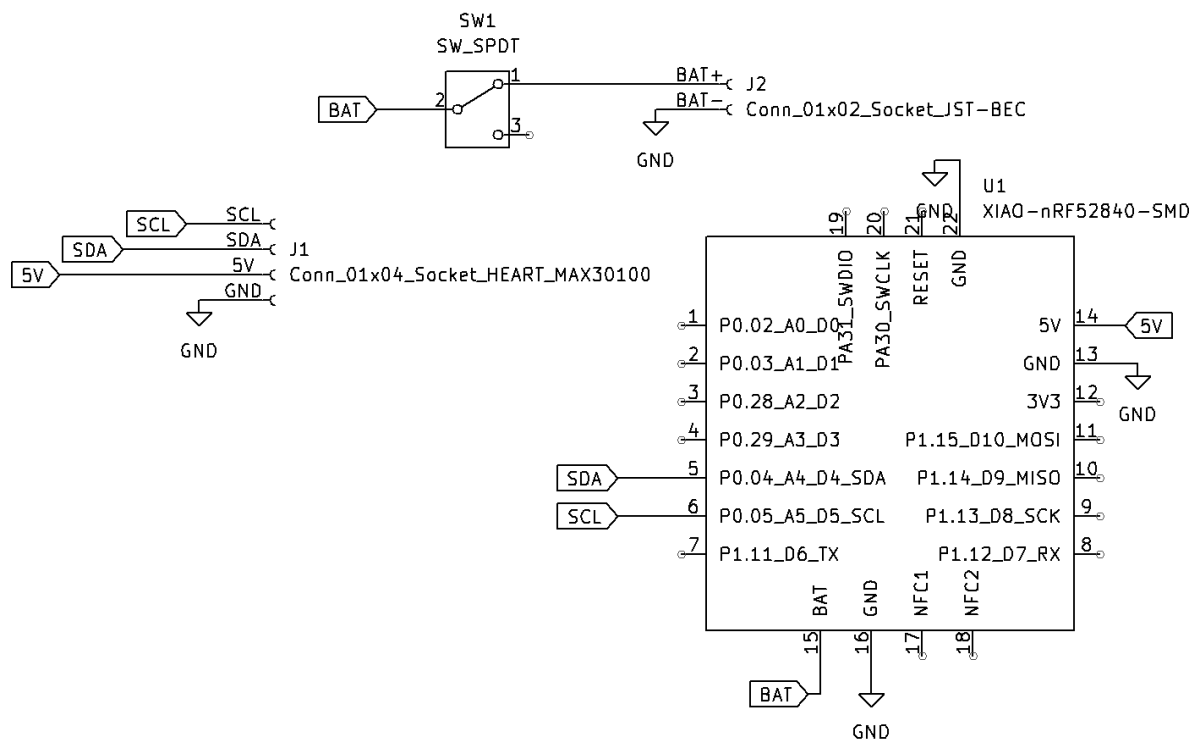
- urządzenie główne schemat ideowy:



- urządzenie dodatkowe schemat blokowy:



- urządzenie dodatkowe schemat ideowy:



## b) Opis funkcji poszczególnych bloków układu:

- urządzenie główne:

Blok Przyciski: przyciski tactile switch podłączone od portu do masy.

Blok port USB: zewnętrzny port USB pozwalający dostarczyć zasilanie, z możliwością wyłączenia urządzenia przyciskiem.

Blok mikrokontrolera: mikrokontroler Raspberry pi pico 2w, zasilany przez pin Vbus z portu USB.

Blok moduł RTC: moduł RTC z baterią CR2032, dzięki baterii przechowuje zapisany czas przy wyłączonym urządzeniu, baterię należy wymienić co ok 10 lat.

Blok moduł czytnika kart microSD: połączony przez SPI moduł pozwalający dołączyć kartę microSD sformatowaną w formacie fat32.

Blok moduł wyświetlacz OLED: wyświetlacz z możliwością komunikowania się przez i2c lub SPI, skonfigurowany przez przelutowanie rezystora na komunikację po i2c. Wymaga podłączenia pinu reset do portu GPIO mikrokontrolera.

Blok moduł czujnik temperatury i wilgotności: moduł BME280, komunikuje się przez i2c. Wynik ciśnienia atmosferycznego nie jest ustandaryzowany do poziomu morza.

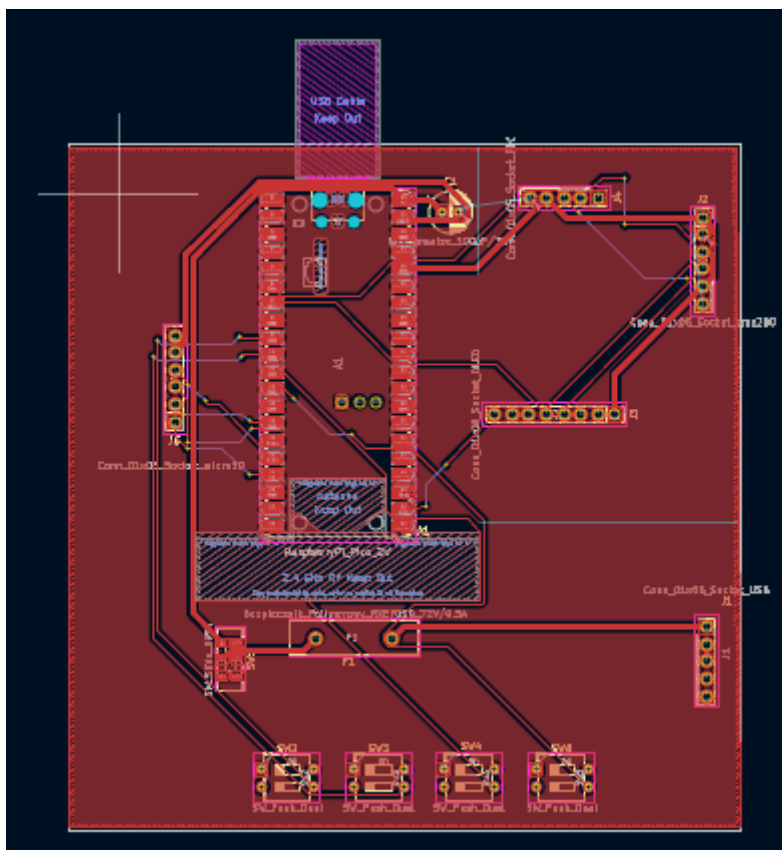
- urządzenie dodatkowe:

Blok mikrokontrolera: Seeed Xiao NRF52840, energooszczędny mikrokontroler, mogący komunikować się poprzez bluetooth, do jego pinów BAT+ i BAT- przylutowany został przewód do akumulatora/

Blok czujnik tętna i saturacji krwi: m5stack heart max 30100, czujnik działa przez podświetlenie diodą podczerwoną palca, lub nadgarstka, i przez czujnik monitorujące tętno i saturację tlenem krwi. Aby mierzyć tętno w nadgarstku trzeba konfigurując czujnik, ustawić zasilanie diody na większe niż domyślne 7,6 mA dla palca, ponieważ nie wystarcza wtedy światła aby przebić skórę. Dla palca natomiast ustawienie 7.6 mA w zupełności wystarcza i większy prąd powoduje że czujnik jest przejaszczony i niepoprawnie odczytuje pomiar.

**c) Schemat montażowy obejmujący projekt płytki drukowanej, rozmieszczenie elementów na płytce, rozmieszczenie elementów urządzenia w obudowie:**

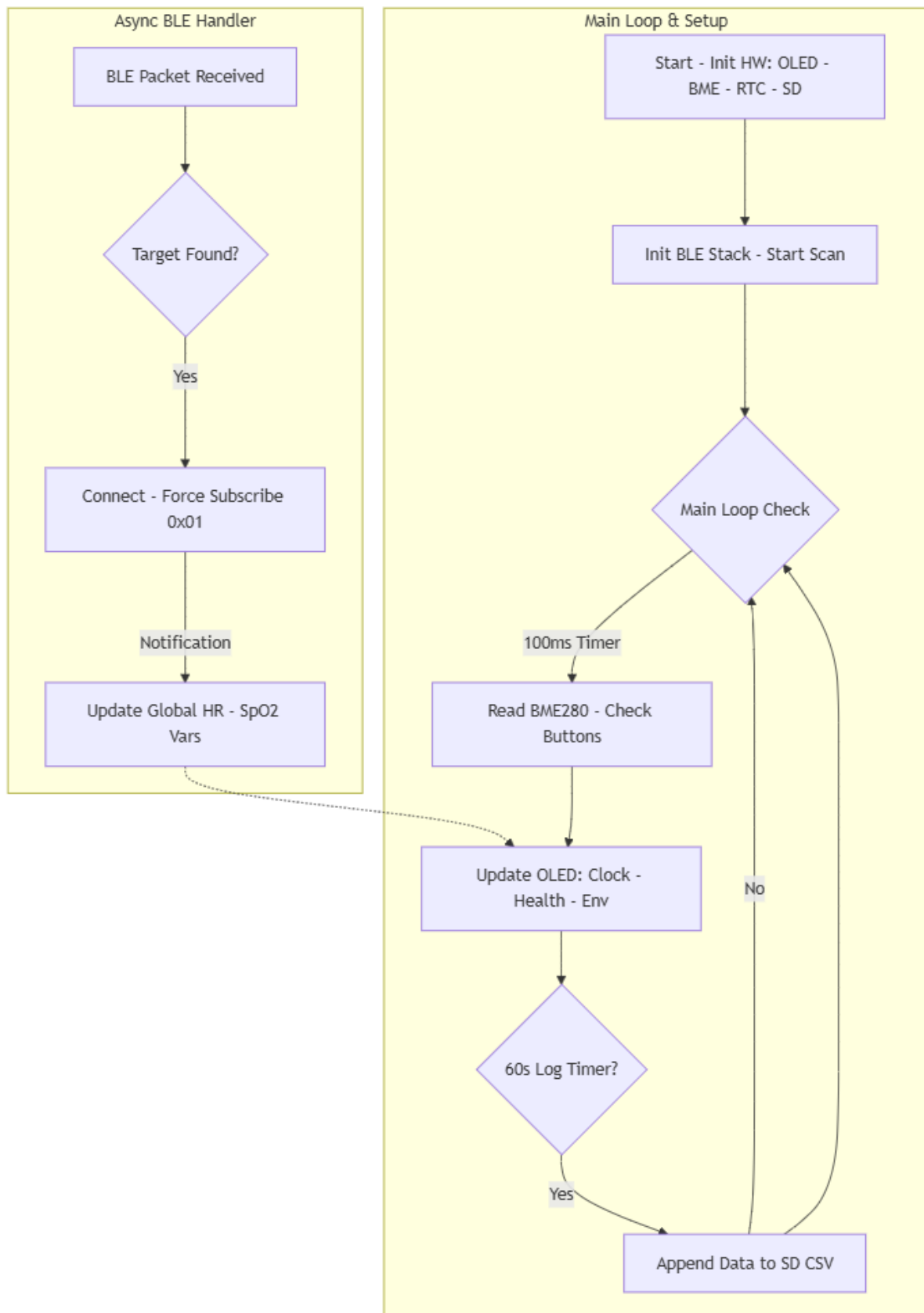
- urządzenie główne:



- urządzenie dodatkowe nie wymaga płytki drukowanej ponieważ składa się jedynie z mikrokontrolera, gotowego przyłącza baterii na nim i 1 czujnika z gotową obudową.

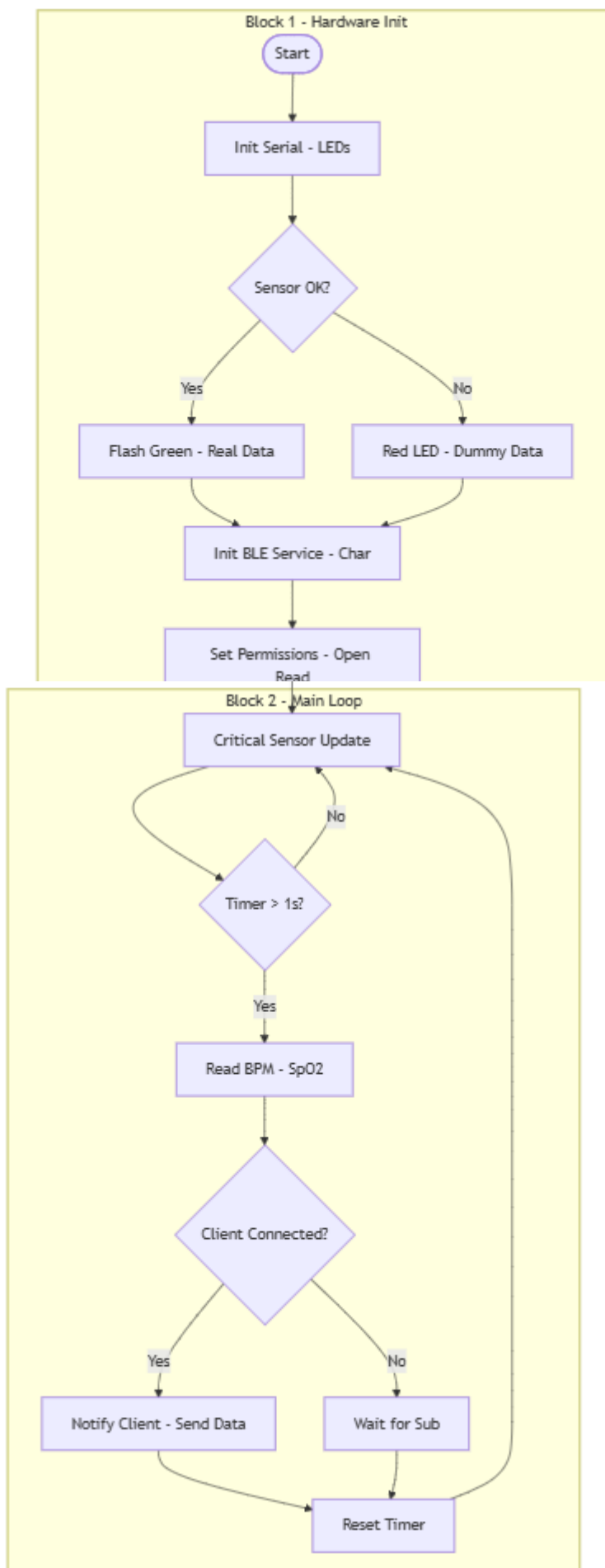
#### d) Algorytm oprogramowanie urządzenia:

- urządzenie główne:





- urządzenie dodatkowe:



## e) Opis wszystkich zmiennych:

### - urządzenie główne:

zmienne potrzebne dla bluetooth:

```
const char* target_name = "XIAO_Health_Native";

const uint16_t target_service_uuid = 0xFF00;
const uint16_t target_char_uuid    = 0xFF01;

static btstack_packet_callback_registration_t hci_event_callback_registration;
static hci_con_handle_t connection_handle = HCI_CON_HANDLE_INVALID;
static gatt_client_notification_t notification_listener;

static gatt_client_service_t found_service;
static gatt_client_characteristic_t found_char;
static bool service_found = false;
static bool char_found = false;
static bool is_subscribed = false;
```

zmienne potrzebne dla wyświetlania wartości przesłanych przez bluetooth:

```
volatile int global_hr = 0;

volatile int global_spo2 = 0;
bool bleConnected = false;
```

zmienne potrzebne dla podłączonych czujników, wyświetlacza i przycisków:

```
// --- HARDWARE SETTINGS ---

// 1. I2C
#define I2C_SDA_PIN 4
#define I2C_SCL_PIN 5
#define OLED_RESET 17

// 2. SPI1
#define SD_SCK 10
#define SD_MOSI 11
#define SD_MISO 12
#define SD_CS 13
```

```
// 3. BUTTONS
#define BTN1_PIN 6
#define BTN2_PIN 7
#define BTN3_PIN 8
#define BTN4_PIN 9

#define BME_ADDR    0x76
#define OLED_ADDR    0x3C

Adafruit_SSD1306 display(128, 64, &Wire, OLED_RESET);
Adafruit_BME280 bme;
RTC_DS3231 rtc;

uint32_t tsLastReport = 0;
uint32_t tsLastLog = 0;
const long REPORT_PERIOD = 100;
const long LOG_PERIOD    = 60000;

bool sdStatus = false;
int  currentMode = 1;
```

### - urządzenie dodatkowe:

wbudowane LED:

```
#define LEDR 11
#define LEDG 12
#define LEDB 13
```

zmienne przechowujące dane z czujnika:

```
PulseOximeter pox;

uint32_t tsLastReport = 0;
bool sensorWorking = false;
```

zmienne potrzebne dla bluetooth:

```
const uint16_t CUSTOM_SERVICE_UUID = 0xFF00;

const uint16_t CUSTOM_CHAR_UUID    = 0xFF01;

BLEService      customService = BLEService(CUSTOM_SERVICE_UUID);
BLECharacteristic customChar  = BLECharacteristic(CUSTOM_CHAR_UUID);
```

## **f) Opis funkcji wszystkich procedur:**

- urządzenie główne:

```
void setup();
```

Funkcja inicjalizuje oprogramowanie.

```
void loop();
```

Główna pętla programu.

```
void logData(DateTime dt, float t, float h, float p, int hr, int spo2);
```

Funkcja logująca dane do karty microSD

```
void updateScreen(DateTime now, float t, float h, float p, bool sdOk, int mode);
```

Funkcja wyświetlająca dane na wyświetlacz OLED.

```
static void packet_handler(uint8_t packet_type, uint16_t channel, uint8_t  
*packet, uint16_t size);
```

Jest to główna funkcja zarządzająca stanami Bluetooth, która odpowiada za skanowanie otoczenia, łączenie się z wykrytym urządzeniem "XIAO\_Health\_Native" oraz ponowne uruchamianie skanowania po rozłączeniu .

```
static void gatt_event_handler(uint8_t packet_type, uint16_t channel, uint8_t  
*packet, uint16_t size);
```

Funkcja ta odbiera przychodzące powiadomienia - notyfikacje z danymi i wpisuje odebrane wartości tętna oraz SpO2 do zmiennych globalnych .

```
static void on_characteristics_discovered(uint8_t packet_type, uint16_t channel, uint8_t *packet, uint16_t size);
```

Weryfikuje znalezienie szukanej charakterystyki - 0xFF01 i po jej wykryciu wymusza subskrypcję powiadomień poprzez ręczny zapis wartości 0x01 do deskryptora .

```
static void on_services_discovered(uint8_t packet_type, uint16_t channel, uint8_t *packet, uint16_t size);
```

Sprawdza, czy połączone urządzenie posiada właściwą usługę – 0xFF00 i jeśli tak, uruchamia proces wyszukiwania w niej charakterystyk .

- urządzenie dodatkowe:

```
void onBeatDetected()
```

Funkcja załączający LEDy po wykryciu tętna.

```
void setup()
```

Funkcja inicjalizująca.

```
void loop()
```

Główna pętla programu.

#### **g) Opis interakcji oprogramowania z układem elektronicznym:**

- urządzenie główne:

Oprogramowanie steruje magistralą I2C w celu inicjalizacji i komunikacji z wyświetlaczem OLED oraz czujnikami środowiskowymi BME280 i modułem czasu rzeczywistego DS3231 . Kod cyklicznie monitoruje cyfrowe wejścia podłączone do czterech przycisków fizycznych i na ich podstawie zmienia tryb

prezentacji danych na ekranie . System zarządza również magistralą SPI do obsługi karty pamięci SD na której regularnie zapisuje logi z danymi pomiarowymi w pliku tekstowym . Równolegle stos Bluetooth Low Energy pracuje w trybie centralnym skanując otoczenie w poszukiwaniu czujnika medycznego i po nawiązaniu połączenia odbiera od niego pakiety danych .

- urządzenie dodatkowe:

Program obsługuje czujnik pulsoksymetru MAX30100 poprzez magistralę I2C i wykonuje w głównej pętli krytyczne czasowo operacje przetwarzania sygnału biometrycznego. Mikrokontroler steruje diodami LED RGB sygnalizując kolorem zielonym poprawną pracę sensora lub czerwonym błąd inicjalizacji i przejście w tryb generowania danych testowych . Oprogramowanie uruchamia stos Bluetooth w trybie peryferyjnym rozgłaszając dedykowaną usługę i po otrzymaniu żądania subskrypcji wysyła do urządzenia nadrzędnego powiadomienia z aktualnym tętnem i nasyceniem tlenem .

#### **h) Szczegółowy opis ważniejszych procedur:**

- urządzenie główne:

```
static void packet_handler(uint8_t packet_type, uint16_t channel, uint8_t *packet, uint16_t size);
```

Jest to główna funkcja zarządzająca stanami Bluetooth, która odpowiada za skanowanie otoczenia, łączenie się z wykrytym urządzeniem "XIAO\_Health\_Native" oraz ponowne uruchamianie skanowania po rozłączeniu .

```
static void gatt_event_handler(uint8_t packet_type, uint16_t channel, uint8_t *packet, uint16_t size);
```

Funkcja ta odbiera przychodzące powiadomienia - notyfikacje z danymi i wpisuje odebrane wartości tętna oraz SpO2 do zmiennych globalnych .

```
static void on_characteristics_discovered(uint8_t packet_type, uint16_t channel, uint8_t *packet, uint16_t size);
```

Weryfikuje znalezienie szukanej charakterystyki - 0xFF01 i po jej wykryciu wymusza subskrypcję powiadomień poprzez ręczny zapis wartości 0x01 do deskryptora .

```
static void on_services_discovered(uint8_t packet_type, uint16_t channel,
uint8_t *packet, uint16_t size);
```

Sprawdza, czy połączone urządzenie posiada właściwą usługę – 0xFF00 i jeśli tak, uruchamia proces wyszukiwania w niej charakterystyk .

- urządzenie dodatkowe:

```
void loop()
```

Główna pętla programu nieustannie wywołuje metodę aktualizacji obiektu pulsoksymetru o ile flaga stanu sensora jest aktywna co pozwala na precyzyjne śledzenie uderzeń serca w czasie rzeczywistym. Co jedną sekundę kod sprawdza licznik czasu i pobiera bieżące wartości tętna oraz nasycenia krwi tlenem lub w przypadku awarii sprzętu podstawia sztywne wartości testowe wynoszące odpowiednio 65 BPM i 98 procent . Następnie procedura weryfikuje stan połączenia Bluetooth i jeśli klient jest połączony oraz aktywował subskrypcję powiadomień to wysyła dwubajtowy pakiet danych do urządzenia centralnego . W przypadku braku aktywnej subskrypcji lub po wysłaniu danych program wypisuje odpowiednie komunikaty diagnostyczne na port szeregowy i resetuje licznik czasu w celu odmierzenia kolejnego cyklu pomiarowego .

## **5. Specyfikacja zewnętrzna:**

### **a) Opis funkcji elementów sterujących urządzeniem:**

- urządzenie główne:

Przyciski tactile switch przełączają urządzenie między czterema różnymi wyświetlanymi ekranami, na 1 znajduje się data i godzina, na 2 znajdują się pomiary z urządzenia dodatkowego przesłane przez bluetooth, na 3 znajdują się wszystkie pomiary na jednym ekranem łącznie z temperaturą, wilgotnością i ciśnieniem, na 4 ekranie nie wyświetla się nic, służy on jako zgaszony ekran i tylko zapisuje na kartę microSD w tle.

Przełącznik suwakowy pozwala na włączenie i wyłączenie urządzenia.

-urządzenie dodatkowe:

Przycisk bistabilny pozwala na włączenie i wyłączenie urządzenia, nie będzie on działał gdy urządzenie podłączymy przez port usb typu C do ładowania.

## **b) Opis funkcji elementów wykonawczych:**

- urządzenie główne:

Wyświetlacz OLED wyświetla 4 różne ekrany opisane zawierające różne dane z czujników.

Moduł kart microSD pozwala na włożenie karty microSD, i zapisywanie na nią pomiarów wraz z znacznikiem czasowym z modułu RTC.

## **c) Opis reakcji oprogramowania na zdarzenia zewnętrzne:**

- urządzenie główne:

Po włączeniu urządzenia wykonuje ono próbę połączenia się przez bluetooth, gdy w pobliżu znajduje się urządzenie dodatkowe włączone, to urządzenia nawiążą połączenie, następuje również inicjalizacja modułu kart microSD, jeśli się nie powiedzie wyświetli się na ekranie komunikat.

Po przyciśnięciu przycisku urządzenie reaguje przełączając tryb pracy – zawartość wyświetlaną na ekranie.

- urządzenie dodatkowe:

urządzenie włączy się po przyciśnięciu przycisku bistabilnego, i zacznie propagację siebie w sieci bluetooth, sygnalizując to diodą LED, po połączeniu dioda LED zgaśnie, po przyłożeniu palca do czujnika m5stack Heart max 30100, dioda LED zapali się po wykryciu pulsu.



#### **d) Skrócona instrukcja obsługi urządzenia:**

Urządzenie główne należy włączyć do zasilania 5V min 1A przez port micro USB dodatkowy z boku urządzenia, nie ten wbudowany w Raspberry pi pico 2w, po włączeniu można wybrać tryb pracy przyciskając odpowiedni przycisk.

Urządzenie dodatkowe należy wyłączać przyciskiem bistabilnym, można je również podłączyć do ładowania przez wbudowany w Seeed Xiao port USB typu C.

Aby połączyć urządzenia należy najpierw włączyć urządzenie dodatkowe, a później włączyć urządzenie główne, połączenie nastąpi automatycznie.

### **6. Opis montażu i uruchamiania:**

#### **a) Jakie problemy wystąpiły podczas montażu i uruchamiania i jak zostały rozwiązane.**

- Problem z wyświetlaczem OLED: wyświetlacz OLED zdawał się nie działać, nie wyświetlał nic, po podłączeniu pinu RES z wyświetlacza do portu GPIO pico i odpowiedniej obsłudze programowej wyświetlacz zaczął działać

- Problemy z połączeniem bluetooth, urządzenie główne z raspberry pi pico 2 w, zdawało się nie działać, łączyło się po bluetooth ale nie pobierało danych przesyłanych, podobnie po skonfigurowaniu jako serwer i połączeniu po bluetooth z telefonem, testowano z użyciem apki NRF connect, nie transmitowało danych. Problem występował przy użyciu biblioteki bluetooth Adafruit. Problem został rozwiązany po przejściu na wspieraną przez raspberry pi pico bibliotekę bluetooth btstack.h

- Głównym problemem była niemożność automatycznego odnalezienia przez Pico właściwego adresu sterującego w pamięci Bluetooth urządzenia XIAO. Standardowa funkcja biblioteki BTstack gubiła się przy próbie zapisu do deskryptora konfiguracji, co uniemożliwiało włączenie powiadomień o nowych danych. Dodatkowo kod na XIAO miał zbyt restrykcyjne ustawienia uprawnień, które blokowały próby zapisu ze strony niesparowanego urządzenia takiego jak Pico. Skutkowało to sytuacją, w której XIAO próbowało wysyłać dane w próżnię otrzymując błąd transmisji, podczas gdy Pico czekało w nieskończoność na potwierdzenie subskrypcji. Rozwiązanie polegało na obejściu automatycznego wyszukiwania i ręcznym wyliczeniu adresu sterującego poprzez dodanie jedynek

do adresu charakterystyki głównej. Dzięki temu mogliśmy wysłać surową komendę aktywacyjną bezpośrednio pod wskazany adres ignorując standardowe procedury negocjacji. Równocześnie zmodyfikowaliśmy kod nadajnika, aby zezwalał na otwarty dostęp do zapisu i wstrzymywał wysyłanie pomiarów do momentu poprawnego zestawienia połączenia.

#### **b) Jakie przeprowadzono testy poprawności działania urządzenia:**

Przeprowadzono testy łączności w obrębie jednego pomieszczenia, przeprowadzono testy poszczególnych komponentów łącząc je tylko z mikrokontrolerem i wgrywając testowy kod.

Przeprowadzono obszerne testy łączności bluetooth z różnymi wersjami kodu łączącymi się z telefonem – aplikacją NRF Connect, lub ze sobą i wypisujące na Serial monitor.

### **7. Wnioski z uruchamiania i testowania:**

- Podczas testowania wyszedł problem w urządzeniu dodatkowym polegający na tym że urządzenie dodatkowe po zasileniu z baterii 3,7 V, nie ustawia zasilania 5v na pinie 5V, przez co urządzenie m5stack Heart max 30100 nie włączało się ponieważ mimo logiki 3,3V urządzenie to jest zasilane z 5V, problem ten można by rozwiązać kupując droższy czujnik oparty na module max 30102, pracujący na zasilaniu 3,3 V. Rozwiązano ten problem przez zasilenie urządzenia przez kabel usb z powerbanka, niestety powoduje to konieczność użycia kabla, i lepszym rozwiązaniem byłoby kupienie droższego czujnika, ponieważ mógłby on być uruchomiony przy zasilaniu z baterii.

- Kluczowym wyzwaniem programistycznym okazała się obsługa stosu Bluetooth Low Energy na mikrokontrolerze Raspberry Pi Pico 2 W który miał problemy z automatycznym wykrywaniem deskryptorów urządzenia podrzędnego

-W obecnej konfiguracji część mobilna wymaga zasilania przewodowego USB co ogranicza komfort użytkowania podczas snu i wymusza w przyszłości wymianę sensora na model kompatybilny z napięciem 3.3V

- Mimo problemów z zasilaniem zrealizowano główny cel projektu jakim jest integracja pomiarów środowiskowych i biometrycznych oraz ich archiwizacja na karcie microSD z wykorzystaniem znacznika czasu .